UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA INDUSTRIAL

# DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA PARA LA CLASIFICACIÓN DE TWEETS SEGÚN SU POLARIDAD

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL INDUSTRIAL

PABLO ANDRÉS TAPIA CARO

PROFESOR GUÍA:
SR. JUAN DOMINGO VELÁSQUEZ SILVA

MIEMBROS DE LA COMISIÓN:
SR. ALBERTO CABEZAS BULLEMORE
SR. IGNACIO CALISTO LEIVA

SANTIAGO DE CHILE
MAYO 2014

# Resumen Ejecutivo

RESUMEN DE LA MEMORIA
PARA OPTAR AL TITULO DE
INGENIERO CIVIL INDUSTRIAL
POR : PABLO TAPIA CARO
FECHA: 26/05/2014
PROF. GUIA: SR. JUAN VELÁSQUEZ

La alta penetración de Twitter en Chile ha favorecido que esta red social sea utilizada por empresas, políticos y organizaciones como un medio para obtener información adicional de las opiniones de usuarios acerca de sus productos, servicios o ellos mismos. Al ser los comentarios en Twitter, por defecto, de carácter público, se pueden analizar con el fin de extraer información accionable. En particular las empresas además de estar interesadas en la información cuantitativa, les interesa saber bajo qué polaridad se efectúan estas menciones, por cuanto una variación positiva en el número de comentarios puede deberse a un mayor número de menciones tanto positivas como negativas.

Si bien existen un número considerable de softwares que vienen con la funcionalidad de detección de polaridad de sentimientos, estos no son de mucha utilidad ya que la forma en que interactúa el usuario chileno con esta plataforma está llena de modismos propios de nuestro lenguaje local y abreviaciones que se deben principalmente a la limitación de caracteres de Twitter.

Al ser esta una industria inmadura en Chile, la tarea de detección de polaridad de sentimientos, se está realizando de forma manual por agencias publicitarias y otro tipo de empresas, pero dado el gran número de comentarios que se producen minuto a minuto, esta tarea resulta muy demandante en tiempo y dinero. Para resolver este tipo de problemáticas se utilizan técnicas de aprendizaje automático con el fin de entrenar un algoritmo que luego pueda determinar si un comentario es positivo, negativo o neutro, campo que se conoce como sentiment analysis. Mientras más datos sean procesados para el entrenamiento del algoritmo, mejor es el desempeño del clasificador y como en Twitter es sencillo obtener comentarios mediante su API, a diferencia de la web, se han formulado técnicas para generar automáticamente la corpora que contiene los tweets de entrenamiento para cada una de las clases y así sacar provecho de esta propiedad.

En este trabajo se profundiza el uso de una metodología semiautomática basada en emoticons para la generación de una corpora de tweets para la detección de polaridad de sentimientos en Twitter. Esto se realiza introduciendo un nuevo enfoque para la consolidación de los datos de entrenamiento mediante filtros que mejoran el etiquetado automático. Esto permite prevenir la aparición de comentarios erráticos y que causan ruido en las fases de entrenamiento y clasificación. Además se introduce una nueva clase de tweets que no se había considerado anteriormente, que consiste de tweets que carecen de información suficiente para clasificarlos como positivos, negativos o neutros, por lo que clasificarlos en alguna de estas clases disminuye la precisión del sistema. Evaluaciones experimentales mostraron que el uso de esta cuarta clase denominada “irrelevante” con el criterio de filtros presentado para la generación de la corpora, mejora el desempeño del sistema. Además se comprobó experimentalmente que el uso de una corpora generada en base a tweets chilenos clasifican mejor a los comentarios originados por usuarios locales.

*I believe it is our fate to be here. It is our destiny. I believe this night holds, for each and every one of us, the very meaning of our lives. This is a war and we are soldiers. What if tomorrow the war could be over? Isn't that worth fighting for? Isn't that worth dying for?*
*- The Matrix*

# Agradecimientos

Este es el final de un proceso del que me siento muy agradecido, por todas personas que me rodean, mis amigos y por todas las oportunidades que se me han presentado a lo largo del camino, de la cuales he intentado aprender y crecer.

Esta memoria va dedicada a mi papá José, por enseñarme a ser una buena persona, a mi madre Patricia que desde pequeño me inculcó la idea de colocarse metas altas, a mi hermano Gabriel, por mostrarme lo que es dar sin esperar nada a cambio, y a mi *Patio*, Harumi que ha sido mi apoyo incondicional desde que la conocí. Los amo.

Pablo Andrés Tapia Caro

# Tabla de Contenido

# Índice de Tablas

# Índice de Figuras

# Capítulo 1

# Introducción

La toma de decisiones es el proceso durante el cual una persona u organización debe elegir entre dos o más alternativas. Todos y cada uno de nosotros ocupamos gran parte de nuestros días tomando múltiples decisiones, algunas de las cuales tienen una importancia relativa para el desarrollo de una organización, mientras otras son totalmente decisivas para ellas.

La información es un recurso esencial para la toma de decisiones. Mientras más información existe, se incurre en un menor riesgo al disminuir la incertidumbre. Este es el principal foco y objetivo de los estudios de mercado, definidos como el proceso sistemático de recolectar y analizar información sobre clientes, competidores y el mercado. Su uso incluye asistencia en la creación de planes de negocio, el lanzamiento de un nuevo producto o campaña, mejoramiento de servicios actuales y la expansión a nuevos mercados, entre otros.

Además, estudios de mercado son utilizados para determinar qué proporción de un mercado compraría un producto o servicio dependiendo de su afinidad o preferencia por una marca o compañía, basado en variables sociodemográficas como sexo, edad, localización y renta. Esta motivación de recolectar tanta información como sea posible para reducir la incertidumbre al momento de tomar decisiones, ha contribuido que la web se transforme en un importante caso de estudio por su casi inagotable cantidad de información (y opiniones) que existen en ella.

## 1.1 Situación Actual

Dada la importancia de Twitter en el mercado chileno y el mundo [1, 2], hay un creciente número de empresas interesadas en analizar los comentarios generados en esta red social. A diferencia de otras plataformas sociales la mayoría de los posts en twitter son públicos, lo cual facilita su análisis y es esto de lo que las empresas de investigación de mercado y agencias de medios digitales quieren tomar ventaja.

El interés de los clientes por contratar este tipo de servicios está justificado por la enorme tasa de uso que posee esta red social el cual puede facilitar la tarea de comprender sobre

qué, cómo y por qué las personas están hablando de ellos en las redes sociales. Herramientas elaboradas a partir de información extraída de estos medios puede responder preguntas como: ¿Cuántas personas están hablando de mi marca? ¿Hay un incremento del número de personas que interactúan con mi cuenta? ¿Cuánto nombran mi marca con respecto al mes anterior? ¿Cuáles son los temas a los que más se asocia mi marca y con los que más aparece?

La mayoría de las herramientas disponibles en el mercado ofrecen análisis y mediciones de datos cuantitativos como por ejemplo: Cantidad de comentarios que incluyen una palabra, detectar la variación en el número de menciones según una unidad de tiempo, variaciones en la cantidad de seguidores de un usuario, etc. Como fue mencionado anteriormente, estudios cuantitativos acerca de menciones pueden ayudar a detectar un suceso importante pero esto es ofrecido por una gran cantidad de softwares disponibles en línea, por lo que actualmente estos datos son considerados commodities en esta industria. Por ejemplo ¿Qué tan relevante para un político es saber que tiene 5000 menciones la última semana? ¿Es esta información accionable por el cliente? ¿Puede tomar decisiones basadas en esta información?

Si bien esta información puede entregar valor al momento de ser comparado este indicador con el del periodo anterior, de esto solo se puede saber si la audiencia ha aumentado o disminuido, pero esta variación en número de comentarios puede ser una consecuencia de un aumento en el número de comentarios positivos (promotores) o negativos (detractores). Esta información puede ser de gran interés para el cliente con tal de determinar cuánto y bajo qué polaridad los usuarios se refieren a su marca o persona en las redes sociales.

Una técnica adoptada por los softwares hoy en día es otorgar la posibilidad a sus usuarios de clasificar manualmente los comentarios de acuerdo a su polaridad con el fin de obtener una estadística que les permita responder este tipo de preguntas. Esto es, si el comentario es positivo, neutral o negativo. Pero el problema es que el gran número de comentarios realizados minuto a minuto, excede la capacidad que tiene un analista, o un grupo de estos para procesar y etiquetar estos datos. Es por esta razón que son utilizadas técnicas de minería de datos y algoritmos de Machine Learning para llevar a cabo esta tarea, la cual es conocida como Sentiment Analysis.

Otros softwares más especializados ofrecen la posibilidad de realizar Sentiment Analysis de forma automática. Las técnicas más efectivas para llevar a cabo estas tareas corresponden a una rama específica del Machine Learning llamada Entrenamiento Supervisado o Supervized Learning Estos algoritmos son entrenados mediante data preclasificada, que en este caso son comentarios (tweets) etiquetados según su polaridad positivos, negativos o neutros. Marrese en [3] define los algoritmos supervisados como la generación de una función matemática que es capaz de colocar o clasificar inputs en un set de outputs dados. Estos outputs son llamados labels y para generar los pares de entrenamiento son usualmente asignados a los documentos mediante un proceso de clasificación manual.

La data preclasificada y etiquetada por cada clase es llamada Corpus, que en su conjunto conforman la Corpora. La Corpora toma especial importancia ya que es uno de los factores más relevantes en la efectividad de un clasificador supervisado. Al ser tweets los datos usados para entrenar el algoritmo, resulta imperativo la generación de una Corpora robusta y adhoc a la realidad del lenguaje en la red social, además de la extracción de las características

principales (key features) que definen un tweet como positivo, negativo o neutro. Es en este punto donde la mayoría de los softwares fallan en su módulo de Sentiment Analysis ya que no consideran las características propias del lenguaje de twitter en su entrenamiento, y no están adaptadas para el lenguaje chileno.

### 1.1.1 Descripción y justificación del proyecto

La penetración de los smartphones en Chile, el incremento de las conexiones 3G, la adopción de su uso como coplataforma a la TV y otros factores han favorecido que Twitter se haya convertido en la segunda red social con mayor cantidad de usuarios en Chile. Considerando que en Twitter los usuarios discuten y conversan de marcas, políticos, empresas etc., esta red social se ha convertido en un espacio lleno de opiniones que son de especial interés para marcas, políticos, agencias de publicidad, etc. Permitiéndoles convertir todos estos comentarios en información accionable que puede apoyar el proceso de toma de decisiones o simplemente alertar cuando se produce una variación inesperada.

Este trabajo contempla la prueba y desarrollo de un proceso semi-automatizado para la generación de un corpus de data preclasificada, y un algoritmo de clasificación que permita etiquetar a los tweets de acuerdo a su polaridad (positivo, negativo o neutro) con el objetivo de otorgar información adicional a las empresas.

## 1.2 Objetivos

### 1.2.1 Objetivo General

Desarrollar un sistema clasificador automático de comentarios realizados en Twitter, según su polaridad (i.e positivo, negativo, neutro), considerando las características especiales del lenguaje local al interactuar con la plataforma.

### 1.2.2 Objetivos específicos

- Generar un Corpus de tweets en español que considere modismos chilenos y las características específicas con que los usua rios chilenos interactúan en esta red social.
- Investigar y poner a prueba distintas metodologías que permitan obtener una mejora al actual estado del arte sobre clasificación de polaridad de sentimientos en Twitter.
- La implementación del sistema clasificador online con la capacidad de procesar y clasificar en tiempo real al menos 100 tweets por minuto.

## 1.3 Hipótesis de investigación

La gran penetración Twitter en el contexto chileno, en conjunto con las propiedades de sus comentarios y su carácter público, hacen de esta red social una fuente ideal para la extracción

de conocimiento. Este trabajo sugiere que es posible generar un corpus mediante una metodología semiautomatizada introducida por Pak y Paroubek en [4] para el entrenamiento de un algoritmo de detección de polaridad, incorporando las características del lenguaje chileno y la forma en que los usuarios interactúan en esta plataforma. Además sugiere que el hecho de considerar únicamente comentarios emitidos por usuarios chilenos como parte de los datos de entrenamiento, permite una mejor clasificación de los mismos.

## 1.4 Metodología

La metodología utilizada considera la necesidad de investigar, seleccionar, implementar y desarrollar distintos mecanismos para llevar a cabo el proceso de Sentiment Analysis. Siendo este una tarea de especial relevancia por los motivos mencionados anteriormente, existen muchos trabajos e investigaciones desarrollados para llevar a cabo esta tarea, pero la gran diferencia en esta memoria es que se realizará para el lenguaje español, considerando además modismos y características especiales de esta red social con el fin de aumentar la precisión en la clasificación.

### 1.4.1 Fases de la metodología

- Investigación
  - Estado del arte del Opinion mining en la web y twitter
  - Identificación de metodologías y técnicas utilizadas
  - Identificar un aspecto del estado del arte donde se puede generar un aporte
- Desarrollo y validación de experimentos
  - Desarrollo de prototipos utilizando distintas metodologías para mejorar el sistema clasificador
  - Experimentación usando estas metodologías
  - Validación y comparación de los resultados
  - Selección de la mejor configuración a partir de los resultados
  - Validación de la hipótesis
- Implementación del sistema

La fase de Investigación tiene por objetivo generar un compilado de un número importante de las técnicas ya desarrolladas para la realización de Sentiment Analysis en Twitter y la Web con el fin de seleccionar aquellas técnicas que hayan tenido mejores resultados y considerando las necesidades y características de este proyecto para su implementación. El objetivo fundamental de esta etapa es identificar un aspecto del actual estado del arte donde se pueda realizar una mejora para la clasificación de polaridad de sentimientos en Twitter, además de comprender como se lleva a cabo esta tarea actualmente.

La segunda fase corresponde a la generación del corpus mediante la definición de un set de reglas y mecanismos que permitan una rápida selección y etiquetado de los tweets para cada clase. Por otro lado considera la generación de un algoritmo de entrenamiento el cual será mejorado aplicando e incorporando las distintas técnicas seleccionadas en la fase previa. Finalmente la configuración de técnicas seleccionadas será aquella que mejor responda al objetivo del algoritmo el cual es obtener el mayor nivel de accuracy en la clasificación de tweets en español para comentarios chilenos.

La evaluación utilizará el concepto de 10-Fold Cross Validation donde el Corpus es dividido entre 10 partes (folds), dejando una de lado para entrenar el algoritmo con todo el resto y evaluarlo con la que fue previamente separada. Este proceso es realizado 10 veces dejando siempre uno de los folds de lado y luego validando el algoritmo con él. El accuracy del modelo es finalmente calculado como el promedio del accuracy obtenido para cada fold. Para este tipo de trabajo los ratios más utilizados para la validación de algoritmos son accuracy, recall y precisión los cuales serán explicados en el Marco Conceptual. Además para contrastar y evaluar el sistema con datos que no sean los recolectados en la fase de entrenamiento, otra validación se llevará a cabo mediante la clasificación de 100 tweets pre-etiquetados manualmente por cada clase. Esto permitirá obtener otros valores de evaluación que mostrarán que tan efectivo es el sistema para comentarios extraídos aleatoriamente de twitter, que no necesariamente poseen la estructura de los datos de entrenamiento.

Finalmente la última fase es la implementación del sistema en el sistema el cual permitirá en tiempo real la clasificación de tweets con tal de saber bajo que polaridad se distribuyen los comentarios relacionados a una marca, persona o industria.

## 1.5 Resultados esperados

Este trabajo propone los siguientes resultados esperados:

- Un marco conceptual que resume las técnicas más importantes y utilizadas para el Web Opinion Mining en un Estado del Arte.
- Una mejora a la metodología basada en reglas para un etiquetado automático de los datos de entrenamiento, para la generación de un Corpus de tweets positivos, negativos y neutros.
- Un Corpus de tweets positivo negativo y neutro para el entrenamiento y evaluación del desempeño de los algoritmos para la detección de polaridad en formato txt y en una base de datos.
- Un conjunto de técnicas y su implementación para el proceso de clasificación mediante Sentiment Analysis

## 1.6 Contribuciones de la memoria

Gracias a la investigación llevada a cabo para escribir el estado del arte, se implementó un sistema para la generación semiautomática de una corpora de tweets, dónde se notaron ciertas problemáticas presentes en el enfoque tradicional. En este trabajo y en el paper adjunto, se proponen una serie de filtros para evitar la aparición de comentarios en los corpus que no hacen más que ruido en el proceso de entrenamiento y clasificación. Por otro lado se considera un nuevo enfoque para la clasificación de polaridad de sentimientos en Twitter al incluir una cuarta clase de tweets para clasificar comentarios que no pertenecen a ninguna de las clases consideradas tradicionalmente. A esta clase pertenecen todos aquellos tweets que carecen de contenido o coherencia y por lo tanto no presentan una polaridad definida. El considerar esta cuarta clase lleva a un aumento en el desempeño del sistema ya que Twitter está lleno de este tipo de comentarios.

## 1.7 Estructura de la memoria

El resto del documento está estructurado de la siguiente manera: El siguiente capítulo de Marco Conceptual (Capítulo 2) contiene todas las definiciones y conceptos acerca del estado del arte actual y que serán utilizados posteriormente. Estos conceptos incluyen algunos aspectos básicos de la historia de la web, data mining, web mining, procesamiento de lenguaje natural y web opinion mining y características, definiciones y componentes de Twitter. En esta sección se pone énfasis en los métodos y componentes que debe tener un clasificador de polaridad de sentimientos, y se muestran enfoques actuales para resolver esta tarea.

Después de las definiciones dadas, en el capítulo 3 se presenta la metodología a utilizar para la generación de la corpora. Esta será utilizada para el entrenamiento del algoritmo. También en este capítulo se hace un análisis de los datos obtenidos mediante la metodología aceptada, encontrándose distintos tipos de tweets que no contribuyen, o simplemente no corresponden a cada corpus. Para resolver esta situación se exponen en este capítulo distintas metodologías para evitar la aparición de estos tipos de tweets en la corpora y se realiza finalmente un nuevo análisis de la estructura que posee cada corpus.

En el capítulo 4 se explican distintos modelos para la extracción de features y los algoritmos a utilizar en el sistema clasificador. Se presentan en detalle los supuestos, técnicas y algoritmos a considerar para la clasificación de tweets.

En el capítulo 5 se detalla cómo se implementará el sistema en base a los requerimientos y objetivos de este. Se detalla la función de cada módulo incluyendo cómo se extraen los datos para generar el primer set de entrenamiento, cómo se aplican los filtros para mejorar esta metodología, la extracción de features y persistencia de los datos generados, el módulo de recolección de tweets a clasificar, el sistema clasificador y por último el evaluador.

Finalmente en el capítulo 6 se presentan los resultados obtenidos en base a dos criterios de evaluación y se agrega un caso de estudio donde se utiliza el sistema para clasificar los comentarios emitidos en la red social para un debate presidencial acontecido el 29 de Octubre del año 2013. El Capítulo 7 presenta las conclusiones obtenidas a partir de este trabajo.

# Capítulo 2

# Marco Conceptual

Este marco conceptual pretende dar al lector una explicación general y clara de los conceptos esenciales y básicos de los cuales se van a desarrollar el proyecto de tesis. Esto implicará un breve repaso de la evolución de la Web, la necesidad que poseemos de guardar información, la evolución de los computadores y como esto ha propiciado el hecho que la Web se llene de opiniones. Además se introducirán conceptos básicos de *Opinion Mining* y como esta puede ser llevada a cabo en Twitter a partir de los trabajos realizados hasta este momento, en un estado del arte del *Web Opinion Mining* en Twitter.

## 2.1 La *World Wide Web* y su evolución

### 2.1.1 La necesidad de información

En la humanidad siempre ha existido la necesidad de guardar información. Desde tiempos antiguos los hombres han almacenado información para resolver requerimientos básicos como los de conservar el conocimiento en sectores tan diversos como oficinas de gobierno, la iglesia, hospitales y negocios. Hasta hace algunas décadas los datos eran almacenados en documentos o libros y organizados en librerías o depósitos de archivos. Estos tipos de almacenamiento tienen distintos tipos de problemas como la conservación y organización de los documentos, actualizarlos, y el gran volumen necesario para conservarlos en un ambiente físico.

Por estas razones la masificación de la computación ha jugado un rol fundamental respecto a este tema. Los computadores se han vuelto más económicos y poderosos. La tasa de crecimiento de esta tecnología fue predicha por Moore en [5] el año 1965 donde estableció que por lo menos por 10 años más el número de componentes en un chip se doblaría cada año, cifra que luego fue reducida a 18 meses. Esto significa que cada 2 años los computadores serían el doble de poderosos. Estos avances tecnológicos vinieron a resolver los problemas de almacenaje y procesamiento de información que tenían las personas.

Una vez que toda esta data está almacenada, la pregunta que surge es qué hacer con toda ella y como conservarla, organizarla y hacerla disponible cuando es requerida. Este

es el principal objetivo de las bases de datos relacionales, que como son descritas en [6] corresponden a un repositorio de datos provenientes de un mismo contexto organizados en tablas que contienen relaciones unas con otras.

Por otro lado el aumento en la capacidad de almacenaje de los computadores ha permitido que el volumen de la data almacenada esté creciendo exponencialmente. Esto está ocurriendo en todas las áreas donde las personas se desenvuelven, desde tareas comunes como transacciones financieras, en supermercados, retail, detalles de llamadas telefónicas, estadísticas e indicadores financieros, así como también en tareas más extravagantes como es el caso de imágenes astronómicas, bases de datos moleculares y registros médicos [7]. Este gran volumen de datos resulta imposible de analizar, dejándolo de lado a la hora de tomar decisiones (proceso en el cual es fundamental contar con toda la información posible para reducir la incertidumbre). Para resolver este problema, se origina el *data mining* que consiste en la extracción de información oculta y predecible a partir de grandes bases de datos.

### 2.1.2 La Web 1.0

Antes de conversar de la Web como la conocemos hoy es necesario volver a sus orígenes. En 1980 la Web nació de la mano de Tim Berners-Lee como un proyecto para almacenar y compartir información para el Centro Europeo de Estudios Nucleares (CERN) [8]. Luego se estandarizaron los protocolos y se hizo accesible a todo el público que contara con un computador y una conexión a Internet. Pero únicamente los usuarios con conocimiento más avanzado de computación pudieron acceder a la confección y desarrollo de sitios Web. Por esta razón y la incipiente penetración de la computación en la sociedad es que en sus inicios la Web para la mayoría de los usuarios tenía una naturaleza pasiva. La Web 1.0 (como se llama a la Web antigua) estaba conformada por páginas estáticas, donde el usuario promedio estaba pasivamente involucrado consumiendo únicamente la información que los usuarios avanzados generaban. Como es descrito por García en [9], la Web estaba limitada a ciertos sectores, áreas o calidad de los autores para hacer uso de ella y difundir el contenido.

### 2.1.3 El prosumer y la Web 2.0

Debido al aumento de la penetración de Internet y por lo tanto el mayor uso de la Web es que la información contenida en ella toma una mayor importancia, generándose grandes volúmenes de datos relevantes para su estudio. Es así como nació un área del *data mining* específico a lo relacionado con el estudio de la Web llamado *Web Mining*. Este concepto reúne todas las técnicas, algoritmos y metodologías usadas para la extracción de información y conocimiento a partir de la data originada en la Web. Velásquez et.al. en [10] define el concepto de Web Data para datos originados en la Web, que son principalmente de tres fuentes: Objetos de contenido que son todos aquellos objetos contenidos dentro de una página Web, Objetos de estructura que corresponden a la estructura de vínculos (textithyperlinks) presentes en un sitio Web y los Objetos de Uso que tienen relación con los registros de *Web logs* que contienen toda la interacción de los usuarios con un sitio Web. Cada uno de estos tipos de Web data son estudiados por ramas específicas del *Web Mining: Web Content Mining, Web Structure Mining* y *Web Usage Mining*.

Por el año 2004 emergió el concepto de *Web 2.0* también llamada la Web de Todos [11]. Esta evolución de la Web es caracterizada por la facilidad de intercambiar contenido, la generación colaborativa de contenido y el hecho que el diseño estaba centrado en el usuario. García en [9] se refiere a las ventajas de la Web 2.0 con los siguientes atributos (entre otros): Interactividad permitiendo total comunicación en ambos sentidos. Aprendizaje colaborativo permitiendo el trabajo en equipo e intercambio de ideas. La multidireccionalidad del contenido generado dado que las opiniones y respuestas son generadas simultáneamente y su distribución es libre considerando el hecho que cualquiera puede editar y compartir ideas que pueden ser difundidas con un solo clic y ser conocidas por una multitud de personas. A diferencia de la Web 1.0, donde los contenidos eran condicionales a la voluntad y deseo de los administradores del sitio, y donde los usuarios participaban únicamente como consumidores, en la Web 2.0 el usuario se convierte en ambos productor y consumidor de contenido, y su producción está solo limitada a la voluntad del usuario. De aquí surge el concepto de *Prosumer* (como una abreviación de *Producer* y *Consumer*).

Esta facultad que otorga la Web 2.0 a los usuarios en el sentido de generar contenido, ha propiciado el hecho que ésta se llene de opiniones, potenciado principalmente por el surgimiento de blogs especializados y redes sociales dedicadas que permiten la interacción entre usuarios intercambiando y exponiendo información y opiniones. El crecimiento de la Web como medio de comunicación también ha contribuido a establecerse como un lugar en donde se discute de productos y servicios, hacer críticas por malos y buenos servicios e incluso como una fuente de opiniones para tomar una decisión al momento de realizar una compra [12, 13]. En este sentido la Web está llena de reseñas las cuales resultan de gran interés para las compañías, de donde ellas pueden obtener *feedback* adicional a los estudios de mercado tradicionales tales como encuestas, de la forma que ha sido realizado por décadas [14]. Principalmente por estos motivos es que nace una nueva rama específica del *Web Content Mining* llamada *Opinion Mining* que tiene por objetivo extraer la información subjetiva desde las opiniones expresadas en la Web mediante técnicas de *Data Mining* más específicamente, Procesamiento de Lenguaje Natural y *Text Mining*.

## 2.2 Data Mining

Como es mencionado en [15] los datos están siendo recolectados a una tasa cada vez mayor. Esta tarea de convertir los datos en conocimiento antes era parte de un proceso manual de análisis e interpretación la cual es muy demandante en tiempo, dinero y es altamente subjetiva. A medida que los volúmenes de data crecen, este tipo de metodologías se vuelven completamente inútiles en muchos campos. Este hecho tiene como consecuencia la urgencia de una nueva generación de teoría computacional y herramientas que ayuden a las personas a extraer información útil de estos crecientes volúmenes de data digital. Estas herramientas pertenecen a un campo conocido como Descubrimiento de Conocimiento desde las Bases de Datos (KDD) que se enfoca en llevar a cabo esta tarea descubriendo patrones en grandes sets de datos utilizando métodos de inteligencia artificial, *machine learning*, estadísticas y sistemas de bases de datos. El principal objetivo de este proceso es la extracción de información desde un conjunto de datos y transformarlos en una estructura entendible para su uso posterior.

### 2.2.1 El proceso KDD

El término Descubrimiento de Conocimiento en Bases de Datos o KDD, por sus siglas en inglés, se refiere al amplio proceso de búsqueda de conocimiento en los datos, y se enfoca en la aplicación de métodos de minería de datos de alto nivel. Es de interés para los investigadores en el aprendizaje automático (*machine learning*), reconocimiento de patrones, bases de datos, estadísticas, inteligencia artificial, la adquisición de conocimientos de los sistemas expertos, y la visualización de datos.

Los dos objetivos primarios de la mineria de datos son la predicción y descripción. La predicción involucra el uso de variables o campos de una base de datos para presidir futuros o desconocidos valores de otras variables de interés. La descripción se enfoca en encontrar patrones interpretables que describan los datos. La importancia de la predicción y la descripción en determinadas aplicaciones de minería de datos es variable. Sin embargo, en el contexto de KDD, la descripción tiende a ser más importante que la predicción, a excepción del reconocimiento de patrones y aprendizaje automático (como el reconocimiento de voz) donde la predicción es a menudo el objetivo principal del proceso de KDD.

Los términos de descubrimiento de conocimiento y minería de datos son distintos. KDD se refiere al proceso general de descubrimiento de conocimiento útil de los datos. Se trata de la evaluación y, posiblemente, la interpretación de los patrones para tomar la decisión de lo que califica como conocimiento. También incluye la elección de esquemas de codificación, procesamiento previo, toma de muestras, y las proyecciones de los datos antes de la etapa de minería de datos. La minería de datos se refiere a la aplicación de algoritmos para la extracción de patrones de datos sin los pasos adicionales del proceso de KDD.

El proceso general de encontrar e interpretar de los patrones de datos implica la aplicación repetida de los siguientes pasos [15]:

1. Desarrollo y entendimiento de
    - El objetivo de la aplicación
    - Las metas del proceso KDD desde el punto de vista del usuario final
2. Creación de un conjunto de datos de destino: la selección de un conjunto de datos, o concentrarse en un subconjunto de variables, o muestras de datos, en la que el descubrimiento se va a realizar.
3. Limpieza y preprocesamiento de los datos
    - Eliminación de ruido y *outliers*
    - Recolección de información necesaria para modelar el ruido
    - Definición de estrategias para manejar datos perdidos (missing values)
4. Reducción y proyección de los datos
    - Encontrar características o *features* útiles que representen la data dependiendo del objetivo del proceso

- Usar reducción de dimensionalidad de los datos o métodos de transformación que reduzcan efectivamente el número de variables a considerar.

5. Seleccionar la tarea de *data mining* a realizar
   - Usar técnicas de regresión, clasificación, *clustering*, etc. Dependiendo del objetivo del proceso KDD
6. Elección del algoritmo de *Data Mining*
   - Selección de el, o los métodos para buscar patrones en la data
   - Decidir cuáles modelos y parámetros pueden ser los apropiados
   - Coincidir el método de minería de datos con el criterio y objetivo del proceso KDD.
7. *Data Mining*
   - Buscar patrones de interés en una representación de los datos incluyendo reglas de clasificación, árboles, regresiones y *clustering*.
8. Interpretación de los patrones obtenidos
9. Consolidación del conocimiento adquirido usándolo directamente, incorporando el conocimiento en otro sistema para su uso posterior, o simplemente documentarlo y reportarlo a las partes interesadas

El proceso KDD puede involucrar varias iteraciones y contener *loops* entre dos etapas cualquieras. El flujo básico está ilustrado en la figura 2.1. La mayoría de los trabajos se enfocan en la fase de *data mining* (etapa 7), pero las demás etapas son también importantes para una aplicación exitosa del proceso KDD.

Los conjuntos de tareas de *data mining* del punto 4 se dividen en: Clasificación, regresión, *clustering*, abreviación, modelamiento de dependencias y detección de cambio y desviaciones. La tarea de clasificación es una función de aprendizaje que mapea un dato en una clase de un conjunto predefinido. La regresión es también una función de aprendizaje, que en este caso mapea un dato en una variable de valor real. El *clustering* es una técnica descriptiva que busca identificar un conjunto finito de categorías o *clusters* para describir los datos. Está estrechamente relacionada con la estimación de densidad de probabilidad que consiste en una técnica de estadística para estimar desde la data la matriz de multivarianza. La abreviación, *summarization* en inglés, se enfoca en encontrar una descripción resumida para un subconjunto de los datos. El modelamiento de dependencias consiste en encontrar un modelo que describa dependencias significativas e importantes entre las variables. Estas dependencias pueden ser del tipo estructural o cuantitativo. Por último la detección de cambio y desviación se enfoca en el descubrimiento de cambios significativos de los datos.

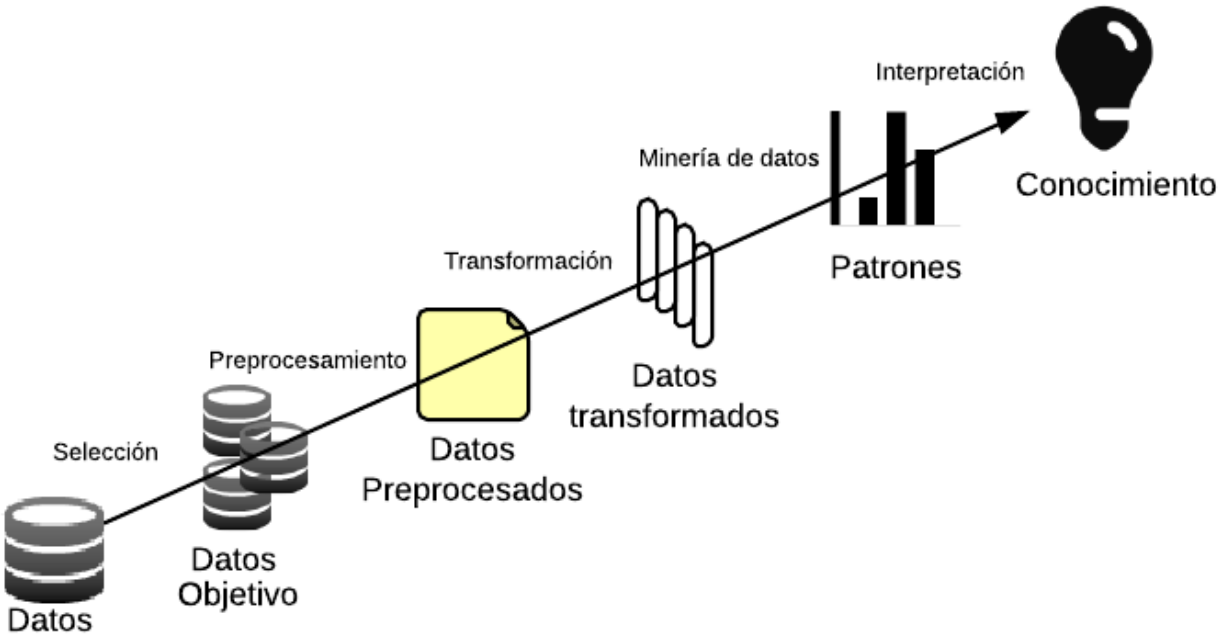



Figura 2.1: Proceso KDD
Fuente: Confección propia

### 2.2.2 *Machine Learning*

Para resolver un problema en un computador es necesario el uso de un algoritmo. Un algoritmo es una secuencia de instrucciones que deben ser llevadas a cabo para llevar un input inicial (entrada) a un output (salida) [16]. Un ejemplo clásico de un algoritmo son los de ordenamiento, los cuales ordenan un conjunto de datos según un criterio especifico como podría ser el alfabético. En este caso el input es un set de palabras y el output es una lista ordenada según el criterio bajo el cual fue construido el algoritmo. Para el mismo problema pueden existir muchos algoritmos que lleven a cabo esta tarea, y el interés está en encontrar aquel que es el más eficiente requiriendo el menor número de instrucciones y uso de memoria.

Por otro lado para algunas tareas no existen algoritmos que por ejemplo puedan discriminar entre mails que corresponden a spams o correos legítimos. En este caso el input corresponde a un documento (*email*) que en su caso más simple es un conjunto de caracteres. Además se pretende que el output o salida del programa sea una clasificación si el correo de entrada corresponde o no a spam, pero no se conoce la forma en que el input puede ser convertido en el output deseado.

Uno de los fundamentos del *Machine Learning* es que existe un proceso que describe los datos que se observan. A pesar que no se conocen los detalles de cómo estos datos son generados, se sabe que este no es aleatorio y quizás no sea posible identificar el proceso completo, pero supone que se puede hacer una buena aproximación [16]. Gracias al *Machine Learning*, lo que no se tiene en conocimiento se puede obtener a partir de los datos. Se

pueden compilar miles de mensajes de entrenamiento en donde ya se sabe a priori que tipo de mail son (spam o no) y lo que se desea es que un sistema "aprenda" que tipo de mensajes corresponden a spam y cuáles no. En otras palabras el objetivo es que un computador extraiga automáticamente el algoritmo para realizar esta tarea. Con los avances en computación en términos de almacenamiento y procesamiento se pueden llevar a cabo este tipo de tareas.

Como es definido en [17] *Machine Learning* estudia algoritmos computacionales que permiten a los computadores aprender a hacer tareas o predicciones con cierto grado de certidumbre. Este aprendizaje es realizado a partir de datos como ejemplos, experiencia directa o instrucciones. Por lo que en general, *Machine Learning* es aprender a hacer ciertas tareas basado en observaciones o ejemplos del pasado.

Los algoritmos de *Machine Learning* se dividen en varios tipos dependiendo del output deseado y el tipo de input entregado al momento de entrenamiento del algoritmo. Entre otros se encuentran:

**Algoritmos Supervisados:** Este tipo de algoritmos generan una función matemática con el fin de asignar inputs a un conjunto de outputs deseados. Estos outputs son frecuentemente llamados etiquetas o *labels* ya que son usualmente el resultado de un proceso de etiquetado manual. Este tipo de algoritmos son entrenados mediante datos preseleccionados llamados datos de entrenamiento (*training examples*), que están conformados usualmente por un par consistente en un objeto de entrada (usualmente un vector de características) y el valor de salida (también llamado *tag*, etiqueta o *label*).Un algoritmo de entrenamiento supervisado, una vez que ya es entrenado produce una función derivada que puede ser utilizada para clasificar o posicionar nuevos ejemplos. El escenario óptimo es aquel en que el algoritmo es capaz de determinar correctamente las clases o etiquetas de nuevos y desconocidos ejemplos. Para esto es necesario que el algoritmo de entrenamiento sea capaz de generalizar desde la data de entrenamiento a ejemplos no contenidos de una forma "razonable" [18].

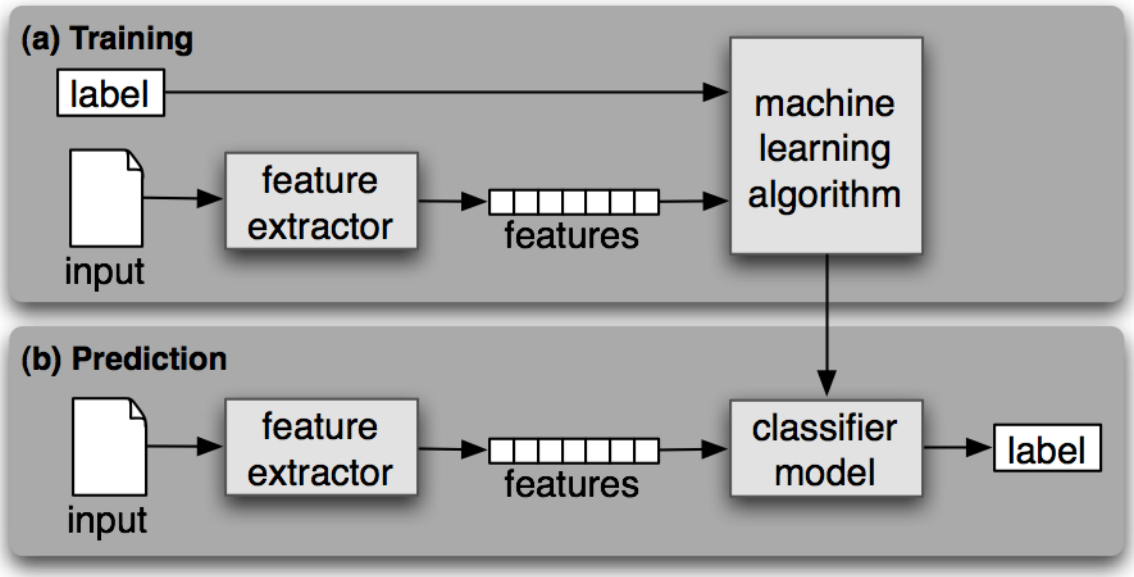



Figura 2.2: Modelo de un algoritmo supervisado
Fuente: Confección propia

**Algoritmos No Supervisados:** Estos algoritmos no intentan asignar un input a un conjunto de valores de salida, sino que simplemente modelar el set de valores de entrada. En este caso los valores de salida no son conocidos y el objetivo es descubrir la estructura que poseen los datos. En otras palabras hay ciertos patrones de estructura que ocurren en los datos y el objetivo es descubrir que patrones ocurren generalmente y cuáles no. En estadística, estos métodos son llamados *density estimation* [16]. Un ejemplo clásico de algoritmos no supervisados es el *clustering* que tiene por principal objetivo encontrar grupos de valores de entrada que dentro del *cluster* contengan características similares y sean lo más disímiles posible con el resto de los *clusters*.

### 2.2.3 Minería de textos

La minería de opiniones encapsula un gran número de técnicas que forman parte del *Text Mining*, la que a su vez también es un sub-campo de la minería de datos. La principal diferencia entre el *text mining* y *data mining* es que en general el tipo de datos utilizados para el primero tipo son documentos compuestos por caracteres alfanuméricos. El *Text Mining* se define como la extracción y descubrimiento de conocimiento no trivial a partir de textos inestructurados. Esto contempla un gran área de estudio desde *Information Retrieval* (IR), clasificación de textos y *Clustering* hasta extracción de entidades, eventos y relaciones de ellos. *Natural Language Processing* (NLP) corresponde al intento por extraer una representación más acabada del significado de los textos. NLP usa típicamente conceptos lingüísticos tales como *part-of-speech* (POS tagging) y estructuras gramaticales además de su intento por tratar con distintos elementos gramaticales como anáforas y ambigüedades. Para realizar esto se usan distintas representaciones y softwares como *lexicons* de palabras que contienen sus significados y sus propiedades gramaticales, además de conjuntos de reglas gramaticales y usualmente otros recursos tales como la ontología de entidades y diccionarios de sinónimos y abreviaciones [19].

El proceso de *Text Mining* consiste principalmente en la aplicación de las siguientes técnicas:

**Tokenización** Las palabras y tokens (conjuntos de caracteres) son considerados en general la unidad básica en casi todas las teorías lingüísticas para procesamiento de textos [20]. La tokenización de frases corresponde al proceso de transformación de cadenas de caracteres a cadenas de palabras, y es el primer paso en el Procesamiento de Lenguaje Natural (NLP) [21].

El objetivo detrás de la tokenización es separar el texto de un documento en una secuencia de tokens o términos. Existen muchas y variadas alternativas de cómo y qué determina estos puntos de separación. Uno de los criterios más utilizados es considerarlos como todos los caracteres no alfabéticos. Esto tiene como resultado tokens consistentes de una sola palabra, que para muchos casos es el la opción más apropiada antes de finalmente construir la representación de un documento como un vector de características (*feature vector*). También si se quieren considerar frases o varias tokens probablemente se realizarán las separaciones con caracteres de puntuación. El método utilizado tiene relación con el objetivo que posee el sistema y el que permita la mejor forma de extraer las características claves (*key features*)

de los textos.

**Borrado de *Stopwords*** Las *Stopwords* son una clase de palabras que son típicamente cortas y ocurren muy frecuentemente en un lenguaje. Usualmente las *Stopwords* tienen una función puramente gramatical en las oraciones y no añaden gran significado al contexto. Aunque estas palabras aparezcan muchas veces en los textos no agregan valor ya que son utilizadas principalmente para unir palabras en una oración. Es comúnmente entendido y aceptado que las *stopwords* no contribuyen al contenido de los textos o documentos [22].

Debido a su alta frecuencia de aparición, su presencia para el *text mining* es un obstáculo para comprender y analizar el contenido de los documentos, ya que muchos algoritmos se basan efectivamente en la frecuencia de las palabras dentro de los documentos para extraer patrones. Para eliminar estos efectos, gran parte de los softwares y metodologías de *Text Mining* en general, usan un listado de *stopwords* para identificarlas y removerlas, y así evitar su procesamiento y análisis. Por lo general las *stopwords* incluyen las siguientes figuras gramaticales: artículos, conjunciones, pronombres, verbos auxiliares y preposiciones [23]. Algunos ejemplos de *stopwords* en español son: "alguna", "el", "así", "cuando", "mientras", "tener". Por otro lado el desarrollo de esta lista de *stopwords* es difícil e inconsistente entre distintas fuentes de datos y lenguajes.

***Stemming* y lematización** Un algoritmo de *stemming* es un procedimiento computacional que reduce todas las palabras con la misma raíz a una forma común, usualmente removiendo los sufijos respectivos a cada palabra. El *stem* (o tallo) no tiene que ser idéntico a la raíz morfológica de la palabra, pero usualmente esto es suficiente ya que las palabras relacionadas por lo general poseen el mismo *stem*, incluso si el tallo no es una raíz válida. Algoritmos para el *stemming* han sido estudiados desde los años 60s. Muchos motores de búsquedas tratan las palabras con los mismos *stems* como sinónimos para obtener resultados expandidos de las búsquedas, proceso que se conoce como *conflation*.

Por su naturaleza computacional, un algoritmo de *stemming* tiene limitaciones inherentes. Los procedimientos son llevados a cabo sobre palabras individuales por lo que no se tiene acceso a la información sobre sus relaciones gramaticales o semánticas entre una palabra y otra, es decir, no considera el contexto. De hecho está basado en un supuesto que las palabras con el mismo *stem* tienen un significado muy similar o el mismo. Si bien este supuesto es correcto en la mayoría de los casos, en inglés es cuando tiene su mejor desempeño por la estructura de sufijos que tienen las palabras. Es una aproximación mejor o peor dependiendo del uso que se le da a los *stems*, variaciones semánticas de las raíces de las palabras y el poder del algoritmo (que tan radicalmente transforma las palabras a su *stem*). Un algoritmo de *stemming* suficientemente poderoso para agrupar todas las palabras con la misma raíz podría ser inadecuado para por ejemplo contabilizar la frecuencia de las palabras. Para ese caso uno esperaría, por ejemplo, que el par neutrón, neutralizar no coincidiera y por esto trabajar con una lista muy limitada de sufijos a truncar [24].

Hoy en día, probablemente uno de los algoritmos más comunes para *stemming* en inglés, y que ha sido repetidas veces probado de ser muy efectivo empíricamente, es el algoritmo de Porter desarrollado por Martin Porter el año 1980 [25]. El algoritmo que Porter propuso consiste en 5 fases de reducción de las palabras que son aplicados secuencialmente. Cada una

de estas fases posee ciertas reglas que son automáticamente seleccionadas y aplicadas a los textos.

Como fue mencionado anteriormente un algoritmo de *stemming* es un proceso de normalización lingüística, donde las formas variantes de una palabra son reducidas a una forma común. Por ejemplo:

$$\left.\begin{array}{c} coneccion \\ conecciones \\ conectar \\ conectando \end{array}\right\} conec$$

Es importante recordar que se utiliza *stemming* con la intención de mejorar la performance de *Information Retrieval* (IR) en los sistemas. Este no se trata de un ejercicio de etimología o gramática, de hecho del punto de vista gramatical o etimológico un algoritmo de *stemming* puede llevar a cometer muchos errores. La finalidad de estos algoritmos es reducir la dimensionalidad de los problemas y agrupar palabras con un significado similar para que posteriormente se puedan asociar y contribuyan de la misma forma en los algoritmos de *data mining*.

Para algunos lenguajes como el Chino, el concepto de *stemming* no es aplicable, pero es ciertamente de mucha utilidad para muchos otros lenguajes del grupo Indo-Europeo. En estos lenguajes las palaras tienden a ser constantes en su raíz y variar al final con las siguientes formas (entre otras):

- ion
- ions
- es
- aba

La parte variable es el final o "sufijo". Remover estos sufijos tiene el nombre de *suffix stripping* (desmontaje del sufijo) o *stemming* y la parte residual corresponde al *stem* o tallo como fue anteriormente mencionado. A continuación se muestra un ejemplo del algoritmo de *Stemming* de Porter en inglés

Frase original

*Do you really think it is weakness that yields to temptation? I tell you that there are terrible temptations which it requires strength, strength and courage to yield to Õscar Wilde*

Frase Stemmed

*Exampl Do you realli think it is weak that yield to temptat I tell you that there ar terribl temptat which it requir strength strength and courag to yield to Oscar Wild*

***POS Tagging (Parts of Speech)*** En la gramática tradicional en español las palabras son clasificadas en 9 partes del discurso (*Parts of speech* en inglés): determinante, sustantivo, pronombre, verbo, adjetivo, preposición, adverbio, conjunción o interjección (a diferencia del Inglés que no tiene el determinante y por lo tanto tiene solo 8 [26]). Cada parte del discurso explica no lo que la palabra es, sino como la palabra es utilizada en la oración. De hecho la misma palabra puede ser usada como un sustantivo en una oración y como verbo o adjetivo en otra. Los próximos ejemplos muestran un ejemplo de *POS tagging* en donde se puede observar el output de un software y como clasifica la palabra según su rol gramatical que desempeña en la oración, pudiendo este variar dependiendo del contexto.

Por ejemplo en la oración el gato come pescado y bebe agua. En esta oración "gato", "pescado" y "agua" son sustantivos (N), por otro lado "come" y "bebe" son verbos (V).

| El | gato | come | pescado | y | bebe | agua | . |
|---|---|---|---|---|---|---|---|
| *el* | *gato* | *comer* | *pescado* | *y* | *beber* | *agua* | *.* |
| DA0MS0 | NCMS000 | VMIP3S0 | NCMS000 | CC | VMIP3S0 | NCCS000 | Fp |

Figura 2.3: Ejemplo de *postags* de una oración
Fuente: Confección propia

En la siguiente oración "Yo bajo con el hombre bajo a tocar el bajo" la palabra bajo está contenida 3 veces, pudiendo ser agrupados como una sola *token*, contribuyendo de la misma forma al entrenamiento del algoritmo. Pero en realidad en esta oración la palabra bajo se usa con tres significados y funciones gramaticales distintas. Esta es la principal funcionalidad que tiene un software de *POS-tagging* en donde se es capaz de discriminar cual es la función gramatical. En este caso se tiene bajo como verbo (V) como adjetivo (A) y como sustantivo (N, *noun*).

| Yo | bajo | con | el | hombre | bajo | a | tocar | el | bajo | . |
|---|---|---|---|---|---|---|---|---|---|---|
| *yo* | *bajar* | *con* | *el* | *hombre* | *bajo* | *a* | *tocar* | *el* | *bajo* | *.* |
| PP1CSN00 | VMIP1S0 | SPS00 | DA0MS0 | NCMS000 | AQ0MS0 | SPS00 | VMN0000 | DA0MS0 | NCMS000 | Fp |

Figura 2.4: Ejemplo de postags de una oración
Fuente: Confección propia

Entonces *Part-Of-Speech tagging*, o *POS tagging* es el proceso que busca colocar a cada palabra en un documento un tag particular al rol gramatical que cumple, basado en la definición de la palabra y la relación que tiene con sus las palabras relacionadas o adyacentes en una frase o párrafo.

Existen muchos *softwares* de *POS-Tagging* para diferentes lenguajes. En este trabajo el utilizado para realizar esta tarea es el llamado *TreeTagger*, el cual es una herramienta para el marcado de texto utilizando la información del contexto y la palabra, y además funciona como *tokenizador* y *lematizador*, funciones que serán utilizadas en algunos de los experimentos a desarrollar y que seran explicadas más adelante. Este *software* fue desarrollado por Helmut Schmid en el TC Project en el Instituto de Lingüísticas Computacionales de la Universidad de Stuttgart. *TreeTagger* ha sido utilizado exitosamente para etiquetar varios lenguajes

incluyendo Alemán, Inglés, Francés, Italiano, Español, Búlgaro, Ruso, Griego, Portugués, Swahili, Latín, Estoniano, entre otros y es adaptable a otros lenguajes si un lexicón y un corpus manualmente etiquetados están disponibles para el entrenamiento [27]. En el anexo se puede encontrar todos los tags utilizados por TreeTagger como resultado del proceso de *POS-Tagging*

## 2.3 *Web Opinion Mining*

Los estudios y softwares de detección de sentimientos y opiniones han sido testigos de un creciente interés en los últimos años debido a la creciente cantidad de reseñas y opiniones online y la necesidad de organizarlos. En relación a este tema existen 4 problemas predominantes para los investigadores del área: clasificación de la subjetividad de documentos, clasificación de sentimientos de una palabra, clasificación de sentimientos de un documento, y la extracción de opiniones. Es un hecho que existen relaciones entre las tareas anteriormente mencionadas como por ejemplo que un clasificador de subjetividad puede prevenir al clasificador de sentimientos considerar textos irrelevantes para su clasificación del punto de vista objetivo [28]. Este hecho será ocupado en los próximos capítulos y experimentos con este fin.

### 2.3.1 Detección de sentimientos

**Clasificación de subjetividad** La subjetividad se refiere a aspectos del lenguaje usados para expresar opiniones, evaluaciones y especulaciones [29]. El problema de clasificación de subjetividad tiene por objetivo reconocer las oraciones utilizadas para expresar opiniones y otras formas de subjetividad, de oraciones que son utilizadas objetivamente y que presentan información factual [28].

**Clasificación de sentimientos** La clasificación de sentimientos según su polaridad incluye dos tipos de formas: estas son clasificación binaria de sentimientos y clasificación multiclase, con más de dos categorías posibles. Dado un conjunto de documentos $D = \{d_1, \ ... \ , d_n \}$ y un conjunto de categorías predefinidas $C = \{positivo, negativo\}$, la clasificación de sentimientos binaria responde a clasificar cada $d_i \in D$ con una etiqueta perteneciente a $C$. Cuando $C$ contiene más de 2 elementos como por ejemplo $C^* = \{muy\ positivo, positivo, neutral, negativo, muy\ n$ el problema cambia a una clasificación multiclase de sentimientos.

El problema de explotar Corpus etiquetados para entrenar clasificadores para el análisis de sentimientos ha sido un gran tema de estudio en los últimos años [30–32]. Una característica común de todos estos trabajos ha sido la tendencia de definir un problema de clasificación de sentimientos binario, i.e positivo vs negativo. Actualmente en todos los problemas de polaridad, incluyendo el análisis de sentimientos existen al menos tres categorías que deben ser distinguidas: positiva, negativa y neutra. No toda opinión de un producto, experiencia o comentario es puramente del tipo positivo o negativo. En la mayoría de los sets de comentarios, algunos de éstos tienen características objetivas, sin ningún sentimiento asociado, mientras que otros presentan una mezcla o conflicto de sentimientos. Con la excepción del trabajo de Pang et. al. [33], la investigación en la categorización automática de sentimientos ha ignorado este tipo de documentos neutrales. En [34] Koppel et.al. muestra por qué es

importante considerar la clase neutral y entre sus conclusiones se encuentran que el entrenamiento utilizando ejemplos positivo y negativos no permite una clasificación precisa de los ejemplos neutrales, y también que el uso de datos neutros contribuye a una mejor distinción entre ejemplos positivos y negativos. Esto conlleva a que el uso de la clase neutral en el entrenamiento aumente significativamente la precisión en la clasificación.

### 2.3.2 Aplicaciones de la detección de sentimientos

**Comparaciones de productos** Es una práctica común entre los sitios de *E-commerce* que habiliten espacios especialmente dedicados para que los compradores de productos realicen *reviews* o reseñas de los productos que han adquirido. Con más y más gente usando la Web para expresar opiniones, el número de reseñas que un producto recibe crece rápidamente. Muchos de los estudios que se enfocan en este tipo de sitios, clasifican los productos entre recomendados y no recomendados [30, 33, 35], pero todos los productos tienen muchas características, pero los clientes pueden estar interesados solo en algunas. Más aún, cuando un producto posee una carencia en un aspecto, lo más probable es que tenga virtudes en otra [36,37]. El poder comparar las características de un producto en donde el usuario pueda claramente ver las ventajas y desventajas de éste, pueden ayudarle a decidir si comprar o no este producto.

**Retroalimentación a las empresas** Si bien los estudios de mercado existen desde tiempos inmemoriales, mientras más información se contenga para la toma de decisiones, se reduce la incertidumbre y con esto el riesgo asociado a esta decisión. Es por esto que la información contenida en la Web, ya sea desde blogs especializados, hasta en las redes sociales toma especial importancia a la hora de evaluar productos o realizar investigaciones para lanzar o posicionar un nuevo producto en el mercado.

**Detección de eventualidades** Un aumento en el número de menciones de un producto, marca o empresa en la Web, puede representar una eventualidad que podría ser manejada si se detecta y atiende a tiempo. En esta tarea la detección de sentimientos juega un rol fundamental a la hora de determinar bajo que polaridad o sentimiento los comentarios están siendo emitidos.

**Resumen de opiniones** Mientras más crece el número de comentarios y opiniones sobre productos en la Web, más complejo se hace para las empresas organizar y analizar todas estas reseñas. Más aún cuando las reseñas son largas y solo algunas partes corresponden a opiniones. Además se torna difícil para el cliente tomar una decisión informada a la hora de comprar un producto. El resumen de opiniones tiene por objetivo realizar un compendio de opiniones, entregando la polaridad de sentimientos y detectando aquellas frases que contienen opiniones de las que no. Según Hu en [38] si una frase contiene una o más características de un producto, y una o más palabras que expresan opiniones, entonces esta frase es una frase de opinión.

## 2.4 *Microblogging*: Twitter y sus características

Antes de hablar de cómo se puede extraer la polaridad de sentimientos de comentarios emitidos en Twitter, resulta relevante comprender su estructura y características principales.

Después de la aparición de los blogs y respondiendo a la necesidad que tenían los usuarios de poder enviar mensajes de forma rápida y fácil, nace el *Microblogging* con su principal actor Twitter. En esta nueva forma de comunicación, los usuarios pueden describir un estado o tweet en un mensaje corto vía Internet, celulares o emails. Esta red social fue lanzada en Octubre del año 2006 y para Abril del 2007 siguiente ya contaba con 94.000 usuarios [39]. Teniendo un crecimiento mantenido para el Diciembre del año 2013 Twitter ya contaba con 200 millones de usuarios activos a nivel mundial y 400 millones de comentarios al día [40]. Como fue mencionado anteriormente, las razones del por qué Twitter es un caso relevante de estudio y que justifican este trabajo se deben a la gran penetración que tiene la red social en el mercado chileno, dónde el número de usuarios es de 4,8 millones para principios del 2012 y 24% de ellos son usuarios activos (definidos según el estudio como los usuarios que han escrito en la red social en los últimos 3 meses antes del estudio) [2], y esta cifra no considera la gran adopción que ha tenido en el último tiempo Twitter en Chile en el contexto de las telecomunicaciones donde es considerado un medio importante para estrategias de marketing, un complemento a la televisión, un canal para montar servicio de atención al cliente, e incluso una forma de comunicación en emergencias y desastres.

En Twitter los usuarios interactúan y se comunican entre ellos mediante tweets. Éstos corresponden a mensajes cortos con un máximo de estrictamente 140 caracteres de largo. Muchos de ellos referencian a otros usuarios de la red social, otro tweet o a una página Web siguiendo las siguientes convenciones:

### 2.4.1 Estructura de Twitter

**Perfil de usuario** Un usuario de Twitter posee una página personal denominada "cuenta". Este perfil público contiene el nombre completo, ubicación, una pequeña biografía de un par de líneas y el número de tweets, seguidores y amigos que posee el usuario. Estas personas que siguen al usuario y aquellos que son seguidos por él también son listados.

***Trending Topics*** Las palabras, frases y hashtags que son mencionados más frecuentemente son llamados *Trending Topics* (temas con mayor tendencia). Un *hashtag* es una convención que utilizan los usuarios de Twitter para crear y seguir un hilo de discusión anteponiendo a una palabra el carácter '#'. Esta convención permite a los usuarios conectarse entre ellos cuando hablan de un tema en común. Por ejemplo cuando un equipo de fútbol está jugando generalmente los usuarios realizan comentarios con un *hashtag* común para que su tweet fácilmente pueda ser visto por otros usuarios conversando del mismo tema. Twitter muestra una lista de los top diez *trending topics* del momento según la ubicación en una barra a la izquierda de la página inicial de todos los usuarios por defecto. El mecanismo que utiliza twitter para obtener estos datos no es público ni conocido.

### 2.4.2 Tipos de tweets

Los mensajes de Twitter pueden ser categorizados por su objetivo y naturaleza según las convenciones de la red social como:

- Self: El tweet por defecto. Es utilizado para comunicar algún hecho propio (opinión, sentimiento, estado, etc) en busca de una audiencia. Compuesto por texto y/o links sin hacer referencia a otros usuarios o mensajes.
- Reply: Respuesta a un tweet de otro usuario de acuerdo al protocolo de respuesta del API de Twitter. Una alta frecuencia de estos tweets implica una conversación activa.
- RT: Como una abreviación de *retweet*, este tipo de mensaje corresponde a la republicación de un tweet de otro usuario que se considera interesante de acuerdo a los protocolos de Twitter.
- Vía: Cita de un tweet de otro usuario. De alguna forma es equivalente al RT, pero fuera del protocolo estándar. Usualmente se incorporan palabras como vía, *by*, por,etc. para indicar la autoría del tweet. Por lo general al final contiene un comentario o un link de referencia.
- Mención: Es un tweet donde se menciona a otro u otros usuarios. Estos tweets no corresponden a respuestas, RT o vía y por lo general se utilizan para comenzar una conversación con otros usuarios.

### 2.4.3 Extracción de tweets mediante la *API*

Para extraer tweets en tiempo real, éstos pueden ser descargador mediante el API que Twitter pone a disposición de los desarrolladores. La conexión con el API de Twitter es solo posible creando una aplicación en el sitio de desarrolladores de Twitter https://dev.twitter.com/apps y usando *OAuth*. *OAuth* es un protocolo de identificación que permite a los usuarios aprobar aplicaciones que usen sus credenciales de una cuenta sin la necesidad de compartir su contraseñas. Como es definido en OAuth.net, el framework OAuth 2.0 permite a una aplicación de un tercero obtener acceso limitado a un servicio HTTP.

La recuperación de tweets en tiempo real se realiza a través de la *Streaming API* utilizando el método de filtro *filter track consumer* y realizando la petición *HTTP* mediante el método *POST*. La respuesta retornará todos los estados públicos que contienen uno o más de los parámetros de filtraje. Pueden ser especificados múltiples parámetros, lo que permite a la mayoría de los clientes utilizar una sola conexión al *Streaming API*. A pesar que ambos tipos de peticiones *GET* y *POST* son soportados, en este proyecto se utilizó *POST* ya que las peticiones *GET* con muchos parámetros pueden ser rechazadas debido a una excesiva longitud de la *URL*, como es explicado en los documentos de desarrolladores de Twitter.

Los parámetros recibidos por el servicio *filter track* son:

- *Follow*: Corresponden a una lista separada por comas, con los identificadores de usuario, indicando de cuáles usuarios se quieren obtener los tweets que generen.
- *Track*: Palabras claves a seguir. Una listado de frases de palabras que son ingresadas separadas por coma. Todas los *tweets* que contengan al menos una de las frases serán retornados mediante este método.
- *Locations*: Especifica un conjunto de polígonos a seguir. Todos los comentarios que sean georreferenciados y su par latitud, longitud esté contenido en alguno de estos polígonos definidos por 4 puntos con sus latitudes y longitudes respectivas, será retornado. Existen restricciones de superficie máxima que pueden ser consideradas al utilizar esta opción.
- *Delimited*: Especifica si los mensajes deben ser delimitados por su largo de caracteres y cuánto.
- *Stall Warning*: Si este parámetro es configurado como verdadero, generará mensajes periódicos al cliente si está en peligro de ser desconectado.

Todas las respuestas de este servicio son en formato JSON. Para comprender la estructura y contenido de un tweet retornado por un filtro se explican sus componentes más importantes a continuación:

- *Tweet text*: El texto completo del *tweet*
- *In reply to user id*: ID del usuario de twitter si el *tweet* le responde a otro usuario. Nulo si no.
- *In reply to user status id*: Id del *tweet* si éste está respondiendo a otro *tweet*. Nulo si no.
- *Source*: Dispositivo o plataforma desde la cual el *tweet* fue generado.
- *Tweet ID*: ID único del *tweet*.
- *Entities*: Un arreglo de posibles menciones, *hashtags* y *URLS* que pueda contener un *tweet*. Además se especifican el nombre de las menciones y las posiciones de las entidades en la cadena de texto del *tweet*.
- *Retweeted*: *Boolean* que especifica si el tweet fue *retwiteado* o no.
- *Cordinates*: Latitud y longitud del *tweet* cuando éste es georreferenciado.
- Created at: Fecha en formato TIMESTAMP del tweet que contiene la fecha de su creación (UTC)
- *User*: Arreglo que contiene múltiples atributos del usuario como:
  - *Statuses count*: Número de tweets que ha generado el usuario
  - *Followers count*: Número de seguidores que posee el usuario.

  - *Profile image*: URL de la imagen que posee el usuario en Twitter.
  - *User screen name*: Nombre del usuario.
  - *Friends count*: Número de personas que sigue el usuario.

- *Retweeted status*: Si el *tweet* fue un *retweet* de un *tweet* de otro usuario, esta propiedad incluye toda la información anterior para el tweet original.

**Problemas de Twitter para realizar *opinion mining*** Los problemas clásicos del *Web Mining* que aplican al *Web Opinion Mining* en twitter son entre otros:

- Los volúmenes de datos son muy grandes.
- Existe mucho ruido y mucha información irrelevante.
- El tiempo de aprendizaje para algoritmos supervisados suele ser grande (horas o días)
- El tiempo de respuesta debe ser muy rápido para atender al gran flujo de peticiones (en línea)

El más importante de todos estos problemas es que al ser estos textos escritos por cualquier persona y al ser de carácter informal, no necesariamente están escritos correctamente. Más aún cuando existe una limitación de caracteres como en el caso de Twitter, se agudiza el uso de abreviaciones. Este hecho implica nuevos desafíos que necesitan ser considerados al hacer minería de opiniones en twitter. Algunos de estos problemas están descritos en mayor detalle en los siguientes capítulos.

## 2.5 Enfoques más conocidos de *Sentiment Analysis* en Twitter

Por las características que presenta el problema de *Sentiment Analysis* es que se utilizan técnicas de *Data Mining* de aprendizaje supervisado. Este tipo de algoritmos consta de cuatro fases principales, la primera de recolección y construcción del Corpus, la segunda la extracción y selección de *features*, la tercera del desarrollo del algoritmo clasificador y por último la evaluación de la aplicación. En esta sección se expondrán las principales técnicas utilizadas en la literatura existente para resolver estas dos problemáticas.

### 2.5.1 Generación del Corpus

En *Natural Language Processing* un *Corpus* es un conjunto de textos que son usualmente guardados y procesados por computadores y usados con fines estadísticos para contrastaciones de hipótesis y así validar reglas lingüísticas. En *Sentiment Analysis* forma parte esencial de las aplicaciones y su desempeño, debido a que los textos contenidos en el Corpus conforman los datos de entrenamiento que se utilizarán para el aprendizaje de los algoritmos de clasificación.

Para los algoritmos de aprendizaje supervisado y categorización de clases, se deben construir Corpus que representen fehacientemente cada una de las clases para que luego de extraídas las propiedades de los textos contenidos en cada una de ellas, el algoritmo sea capaz de clasificar correctamente nuevos textos no procesados anteriormente. Como la tarea de *Sentiment Analysis* en Twitter contempla la categorización de comentarios entre positivos, negativos y neutros, objetivos y subjetivos entre otros, es necesario la creación de un corpus por cada una de estas clases que en su conjunto conforman la *Corpora*.

Cuando una *Corpora* es procesada con un algoritmo de *POS-Tagging* u otro *NLP* a este resultado se le denomina *annotated-corpora* en donde cada rol que juegan las palabras en los textos son agregados al corpus. Otro ejemplo es el de incluir la raíz de las palabras, etc.

Por último el *Corpus* y los textos que lo conforman son fundamentales para uno de los objetivos de este trabajo: considerar las características propias de la red social e idioma español. Es por esto que no se considerarán ningún tipo de Corpus o repositorios que no tengan como base comentarios de Twitter. Además para incorporar la forma en que los chilenos interactúan con la plataforma se utilizarán tweets únicamente chilenos (se asume que con esto se agregarán los modismos propios del lenguaje local).

Entre los principales métodos de conformación de Corpus se encuentran:

**Lingüistas** Consiste en la elaboración de un diccionario de palabras y frases por medio de lingüistas. Si bien esta es una alternativa muy completa, resulta muy demandante en tiempo y dinero. Además como el lenguaje es dinámico y va evolucionando en las comunidades con la aparición de nuevas expresiones, es necesario que sea constantemente actualizado para que el trabajo mantenga su calidad.

**Repositorios online** En Interne existen repositorios y Corpus ya generados y anotados para su uso en problemas de *Text Mining*. Si bien esta es una alternativa viable mediante repositorios como por ejemplo *WordNet* en inglés, utilizando esta opción no es posible capturar las características propias de Twitter y menos aún los modismos chilenos.

**Etiquetado Manual (*Manual Labelling*)** Esta técnica ha sido ampliamente utilizada en muchos trabajos, y consiste en una preclasificación manual de los datos de entrenamiento por individuos. El problema de esta técnica es que el número de datos de entrenamiento que contenga el Corpus será limitado por el número de horas hombres que se tengan disponibles para el proceso de etiquetado, y un Corpus pequeño y poco robusto puede tener como consecuencia una baja precisión para clasificar nuevos datos. Además al ser el etiquetado un proceso manual y posiblemente realizado por varias personas, éste se torna subjetivo, pudiendo cambiar el criterio de clasificación sucesivas veces durante su ejecución.

**Enfoque de *Emoticons*** Esta técnica presentada en varios trabajos, es introducida por Read en [41]. En este trabajo Read concluye que los *emoticons* pueden determinar la orientación o polaridad de las frases y textos que los contienen, pero uno de los principales problemas que presenta la técnica, al igual que otras de *Sentiment Analysis* es que muchas veces en *reviews* o foros existen cambios en la polaridad del sentimiento dentro del documento y por eso es difícil asignar una polaridad general a éstos. Gracias a la estructura de Twitter,

no se tienen este tipo de problemas ya que un comentario está formado por 140 caracteres, los que por lo general corresponden a una sola frase o idea. Este enfoque de usar *emoticons* en la recolección del corpus para realizar *Sentiment Analysis* en Twitter fue introducida por Go et. al 2009 en [42], considerando varios aspectos propios de la plataforma que lo hacían distinto a los trabajos de *Sentiment Analysis* realizados hasta ese momento:

- El largo, como fue mencionado anteriormente los tweets poseen un máximo de 140 caracteres.
- La data disponible. Gracias al API de twitter es relativamente sencillo obtener millones de datos para que conformen el Corpus. A diferencia del trabajo presentado por Pang & Lee en [33], donde el tamaño del corpus era e 2053 datos.
- El lenguaje utilizado. Los comentarios de twitter se realizan desde distintos tipos de plataformas y dispositivos, como el celular, lo que en conjunto con la restricción de caracteres fomenta el uso de abreviaciones y aumentan las faltas de ortografías y modismos.

Si bien el tercer punto no favorece el uso de técnicas de *Sentiment Analysis*, los primeros dos, en conjunto con lo propuesto por Read, sí. La cadena de supuestos que un *emoticon* presenta la polaridad del sentimiento en una frase y que un tweet está formado usualmente por una sola frase, en conjunto implica que un tweet con un *emoticon* polar, posee la polaridad de ese *emoticon*. Gracias a esto se puede realizar un preetiquetado automático de tweets para generar fácilmente un Corpus de miles de datos de entrenamiento.

Sin embargo el trabajo de Go et.al 2009 tenía foco en el análisis de polaridad de sentimientos en dos clases (positivo y negativo), y solo realiza un análisis menor considerando 3 clases al incluir como ejemplos neutrales aquellos tweets que no tuviesen un *emoticon*, lo que es un supuesto muy débil. Como fue anteriormente mencionado, en [34] Koppel et.al. Se verifica por qué es importante considerar la clase neutral, ya que la utilización de ejemplos positivos y negativos no permite una clasificación precisa de los ejemplos neutrales, los cuales si pueden estar presentes en comentarios de personas y hechos noticiosos.

Para resolver este problema Pak & Paroubek en [4] utilizaron una aproximación que trajo buenos resultados para confeccionar un clasificador de 3 clases, la cual consistía en tomar como textos objetivos o neutrales todos aquellos tweets generados por diarios y revistas informativas. Aunque esta metodología tiene ciertos problemas que se detallará en el Capítulo 3 que trata de la confección del Corpus

Por las bondades ya presentadas y los resultados que han tenido los trabajos que utilizan esta técnica es que aquí se adoptará esta metodología de pre-etiquetado en base a emoticons y tweets de cuentas de diarios para la confección del Corpus.

**Recolección de tweets en base a *Hashtags*** Este enfoque tiene una metodología parecida al caso anterior en donde se conforma cada corpus de forma semiautomática utilizando *hashtags* polares como términos de búsqueda. En [43] Davidov et.al. Utiliza esta metodología

para elaborar un clasificador multiclase de 5 sentimientos: Altamente subjetivo, subjetivo, focalizado, sin sentimiento, y dependiente del contexto. Y luego realiza una comparación con el enfoque de *emoticons* obteniendo correlaciones interesantes acerca de *hashtags* como #happy con *emoticons* polares.

### 2.5.2 Extracción de *Features*

Una vez que los documentos han sido etiquetados, se procede a extraer aquellos aspectos contenidos en los textos que sean indicadores de orientación semántica, los cuales se utilizarán para entrenar el algoritmo de clasificación. Entre las principales *features* consideradas para realizar *Sentiment Analysis* se encuentran los *N-grams* de palabras y el uso de herramientas de etiquetado morfosintáctico o *POS-taggers*. Estos dos tipos de técnicas serán brevemente explicadas a continuación.

***N-grams*** Generalmente en *Text Mining* se considera la unidad básica de los documentos a una palabra y se le denomina *token*. Cuando las *features* están formadas por una palabra se les conoce con el nombre de *unigram*, de dos como *bigram* y así sucesivamente. El hecho de seleccionar más de una palabra para la extracción de *features* tiene como motivación la conservación del contexto para así conservar la relación que existen entre las palabras de una oración (*tweet*). Por ejemplo la frase "estoy feliz" tiene una polaridad completamente opuesta a "no estoy feliz". Es este tipo de efectos los que se pretenden modelar aumentando el número de *tokens* en las *features*. En [42] se experimenta con *unigrams* y *bigrams* para la conformación de las *features* y en [4] con *unigrams, bigrams* y *trigrams*, teniendo mejores resultados con *bigrams*.

**POS-tags** Como fue explicado anteriormente, el *POS-Tagging* tiene por objetivo etiquetar cada una de las palabras y caracteres dentro de una oración, con la función gramatical que cumple dentro del *tweet*. Las utilidades son principalmente agregar una parte del contexto a cada palabra considerando su *POS-Tag* y además poder discriminar entre palabras que se escriben igual y representan cosas distintas. Por ejemplo en la palabra río, usada como sustantivo puede tener una connotación neutral mientras que río como verbo podría tener una polaridad positiva o negativa, por lo que resulta importante poder separar ambas palabras en distintas *features*.

**Enfoques alternativos** Existen muchas otras formas para la extracción de *features* como las realizadas en los trabajos [43, 44], donde por ejemplo se consideran como *features* la presencia de negaciones largo de los textos, uso de letras mayúsculas, símbolos de puntuación, etc.

### 2.5.3 Selección de *features*

Seleccionar las *features* relevantes y decidir cómo codificarlas para la fase de entrenamiento, puede tener un enorme impacto en la habilidad que tiene el algoritmo de aprendizaje para generar un buen modelo. Gran parte del trabajo de generar un clasificador, es decidir cuáles *features* pueden ser relevantes, y cómo se pueden representar. Aunque a menudo es posible obtener un desempeño decente de los algoritmos mediante el uso de un conjunto bastante

simple y obvio de *features*, por lo general hay un aumento significativo en la performance con el uso de características cuidadosamente construidas.

**Selección de *features* basados en la presencia** Es la forma más simple de selección de *features*. Consta de seleccionar todas las *features* que existen en el Corpus de datos de cada clase para la construcción de los vectores, asumiendo que todas aportan valor y representan esa clase.

**Selección de *features* basados en la frecuencia** En este caso son consideradas únicamente las *features* que presentan mayor frecuencia en cada clase, o que al menos aparezcan más veces que un umbral dado. Por ejemplo si el número de veces que aparece una *feature* en una clase que 40, podría ser posible decir que esta *feature* contribuye a la representación de la clase. Para determinar este umbral o el número máximo de *features* a considerar en cada clase se pueden realizar pruebas variando estos valores.

**Selección de *features* basados en algoritmo de *Mutual Information*** El concepto detrás de *Mutual Information* es determinar para cada clase C y *feature* F, un valor que mida cuánto F puede contribuir a realizar una decisión de clasificación correcta en la clase C [45]. La fórmula del puntaje *MI* es:

$$MI(C,F) = \sum_{ef} \sum_{ec} P(C = ec, F = ef) \log \frac{P(C = ec, F = ef)}{P(C = ec)P(F = ef)} \tag{2.1}$$

Luego de calcular el puntaje *MI*, el criterio de selección de *features* consiste en ordenar las *features* según sus puntajes por clase y considerar solo las k *features* con mayores puntajes para cada clase.

**Selección de *features* basados en criterio de Chi-Cuadrado** Este criterio utilizado es similar a la de *Mutual Information*. Para cada *feature* en una clase, también se construye un puntaje, pero lo que se intenta medir en este caso es si la *feature* y la clase son independientes una de otra. Se usa el *test* de *chi-cuadrado*, ya que en estadística éste se utiliza para determinar si dos eventos son independientes entre sí. En este test, se asume como hipótesis que la *feature* y la clase son independientes y se calcula el valor de $X^2$. Un valor alto, implica que no son independientes. Por ejemplo el valor crítico de 0.001 es 10.83. Esto implica que si ambos eventos son independientes, entonces la probabilidad que el puntaje sea mayor a 10.83 es de 0.001. Por lo tanto mientras más grande es el puntaje, existe mayor probabilidad que ambas, la *feature* y la clase, sean dependientes. Entonces lo que se desea es conservar las *features* en cada clase que posean los mayores puntajes de $X^2$. Su fórmula del puntaje es:

$$X^2(F,C) = \frac{N(N_{11}N_{00} - N_{10}N_{01})^2}{(N_{11} + N_{01})(N_{11} + N_{10})(N_{10} + N_{00})(N_{01} + N_{00})} \tag{2.2}$$

Donde N es el número total de comentarios o tweets de entrenamiento. $N_{11}$ es el número de co-ocurrencias de la *feature* F y la clase C. $N_{10}$ es el número de *tweets* que contienen la

*feature* F que no están en la clase C. $N_{01}$ es el número de *tweets* en la clase C que no contienen la *feature* F. $N_{00}$ es el número de tweets que no están en C y que no contienen la *feature* F. Entonces el criterio de selección consiste en seleccionar para cada clase las K *features* con mayores puntajes.

### 2.5.4 Algoritmos de entrenamiento

Por mucho tiempo los científicos estuvieron interesados en comprender de forma rápida cuáles eran los principales tópicos o temas que estaban contenidos en un documento, o en un gran número de ellos, más que entender cuáles eran los sentimientos o polaridades que se encontraban más presentes. Con el fin de utilizar algunas de las técnicas desarrolladas para estos fines, es que se considera el problema de clasificación de polaridad de sentimientos como un caso especial de la categorización de temas.

Algunos de los experimentos que se llevaran será la elaboración de algoritmos con distintos número de clases a considerar como son: Positivo vs Negativo, Positivo vs Negativo vs Neutro, Subjetivo vs, Obejtivo donde los textos (o tweets) subjetivos son ambos positivos y negativos, además de combinaciones entre éstos.

Como en [33], se experimentará con uno de los principales algoritmos de clasificación: Clasificador Bayesiano (*Naïve Bayes Classifier*). Los fundamentos detrás de cada uno de estos algoritmos son muy diferentes pero como ha quedado demostrado en [4, 33, 42] éstos se desempeñan de buena manera en problemas de clasificación de documentos de texto.

Para explicar algunas de las técnicas más comunes de *Machine Learning* se utilizará la misma nomenclatura que en [33]. Sea f_1,. . . ,f_m el conjunto de m características (*features*) que pueden aparecer en un documento. Estas características pueden ser *unigrams* o *bigrams* conformados por una o dos *tokens* respectivamente. Sea n_i (d) el número de veces que f_i ocurre en el documento d. Entonces cada documento es representado por el siguiente vector:

$$\vec{d} := (n_{1(d)}, n_{2(d)}, ..., n_{m(d)}) \tag{2.3}$$

Explicaciones más acabadas de estos algoritmos pueden ser encontradas en el Capítulo 5, pero también serán brevemente explicados en el marco conceptual para facilitar la lectura de los próximos capítulos.

**Clasificador Bayesiano Ingenuo (*Naive Bayes Classifier*)** Este enfoque de clasificación tiene relación con asignar a un documento dado d, la clase con la mayor probabilidad que el documento d corresponda a ella. La expresión que determina esta relación es la siguiente: $arg\max_c P(c|d)$

$$P(c|d) = \frac{P(c)P(d|c)}{P(d)} \tag{2.4}$$

Donde P(d) no tiene influencia al escoger el argumento máximo ya que todas las clases tienen el mismo valor P(d) en el denominador. Para estimar el valor del término P(d—c), el Clasificador Bayesiano asume que las características f_i son condicionalmente independientes dadas la clase, resultado:

$$P_{NB}(c|d) := \frac{P(c)(\prod_{i=1}^{m} P(f_i|c)^{n_i(d)})}{P(d)} \tag{2.5}$$

Independiente que el supuesto de independencia condicional claramente no se mantiene en situaciones del mundo real ya que todas las frases tienen un contexto y una estructura semántica que hace que las palabras sean dependientes unas de otras, la clasificación basada en el algoritmo de *Naïve Bayes* se desempeña de forma sorprendentemente bien en problemas de recuperación de información (IR) y clasificación [46, 47].

**Máxima Entropía** El clasificador de máxima entropía es un clasificador probabilístico que pertenece a la clase de modelos exponenciales. Este clasificador no asume supuestos de independencia condicional de las variables como lo hace el clasificador de *Naïve Bayes*, y está basado en el Principio de Máxima Entropía, trabajo elaborado por Maxwell, Boltzmann y Gibbs [48]. Como este clasificador no asume independencia, es muy utilizado en problemas de clasificación de textos, en donde las características y vectores están conformados por palabras las cuales obviamente son dependientes unas de otras.

Como se hizo anteriormente, el objetivo es usar las características de los documentos de texto como información contextual para categorizarlos en un conjunto de clases o etiquetas objetivo (positivo vs negativo vs neutro, subjetivo vs objetivo, etc.) Usando la misma nomenclatura que antes, sea f_1,. . . ,f_m las m características que aparecen en un indican si una característica en particular f_i existe o no en el documento dado. Este enfoque fue propuesto por Bo Pang y Lillian Lee en [33].

Como es descrito en [49] por Adam Berger, el objetivo detrás de este algoritmo es construir un modelo estocástico que represente precisamente el comportamiento de un proceso aleatorio: toma como input un conjunto de características (*features*) y produce un output y con las etiquetas o clases asignadas.

***Support Vector Machines*** Estos algoritmos de entrenamiento supervisado se enfocan en el reconocimiento de patrones y son utilizados para análisis y tareas de clasificación y regresión. La idea detrás de estos algoritmos es encontrar un hiperplano que pueda separar distintos vectores de una clase, de los de la otra, maximizando la separación o margen entre ellos. Si los datos no son linealmente separables en un espacio dado, entonces el hiperplano óptimo es aquel que maximiza el margen y minimiza el número de clasificaciones incorrectas. De manera intuitiva, el modelo de este clasificador representa al conjunto de entrenamiento como puntos en un espacio n-dimensional, siendo n la cantidad de *features* a considerar, y busca encontrar un hiperplano que maximice la distancia entre los puntos de cada categoría.

El cálculo del hiperplano óptimo corresponde a un problema de optimización que puede ser resuelto con técnicas adhoc como Multiplicadores de Lagrange. Por ejemplo para el caso de clasificación de textos, siendo c_j la clase asignada al documento d_j, y w el hiperplano óptimo, su solución puede ser escrita como

$$\vec{w} := \sum_j \alpha_j c_j \vec{d_j}, \alpha_{j>0} \tag{2.6}$$

Donde los valores alpha_j son obtenidos resolviendo el dual del problema de optimización. Los *Support Vectors* son aquellos (d_j ) donde alpha_j es mayor a cero, y son los únicos vectores del documento que contribuyen a w. Finalmente la regla de clasificación corresponde a qué lado del hiperplano cae el texto a clasificar.

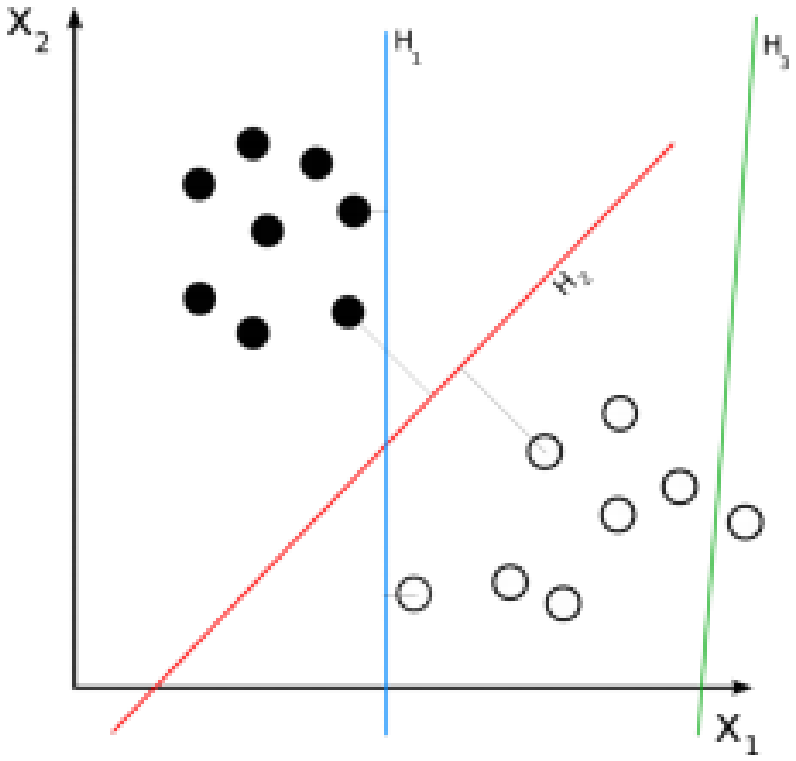



Figura 2.5: Representación de un plano y sus *support vectors*
Fuente: Confección propia

Una diferencia importante entre regresiones lineales y el Clasificador Bayesiano con los *Support Vector Machines* es que en las primeras técnicas, todos los puntos determinan el el resultado óptimo del clasificador, mientras que en SVM solo los "puntos difíciles" cerca de los bordes (*support vectors*) de la regla de clasificación (el hiperplano), son los que definen la condición de optimalidad.

### 2.5.5 Ratios de evaluación: *Accuracy*, *Recall* y *Precision*

En el análisis de documentos, una tarea común es probar la utilidad de un componente mediante una evaluación experimental. Aplicando algoritmos respectivos a un subconjunto de prueba, algunas mediciones importantes como *recall, accuracy* y *precision* pueden ser obtenidas. Los objetivos de estas evaluaciones son: Por un lado probar la efectividad del

algoritmo para ser usado en la práctica, y por otro lado la evaluación puede probar que un algoritmo es mejor o peor que algún otro [50].

El *recall* en un sistema de *Information Retrieval* puede ser definido como el cociente del número de documentos relevantes retornados, sobre el total de documentos relevantes utilizados en el subconjunto de validación. En simples palabras *recall* es la fracción de documentos relevantes que son satisfactoriamente recuperados. En este caso de comentarios de Twitter el cociente de *recall* para un algoritmo de clasificación de dos clases sería el número de etiquetas correctamente asignadas (tweets recuperados), dividido en el número del total de tweets que debieron ser recuperados (todos los tweets relevantes a la consulta).

$$Recall = \frac{VerdaderosPositivos}{VerdaderosPositivos + FalsosNegativos} \tag{2.7}$$

El cociente de *Precision* corresponde a la fracción de documentos recuperados que son relevantes. Esto es, de todos los documentos recuperados, cual es la razón entre los correctamente recuperados y todos los que fueron recuperados. En el mismo caso anterior de un algoritmo de clasificación de dos clases la precisión correspondería a cuántos de los tweets fueron correctamente recuperados dividido entre todos los tweets que fueron recuperados.

$$Precision = \frac{VerdaderosPositivos}{VerdaderosPositivos + FalsosPositivos} \tag{2.8}$$

El último de los 3 comúnmente típicos ratios de validación en *Machine Learning* es el *accuracy*. El *accuracy ratio* corresponde a la proporción de documentos correctamente clasificados para todas las clases dividido sobre el total de documentos clasificados. En el ejemplo de 2 clases, el *accuracy* corresponde al número de tweets correctamente clasificados, sobre todas las clasificaciones realizadas.

$$Accuracy = \frac{VerdaderosPositivos + FalsosNegativos}{VerdarosPositivos + FalsosPositivos + VerdaderosNegativos + FalsosNegativos} \tag{2.9}$$

La tabla 2.1 se denomina matriz de confusión y se resume en qué consiste cada uno de los valores para determinar los ratios anteriormente descritos.

| | Condición Positiva | Condición Negativa |
|---|---|---|
| Clasificación Positiva | Verdadero Positivo (VP) | Falso Positivo (FP) |
| Clasificación Negativa | Falso Negativo (FN) | Verdadero Negativo (VN) |

Tabla 2.1: Matriz de confusión
Fuente: Elaboración Propia

# Capítulo 3

# Caracterización del Corpus

En este capítulo se explicará la metodología seleccionada para la conformación de los corpus de tweets para cada clase. Además se validará mediante la técnica de Muestreo Aleatorio Simple, el grado de efectividad que posee la metodología aceptada, encontrándose diversos problemas en el etiquetado automático. Esto nos lleva a proponer distintas reglas para evitar aquellos problemas, las que se detallan al final de este capítulo. Por último se hace un nuevo experimento para determinar si las reglas presentadas mejoraron el desempeño del etiquetado automático y un análisis a la estructura final de la *corpora*.

## 3.1 El concepto de Corpus y su caracterización para el análisis de sentimientos en Twitter

En lingüística, un *Corpus* es un set estructurado de textos. Éstos son utilizados para validar hipótesis e identificar patrones intrínsecos en su estructura utilizando metodologías de *Data Mining*, *Machine Learninig* y *Natural Language Processing*. Como el desempeño de estos algoritmos mejora a medida que se tienen más datos de entrenamiento (evitando el *overfitting*), es que se desea obtener un corpus con un gran número de datos.

## 3.2 Creando el Corpus

La recolección del Corpus es una de las fases más importante en el proceso de *Sentiment Analysis* en conjunto con el mecanismo de extracción de *features* para la generación de los *feature vectors*. Considerando el hecho que la data de entrenamiento debe ser lo suficientemente robusta para que se pueda generar un modelo que posteriormente permita clasificar nuevos tweets, existe el interés por recolectar tantos tweets como sean posibles para el entrenamiento. Además necesitamos que esta data sea preclasificada o etiquetada de acuerdo a las distintas clases en los que serán clasificados los tweets una vez que el algoritmo clasificador esté construido. Debido a estas características es que el etiquetado manual de los *tweets* resulta una opción no viable por la gran cantidad de personas y horas hombres que serían necesarias para etiquetar todos los tweets usados para el entrenamiento.

Para la recolección de los tweets que conforman parte de la *corpora* se desarrolló un módulo para extraer *tweets* mediante la *API* de Twitter. De aquí una *corpora* de *tweets* fue recolectada formando un *dataset* de tres clases: Positivo, Negativo y Neutro. Para la recolección de *tweets* negativos y positivos se usó el mismo procedimiento propuesto por Go et. al. 2009 en [42] que consiste en la recolección de tweets con *emoticons* polares justificado por el trabajo de Read 2005 en [41] donde se concluye que un *emoticon* polar tiene una alta correlación con la polaridad del texto que lo contiene.

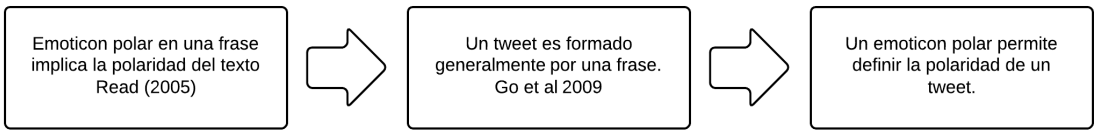



Figura 3.1: Implicancia fundamental para la generación automática de la corpora
Fuente: Elaboración propia

En la ilustración 3.1 se muestra la secuencia de afirmaciones que llevan a considerar que un *emoticon* polar presente en un *tweet*, tiene la misma polaridad que ese *tweet*. Mediante un *software* de monitoreo que extrae tweets mediante la API de Twitter proporcionado por la empresa de *Big Data Analysis, Artool*, se consultaron por comentarios que tuviesen dos tipos de *emoticons*:

- Emoticons Positivos: “:)”, ”=)”, “=D”
- Emoticons Negativos: “:(”, “=(”, “;(”

Estos dos tipos de *tweets* recolectados serán usados para entrenar el clasificador e identificar polaridades positivas y negativas.

Este método semiautomático para el etiquetado de *tweets* permite recolectar y etiquetar una gran cantidad de comentarios para las clases positivas y negativas sin mucho esfuerzo, y en poco tiempo. Pero como ya fue mencionado es importante considerar la clase neutral en nuestro trabajo debido a que no todos los tweets necesariamente presentan una polaridad [34]. En el estudio realizado por Go et al en [42] ellos hacen un pequeño acercamiento para generar un Corpus de clase neutral considerando que los tweets que no tenían *emoticons* pertenecían a esta clase. Sin duda este supuesto es demasiado débil concluyendo en la evaluación del algoritmo de 3 clases era poco efectivo, ya que tuvo resultados muy deficientes. Otro enfoque introducido por Pak & Paroubek en [4] sugiriere que se pueden obtener tweets objetivos en una forma semiautomática, considerando tweets generados por cuentas oficiales de twitter de diarios y revistas, bajo el supuesto que estos medios de comunicación emiten sus noticias en forma objetiva sin emitir juicios. Si bien este enfoque igualmente presenta un sesgo al considerar solo diarios, el esfuerzo se encuentra en encontrar una forma semiautomática de tweets objetivos, y esta lo logra de buena manera según los resultados mostrados en su estudio. El desafío se encuentra en extraer las *features* que mejor determinan las características de los textos neutrales.

Las cuentas que se consultaron para la extracción de tweets neutrales corresponden a los siguientes medios de prensa: @latercera, @emol, @lasegunda, @biobio, @cooperativa entre otros, los cuales suponen ser objetivos y no expresar ningún tipo de opinión acerca de las noticias que emiten. Para cada una de las clases se seleccionaron 80,000 tweets mediante las reglas y procedimientos descritos.

## 3.3 Analizando la estructura del *corpus*

Previo a la realización de pruebas para determinar la efectividad de la metodología propuesta, para el etiquetado automático de tweets, se analizará la Corpora para determinar si cumple con distintas propiedades que poseen los corpus de lenguaje natural.

### 3.3.1 Ley de Zipf

Zipf declaró que la distribución de frecuencias de las palabras en inglés, si estas están ordenadas de acuerdo a la cantidad de veces que aparecen en un texto, siguen una ley de potencia inversa con exponente muy cercano a 1. En otras palabras, si las palabra más recurrente en un texto ocurren con frecuencia P(1), la siguiente palabra más frecuente ocurre con frecuencia P(2), y la palabra con ranking r según su frecuencia ocurre P(r) veces, entonces su distribución es la siguiente:

$$P(r) = \frac{C}{r^{\alpha}} \tag{3.1}$$

Con C $\approx 0.1$ y $\alpha \approx 1$. Esta distribución se conoce como la Ley de Zipf y ha sido comprobada para corpus estándares en inglés actual con muy buenos resultados [35]. Validar y comprobar esta regla en un corpus es muy importante ya que confirma una característica propia del corpus recolectado, que es una propiedad de los textos naturales generados aleatoriamente [36].

De acuerdo al gráfico se puede concluir que la distribución de frecuencias de las palabras presentes en el *Corpus* siguen la ley de Zipf, lo que confirma una apropiada característica del corpus recolectado. Para llevar a cabo esta tarea y verificar esta propiedad, un script en PHP fue desarrollado, teniendo en consideración que *hashtags*, *URLs*, y las *menciones* a otros usuarios (@users) no forman parte del lenguaje natural y por esta razón este tipo de expresiones no fueron consideradas.

### 3.3.2 Otras propiedades del corpus recolectado

Para comprender la estructura interna del *corpus* recolectado, se grafican a continuación la distribución del número de palabras que posee cada tweet por cada clase, al igual que la distribución de caracteres presentes en los comentarios

A partir de la ilustración 6 se puede visualizar que la distribución que presentan los corpus positivo y negativo para el número de palabras es relativamente similar, manteniendo una

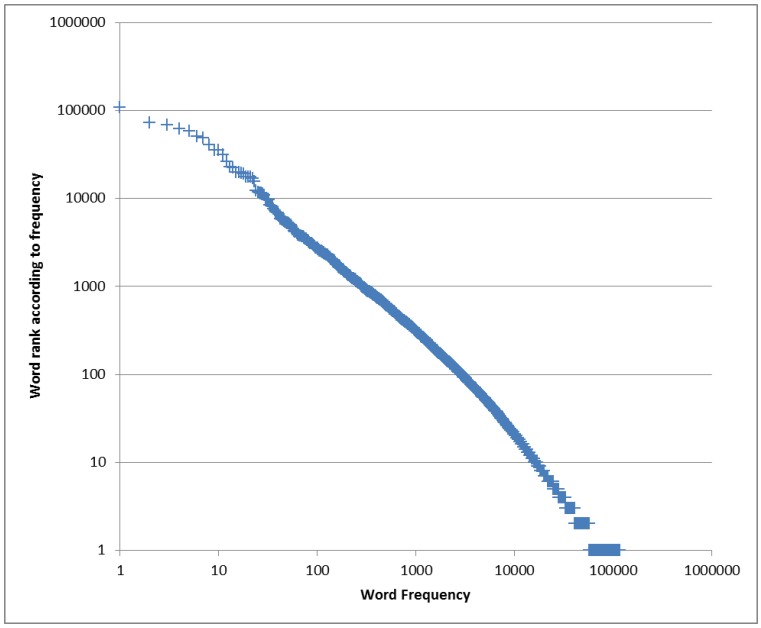



Figura 3.2: Distribución de frecuencia de palabras en la corpora
Fuente: Elaboración propia

frecuencia homogénea de tweets entre los 5 y 25 caracteres. En cambio para la clase neutral se tiene que su comportamiento es similar al de una distribución normal con media 19 y desviación estándar de 3,89 palabras. Este comportamiento distinto al de las demás clases es consecuencia de la fuente y criterio de selección de los *tweets* para esta clase. Al estar conformado el corpus neutral por comentarios de cuentas oficiales de medios de prensa, éstos siempre utilizan las mismas formas para describir la noticia, sin abreviaciones y ocupando todos gran parte de los caracteres disponibles. En las tablas 1 y 2 donde se describen la media y desviación estándar de palabras y caracteres para cada clase, se puede apreciar que la clase neutra posee en promedio 5 palabras más que las demás clases y 25 más caracteres.

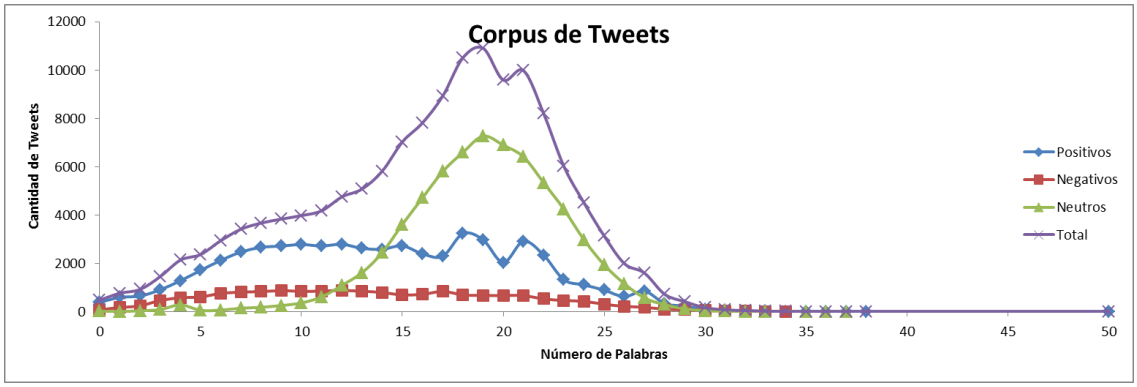



Figura 3.3: Distribución de frecuencia de tweets según número de caracteres
Fuente: Elaboración propia

En las siguientes tablas se pueden apreciar los valores de promedio y desviación estándar para cada corpus, y también para *Corpora* recolectada como un conjunto de los anteriores.

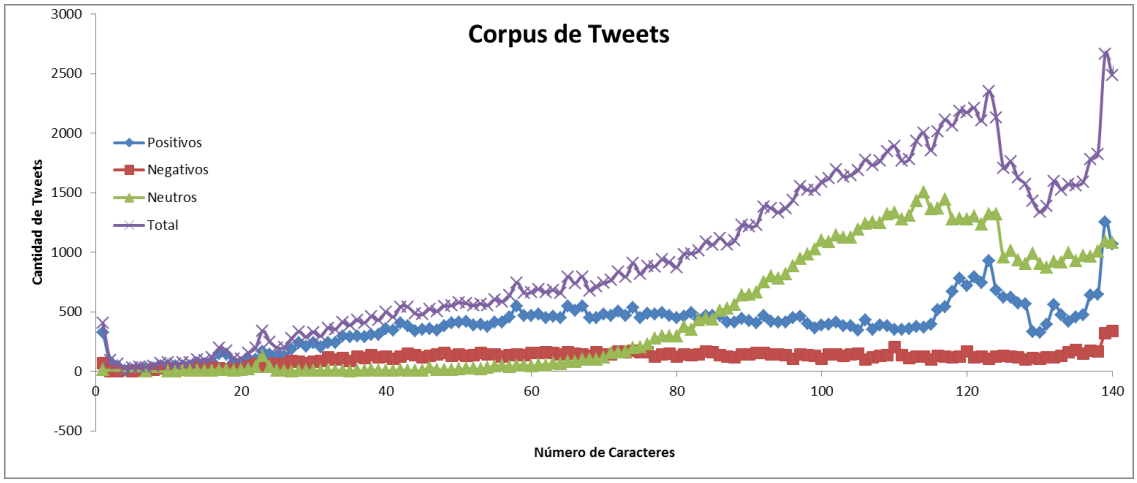



Figura 3.4: Distribución de frecuencia de tweets según número de caracteres
Fuente: Elaboración propia

| Corpus | AVG | STD |
|---|---|---|
| Positivo | 14,01 | 6,58 |
| Negativo | 13,54 | 6,75 |
| Neutro | 18,96 | 3,89 |
| Total | 15,50 | 5,74 |

Tabla 3.1: Características de la distribución de palabras de la *Corpora*
Fuente: Elaboración propia

### 3.3.3 Análisis estadístico del enfoque de etiquetado semiautomático

Para hacer una primera evaluación del enfoque de etiquetado de los tweets utilizando el criterio de los *emoticons*, se realizó un Muestreo Aleatorio Simple (MAS) para cada una de las clases de la *Corpora* recolectada. Para esta tarea se seleccionaron aleatoriamente 100 tweets de cada clase y se verificó uno a uno si las etiquetas asignadas mediante la metodología anterior fueron correcta. En la siguiente tabla se muestran los resultados de este experimento obteniendo que el mejor resultado fue para la clase neutral y que el peor para la clase positiva. Además el *accuracy* global para estas tres clases conjuntas fue de un 81%, lo cual es bastante bajo para un intento de obtener una metodología que pueda obtener una clasificación de polaridad de tweets con una precisión mayor al 85%.

Para comprender qué pudo causar este bajo desempeño del etiquetado semiautomático, el corpus de cada clase fue analizando profundamente, encontrando una serie de problemas, los cuales serán presentados en la siguiente sección.

| Corpus | AVG | STD |
|---|---|---|
| Positivo | 88,35 | 35,75 |
| Negativo | 84,33 | 35,58 |
| Neutro | 110,67 | 19,65 |
| Total | 94,45 | 5,74 |

Tabla 3.2: Características de la distribución de caracteres de la Corpora
Fuente: Elaboración propia

| Clase | Positiva | Negativa | Neutral | Total |
|---|---|---|---|---|
| Correct-tagged cases | 75% | 82% | 86% | 81% |

Tabla 3.3: Muestreo Aleatorio Simple para evaluar el enfoque semiautomático de generación de la *corpora*
Fuente: Elaboración propia

## 3.4 Problemas del enfoque presentado

A raíz de los resultados obtenidos por el muestreo aleatorio simple, se revisó cada uno de los corpus para obtener una medida de desempeño en el etiquetado semiautomático y así detectar posibles problemas de la metodología presentada. Al revisar los casos que quedaron mal etiquetados según el muestreo, se encontraron diversos problemas que generan ruido y en consecuencia, una clasificación equivocada de los *tweets*.

**Tweets sin importancia**
Solo filtrando los tweets por el *emoticon* presente en su contenido, resulta ser una regla demasiado simple para el etiquetado eficiente de tweets. Al analizar la *corpora* recolectada se encontraron tweets del tipo:

1. #AvenidaBrasil :)
2. #FelizLunes a @Brennedy2 :) #TuitUtil http://t.co/sNd2Fmtsxe

En el primer caso el tweet está conformado solo por un *hashtag* y un *emoticon*. Para el segundo caso el *tweet* contiene 2 *hashtags*, una mención a un usuario, una palabra, el *emoticon* y una *URL*. El problema es que estos tipos de tweets no son útiles para el entrenamiento ya que no agregan información distintiva al Corpus al solo considerarse como *hashtags*, *URLS* y menciones luego de aplicadas las trasformaciones a estas *keywords*. Tampoco resulta relevante utilizarlos debido a que no corresponden a textos de lenguaje natural.

A las menciones (*@username*), *hashtags* (*#topic*), *links (Urls)* y abreviación de *retweet (RT)*, los definimos como *Twitter Words* para referirnos a ellas como conjunto. Estas palabras son muy relevantes ya que como el objetivo de los usuarios en Twitter es difundir sus comentarios y captar audiencia, la gran mayoría de los Tweets tiene al menos un Twitter Word.

**Tweets Neutrales**
Al ser obtenidos los comentarios con etiquetas neutrales desde cuentas de diarios y radios,

estos poseen una considerable cantidad de palabras adicionales en comparación a los comentarios positivos y negativos (como se puede apreciar en el gráfico anterior). Esto se debe, principalmente, a que los medios de comunicación deben resumir una noticia en un tweet y además agregar un link a ella en su página, por lo que utilizan generalmente la mayor parte del reducido espacio disponible para comunicar el contenido.

**Retweets**

Otro gran problema detectado en los corpus son los *retweets*. Un *retweet* emitido por un personaje influyente o que refiera a una noticia muy importante puede ser *retwitteado* muchas veces. El *tweet* que ha sido más veces compartido en la historia de Twitter fue el emitido por Barack Obama cuando fue reelecto como Presidente de los Estados Unidos con más de 750 mil *retweets* y 290 mil favoritos [37]. El principal problema que presenta este tipo de tweets es que poseen exactamente el mismo contenido, no aportando a la variabilidad del corpus, y además, profundizando la característica polar que poseen las palabras dentro de ese tweet. Por ejemplo el tweet "Estoy aburrido de estudiar, ahora voy a jugar :)" que dada la metodología de etiquetado semiautomático, supone que cada palabra contenida en el tweet tiene una polaridad positiva, entonces al ser los *retweets* una copia del tweet original, se acentúa la polaridad positiva que el algoritmo extrae de las palabras de este tweet en particular.

**Tweets repetidos**

Otra característica importante encontrada en los tweets de la *corpora* recolectada fueron los relacionados con concursos. Dada la alta penetración de la plataforma y rápida *viralización* de las campañas, es que las empresas y marcas han comenzado a utilizar Twitter como medio para realizar concursos y sorteos. Los usuarios con el fin de ganar o aumentar sus posibilidades de ganar son capaces de emitir decenas de comentarios iguales agregando un número al final del comentario con tal de evitar la restricción de comentarios duplicados de Twitter.

*@LollapaloozaCL #ViveLollapalooza #Cocacolafmcl @CocaCola_CL Quiero ir!! asi que a poner tweets se ha dicho!! :D 1*

⋮

*@LollapaloozaCL #ViveLollapalooza #Cocacolafmcl @CocaCola_CL Quiero ir!! asi que a poner tweets se ha dicho!! :D 92*

Otro tipo de tweets con la misma característica son aquellos en los usuarios pretenden mostrar o conseguir algo en la red social. En el siguiente ejemplo extraído del corpus de comentarios positivos, un admirador de Justin Bieber menciona al cantante solicitándole que lo siga en Twitter.

*@justinbieber TE AMO MUCHO MI AMOR PLEASE MAKE MY DREAM COME TRUE, FOLLOW ME PLEASE :) 102*

El problema que presentan este tipo de tweets a la conformación del corpus siguiendo la metodología semiautomática propuesta, es el mismo que el efecto que provocan los *retweets*, donde las palabras contenidas en estos comentarios se vuelven muy polarizadas según el criterio de selección que cumplían y además no aportan a la variabilidad de palabras en el Corpus al tener prácticamente las mismas palabras . Los *peaks* anómalos que se pueden visualizar en la figura 6 y 8 para la cantidad de palabras en los corpus de clases positivas, son efectivamente una consecuencia de una campaña local que se estaba realizando para ganar entradas a un concierto.

## 3.5 Soluciones a los problemas encontrados y nueva estructura de la *Corpora*

A raíz de los problemas encontrados descritos en la sección anterior, se hicieron una serie de modificaciones a la metodología de recolección de comentarios de entrenamiento para solucionarlos. Si bien en esta sección no se mostrarán los resultados de cómo afectan estas mejoras en la clasificación de los comentarios, a continuación se enumerarán las soluciones abordadas, cómo éstas afectaron la estructura interna del corpus y un nuevo experimento de Muestreo Aleatorio Simple para determinar si mejoró el.

**Tweets sin importancia**
Como fue mencionado en la sección anterior, este tipo de tweets no contribuye con características distintivas a cada corpus al tratarse de comentarios que gran parte de su contenido son *Twitter Words*. Es por esta razón que se considera eliminar de los datos de entrenamiento todos aquellos tweets que tengan esta estructura. El criterio utilizado para identificar estos comentarios consiste en extraer todas las *Twitter Words* de los *tweets* y luego contar el número de palabras que poseen (considerando solo caracteres alfabéticos) y descartar aquellos tweets que no sobrepasen cierto umbral de palabras ya que se consideran tweets irrelevantes. Este umbral será definido en la parte de los experimentos considerando el umbral que mejores resultados de precisión se obtengan para el desempeño general del sistema.

**Retweets**
Para evitar este tipo de tweets en los datos de entrenamiento, éstos serán filtrados usando el atributo de *rt_twitter_user_id* presente en el *JSON* obtenido desde la *API* y que fue extraído al momento del *parsing* del tweet. Cuando este atributo, presente en todos los tweets, es distinto de *null* implica que el tweet en cuestión fue originado haciendo un *retweet* a otro.

**Tweets Repetidos**
Al tener este tipo de tweets el mismo efecto que los *retweets*, también serán filtrados. De acuerdo a la estructura genérica que presentan, se realizan búsquedas en cada corpus para encontrar tweets que terminen con números correlativos y que venga del mismo usuario, presentando una alta frecuencia de *posts*.

En la siguiente tabla se puede observar como variaron los promedios de cantidad de palabras y caracteres, por clase, aplicando estos filtros. Como resultado se obtuvo que la diferencia en el promedio de cantidad de palabras entre la clase neutral y las clases subjetivas se redujo considerablemente, por lo que ya no se presenta una diferencia estructural de los

tweets para las dos metodologías utilizadas para la recolección de los corpus.

| | Type | Words | | Characters | |
|---|---|---|---|---|---|
| Class | | Mean | STD | Mean | Std |
| Positive | Without Filter | 14,01 | 6,58 | 88,35 | 35,75 |
| | With Filter | 10,55 | 6,23 | 59,73 | 31,82 |
| Negative | Without Filter | 13,54 | 6,75 | 84,33 | 35,58 |
| | With Filter | 10,87 | 6,5 | 61,48 | 33,16 |
| Neutral | Without Filter | 18,96 | 3,89 | 110,67 | 19,65 |
| | With Filter | 13,48 | 3,87 | 79,24 | 20,19 |

Tabla 3.4: Corpora structure applying the tweet filters
Fuente: Elaboración propia

Por otro lado para determinar si el uso de los filtros tuvo un resultado favorable en el etiquetado automático de *tweets* utilizando las metodologías descritas en la sección de generación del corpus, se realizó nuevamente un muestreo aleatorio simple a los corpus filtrados según los criterios definidos anteriormente. En la tabla III se muestran los nuevos resultados en comparación con los obtenidos sin aplicar el filtro, donde se obtiene un aumento del 4% total en el desempeño del etiquetado automático de *tweets*. Se observa además que se tiene una disminución del 2% en el desempeño para la clase neutral, lo que se atribuye a la variabilidad propia de la metodología utilizada.

| Clase | Positiva | Negativa | Neutral | Total |
|---|---|---|---|---|
| Casos correctos | 86% | 84% | 84% | 85% |

Tabla 3.5: Nuevo experimento de muestreo aleatorio simple con la corpora filtrada

Antes de finalizar la etapa de construcción de la corpora la pregunta que surge luego de los experimentos, es que hacer con los tweets irrelevantes. Si bien éstos no van a ser considerados en la generación del corpus ya que agregan ruido a éstos, sabemos que Twitter está lleno de este tipo de comentarios y al intentar clasificarlos éstos serán incorrectamente asignados a alguna clase disminuyendo el *accuracy*. A diferencia de los otros tipos de *tweets* filtrados (i.e *tweets* repetidos y *retweets*) que pueden ser clasificados correctamente debido a que su estructura es considerada en el corpus para el entrenamiento de los algoritmos, los *tweets* irrelevantes son eliminados completamente, de modo que no hay *tweets* de este tipo en toda la corpora. Es por esto que se propone una simple regla de clasificación de *tweets* para evitar este problema. Ésta consiste en que antes de la clasificación de un *tweet* en positivo, negativo o neutro, se verifique si el *tweet* cumple con las características de un *tweet* irrelevante (i.e. posee más de 3 palabras sin considerar *Twitter Words*). De ser así entonces en vez de ser clasificado en alguna de las categorías descritas, es automáticamente etiquetado con la categoría de *tweets* irrelevantes.

# Capítulo 4

# Extracción de *Features* y Algoritmos de Clasificación

Los pasos fundamentales para la construcción de un clasificador supervisado son 1) escoger el algoritmo de aprendizaje, 2) escoger las *features*, 3) recolectar la data de entrenamiento, 4) entrenar el algoritmo y 5) evaluar el clasificador. Para mejorar el desempeño global del sistema, la aplicación de técnicas en cualquiera de estas etapas puede ser utilizado, ya sea alternando entre distintos algoritmos de entrenamiento y clasificación, escogiendo distintas metodologías para la extracción de *features*, y la selección de cuáles de éstas conformarán parte de la *Corpora* para entrenar los algoritmos.

En el capítulo anterior se abordó el punto 3 sobre cómo recolectar los datos desde Twitter mediante una metodología que permitiese obtener una cantidad tan grande como fuese posible para entrenar los algoritmos. Además se realizaron mejoras sobre la metodología utilizada para el etiquetado automático de datos. En este capítulo se abordarán los puntos 1), 2) y 4), escogiendo el algoritmo de aprendizaje que se utilizará, cuáles serán las técnicas de extracción de características utilizadas y se definirán los procedimientos para el entrenamiento del algoritmo.

## 4.1 *Feature Extraction*

El proceso de extracción de las características de los datos es uno de los más complejos ya que en base a éstos se entrenan los algoritmos que posteriormente realizarán la clasificación.

Existen 4 categorías de *features* que han sido utilizadas en trabajos previos. Éstos incluyen *features* del tipo sintáctico, semántico, basado en links y de estilo. Entre estas *features* las más utilizadas son las semánticas y sintácticas para la tarea de *Sentiment Analysis* [51]. Estas incluyen el uso de *n-grams* [31, 52], *POS-Tags* [31, 53, 54], y signos de puntuación [55]. Otras *features* utilizadas incluyen patrones sintácticos utilizando *Pos-tags* en combinación con *n-grams* determinando que el par adjetivo polar y adverbio generalmente define la polaridad de una oración.

*Features* del tipo semántico pueden incorporar técnicas de anotación del *corpus* manuales, semiautomáticas o completamente automáticas: *Sentiment lexicons* generados manual o semiautomáticamente [56], generalmente usan un set inicial de datos para generar automáticamente términos que son manualmente filtrados y codificados de acuerdo a su polaridad e intensidad de la información. Las etiquetas definidas son incorporadas para indicar la polaridad positiva o negativa del sentimiento de las frases. Rilof et. al. en [57] utilizó herramientas para la generación semiautomática de un lexicón para construir sets de subjetividad fuerte, subjetividad débil, y sustantivos objetivos. Este enfoque supera el uso de otras técnicas de extracción de *features*, incluyendo el uso de "*bag of words*" para la clasificación de textos objetivos versus subjetivos en inglés. Otra técnica que ha mostrado buenos resultados para la anotación de palabras y frases es la de *Appraisal Groups* abordada por Whitelaw et. al en [58], donde los términos iniciales son generados utilizando *WordNet*, los cuales son luego filtrados manualmente para construir el lexicón. Luego cada expresión es manualmente clasificada entre varias *Appraisal Classes*. Estas clases consideran la orientación y polaridad de las frases. Whitelaw 2005 obtuvo buenos resultados utilizando esta técnica sobre un *corpus* de reseñas de películas, superando la precisión obtenida por otros estudios como los de [59], el enfoque de *Mutual information* de Turney et. al. 2002 en [60] y también el uso de *features* sintácticas por parte de Pang et. al. 2002.

El enfoque basado en links o citas, utiliza éstos para determinar sentimientos presentes en los documentos que los contienen. Por ejemplo [61], concluyó que las páginas que tiene muchos links entre ellas, generalmente presentan sentimientos generales. Por otro lado Agarwal et. al 2009 en [62] observó que en los foros, las respuestas son generalmente antagonistas, en el sentido de que tienen el sentimiento opuesto a las preguntas. Debido al poco uso que tiene el uso de este tipo de generación de *features* es que no se puede establecer que tan efectivo resulta para la detección y clasificación de sentimientos.

Las *features* extraidas desde el enfoque el de estilo incluyen los atributos estructurales que tienen los textos. Wiebe et. al. en [63] utilizó el hecho de palabras que ocurren una sola vez para determinar la subjetividad y opinión de documentos observando que existe una mayor presencia de palabras que aparecen una única vez en textos del tipo subjetivo que objetivos. Gamon 2004 utilizó otros atributos de estructura como el largo de las oraciones en conjunto con otro tipo de *features* para mejorar la clasificación de sentimientos [53].

En la siguiente tabla se puede visualizar un resumen de los tipos de *features* utilizados para *Sentiment Analysis* en textos elaborado por Abbasi et.al 2002 donde se hace un resumen exhaustivo de éstos.

| Categoría | Ejemplos |
|---|---|
| Sintácticos | n-grams de palabras /Postags, puntuaciones |
| Semánticos | Tags polares, appraisal groups, orientación polar |
| Links | Web links, patrones de mensajes y respuestas |
| De estilo | Medidas de estructura y estilo del léxico |

Tabla 4.1: Resumen de tipos de features
Fuente: Abbasi 2002 [51]

El enfoque abordado para la extracción de *features* en este trabajo considera el uso de

*features* del tipo sintáctico, que son más genéricos y aplicables a cualquier tipo de textos. En particular se utilizará *Treetagger* para las tareas de anotación de los datos y obtener los *Pos-tags* presentes en los tweets. Para la extracción de *n-grams*, se utilizará un script desarrollado ya que permite más flexibilidad al poder considerar *bigrams*, y tokens definidas de acuerdo a las necesidades de los experimentos.

En la gran mayoría de los trabajos se considera el uso de *n-grams* y *POS-taggers* para la extracción de *features* ya que han mostrado tener buenos resultados en la tarea se *Sentiment Analysis*. En los trabajos de Pak et al 2010, Pang et al 2008, entre otros, se consideraron el uso de *unigrams, bigrams y trigrams*. Para algunos han tenido mejores resultados con *uni-grams* mientras que otros con *bigrams*. Las principales conclusiones apuntan a que el uso de un mayor número de palabras para la conformación de *n-grams* tiene como efecto que se aporta con más contexto de la frase al clasificador. Pero también aumenta el número de *features* distinas en la *Corpora*, al aumentar el número de combinaciones, por lo que también al considerar trigrams se tienen peores resultados [4]. En este trabajo el enfoque está en la utilización de *unigrams* de palabras en conjunto con los *POS-tags* extraídos de las oraciones, en donde además se consideran los tags de signos de puntuación. El proceso para obtener los *unigrams* desde los comentarios de twitter se realiza de la siguiente manera:

1. *Filtering*: Todos las *Twitter Words* (i.e. @mentions, #Hashtags, Urls y Rts) son reemplazados de los *tweets* colocando en su lugar los siguientes términos respectivamente: [@], [#], [U], [RTs]. Además los *emoticons* también fueron removidos de tal forma que el modelo generado no dependa de ellos, al ser estos precisamente el criterio utilizado para la conformación de los corpus positivo y negativo.

2. *Tokenization*: Los criterios de tokenización consisten en separar los *strings o comentarios* a partir de los espacios y signos de puntuación.

3. *Stopwords removal*: Se utilizó un listado de *stopwords* obtenido de [64] para filtrar de las tokens todas aquellas tokens que estén contenidas en éste.

Para extraer las *features* de los *POS-tags*, se utilizó el software *TreeTagger* que entrega para cada frase procesada, devuelve un listado de las palabras contenidas con su *POS-tag* asociado. En este caso nos interesa ingresar todo el tweet, reemplazando las menciones del tweet por un nombre (utilizamos Pablo), de forma que el software lo reconozca como un sustantivo propio. Por otro lado los *hashtags*, *Urls*, y *RTs* fueron filtrados antes de someterlos al *POS-tagging*. El proceso quedó finalmente de la siguiente manera:

1. *Filtering*: Se filtran de los tweets todos las *Twitter Words* a excepción de las @mentions que se reemplazan por el nombre propio. Además se filtran los emoticons para que no sean etiquetados como signos de puntuación.

2. *Tagging*: El anotado de las palabras presentes en los tweets se realiza mediante el software *TreeTagger*, que asigna etiquetas a palabras, signos de puntuación y números presentes.

En este caso no se hace un filtro de los *POS-tags* asociados a *stopwords*, ya que como se muestra en el trabajo de Pak et al. Los tipos de artículos, formas verbales y otros aspectos gramaticales son diferentes para textos objetivos y subjetivos, por lo que esto son características distintivas para clasificación.

## 4.2 Algoritmos de Clasificación de Polaridad de Sentimientos

Las técnicas utilizadas para la detección de polaridad de sentimientos pueden ser clasificadas entre tres categorías. Estas incluyen algoritmos de *Machine Learning*, análisis de links y enfoques relacionados con la asignación de puntajes.

Muchos estudios se han enfocado en el uso de algoritmos de *Machine Learning* entre los que destacan *Support Vector Machines*, Clasificadores Bayesianos y Máxima Entropía, siendo el Clasificador de Naive Bayes el más utilizado entre estos. Pang Lee 2004, Pang et al 2002 ha usado *SVM* para reseñas de películas, mientras que Whitelaw 2005 ha utilizado NB para análisis de discurso en la *web*. Pang ha determinado que el *SVM* tiene mejores resultados que el NB para la detección de polaridad, mientras que Go et al. 2009 ha determinado que NB tiene mejores resultados aplicando otras técnicas para mejorar el *accuracy*.

Los métodos basados en puntajes son comúnmente utilizados en conjunto con *features* del tipo semánticas. Con estas técnicas generalmente se clasifican los sentimientos de los mensajes de acuerdo a la ponderación o suma total de polaridad positiva o negativa asociados a los sentimientos de las palabras o *features* que poseen los documentos. Para la detección de patrones se han utilizado enfoques como la creación de frases polares (positivas o negativas). A las frases positivas se les asigna un +1 mientras que a frases negativas un -1. Todos los mensajes con suma positiva, son considerados positivos y negativos los que tienen suma negativa. Podría agregar un poquito extra.

En este trabajo se utilizó el clasificador de Naïve Bayes que ha probado tener buen desempeño para la clasificación de sentimientos en Twitter. El clasificador de Naïve Bayes está basado en el teorema de Bayes, que para el caso de este trabajo asigna a un tweet T representado por un vector de *features* F, la clase con la mayor probabilidad de pertenencia.

$$P(c|F) = \frac{P(c)(F|c)}{P(F)} \tag{4.1}$$

El prior $P(c)$ es el mismo para cada clase en nuestros experimentos ya que se extrae el mismo número de tweets de entrenamiento para cada Corpus. Además se asume independencia condicional entre las *features*, aunque se sabe que en los textos de lenguaje natural las palabras son dependientes unas de otras, se ha probado que el clasificador de Naïve Bayes (Ingenuo) se desempeña de buena forma para la clasificación de polaridad de sentimientos incluso bajo este supuesto.

$$P(F) = \sum_{c \in C} P(F|c)P(c) \tag{4.2}$$

$$P(c|F) = \frac{P(c) \cdot P(F|c)}{\sum_{c \in C} P(F|c) \cdot P(c)} \tag{4.3}$$

Al tener el mismo número de *tweets* para cada clase entonces la probabilidad de obtener un *tweet* de una clase aleatoria es igual a 1/card(S). Por esta razón se tiene que el prior de cada clase es la misma.

$$P(c) = \frac{1}{card(C)}, \forall c \in C \tag{4.4}$$

Por ejemplo para el caso de tener 2 clases para un problema de clasificación de polaridad entre positivos y negativos, entonces la probabilidad de obtener un tweet positivo o uno negativo es la misma, al tener misma cantidad de tweets de cada clase que conforman la Corpora. $P(positive) = P(negative) = 0.5$.

Entonces el Posterior puede ser escrito de la siguiente forma:

$$P(c|F) = \frac{P(c) \cdot P(F|c)}{P(c) \sum_{c \in C} P(F|c)} = \frac{P(F|c)}{\sum_{c \in C} P(F|c)} \tag{4.5}$$

Como para cada clase la expresión $P(c|F)$ tiene el mismo denominador, entonces la clase que tiene mayor probabilidad de pertenecer un tweet, dado el vector de *features* que lo representan, está dada por:

$$arg \max_{c \in C} P(c|F) = arg \max_{c \in C} P(F|c)$$

Este fue el criterio utilizado para construir dos clasificadores utilizando los dos tipos de *features* considerados: *unigrams* de palabras y *Part of Speech Tags*. Uno de estos clasificadores utiliza un enfoque combinado de ambas *features* para determinar la polaridad de un comentario, y el otro un algoritmo de dos niveles que primero clasifica un *tweet* entre subjetivo u objetivo utilizando solamente los POS-tags como *features*, y luego los tweets subjetivos son clasificados entre positivo y negativo considerando *unigrams* de palabras como las *features* de este problema.

### 4.2.1 *Clasificador de Naïve Bayes para 3 clases*

Para utilizar ambos tipos de *features* y confeccionar un algoritmo de 3 clases, la probabilidad posterior puede ser escrita como la combinación de los posteriores de cada modelo al asumir independencia entre los *POS-Tags* y *unigrams* basados en tokens:

$$P(c|T) =\sim P(U|c) \cdot P(P|c) \tag{4.6}$$

Donde U es un vector de *unigrams* de palabras que representan al Tweet T y P es un conjunto de *POS-tags* presentes en él. Como se dijo anteriormente, al considerar independencia entre ambos tipos de *features*, y entre *features* del mismo tipo se tiene:

$$P(U|c) = \prod_{u \in U} P(u|c) \tag{4.7}$$

$$P(P|c) = \prod_{p \in P} P(p|c) \tag{4.8}$$

$$P(c|T) \sim \prod_{u \in U} P(u|c) \cdot \prod_{p \in P} p(u|c) \tag{4.9}$$

El término $P(U|c)$ para obtener la clase con máxima probabilidad de pertenencia, dado un vector de *unigrams* de palabras se calcula como:

$$P(u|c) = \frac{P(u,c)}{P(c)} = \frac{card(u \in c)}{card(U)} \cdot \frac{card(Tweets)}{card(Tweets \in c)} \tag{4.10}$$

**Suavización de Laplace** Un problema que tiene este enfoque es que dada la naturaleza de los textos en lenguaje natural y su distribución (*Zip's Law*), es muy probable encontrar *unigrams* o palabras que no se encuentran en el corpus, por lo que no han formado parte del entrenamiento [**?**]. Si no existen tweets que contengan una palabra, entonces la probabilidad condicional asociada a esa palabra al ser clasificada en alguna clase sería:

$$P(u = word|c = class) = \frac{card(word, c = class)}{card(c = class)} = 0 \tag{4.11}$$

Las probabilidades con valor cero no pueden ser condicionadas, independiente de otra evidencia, por lo tanto la probabilidad del tweet de pertenecer a una clase donde una palabra de su contenido no existe es:

$$\prod_{u \in U} P(u|c) = 0 \tag{4.12}$$

Para evitar este problema de asignar una probabilidad cero a un tweet, se pueden utilizar diversas técnicas para suavizar la probabilidad. En este trabajo se utilizó la suavización de Laplace que en la práctica consiste en asumir que todos los *unigrams* se han visto al menos una vez, esto es:

$$P(u|c) = \frac{card(u \in c) + a_j}{card(c) + a} \tag{4.13}$$

Donde $a = \sum_j a_j$. Un caso especial de la suavización de Laplace es la llamada "*add one smoothing*" y que se obtiene al establecer $a_j = 1$. Aunque en [35] se muestra la suavización de Laplace no es tan efectiva como otras técnicas de suavización, se utilizará para evitar este problema en nuestro clasificador.

Aplicando la suavización de Laplace de *add one* a nuestro clasificador bayesiano obtenemos:

$$P(u|c) = \frac{card(u \in c) + 1}{card(c) + k} \tag{4.14}$$

Donde$k$ es el número distinto de distintos valores de *unigrams*. A esto se le conoce como vocabulario o cardinalidad del corpus.

### 4.2.2 Clasificador de Naïve Bayes de dos niveles

Para el clasificador Bayesiano en dos niveles la metodología consiste en separar el proceso de clasificación en dos etapas. Dadas las diferencias que existen en las características de los textos objetivos y subjetivos, se plantea que las etiquetas de *POS-tagging* definen de buena forma cada una de estas clases, por lo que serán éstas las utilizadas para la elaboración de un clasificador de Naive Bayes a partir de las cuales se determinará si un tweet es objetivo o subjetivo.

La segunda etapa de este proceso consiste en clasificar aquellos tweets subjetivos entre las clases positivas y negativas. Para esto se utilizó un clasificador que ocupa como *features*, *unigrams* basados en las palabras presentes en los *tweets*. Para ambas partes del clasificador se utilizan los mismos algoritmos que para el clasificador Bayesiano de 3 clases, considerando las *features* mencionadas para cada etapa. Además para ambas partes del proceso utiliza la suavización de Laplace con la variante de *add one*.

Al igual que para el caso de los *unigrams* de palabras, la probabilidad que un set de *POS-tags* $P$ pertenezca a una clase $c \in C$ está determinada por:

$$P(P|c) = \prod_{p \in P} P(p|c) \tag{4.15}$$

Donde la probabilidad $P(p|c)$ es equivalente a preguntar la probabilidad que un postag pertenezca a la clase c en cuestión. Además aplicando el mismo criterio de *add one smoothing* se tiene que

$$\prod_{p \in P} P(p|c) = \prod_{p \in P} \frac{card(p \in c) + 1}{card(p) + k} \tag{4.16}$$

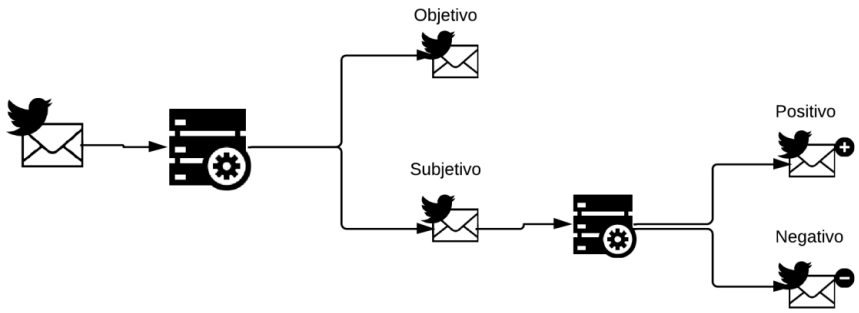



Figura 4.1: Diagrama del algoritmo en 2 niveles
Fuente: Elaboración propia

En la figura anterior 4.1 se esquematiza la clasificación propuesta por el algoritmo de dos niveles basado en el clasificador de Naïve Bayes. En un principio, los *tweets* son clasificados entre subjetivos y objetvos, utilizando los *POS-tags* como *features* y luego, a partir de los *tweets* subjetivos, se realiza una nueva clasificación para determinar cuáles de éstos son positivos y negativos. Esta última clasificación utiliza *unigrams* de palabras como features del clasificador.

# Capítulo 5

# Implementación

En este capítulo de explican los softwares y aplicaciones desarrolladas para llevar a cabo las tareas de conformación del corpus, clasificación de los tweets, evaluación de los algoritmos e implementación general del sistema online. Estas aplicaciones siguen las especificaciones definidas en los Capítulos 3 y 4. Además se detallan con el fin de hacer más fácil la comprensión de las técnicas utilizadas y para modificaciones en trabajos futuros.

Primero se detallan las herramientas, lenguajes de programación y softwares utilizados para el desarrollo del sistema. Éstas deben cumplir con los requerimientos definidos en los objetivos y que permitan y faciliten la construcción del sistema.

Además se describirá la implementación de cada módulo considerado en el diseño del sistema.

## 5.1 Herramientas utilizadas en el desarrollo del sistema

Para la implementación de un sistema, es importante escoger aquellas que permitan alcanzar los objetivos propuestos para la aplicación, y que también resulten sencillas de utilizar para que la programación no presente un obstáculo.

A continuación se detallan las principales herramientas utilizadas para la implementación del sistema:

**Sistema operativo Debian:** Es una distribución de Linux de código abierto. Es muy utilizado entre los usuarios avanzados debido a su excelencia técnica y compromiso con las necesidades de la comunidad de Linux. Esta distribución forma parte de la base del sistema operativo conocido como Ubuntu que se enfoca más en la interfaz y experiencia de usuario más que de la estabilidad.

**Perl:** Es un lenguaje de programación con características similares a C, y de menor nivel que muchos otros lenguajes de programación. Estructuralmente está basado en un estilo

de bloques y es ampliamente utilizado por su desempeño en el procesamiento de textos y debido a sus características. Se previó que fuera práctico (fácil de usar, eficiente y completo) en lugar de hermoso (pequeño, elegante y mínimo). Soporta programación estructurada como programación orientada a objetos y la programación funcional. Tiene incorporado un poderoso sistema de procesamiento de texto y existen muchos módulos disponibles. Esta última es la principal razón por qué se utilizará este lenguaje.

**PHP:** Del inglés *Hypertext PreProcesor*, PHP es un lenguaje de programación de uso general, interpretado en el lado del servidor que originalmente fue diseñado para el desarrollo web de contenido dinámico. Las ventajas que posee este lenguaje es que fue de los primeros que se podían incorporar directamente con HTML en lugar de llamar un archivo externo. El código PHP es interpretado por un servidor web como Apache y en el caso de una página web, devuelve el código que la genera al cliente. Este lenguaje es utilizado porque es sencillo, admite la programación orientada a objetos y es gratuito. Las características generales del lenguaje PHP son:

- Está orientado al desarrollo de aplicaciones web dinámicas.
- Admite conexión a bases de datos mediante comandos sencillos.
- Es considerado un lenguaje sencillo de aprender.
- El código de fuente es invisible en el navegador web del cliente una vez interpretado por el servidor.
- Capacidad de conexión con la mayoría de los motores de bases de datos. Entre ellos *PostgreSQL* y *MySQL*
- Capacidad de utilizar librerías y módulos para añadir funcionalidades.
- Posee una amplia documentación.
- Permite la aplicación de programación orientada a objetos.
- No requiere de definición de tipos de variables.
- Manejo de sockets para la creación de aplicaciones cliente-servidor.

**PostgreSQL:** Es un Sistema de gestión de Bases de datos orientado a objetos y de código abierto, por lo que su desarrollo no es manejado por una persona o empresa, como es el caso de Oracle DB y Microsoft SQL Server, sino que es dirigido por una comunidad de desarrolladores que trabajan de forma desinteresada. Este SGBD es orientado en este proyecto ya que admite una fácil incorporación de índices y además es gratuito.

**JSON:** *Javascrit Object Notation* es un formato estructurado para el intercambio de datos. Su uso se ha masificado gracias a la simpleza que posee el formato especialmente como alternativa a XML en AJAX. JSON se utiliza ya que respuesta que envía la *API* de Twitter a la petición de los tracking son en este formato.

**TreeTagger:** Es una herramienta para anotar texto con la información de sus *POS-tags* y lematizar las palabras. Fue desarrollado por Helmut Schmid en el instituto de lingüísticas computacionales de la Universidad de Stuttgart. Esta herramienta está disponible para más de 10 idiomas y puede ser adaptada entregándole un lexicón etiquetado manualmente. Tiene extensiones para ser utilizado en varios lenguajes de programación entre los que se encuentra Perl.

| Palabra | POS | Lema |
|---|---|---|
| The | DT | the |
| TreeTagger | NP | TreeTagger |
| is | VBZ | be |
| easy | JJ | easy |
| to | TO | to |
| use | VB | use |

Tabla 5.1: Ejemplo de uso de *TreeTagger*
Fuente: Sitio oficial de *TreeTagger*

**Amazon Web Services** Es un servicio de infraestructura *TI* para empresas en forma de servicios web, más conocido como *servicios en la nube*. De todos los servicios que ofrece AWS, se utilizó el denominado EC2 que consiste en el arriendo de instancias virtuales de computadores. La principal razón por la que se consideró esta opción para el desarrollo e implementación de la arquitectura, es el largo periodo que implica la fase de entrenamiento debido al limitado número de comentarios que pueden ser lematizados y anotados mediante el *software TreeTagger*, por lo que resulta de especial interés que el proceso de entrenamiento no sea interrumpido. En caso de ocurrir esto, implicaría reiniciar el proceso de entrenamiento y para los 240 mil tweets que posee la corpora, extraer y procesar las features de acuerdo a los criterios mencionados, proceso el cual dura en total 22 horas. Es por esto que en una primera instancia se consideró Amazón por la estabilidad que posee en términos de conexión y disponibilidad. Aunque luego para evitar el largo tiempo de etiquetado, los tweets comenzaron a ser guardados en la base de datos con sus *postags, lematizaciones y unigrams* una vez que eran procesados por TreeTagger.

## 5.2 Módulo de Recolección y *parseado* de *tweets*

Para la construcción del módulo de recolección de tweets se utilizaron las librerías *Phirehose* para la conexión a la *REST API* de Twitter y la librería PHP curl, para el envío de peticiones vía protocolo HTTP. Los parámetros *track* utilizados para la descarga de tweets fueron los *emoticons* anteriormente mencionados. Además se utilizó el atributo *follows* para obtener los *tweets* de las cuentas que están relacionados con los medios de prensa seleccionados.

| | |
|---|---|
| Emoticons Positivos | :) , =) , ;) , :D |
| Emoticons Negativos | :( , =( , ;( |
| Cuentas de prensa | @biobio, @cooperativa, @emol, @latercera, @cnnchile |

Tabla 5.2: Criterios de recolección de tweets para entrenamiento
Fuente: Elaboración propia

Una vez descargados los *tweets*, éstos son almacenados en una tabla llamada *raw_tweets* que posee: el *JSON* con todos los atributos del *tweet*, la fecha en que ingresó a la base y un *boolean* que indica si el *tweet* fue *parseado* o no. El proceso de *parseado* implica la extracción de todos los campos del *JSON* que son de interés para la realización de los experimentos e implementación del sistema. Para esto, un *software* desarrollado en Perl es el encargado de consultar a la base por todos aquellos *tweets* no *parseados* en la tabla *raw_tweets*, para luego extraer todos los campos de nuestro interés e insertarlos en la tabla *parsedtweets_train*. Una vez *parseado* este *tweet*, se cambia el atributo *boolean: parsed* en la tabla *raw_tweets* a true. También podría ser borrado de la tabla *raw_tweets*, pero los mantenemos por si en el futuro deseamos ocupar alguno de los atributos no *parseados*. A continuación se puede visualizar el modelo de las tablas utilizadas para este módulo.

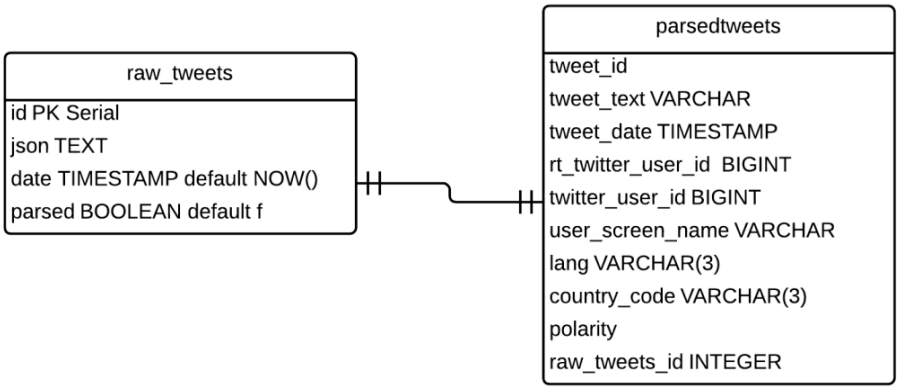



Figura 5.1: Tablas módulo de recolección
Fuente: Elaboración propia

Al momento del parseo del tweet se verifica por cuál de los criterios entró a la base examinando el atributo *tweet_text* en el *JSON*. Si este posee alguno de los *emoticons* positivos, se fija el atributo *sentiment* en 1, si tiene un *emoticon* negativo en 2 y 3 si el *tweet* tiene como atributo *user_screen_name* alguno de los medios de prensa seleccionados. Además si algún tweet cumple simultáneamente dos de los criterios, este es descartado de la base en *raw_tweets* y *parsedtweets_train* ya que provoca ambigüedad en el criterio de etiquetado automático.

Para el experimento que contrasta el uso de un Corpus en español con otro que considera solo tweets de usuarios chilenos, se realizan ciertas modificaciones a esta estructura incorporando dos tablas de *parsedtweets_train* con el fin de poder almacenar de forma sencilla ambos conjuntos de datos de entrenamiento. En una se colocarán sólo *tweets* emitidos por usuarios chilenos, y en la otra los tweets en *lenguaje español* que no tengan el atributo *country_code* con valor CL. Este procedimiento sólo se realiza de esta forma para las clases subjetivas, ya que para la objetiva se utiliza el mismo conjunto de usuarios pertenecientes a medios de prensa locales.

El procedimiento que sigue el algoritmo para la inserción de los tweets en las tablas es el siguiente: Al momento del *parseado* se verificarán 3 cosas:

1. Si el *tweet* tiene atributo *country_code* con valor 'CL'

2. Si el *tweet* tiene atributo lang con valor 'es'

3. Que el *tweet* contenga estrictamente *emoticons* del tipo positivo o negativo, y no ambos.

- Si se cumple 1 y 3, se guardan los datos *parseados* en una tabla específica para *tweets* con *country code* CL denominada *parsedtweets_train_cl*.
- Si se cumple 2, 3 y no se cumple 1, se guardan los datos *parseados* en una tabla específica con *tweets* con *lang* 'es' y que *country_code* es distinto de 'CL' denominada *parsedtweets_train_es*
- Si el *tweet* no cumple 1 ni 2 entonces no se guarda en el sistema.

## 5.3 Módulo de entrenamiento

El módulo de entrenamiento es aquel que se encarga de la selección de tweets, extracción de las *features* de los *tweets* seleccionados, el entrenamiento del algoritmo y finalmente persistencia de los datos entrenados. A continuación se detallará la implementación para cada una de estas tareas.

**Selección de tweets:**
Para relajar ciertos supuestos del algoritmo de clasificación, es necesario que para la data de entrenamiento se utilice la misma cantidad de tweets por cada corpus. Utilizando los filtros explicados en el capítulo 3 el proceso de selección de *tweets* consta de obtener solo aquellos *tweets* que cumplan con las siguientes tres condiciones:

1. El atributo *rt_twitter_user_id* debe ser igual a null, para evitar problemas con la repetición excesiva de *features* en los corpus.

2. No debe ser un *tweet* repetido, que como se definió anteriormente consisten en aquellos comentarios que poseen el mismo contenido, y cambian únicamente en el número que poseen al final del *tweet*.

3. No deben ser clasificados como *tweets* irrelevantes, que son todos aquellos comentarios que tienen menos de 4 palabras, sin contabilizar las *Twitter Words*.

Para obtener tweets que cumplan con la condición 1 se realiza la siguiente consulta SQL:

```
SELECT *   FROM parsedtweets_train   WHERE rt_twitter_user_id IS NULL
```

Para filtrar los tweets repetidos, mediante un script se verifican aquellos tweets de los mismos usuarios y que termina con números. Si se verifica que presentan serialidad, entonces se descartan y se mantiene solo el tweet original.Para facilitar la búsqueda de tweets repetidos, la consulta anterior para la extracción de tweets se modifica a:

```
SELECT * FROM  parsedtweets_train WHERE rt_twitter_user_id IS NULL
 ORDER BY user_screen_name, tweet_text
```

Para filtrar aquellos tweets que no cumplen con la condición 3, de *tweets* irrelevantes, se elaboró un script que mediante expresiones regulares divide un *tweet* en sus palabras (sólo aquellas alfabéticas), considerando como separaciones entre palabras espacios y signos de puntuación, para posteriormente obtener el número de cuántas posee. Si este número es inferior a 4 entonces el tweet se descarta y no forma parte del conjunto de entrenamiento. Este criterio de preselección de tweets es el que se utiliza para obtener 80.000 tweets por cada clase para los experimentos considerando filtros, identificando en la base de datos con el atributo corpus = true para los tweets seleccionados.

**Extracción de *features* y entrenamiento del algoritmo:**
Para la extracción de *features* se elaboró un *script* en PHP que es el encargado de extraer los *tweets* seleccionados para el corpus, desde la base de datos. Este mismo *script* envía cada *tweet* al servidor encargado de realizar el *POS-tagging* y lematización efectuado por el TreeTagger, para luego devolver cada palabra con su lematización y el *POS-Tag* asociado. Además el *script* lleva una cuenta del número total de *tweets*, el número de *unigrams* de palabras, *POS-tags* de cada clase, y la frecuencia de cada *feature* dentro de cada corpus. Esto para luego generar la matriz de probabilidad que permite la clasificación de los comentarios.

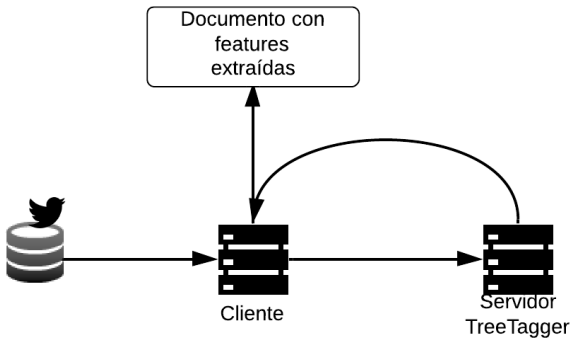



Figura 5.2: Diagrama extracción de *features* y entrenamiento del algoritmo
Fuente: Elaboración propia

**Persistencia de los datos de entrenamiento:**
Por último como los procesos de lematización y *POS-tagging* llevado a cabo por TreeTagger necesitan de un tiempo considerable, todas las *features* extraídas de los tweets son almacenadas en un documento, de tal forma que si el servidor sufre alguna caída, los datos pueden ser cargados rápidamente para seguir con la clasificación en tiempo real de los tweets, y no tener que realizar nuevamente el entrenamiento.

Posteriormente para agilizar el mismo proceso de entrenamiento y no tener que realizar el *POS-tagging* y lematización por cada set que se desee entrenar y clasificar, todo la *corpora* seleccionada fue anotada y el resultado de este proceso fue almacenado en la base de datos haciendo referencia a los tweets que pertenece cada *unigram* lematizado y *POS-tag*. La estructura de base de datos de este módulo se muestra en la figura 5.3.

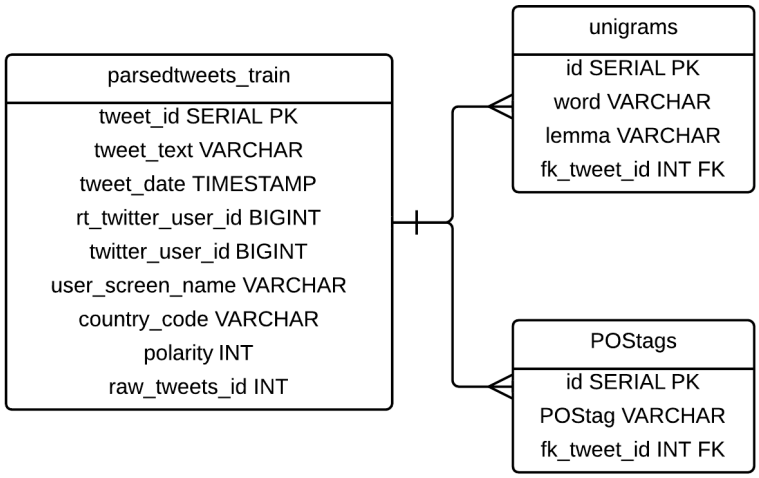



Figura 5.3: Diagrama tablas para la persistencia
de los datos lematizados y anotados
Fuente: Elaboración propia

## 5.4 Módulo de clasificación

El módulo de clasificación es el encargado de determinar la clase más probable a la que pertenece un tweet que ha ingresado al sistema. Para esto se utilizan las dos variantes descritas en el capítulo 4 del clasificador Bayesiano. Los tweets que ingresan al sistema por cualquier término o usuario que se esté monitoreando, se ingresan a la tabla *parsedtweets*, que posee atributos similares a los de parsedtweets_train con la diferencia que para este caso el campo *sentiment* es nulo por defecto. Una aplicación cliente es la encargada de obtener todos aquellos tweets que cumplan con esta condición para luego someterlos 1 a 1 al servidor que juega el rol de clasificador cumpliendo las siguientes etapas:

1. Si se considera el uso de la cuarta clase definida por la regla de tweets irrelevantes, entonces se contabiliza el número de palabras que posee el tweet sin considerar las *Tweet Words* y si son menos de 4 entonces se devuelve el valor 4 al cliente 4 que corresponde a la clase de tweets irrelevantes.
2. El tweet sin clasificar es procesado por el TreeTagger
3. Se obtienen los *POS-tags* y los *unigrams* conformados por las palabras lematizadas.
4. Se aplica el algoritmo escogido para la clasificación (i.e NB de 3 clases o NB en 2 niveles).
5. Se devuelve el valor de la clasificación al cliente: 1 para positivo, 2 negativo y 3 neutro.

Una vez clasificado el tweet, el cliente actualiza el campo del *sentiment* con el valor de respuesta enviado por el servidor clasificador.

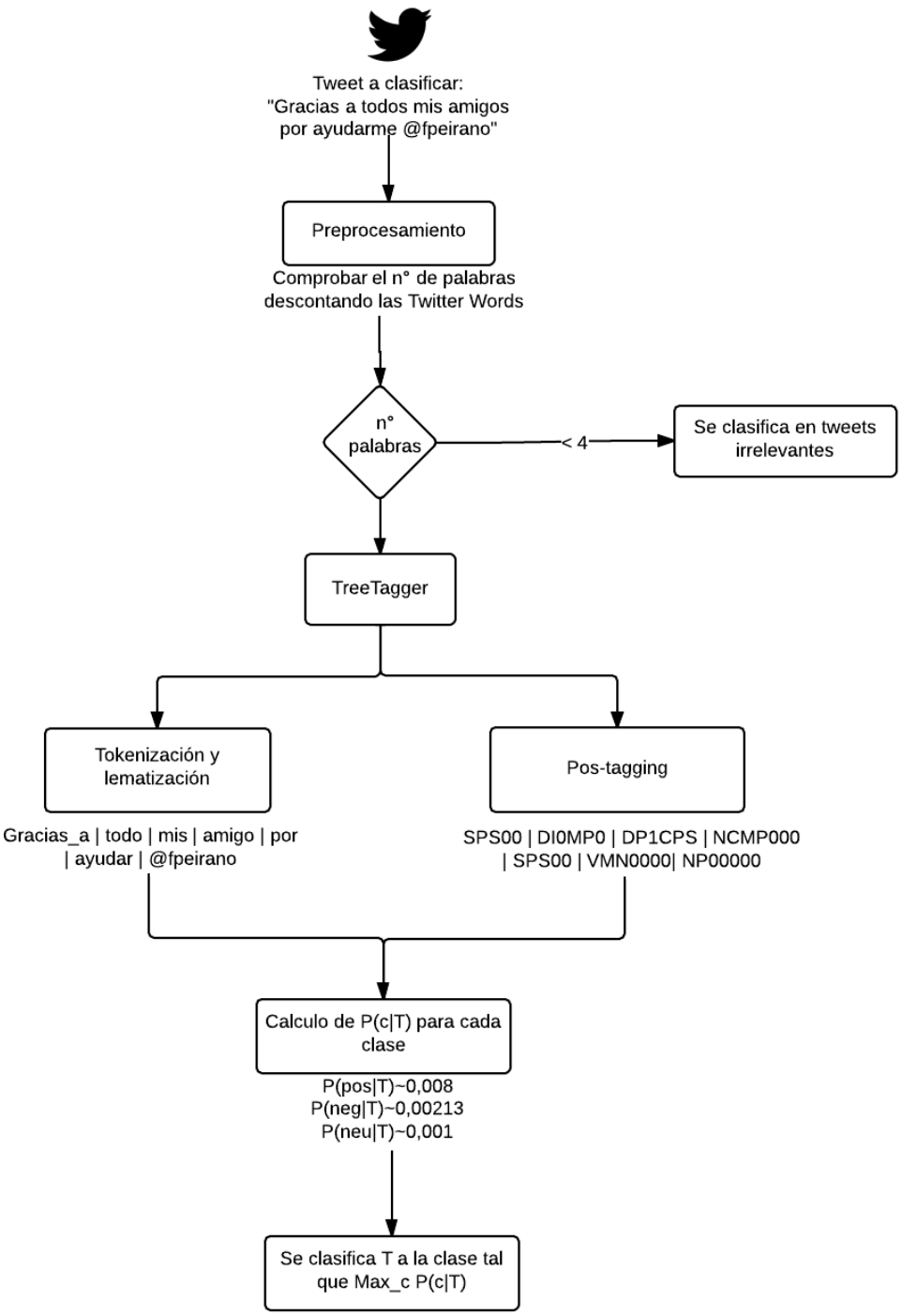



Figura 5.4: Detalle del proceso de clasificación
Fuente: Elaboración propia

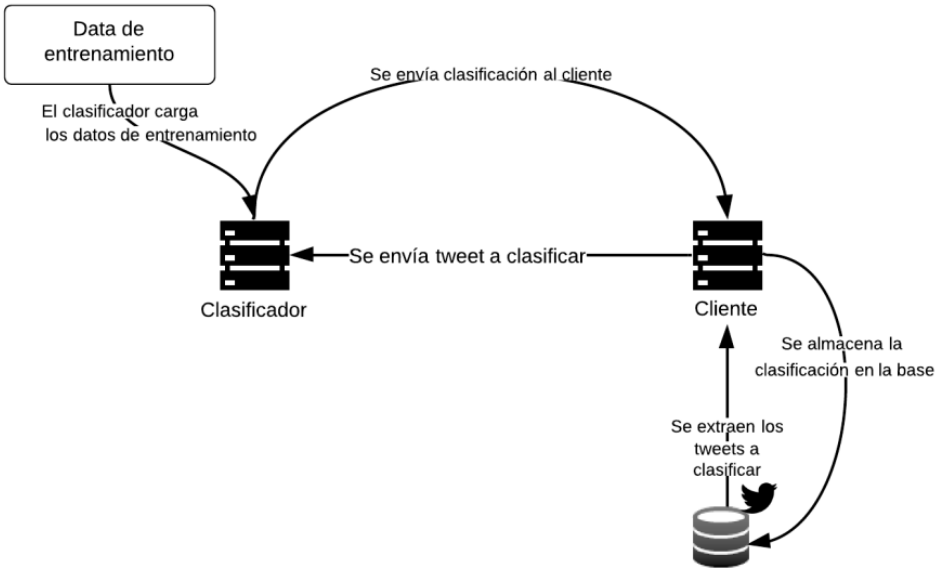



Figura 5.5: Diagrama configuración del sistema de clasificación
Fuente: Elaboración propia

## 5.5 Módulo de evaluación

Este módulo es el encargado de realizar las validaciones de los experimentos calculando el ratio de evaluación seleccionado para las dos técnicas de validación utilizadas. Para la evaluación del corpus y el algoritmo se utilizó el ratio conocido como *predictive accuracy* que consiste en la siguiente fórmula:

$$accuracy = \frac{N(clasificaciones\ correctas)}{N(casos\ de\ prueba)} \tag{5.1}$$

Además se utilizaron dos distintas metodologías de evaluación, las que se presentan a continuación.

**10-Fold Cross Validation**
El *10 Fold cross validation* es un método estadístico para evaluar algoritmos de aprendizaje dividiendo los datos de entrenamiento en 10 *folds* o particiones. Una de estas particiones es separada y el algoritmo es entrenado con todo el resto de los *folds*. Una vez que el proceso se ha completado, el modelo es evaluado usando la partición que había sido separada, obteniendo los ratios de *accuracy*, *recall* y precisión. Este procedimiento es repetido dejando de lado cada vez una de las particiones, y el resultado final para los ratios de evaluación es el promedio de los valores obtenidos por la evaluación de cada partición. Si bien el *10-Fold cross validation* es un caso particular del *X-cross fold validation*, donde su caso más extremo es el de *Leave One out*, donde cada partición tiene tamaño 1, la variante más utilizada es la de 10 particiones ya que ofrece un equilibrio entre el uso de distintas partes del corpus para evaluar el algoritmo, y no resulta tan demandante en recursos como el caso de *Leave One Out*.

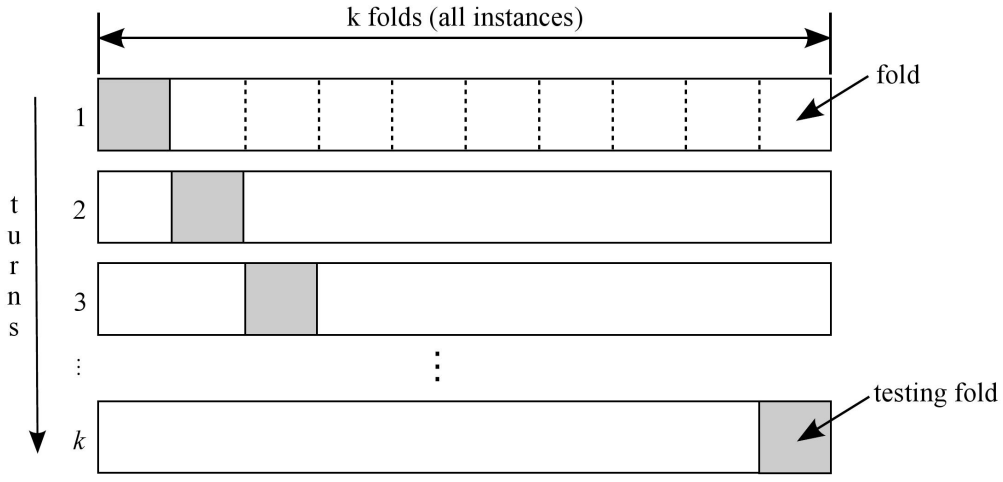



Figura 5.6: Modelo de evaluación K-fold cross validation
Fuente: Elaboración propia

**External evaluation dataset** El otro criterio usado para realizar una evaluación del sistema en términos de *accuracy*, fue obtener un nuevo set de tweets para obtener una medida de precisión del clasificador. Para obtener este nuevo conjunto de tweets, era importante que estos no estuviesen sesgados por algún término de búsqueda en particular, por lo que se usó

el criterio de búsqueda por ubicación geográfica para descargar los tweets mediante la *API*. Se etiquetaron manualmente los comentarios hasta completar 100 positivos, 100 negativos y 100 neutros.

Una gran diferencia que posee este enfoque de evaluación con el de *10-fold cross validation*, es que en este último se asume que todas las *labels* están bien colocadas para los comentarios que conforman la *corpora*, y en base a este supuesto se realiza la evaluación. En cambio para la evaluación utilizando un set externo, se pone a prueba además el desempeño de la metodología para la generación de la *corpora*. Por otro lado para evitar al máximo las subjetividades al generar este set de *tweets* de evaluación, éstos fueron etiquetados por la misma persona y al momento de cualquier duda sobre el tipo de polaridad que presentaban, el *tweet* era omitido, y otro clasificado en su lugar.

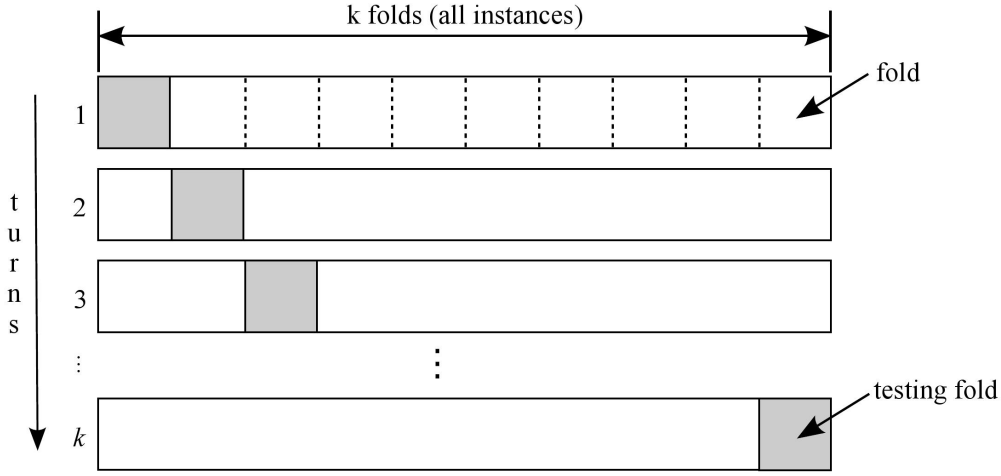



Figura 5.7: Modelo de evaluación K-fold cross validation
Fuente: Elaboración propia

# Capítulo 6

# Resultados

En esta sección se abordarán los experimentos confeccionados y realizados para probar las hipótesis de la investigación. En ellas se sugería que era factible realizar un clasificador de polaridad de sentimientos en base a la metodología semiautomática para la generación de la *corpora*, y que sería replicable para tweets en español. Además de probar que el entrenamiento de un algoritmo de clasificación con tweets chilenos, mejoraría la clasificación de tweets originados en el mismo lugar.

A continuación se expondrán los resultados obtenidos para las distintas metodologías de conformación de la *corpora* planteadas, evaluaciones de los algoritmos de clasificación para cada caso y por último una aplicación del sistema clasificador a un caso real de un evento que tuvo relevancia en Twitter para el medio Chileno.

## 6.1 Propiedades de la Corpora

Si bien en el capítulo 3 se abordó el cambio de estructura que sufrió la *corpora* al aplicar los filtros descritos, en esta sección se hará un análisis más acabado de la estructura presente en cada Corpus, las palabras y *POS-tags* que presentan una polaridad más marcada y la distribución de frecuencia de los *POS-tags* para cada una de las clases.

En el siguiente grupo de imágenes 6.1 se tienen la distribución de palabras para cada corpus, utilizando o no el filtro de tweets propuesto en el capítulo 3. Se puede observar que cada una de las curvas con filtro está desplazada a la izquierda respecto a la sin filtro. Esto tiene relación con que la media de palabras por tweet disminuye al filtrar todos los tweets definidos como tweets irrelevantes, que son aquellos que tienen un bajo número de palabras sin considerar las *Tweet Words*. Además para el caso del corpus de *tweets* positivos se observan 2 *peaks* anómalos en la distribución de palabras. Haciendo una inspección de este corpus, se apreció que correspondían a grupos de tweets repetidos, los cuales respondían a una campaña realizada por una famosa compañía de bebidas de fantasía. El problema de este tipo de *tweets*, como fue mencionado anteriormente, causan ruido al polarizar ciertas *features* contenidas en esos tweets y por lo tanto afectando el desempeño del clasificador.

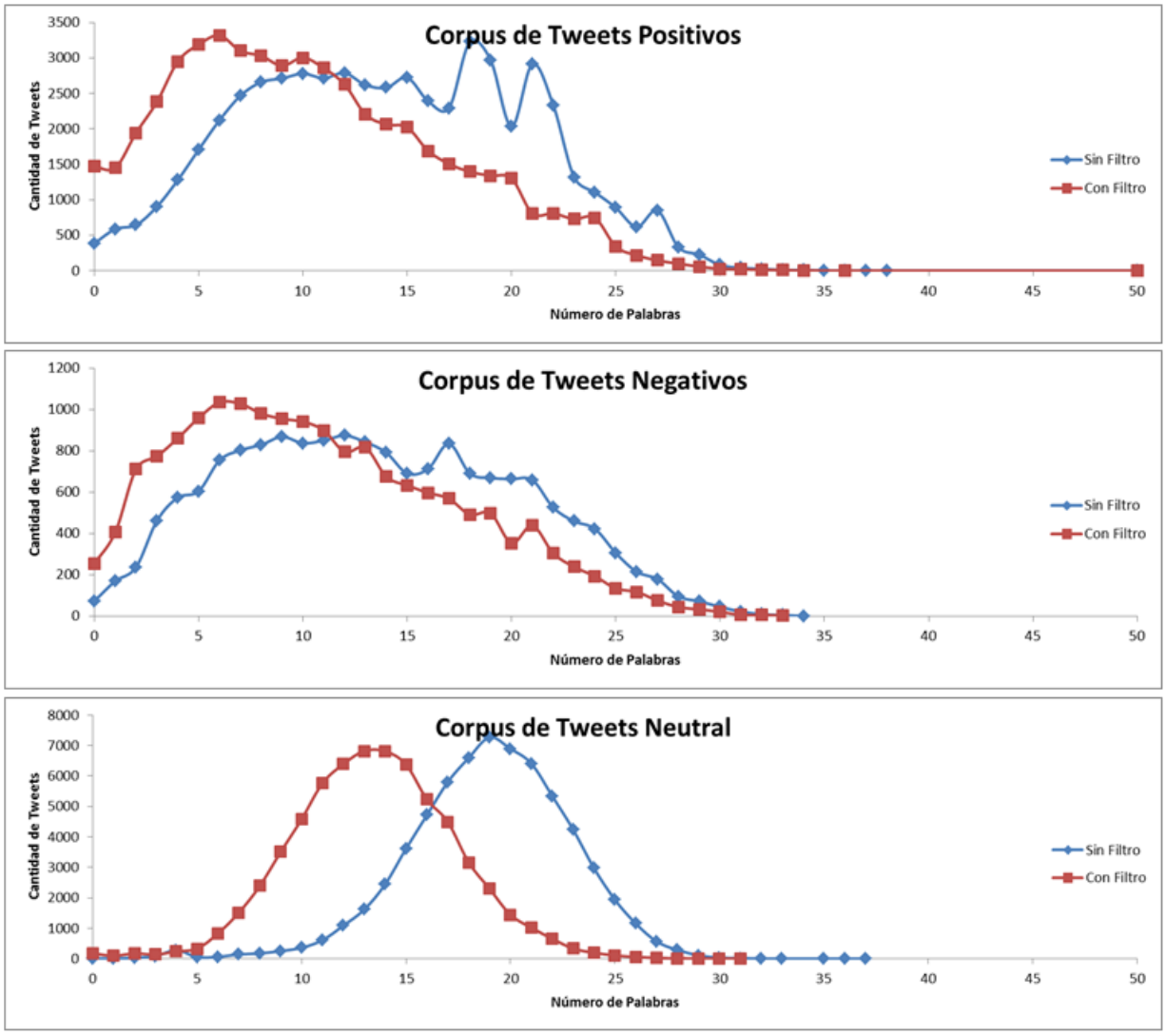



Figura 6.1: Frecuencia de número de palabras en tweets antes y después de aplicar los filtros
Fuente: Elaboración propia

La figura 6.1 muestra la distribución del número de palabras y cantidad de caracteres presentes en el corpus de cada clase. Se grafican dos curvas que representan las distribuciones antes y después de aplicados los filtros presentados en el capítulo 3.

Para mostrar la relevancia que tienen los tipos de tweets abordados en el capítulo 3, se elaboró la tabla 6.1, donde se muestra que porcentaje de cada corpus corresponde a los tweets filtrados

| | Positivo | Negativo | Neutro |
|---|---|---|---|
| Tweets irrelevantes | 11% | 8% | 2% |
| Tweets repetidos | 4% | 2% | 0% |
| Retweets | 3% | 1% | 0% |
| Total | 18% | 11% | 2% |

Tabla 6.1: Presenca de tweets problemáticos en la corpora sin filtrar

De la tabla 6.1 se concluye que la clase más afectada al no considerar un filtro de *tweets* para la conformación de la *corpora*, es la positiva. En el conjunto inicial más de un 10% de los *tweets* presentes en este corpus correspondían a *tweets irrelevantes*, afectando el proceso de entrenamiento, y por ende la categorización.

| Positivo | Negativo |
|---|---|
| Alegrar | No |
| Feliz | Malo |
| Ayudar | Feo |
| Agradable | Triste |
| Amigo | Llorar |
| Mejor | Fome |
| Lindo | Mentira |

Tabla 6.2: Palabras con mayor polaridad

En la tabla 6.2 se muestran las palabras que presentan mayor polaridad para las clases positivas y negativas. Dentro de las palabras con mayor polaridad negativa, se encuentra la palabra "fome" que es solo utilizada en el contexto chileno. Con este resultado se podría empezar a suponer que un algoritmo entrenado solo con *tweets* chilenos podría tener un mejor desempeño clasificando los mismos. Más adelante se realizará un experimento para probar si efectivamente un clasificador entrenado con tweets chilenos, se desempeña mejor en la clasificación de comentarios de usuarios nacionales, y cuánto mejor resulta.

En la ilustración **??** se puede observar la distribución de cada *POS-tag* presentes en la *Corpora*. Se aprecian diferencias sustanciales entre los Corpus de comentarios subjetivos (positivo-negativo) con los objetivos (neutrales) ya que en los primeros existen un mayor número de sustantivos propios, además que los verbos y artículos utilizados son en mayor número en tercera persona, mientras que para los subjetivos, en primera persona.

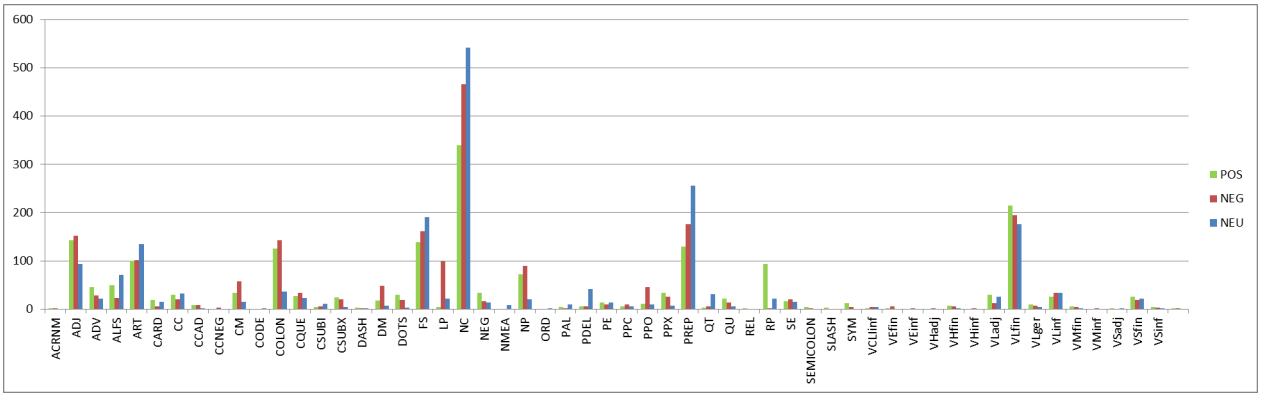



Figura 6.2: Distribución de postags por corpus
Fuente: Elaboración propia

En base a estos dos tipos de *features* (*POS-tags* y *Word-based unigrams*) extraídos se construyen los clasificadores explicados en el capítulo 5. A continuación se mostrarán los resultados obtenidos en cada experimento con las variantes introducidas.

## 6.2 Evaluación y resultados de los clasificadores

En este trabajo se consideraron dos algoritmos en base a Naïve Bayes para clasificar los *tweets* entre positivos, negativos y neutros. Además por lo observado en la *Corpora* se introdujo una clase adicional, que su clasificación se basa en una regla predefinida, con el objetivo de aumentar el desempeño del clasificador. Los algoritmos considerados fueron el clasificador de tres clases y el clasificador en dos niveles. El primero asume una independencia entre los *POS-tags* y *unigrams* de palabras para determinar la clase más probable de un *tweet* dado. Por otro lado el clasificador en dos niveles, pone a prueba qué tan bueno es el criterio de clasificación subjetivo-objetivo en base a los *POS-tags*, definiendo así en un primer nivel si un *tweet* es subjetivo u objetivo utilizando únicamente los *POS-tags* como *features* en un algoritmo de clasificación basado en Naïve Bayes. Aquellos *tweets* que son clasificados como objetivos son considerados neutrales, y los subjetivos son sometidos a un nuevo clasificador utilizando los *unigrams* de palabras como *features*, obteniendo así los *tweets* positivos y negativos.

Los criterios de evaluación considerados fueron los explicados anteriormente, que consisten en *10-fold cross validation* y la evaluación mediante un set de *tweets* etiquetado manualmente, cuya conformación se detalla en el capítulo 5 sección 5.

| 10-Fold Cross Validation | 2 Level Naïve Bayes | 3 Classes Naïve Bayes |
|---|---|---|
| Without Filter | 0.575 | 0.721 |
| With Filter | 0.614 | 0.787 |
| With Filter + 4th class | 0.622 | 0.806 |

Tabla 6.3: Accuracy evaluation with 10-fold cross validation

| Hand Tagged Set | 2 Level Naïve Bayes | 3 Classes Naïve Bayes |
|---|---|---|
| Without Filter | 0.432 | 0.688 |
| With Filter | 0.484 | 0.707 |
| With Filter + 4th class | 0.535 | 0.782 |

Tabla 6.4: Accuracy evaluation with the hand-tagged set

En las tablas 6.3 y 6.4 se muestran los resultados de ambos algoritmos bajo las distintas metodologías utilizadas para la evaluación. Se puede observar también que el desempeño de los algoritmos aumenta a con la inclusión del uso de los filtros propuestos en alrededor de un 4%, y que el algoritmo de clasificador de 3 clases resultó ser más efectivo para la clasificación que el de dos niveles. Además el uso de la regla para clasificar los *tweets irrelevantes* en una cuarta clase tuvo importantes mejoras como se puede apreciar en ambas tablas de resultados. Por último la evaluación de *10-fold cross validation* es mejor en todos los casos que la del conjunto de etiquetado manual, debido a que en la primera se asume que todas las etiquetas están correctamente colocadas en el corpus al usarse parte del mismo corpus en cada iteración del algoritmo para la evaluación.

## 6.3 Incidencia del lenguaje local en la clasificación de un tweet

Para generar un sistema clasificador polaridad de sentimientos en los tweets, que considerara los modismos propios del lenguaje local en la interacción con esta red social, se propuso utilizar como datos de entrenamiento únicamente tweets de usuarios Chilenos de Twitter. Una de las hipótesis de este proyecto plantea que el uso de un algoritmo de clasificación que considera una *corpora* generada por *tweets* locales, tendría un mejor desempeño para clasificar *tweets* del mismo lugar. Para el desarrollo de este experimento se generaron dos distintas *corporas* utilizando el criterio de filtrado de tweets propuestos, y la 4ta clase de tweets irrelevantes usando un umbral de 4 palabras. Una de estas *corpora* es la denominada chilena, generada por *tweets* que solo tienen el atributo *country_code* de valor 'CL'. La otra *corpora* es generada por *tweets* que tienen el atributo *lang* con valor 'ES' y no tienen *country_code* 'CL'. Además para mantener los criterios de generación de la *corpora* y que los resultados de este experimento sean comparables, se utilizaron los mismos *emoticons* para obtener de Twitter los comentarios que conformarán la *corpora*, y manteniendo el *corpus* generado para tweets neutrales, que consiste en comentarios emitidos por medios de prensa chilenos. Más detalles de cuáles fueron los criterios de recolección se detallan en el capítulo 5, en el apartado del módulo de recolección. Ya que el criterio para la conformación de la *corpora* de tweets chilenos es mucho más restrictivo considerando el hecho que no todos los tweets poseen *country_code* definido, entonces el conjunto de datos recolectados para la *corpora* de tweets en español genérico es mucho más grande (unas 40 veces). Gracias a esto se conformaron 5 distintas *corporas* de 80.000 tweets aleatoriamente, cumpliendo con las restricciones de filtros anteriormente mencionadas.

La técnica de validación utilizada para obtener los resultados y comprobar la hipótesis, fue la evaluación mediante el set de datos etiquetado manualmente. No se utilizó *K-cross fold validation* ya que es importante para este experimento usar el mismo set de evaluación para que los resultados sean comparables, y la técnica recién mencionada utiliza parte de cada *corpora* como datos de validación por lo que no cumple con este requisito.

| Corpora | Accuracy |
|---|---|
| CL | 0,793 |
| ES1 | 0,752 |
| ES2 | 0,747 |
| ES3 | 0,733 |
| ES4 | 0,756 |
| ES5 | 0,753 |

Tabla 6.5: Evaluación de clasificación de tweets chilenos usando distintas corporas de entrenamiento

En la tabla anterior se muestran los resultados de este experimento, obteniendo una importante diferencia en los ratios de validación al entrenar el algoritmo con tweets en español y otros estrictamente chilenos. Este resultado avala la hipótesis planteada sosteniendo que el uso de una *corpora* de tweets locales para el entrenamiento de un algoritmo clasificador de

sentimientos, produce mejoras considerables en el desempeño del sistema. En este caso se el aumento en el *accuracy* del sistema fue de aproximadamente un 4%.

## 6.4 Caso de estudio: *Sentiment Analysis* Primer debate presidencial ANATEL Octubre 2013

Como trabajo adicional y con el objetivo de utilizar el sistema clasificador en un caso de estudio relevante, se procesaron y clasificaron según su polaridad, todos los tweets obtenidos por la empresa Artool S.A respecto al debate presidencial que tuvo lugar el 29 de octubre del 2013.

A partir de los tweets clasificados, luego se obtuvieron como se componían las menciones a cada candidato según su polaridad.

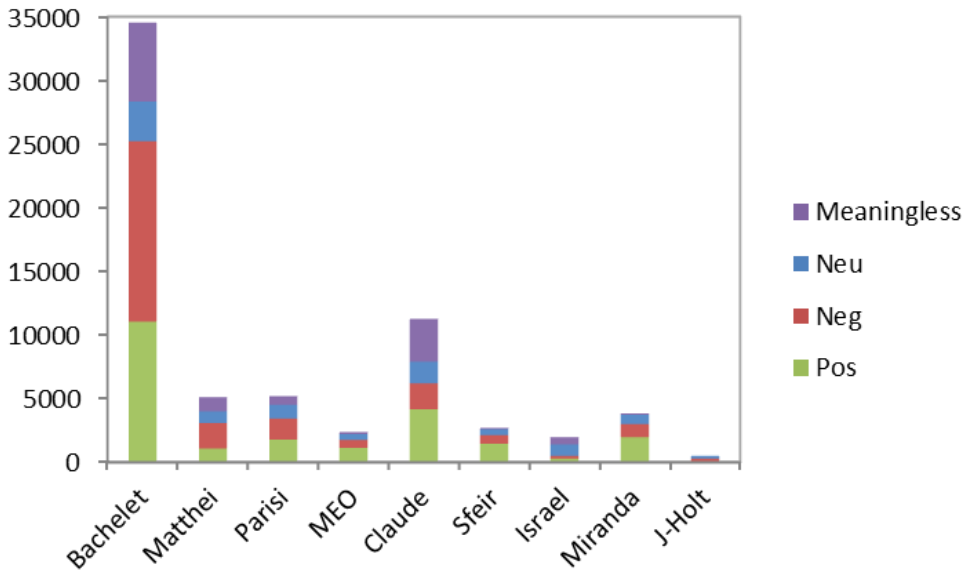



Figura 6.3: Distribución de polaridad de sentimientos para cada candidato
Fuente: Elaboración propia

En la figura 6.3 se presenta la cantidad de comentarios positivos, negativos, neutros e irrelevantes para cada candidato a partir de la base de comentarios del debate de ANATEL. Con la ayuda del sistema se puede determinar rápidamente cuáles son los candidatos con mayor número de menciones por cada tipo. Además se confeccionó un análisis porcentual por cada candidato para identificar la distribución interna de la polaridad de los comentarios de cada candidato. El resultado de este ejercicio se muestra en la figura 6.4 donde se observa claramente que los candidatos con mayor porcentaje de comentarios positivos respecto de su total, son Miranda, Sfeir y Enríquez-Ominami.

Con el fin de establecer un indicador que permita llevar un monitoreo del sentimiento de un evento, marca o persona en el tiempo se confeccionó el ratio que denominamos *Polarity score*. Como se define en la ecuación 6.1, este valor corresponde a una medida de cuántos de los comentarios subjetivos son positivos. El *Polarity score* toma valor 0 cuando todos

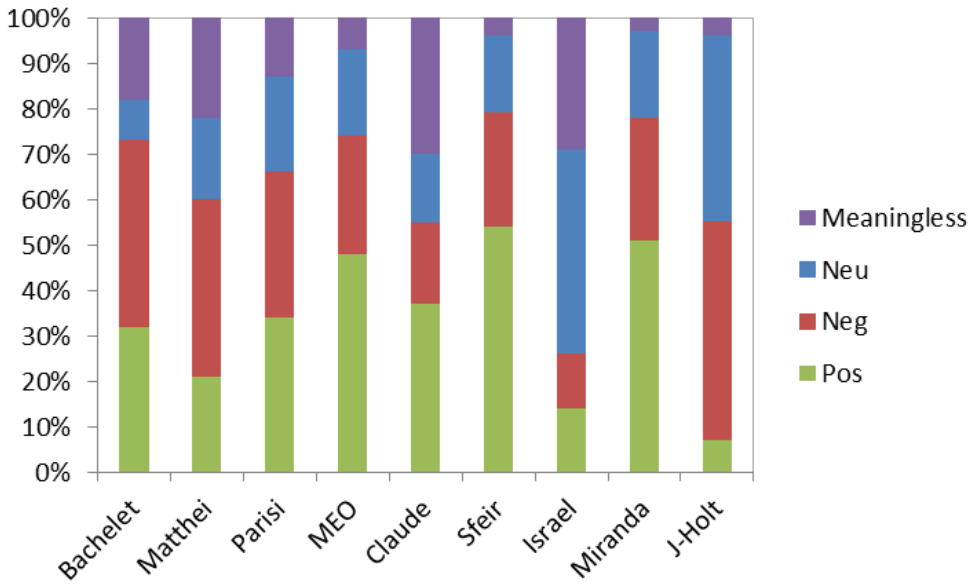



Figura 6.4: Distribución porcentual de polaridad de sentimientos de cada candidato
Fuente: Elaboración propia

los tweets subjetivos corresponden a comentarios negativos, y 1 cuando todos los tweets son positivos. En la tabla 6.6 se calculó el *Polarity Score* para cada candidato obteniendo el mejor valor para Sfeir con un 68% de comentarios positivos del total de comentarios subjetivos.

$$polarity\ score = \frac{(positive)}{(positive + negative)} \tag{6.1}$$

| **Candidate** | **Sentiment Ratio** |
|---|---|
| Bachelet | 0.44 |
| Matthei | 0.35 |
| Parisi | 0.52 |
| Enriquez-Ominami | 0.65 |
| Claude | 0.67 |
| Sfeir | 0.68 |
| J-Holt | 0.54 |
| Miranda | 0.65 |
| Israel | 0.13 |

Tabla 6.6: Polarity score para cada candidato sobre el primer debate presidencial organizado por ANATEL

# Capítulo 7

# Conclusiones

Este trabajo propone una extensión a los trabajos de Go et al 2009 y Pak et al 2010 basados en la confección de una *corpora* en base a metodologías de etiquetado semiautomáticas para la detección de polaridad de sentimientos en Twitter. Las extensiones propuestas incluyen la incorporación de filtros basados en reglas para la consolidación de la *corpora*, definiendo 3 tipos de tweets a excluir debido a que no aportan a la variabilidad de cada corpus, o simplemente no deben ser clasificados en ninguna de las tres clases. Estos son: *retweets*, tweets repetidos y tweets irrelevantes. Este resultado se obtuvo a partir de un experimento que tenía por objetivo determinar el desempeño de la metodología de generación de *corpora* aceptada para la clasificación de tweets en español. Como consecuencia de este experimento se determinó que estos tipos de comentarios no deben formar parte de ningún corpus debido a un experimento realizado mediante muestreo aleatorio simple, en donde un 11% de los comentarios mal etiquetados correspondían a tweets irrelevantes. Por otro lado los tweets repetidos y *retweets*, fueron también descartados ya que las palabras presentes en ellos y las estructuras de las oraciones son muy similares o iguales, no aportando a la variabilidad del corpus y acentuando las características polares de las *features* presentes en esos comentarios. Este hecho también fue corroborado al realizar experimentos con 2 algoritmos distintos de clasificación, considerando y prescindiendo de los filtros de tweets y la cuarta clase de tweets irrelevantes. Para ambos algoritmos se obtuvo el mejor desempeño cuando se incluían ambas *técnicas* propuestas en este trabajo.

El caso más relevante de los tres tipos de tweets filtrados corresponde a los tweets irrelevantes. Estos *tweets*, al tener una naturaleza completamente distinta a los demás tweets filtrados, y considerando que Twitter contiene un sinnúmero de comentarios de este tipo, es que se definió una clase específica para ellos. Esta consideración implicó un aumento de aproximadamente un 7% en el *accuracy*, lo que demuestra lo relevante que resulta su consideración al momento de elaborar un clasificador de polaridad para Twitter.

Por otro lado la aplicación de técnicas de Procesamiento de Lenguaje Natural para la detección de polaridad de sentimientos demostró ser muy efectiva en conjunto con otras técnicas de minería de textos, obteniendo un *accuracy* sobre el 80% para el caso de evaluación mediante *10-Fold Cross Validation*. Sin embargo el uso únicamente de *POS-tags* para la

clasificación de textos entre subjetivos u objetivos no arrojó resultados satisfactorios, por lo que se sugiere la implementación del mismo algoritmo en dos niveles esta vez utilizando ambos tipos de *features* en las 2 etapas.

En relación al algoritmo de Clasificación de Naïve Bayes se comprobó su efectividad incluso con ambos supuestos de independencia inter *features* e *intra features*. Esto es, a pesar que se asume que los *POS-tags* y palabras son independientes entre ellos, y que las palabras dentro de un comentario son independientes, al igual *POS-tags* dentro de un comentario, los resultados que se tienen para la clasificación de tweets son satisfactorios pudiendo reducir enormemente los tiempos de clasificación en comparación con el etiquetado manual, que también está sujeto a subjetividades humanas.

Por otro lado la arquitectura propuesta para el sistema permite cumplir con los objetivos de clasificación en tiempo real, pudiendo procesar 3 tweets por segundo para la implementación mono-cliente, en donde solo se acepta una petición de clasificación a la vez, la que podría ser aumentada mediante una aplicación multicliente. Esta implementación permitió clasificar los más de 70 mil tweets del primer debate de ANATEL en menos de 7.2 horas, lo cual pudo haber tomado 108 horas consecutivas considerando que un codificador profesional demora 5 segundos en clasificar cada comentario y está sujeto a condiciones de fatiga y subjetividades en el criterio de clasificación.

Como conclusión principal de este trabajo se tiene que la metodología semiautomática de generación de *corpora* basada en *tweets* con *emoticons* y *tweets* de medios de prensa es efectiva para el desarrollo de un sistema clasificador de tweets según su polaridad, y que los nuevos enfoques propuestos en base a filtros y el uso de una nueva clase de tweets irrelevantes, sí presentan una mejora en el clasificador de polaridad. Además se probó la hipótesis que sugería que el considerar una *corpora* generada a partir de comentarios chilenos permitía una mejor clasificación de los tweets locales.

Además, a partir de los comentarios disponibles en la web, y gracias a la API proporcionada por Twitter, se puede obtener información relevante y accionable acerca de qué, cuánto, cómo y bajo qué polaridad se está conversando de un tema en Twitter. Esto puede traer beneficios tanto a los proveedores, como a los usuarios de los productos o servicios:

- A los proveedores al poder analizar íntegramente todos los comentarios que se refieren a sus marcas y poder determinar cuáles son los aspectos que presentan polaridades negativas, pudiendo solucionar aquellos que son gravitantes.
- A los usuarios ya que al tomar cada vez más relevancia esta red social como medio de comunicación formal, las empresas si se preocupan de lo que se comenta en ella y ofrecen soluciones a los problemas presentados.

## 7.1 Trabajo Futuro

A continuación se incorporan los distintos trabajos que podrian ser llevados a cabo a partir de esta memoria con tal de mejorar el desempeño global del sistema:

1. La incorporación de técnicas de selección de *features* tales como el criterio de frecuencia, Chi-cuadrado o *Mutual Information* para aumentar el desempeño del sistema.

2. Considerar el uso de otros tipos de algoritmos supervisados tales como SVM o Redes Neuronales como clasificadores para contrastarlos con el clasificador de Naïve Bayes.

3. Utilizar otros tipos de *features*, como los semánticos y de puntuación, ya que por ejemplo muchas veces se relacionan signos de exclamación con textos subjetivos.

4. La respectiva modificación al módulo de *postagging* que permita que la aplicación sea multicliente para así agilizar aún más el proceso de entrenamiento, clasificación y aumentar la capacidad de procesamiento de tweets del sistema.

5. Variar la cantidad de tweets de entrenamiento en cada corpus, con tal de determinar cuál es el tamaño óptimo que debe poseer éste para maximizar el desempeño del clasificador.

6. Por otro lado se propone incorporar los los enfoques presentados en el trabajo realizado por Pak et al 2010, usando los mismos datos de entrenamiento y evaluación con tal de determinar y hacer más claro en cuánto se mejoró el clasificador.

7. Finalmente a raíz de lo estudiado, se observó que en Twitter se comenta mucho de forma irónica, y aunque este es un problema totalmente distinto y que no es capaz de ser detectado por el sistema presentado en este trabajo, sería de una enorme utilidad y un aspecto diferenciador el hecho que se considerase la ironía en la clasificación de tweets.

# Apéndice

## A Paper *Twitter Sentiment Polarity Analysis: A novel approach for Improving the automated labeling in a text corpora*

Véase la página siguiante

# Twitter Sentiment Polarity Analysis: A novel approach for Improving the automated labeling in a text corpora

Pablo A. Tapia
*Department of Industrial Engineering*
*University of Chile*
*Santiago, Chile*
*patapia@ing.uchile.cl*

Juan D. Velásquez
*Department of Industrial Engineering*
*University of Chile*
*Santiago, Chile*
*jvelasqu@dii.uchile.cl*

***Abstract*—The high penetration of Twitter in Chile has favored the use of this social network for companies and brands to get additional information on user opinions and feedback about their products and services. In recent years there have been many studies to determine the polarity of a comment on Twitter mainly considering three classes: positive, negative and neutral. One big difference inherent in the problem of Sentiment Analysis on Twitter as opposed to the Web is the ease with which you can obtain data to perform a supervised training algorithm with its API. To take advantage of this characteristic it is necessary to find a semi-automatic method for obtaining tweets to generate the corpora and avoid the traditional method of manual labeling which is very demanding in time and money. This paper goes deeper into the work of using a semi-automated generated corpora for Twitter sentiment polarity classification, introducing a novel approach of tweet selection for corpora consolidation and the addition of a fourth class of tweets that doesn't correspond to any of the above. This new class includes tweets that are irrelevant for classification and do not contain much information, a type of posts that Twitter is full of. Experimental evaluations show that the usage of the fourth class of the denominated meaningless tweets together with the tweet filter criteria for corpus generation improves the system accuracy.**

***Keywords*-Twitter sentiment analysis; opinion mining; polarity detection;**

## I. Introduction

Companies have always been worried about the perception that customers have of their brands, and have conducted marketing studies such as surveys or focus groups to obtain information that supports decision making. Some time ago, when a person wanted to buy a product, he was limited to asking friends and family opinions. Recent studies have shown that with the explosive growth of social media on the Web (reviews, forums, blogs, micro-blogs, etc), individuals are increasingly using opinions present in this media before making a purchase, and are greatly influenced by the reviews and comments that can be found on the Web [1]. It is therefore of great interest for companies to know what users of their products say, but given the vast quantity of available reviews and the specific characteristics of the Web, specialized tools are needed to extract the opinions of Websites. Twitter is the most popular microblogging platform, and offers a simple way for users to express their feelings and thoughts by means of short messages with a strict maximum of 140 characters that can be freely used. One of the most important features that makes Twitter an important case study is that by default the comments made by users on this platform are public, being available to anyone. On the other hand Twitter has a very robust API that allows the easy extraction of fully-indexed comments from its content, including the relation between user posts [2]. Because of Twitter structure, a message written in Twitter can also propagate rapidly, making it visible to specific communities in almost real time [3]. Because of this, Twitter data is a powerful source of information for assessing and predicting large-scale facts. For example O'Connor et al. 2010 in [4] capture large-scale trends for analyzing political opinion in tweets as a supplement for traditional polling.

In recent years the penetration of this social network has been constantly increasing at an enormous rate in Chile and the world [5], [6]. For this reason, different industries such as banks, retail and others are using this platform for customer service and to evaluate how and how much users are talking about them. Nowadays, because of the specific way that users interact in this platform, common sentiment analysis tools are not very effective at performing this task and the labeling is done by a human tagging process. This paper goes deeper into the emoticon-based approach for generating both positive and negative corpora, an automated method for collecting a neutral class corpus and the addition of a fourth class of tweets to improve the classifier's accuracy, which corresponds to tweet posts that cannot be classified into any of the previous classes because they are meaningless or lack content, a type of tweets that Twitter is full of.

The paper is organized as follows. Section 2 overviews the related work previously done for attempting the sentiment analysis task in Twitter. Section 3 presents a brief description of how the corpora were collected using previous approaches. In Section 4 the corpora are analyzed in order to evaluate and present some problems found following the accepted corpora-generation approach. Section 5 presents

some solutions to the problems found in the previous section and the new corpus structure for each class applying them. Section 6 describes the classification algorithms based on the Naïve Bayes classifier that are going to be used for the experiments. Section 7 explains the implementation and design of the system. Section 8 examines the results of the classifiers using the proposed improvements, Section 9 includes the analysis of a Chilean presidential debate where the system was used to obtain the sentiment polarity of each candidate, and Section 10 presents some conclusions and discusses future work.

## II. Related Work

The ease of collecting Twitter comments makes the problem of sentiment analysis on Twitter very different than doing the same task focused on the Web. Thanks to the Twitter API we are able to extract millions of tweets to train a supervised machine learning algorithm. The set of text posts used for this purpose is called the corpus. The collected corpora must be labeled in order to train an algorithm which can later receive a tweet as input, and is able to correctly assign a class from the ones in the training data. The traditional method of labeling the corpora is reading all training data and labeling it manually. This approach has two problems, the first related to the subjectivity present in assigning labels, this task usually being performed by various people, even on different days and at different fatigue levels, which affects the labeling criteria. The other problem is that this methodology cannot take advantage of the Twitter properties in order to get a large number of tweets for the generation of the corpora due to the large amount of time and money that is required to perform the task of manual labeling. For these reasons researchers have recently been developing studies for a methodology that allows automatic generation of a corpus, i.e. a corpus of tweets for each class with an automatic labeling approach.

Go et al. 2009 in [7] pioneered this problem using a study by Read 2005 [8] that refers to the extraction of opinions in movie reviews. One of Read's conclusions is that a polar emoticon defines the polarity of texts that contain it. The main problem of this approach is that in the same review many fragments can be presented with different types of polarity, so it is a problem to determine the overall polarity of a review. For Twitter this fact does not present a big problem as a comment on the social network has a maximum length of 140 characters and they most often consist of a single idea. This is the approach taken by Go et al. to establish that a tweet with a polar emoticon presents the polarity of the emoticon. However, their work had focused on the polarity of sentiment analysis into two classes (positive and negative), and only makes a minor analysis considering three classes, including for a neutral class tweets that don't present any emoticon in their content, which is a very weak assumption. In [9] Koppel et al. show why it is important to consider the neutral class, and among its findings are that training using only positive and negative examples does not allow an accurate neutral example classification, and that classification using neutral data contributes to a better distinction between positive and negative examples. This implies that considering a neutral class corpus for training and classification significantly increases the overall system accuracy.

Since then, many other studies have been done improving sentiment polarity classification using semi-automated corpora generation, but mainly focusing on improving the training algorithm and feature extraction rather than the collection and selection of the corpora.

To use a large number of tweets for training requires then a selection approach that allows automatic labeling of neutral-type comments. To resolve this issue Pak et al. proposed in [10] to extract opinions issued by official accounts of newspapers, making the assumption that these tweets are neutral. While this approach produced good results for classification in general, fundamental differences with the structure of the obtained subjective tweets were not considered using the explained subjective and objective corpora collection methods.

In this paper we work with these differences and problems of the automatic corpora collection approach and propose solutions to improve the system classification. Also as a consequence of the proposed solutions addressed we considered a fourth kind of tweet, which is based on a classification rule, and that includes all the tweets that are not relevant to be classified in any other class because they lack content or are meaningless, comments of which Twitter is full. Assigning this type of tweet to its own class improves the overall performance of the classifier by avoiding bad training examples, noise in the other corpus classes and misclassifications.

## III. Collecting the Corpora

Corpora collection is one of the most important tasks in the process of Sentiment Analysis. Considering the fact that training data should be sufficiently robust so it can generate a model for later classifying new tweets, there is an interest in collecting as many tweets as possible for the algorithm training. We also need this data labeled according to the different classes in which the tweets will later be classified. Because of these characteristics the manual labeling of tweets is not a viable option due to the large numbers of people and man hours that would be required to label all tweets used for training.

In this work a corpus of tweets was collected to build a dataset of three classes: Positive, Negative and Neutral. For the collection of positive and negative tweets the same methodology proposed by Go et al. 2009 [7] was followed, which involves collecting tweets with polar emoticons justified by the work of Read 2005 in [8] which concludes that

a polar emoticon has a high correlation with the polarity of text containing it.

Using the Twitter API all the comments were collected for 2 months that come with the country code attribute referencing CHILE and that contain the following emoticons:

- Positive Emoticons: ” :) ”, ” =) ”, ” ;) ”
- Negative Emoticons: ” :( ”, ” =( ”, ” ;(”

To collect the neutral tweet class, the same procedure proposed by Pak & Paroubek was followed, selecting tweet posts from popular newspapers and Twitter press accounts. Some of these media are: @latercera, @emol, @lasegunda, @biobio and @cooperativa among others, which are supposed to be objective and not express any opinion about the news they broadcast.
For each one of the classes 80,000 tweets were selected according to the described rules and procedures.

## IV. Analyzing the Emoticon-based Tagging Approach

In previous work, the emoticon-based labeling approach has been accepted for collecting the corpora and the efforts have focused on improving the accuracy of the classifier algorithm using text mining and natural language processing techniques. However in this work we wanted to test the proposed approaches for collecting the corpora and establish how accurate these semi-automated labeling methods are.

To make a first evaluation of the described semi-automated approach, a Simple Random Sampling was performed for each of the classes of the collected corpora. For this task 100 tweets were randomly selected from each class and they were checked one by one to determine if the labels assigned by the above methodologies were correct. The following table shows the results of this experiment, obtaining the best result for the neutral class, and the worst for the positive class. The overall accuracy for these three classes combined was 81%, which is quite low for an attempt to obtain a methodology that can classify tweets according to their polarity with an accuracy of at least 85%.

Table I
Simple Random Sampling to evaluate the semi-automated generated corpora

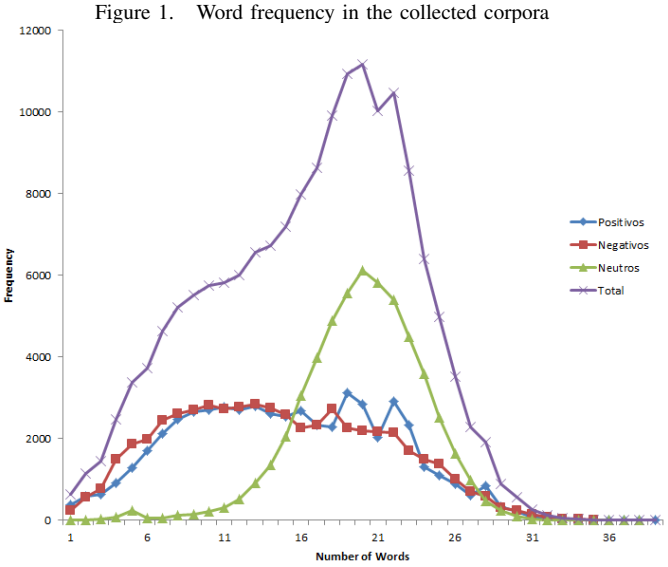
| Class | Positive | Negative | Neutral | Total |
|---|---|---|---|---|
| Correct-tagged cases | 75% | 82% | 86% | 81% |

Because of the results obtained by the simple random sampling experiment, each corpus was analyzed in depth in order to find any problems the proposed methodology might have. Inspecting the wrong-tagged cases, various problems were found that generate noise and, in consequence a bad tweet labeling, which are specified below:

- **Meaningless tweets:**
  Only filtering tweets by the emoticon present in their content (for the subjective classes), is a very simple rule for a tweet labeling approach. When analyzing the collected corpora, we found tweets like the following:
  1) #AvenidaBrasil :)
  2) #FelizLunes a @Brennedy2 :) #TuitUtil http://t.co/sNd2Fmtsxe

  For the first example, the tweet is made up of one hashtag and one emoticon. For the second example the tweet consists of a word, two hashtags, a mention of another user, a link and the emoticon. The problem lies in the fact that these tweets are not useful for training the algorithm because they do not add distinct information to each corpus. In addition, the number of features that can be extracted from them through processing hashtags, links and references to standard codes are small. Meaningless tweets are thus not useful for training the algorithm because they do not contribute with distinct features to each corpus.

Figure 1. Word frequency in the collected corpora




- **Neutral tweets:** Being neutral texts obtained from newspapers and press accounts, these have a considerable number of additional words compared to positive and negative comments (as seen in figure 1). This is mainly because these types of accounts have to summarize the news in a tweet and also add a link to it to a Web page, so they generally use most of the limited space available to communicate content. Also because of this, they have more features making a tweet more likely to be classified as a neutral type.

- **Retweets:** Another major problem identified in the corpus is retweets. A retweet sent by an influential person or that refers to very important news can be retweeted many times. The tweet that was shared more times in the history of Twitter was posted by Ellen Degeneres, a famous American TV host

for the Oscars with more than 3.3 million retweets and 1.9 million favorites [11]. The second one was Barack Obama's tweet when he was re-elected as President of the United States with over 750 thousand retweets and 290 thousand favorites [12]. The main problem with this type of tweets is that they have exactly the same content, not contributing to the variability of the corpus, and further deepening the polar characteristic possessed by the words within that tweet. For example, the tweet "I'm bored of studying, now I'll play :)", using the semi-automatic labeling methodology assumes that each word in the tweet has a positive polarity , then because retweets are copies of the original tweet, it accentuates the positive polarity that the training algorithm extracts from the words of this particular tweet.

- **Repeated Tweets:** Another important problem found in the tweets of the collected corpora was related to marketing campaigns in Twitter. Given the high penetration of the platform and quick viralization of campaigns, companies and brands have begun to use Twitter as a means to conduct raffles and sweepstakes. In order to gain or increase their chances of winning users are able to emit dozens of equal comments by adding a number to the end of the comment in order to avoid the duplicate restriction of Twitter comments. Below there is an example of a tweet that was posted over 90 times by the same user in order to win a ticket for a concert.

*@LollapaloozaCL #ViveLollapalooza #Cocacolafmcl @CocaCola_CL Quiero ir!! asi que a poner tweets se ha dicho!! :D 1*

⋮

*@LollapaloozaCL #ViveLollapalooza #Cocacolafmcl @CocaCola_CL Quiero ir!! asi que a poner tweets se ha dicho!! :D 92*

The problem with this kind of tweet to the conformation of the corpus following the semi-automatic method proposed, is essentially the same as the effect that is caused by retweets, where the words contained in these comments become very polarized according to the emoticon labelling criteria, and also they do not contribute to the variability of words in the Corpus because they have almost the same. The anomalous peaks that can be seen for the number of words in the corpus of the positive class in Figure 1 is actually a consequence of a local campaign to win tickets to a famous singer's concert.

## V. Proposed Solutions for the identified problems

- **Meaningless tweets:** To handle this kind of tweet, the approach followed is to make a filter of hashtags, RT mentions (initials that refer to retweets) and URLs. After removing these kinds of strings, and according to the number of words, the tweet is selected to be part of the corpus only if it has more than a certain number of words. This number is then defined later in the experiments. We call these kind of Twitter strings *Twitter words*

- **Retweets:** To avoid this type of tweet, they are filtered using the rt_twitter_user_id attribute and excluded if it is not null because it will be then a retweet of another user's tweet. This attribute comes within the ones that Twitter API includes.

- **Repeated Tweets:** Since these tweets cause the same effect as retweets, they will also be removed. According to the general structure that they have, we queried each corpus in order to find tweets that end with correlative numbers and from users that have a high frequency of posts.

In the next table it can be observed how varied the number of words and characters after applying these filters. It reduces, on average, 3.5 words per tweet for the positive class, almost three words for the negative class and five words for the neutral class. So tweets in the corpus no longer have fewer than a certain threshold of words (excluding the strings mentioned) which is defined later after experimenting with a different number and selecting the one that presents the best results for the classification.

Table II
Corpora structure applying the tweet filters

| | Type | Words | | Characters | |
|---|---|---|---|---|---|
| Class | | Mean | STD | Mean | Std |
| Positive | Without Filter | 14,01 | 6,58 | 88,35 | 35,75 |
| | With Filter | 10,55 | 6,23 | 59,73 | 31,82 |
| Negative | Without Filter | 13,54 | 6,75 | 84,33 | 35,58 |
| | With Filter | 10,87 | 6,5 | 61,48 | 33,16 |
| Neutral | Without Filter | 18,96 | 3,89 | 110,67 | 19,65 |
| | With Filter | 13,48 | 3,87 | 79,24 | 20,19 |

Thanks to only the use of these three filter criteria for the selection of tweets in corpora generation, the overall accuracy for a new simple random sample experiment increased by almost 5%, which is fairly good considering the simplicity of the rules applied.

One question that was caused after this experiment was what we should do with the meaningless tweets. Despite not being considered for the corpora generation, we know that Twitter is full of them and in any attempt to classify

Table III
NEW SIMPLE RANDOM SAMPLING EXPERIMENT WITH FILTERED CORPORA

| Class | Positive | Negative | Neutral | Total |
|---|---|---|---|---|
| Correct-tagged cases | 86% | 84% | 84% | 85% |

them they will be incorrectly assigned to a class making noise and decreasing accuracy. Unlike other types of filtered tweets, i.e. repeated tweets and retweets that can be well classified because original examples are considered in corpora generation, for meaningless tweets it isn't like that, and there is no tweet like this in the entire corpora. This is why we propose the use of a particular class that only meaningless tweets belong to. In this study we will use the same filtering approach to classify these tweets into the new category, but further work can be done using the filtered examples of the original corpus to train a classifier in order to learn more characteristics about meaningless tweets. The approach considered had good results improving the overall accuracy of the classifier and the system. In this work we will use unigrams to buid the classifier in order to find out if the filters criteria improves the system accuracy. The results of these experiments will be presented in the following sections.

## VI. FEATURE EXTRACTION AND CLASSIFICATION ALGORITHMS

**Feature Extraction:**
The collected corpora are used to extract features that will be used to train the sentiment classifier. Pang et al. and Pak et al. among others have experimented with unigrams and bigrams, some obtaining better results using unigrams for movie review and tweets polarity extraction while others found that bigrams outperform unigrams. The process for obtaining the unigrams from the Twitter posts is as follows:

1) Filtering: All type of Twitter words (i.e. @mentions, #Hashtags, URLS, RTs) were removed from the tweets replacing them by a very distinct string: [@] for mentions, [#] for hashtags, [U] for Urls and RT. Emoticons were also extracted from tweets so the model does not depend on them.
2) Tokenization: The tokenization criteria consist of splitting strings by spaces and punctuation marks.
3) Stopwords removal: we use the stopwords list provided in [13] and remove the tokens matching any of those terms.
4) The same procedure was followed to extract the features from tweet POS tags using Treetagger

**Classification Algorithhms:**
In this work we used Naïve Bayes Classifier because it was proven to have good performance in sentiment classification problems for Twitter. The Naïve Bayes Classifier is based on the Bayes Theorem and for this work assigns to a tweet T, represented by a vector of features F, the class from the set of classes C which maximizes the following expression:

$$P(c|F) = \frac{P(c)(F|c)}{P(F)} \tag{1}$$

The prior term *P(c)* is the same for every class in our experiments because we use an equal number of training tweets per corpus. Conditional independence between the features was also assumed. Even when it is known that a natural language text's words are dependent on each other, it is proven that the Naïve Bayes classifier performs very well for text classification [] under this assumption. We can simplify the equations to:

$$P(c|F) = \frac{P(F|c)}{P(F)} \tag{2}$$

$$P(F) = \sum_{c \in C} P(F|c)P(c) \tag{3}$$

And choose then the class that maximizes:

$$arg \max_{c \in C} P(c|F) \sim P(F|c)$$

These criteria were used to build two classifiers using both types of features explained above, unigrams and parts-of-speech (POS) tags.

We developed two classification algorithms: one using a combined approach of the two features selected, proposed by Pak et al. in [10] and a two-level algorithm that first classifies a tweet into subjective or objective classes using only POS tags as features. Subjective tweets are then classified into positive and negative considering unigrams of words as features for this problem.

1) **Naïve Bayes Classifier for three classes:**
To use both types of feature and generate a three-class algorithm the posterior probability can be written as a combination of the posteriors for each model if we assume conditional independency between POS tags and unigrams of words:

$$P(c|T) \sim P(U|c) \cdot P(P|c) \tag{4}$$

Where *U* is the vector of unigrams of words representing the tweet T and P is the set of POS tags present in it. As was said before we assume every feature of each type conditionally independent from each other having:

$$P(U|c) = \prod_{u \in U} P(u|c) \tag{5}$$

$$P(P|c) = \prod_{p \in P} P(p|c) \tag{6}$$

$$P(c|T) \sim \prod_{u \in U} P(u|c) \cdot \prod_{p \in P} p(u|c) \qquad (7)$$

2) **Two-level Naïve Bayes Classifier:**
For this attempt we used both types of features for different classification problems. From the work in [10], [14] we know that objective texts are more likely to talk in third person among other characteristics, so objective vs subjective texts could be distinguished by the POS tags present in them. Because of this we will consider for objective and subjective classification, the POS tags present in tweets as features. Therefore, subjective tweets are classified into positive or negative classes considering unigrams as features.

## VII. System Design

For the collecting module we implemented an application that connects to Twitter API tracking all tweets that contain the defined positive or negative emoticons, and all tweets that a specified set of Twitter press accounts posts. This initial training set was stored in a database and then parsed in order to easily extract the text of the tweet, date, country_code, language, and rt_twitter_user_id, which are fields that are used later. Because we are interested in Spanish tweets, we selected for the training set all tweets that were in language 'es' which corresponds to Spanish.

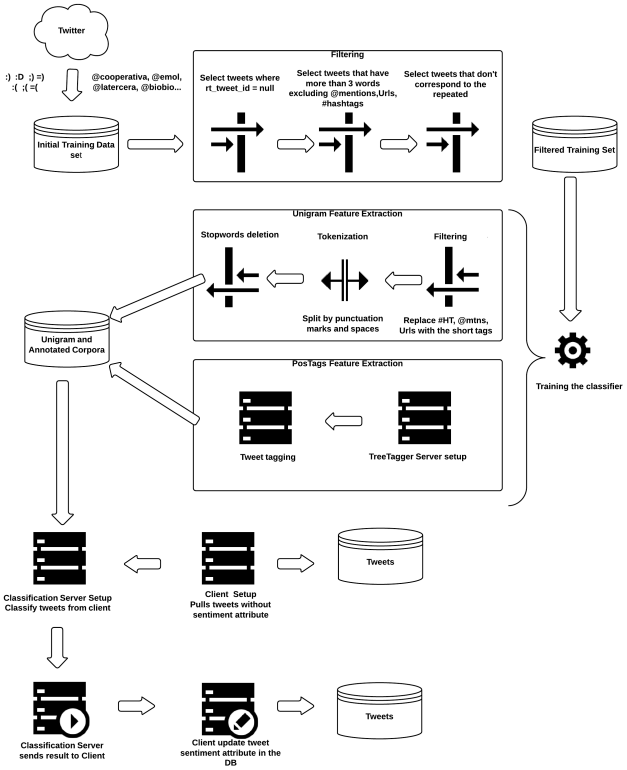
One of the goals of the classification module is that it can classify in real time incoming tweets that are being tracked by a monitoring system. This monitoring system is in charge of downloading all tweets and parsing them in a database that we will call clientdb.

For the implementation of the training and classification system we design a client-server application. The first time the server runs, it is set to training mode with the option of running or not the tweet filters explained in the corpora section. Also this first time, the server outputs a file with all the extracted features so if the server goes down, the system can be rapidly restored by loading the trained data, avoiding the long training time because of the POS-tagging phase. The client program is in charge of getting the tweets recently added to the database clientdb and sending them to the server in order to classify them and update their sentiment attribute with the output given by the server.

The process steps for the training and classification tasks are mainly the following:

1) Extract the same number of tweets for each of the three classes already stored in the DB. If tweet filtering is set to true, then check for tweets that do not have a rt_twitter_user_id, which means that the tweet is not a retweet, filter meaningless tweets which are the tweets with less than four words excluding @mentions, #hashtags, RT tag and URLs, and also filter repeated tweets from the selection.
2) Output the training data and after this, continue running the server in classification mode, defining which of the classification algorithms is going to be used between the three-class Naïve Bayes classifier or the two-level Naïve Bayes classifier explained above.
3) Launch the client program.
4) The client program checks the database to find if there are tweets that have not been classified, sending them to the server so they are classified according to the algorithm that was previously set up, and sleeping for 10 seconds if there are no more tweets to classify.
5) The server returns to the client the classification of the tweet, updating it in the database by setting the sentiment attribute to the server classification response.
6) Iterate four and five while the classification task is needed.

Figure 2. General design of the complete system.




## VIII. Evaluation and Results

For the evaluation we use the accuracy ratio with two distinct validation methodologies:

- We used 10-Fold Cross Validation which is a technique that is useful to evaluate a classification algorithm for a given corpora, splitting and evaluating the training set several times.

- We also manually marked 100 tweets per class in Spanish and classify them with the algorithms in order to get the same accuracy ratio, but with this technique we evaluate the accuracy of the whole system, from the data collection and feature extraction to the classification algorithms.

The accuracy ratio corresponds to:

$$accuracy = \frac{N(Correct\ classifications)}{N(All\ classifications)} \quad (8)$$

Table IV
ACCURACY EVALUATION WITH 10-FOLD CROSS VALIDATION

| 10-Fold Cross Validation | 2 Level Naïve Bayes | 3 Classes Naïve Bayes |
|---|---|---|
| Without Filter | 0.575 | 0.721 |
| With Filter | 0.614 | 0.787 |
| With Filter + 4th class | 0.622 | 0.806 |

Table V
ACCURACY EVALUATION WITH THE HAND-TAGGED SET

| Hand Tagged Set | 2 Level Naïve Bayes | 3 Classes Naïve Bayes |
|---|---|---|
| Without Filter | 0.432 | 0.688 |
| With Filter | 0.484 | 0.707 |
| With Filter + 4th class | 0.535 | 0.782 |

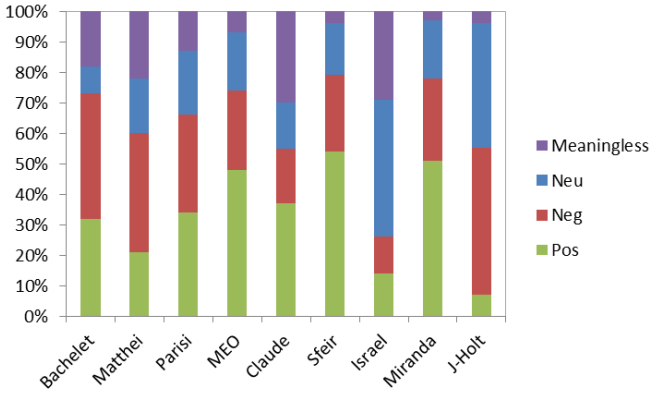
In table IV the results of the experiments with the 10-fold cross validation evaluation are shown. Independent of which classification algorithms were used, the accuracy of the system increased if we used the proposed filters. Also with the addition of the four-class rule for classifying meaningless tweets, the overall accuracy of the system showed further improvements. Regarding the algorithms, the three-class Naïve Bayes Classifier had better results than the two-level algorithm for all the experiments. Also in table V the results of the evaluation made by the hand-tagged set of examples are shown. For this set we also obtained best results for the combined Naive Bayes classifier algorithm, with the filtered corpora and using the fourth class of meaningless tweets classification rule.

## IX. CASE STUDY

We satisfactorily used the implemented system to analyze 74,066 tweets sent by Twitter users about a Chilean presidential debate held on October 29th 2013. This database was generated by querying Twitter API for all posts containing the hashtag #DebateAnatel between 8PM of October 29th and 2AM October 30th, local time. From this database we classified all tweets using the three-class Naïve Bayes classification algorithm with filters and the meaningless tweet rule for the fourth class. After the classification we grouped all tweets referring to each candidate and obtained the number of positive, negative, neutral and meaningless comments. The result of this exercise is shown in figure 3.

We also defined a polarity score ratio to get an overall value of the sentiment polarity of tweets for each candidate. In table VI is shown the calculated polarity score for each candidate.

Figure 3. Sentiment Polarity of candidate's tweets




$$polarityscore = \frac{(positive)}{(positive + negative)} \quad (9)$$

Table VI
SENTIMENT RATIO FOR EACH CANDIDATE ABOUT THE FIRST PRESIDENTIAL DEBATE HELD BY ANATEL

| **Candidate** | **Sentiment Ratio** |
|---|---|
| Bachelet | 0.44 |
| Matthei | 0.35 |
| Parisi | 0.52 |
| Enriquez-Ominami | 0.65 |
| Claude | 0.67 |
| Sfeir | 0.68 |
| J-Holt | 0.54 |
| Miranda | 0.65 |
| Israel | 0.13 |

The objective of this exercise was to derive some analysis that can be generated from the developed Sentiment Classification System. The sentiment ratio explained above does not consider the number of tweets (popularity) of the candidate and it is only a measure of how many of the subjective tweets are positive. Some other ratios that can weight this value with the total number of posts in order to consider popularity may be introduced in future work.

## X. CONCLUSION

In this study, we presented an automated approach for generating corpora for Twitter sentiment polarity detection, which is an improvement of the corpora composition by applying different filters in recognition of the problem that this methodology entails. We also found that many posts in the initial collected training set for positive and negative classes were meaningless or lacking in content, effectively

causing noise within these corpora, preventing the best training results and also decreasing the system classification accuracy when trying to classify this type of tweets into a class. To solve this issue we proposed to filter those tweets that had fewer than four words excluding @mentions, #hashtags, RT tags and URLS from the collected corpora, and in the classification using this same rule to assign them to the meaningless or unclassified class. We also used our proposals to successfully implement the system and create an online application that can classify a continuous flow of tweets in real time. The results obtained validate the proposed improvements of an automated corpora collection in order to increase the number of training examples for a Twitter Sentiment Analysis algorithm. For future work we propose to train an algorithm that can generate a model to classify these meaningless tweets, and also improve the accuracy of the system by implementing bigrams and other feature-extraction algorithms.

## ACKNOWLEDGMENTS

This work was supported partially by the Corfo Innova project 13/DL2-23170, entitled *OpinionZoomt* and the Millennium Institute on Complex Engineering Systems (ICM: P-05-004-F, CONICYT: FBO16).

## REFERENCES

[1] W. Duan, B. Gu, and A. B. Whinston, "Do online reviews matter?—an empirical investigation of panel data," *Decision Support Systems*, vol. 45, no. 4, pp. 1007–1016, 2008.

[2] Twitter, "Twitter Api," http://aiweb.techfak.uni-bielefeld.de/content/bworld-robot-control-software/, 2008, [Online; accessed January-2014].

[3] D. M. Romero and J. M. Kleinberg, "The directed closure process in hybrid social-information networks, with an analysis of link formation on twitter." in *ICWSM*, 2010.

[4] B. O'Connor, R. Balasubramanyan, B. R. Routledge, and N. A. Smith, "From tweets to polls: Linking text sentiment to public opinion time series." *ICWSM*, vol. 11, pp. 122–129, 2010.

[5] Twitter, "Who is in twitter," https://business.twitter.com/whos-twitter/, 2014, [Online; accessed January-2014].

[6] Semiocast, "Twitter penetration in Chile," http://semiocast.com/publications/2012_01_31_Brazil_becomes_2nd_country_on_Twitter_superseds_Japan, 2012, [Online; accessed January-2014].

[7] A. Go, L. Huang, and R. Bhayani, "Twitter sentiment analysis," *Entropy*, vol. 17, 2009.

[8] J. Read, "Using emoticons to reduce dependency in machine learning techniques for sentiment classification," in *Proceedings of the ACL Student Research Workshop*. Association for Computational Linguistics, 2005, pp. 43–48.

[9] M. Koppel and J. Schler, "The importance of neutral examples for learning sentiment," *Computational Intelligence*, vol. 22, no. 2, pp. 100–109, 2006.

[10] A. Pak and P. Paroubek, "Twitter as a corpus for sentiment analysis and opinion mining." in *LREC*, 2010.

[11] E. DeGeneres, "Ellen DeGeneres photo tweet at the oscars," https://twitter.com/TheEllenShow/status/440322224407314432, 2014, [Online; accessed March-2014].

[12] B. Obama, "Obama's 4 more years tweet," https://twitter.com/BarackObama/status/266031293945503744, 2012, [Online; accessed January-2014].

[13] S. T. website, "Stopwords list," http://snowball.tartarus.org/algorithms/spanish/stop.txtl, [Online; accessed January-2014].

[14] J. Spencer and G. Uchyigit, "Sentimentor: Sentiment analysis of twitter data," in *Proceedings of European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, 2012, pp. 56–66.

# Bibliografía


[1] Twitter, "Who is in twitter." https://business.twitter.com/whos-twitter/, 2014. [Online; accessed January-2014].

[2] Semiocast, "Twitter penetration in Chile." http://semiocast.com/publications/2012_01_31_Brazil_becomes_2nd_country_on_Twitter_superseds_Japan, 2012. [Online; accessed January-2014].

[3] E. Marrese Taylor, "Diseño e implementación de una aplicación de web opinion mining para identificar preferencias de usuarios sobre productos turísticos de la x región de los lagos," 2013.

[4] A. Pak and P. Paroubek, "Twitter as a corpus for sentiment analysis and opinion mining.," in *LREC*, 2010.

[5] G. E. Moore *et al.*, "Cramming more components onto integrated circuits," 1965.

[6] K. L. Berg, T. Seymour, and R. Goel, "History of databases.," *International Journal of Management & Information Systems*, pp. 29–36.

[7] D. J. Hand, H. Mannila, and P. Smyth, *Principles of data mining*. MIT press, 2001.

[8] T. J. Berners-Lee, "The world-wide web," *Computer Networks and ISDN Systems*, vol. 25, no. 4, pp. 454–459, 1992.

[9] L. G. Aretio, "Web 2.0 vs web 1.0.," *DIM: Didáctica, Innovación y Multimedia*, no. 10, 2007.

[10] J. D. Velásquez and L. Donoso, "Web mining: Análisis sobre la privacidad del tratamiento de datos originados en la web," *Revista Ingeniería de Sistemas Volumen XXIII*, 2009.

[11] T. O'reilly, "What is web 2.0: Design patterns and business models for the next generation of software.," *Communications & strategies*, no. 65, 2007.

[12] W. Duan, B. Gu, and A. B. Whinston, "Do online reviews matter?—an empirical investigation of panel data," *Decision Support Systems*, vol. 45, no. 4, pp. 1007–1016, 2008.

[13] Brightlocal, "Consumptions of online reviews." http://www.brightlocal.com/wp-content/uploads/2013/06/Local-Consuner-Review-Survey-2013.pdf, 2013. [Online; accessed October-2013].

[14] A. C. Martínez, *Estrategias empresariales en la Web 2.0. Las redes sociales online*. Editorial Club Universitario, 2010.

[15] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth, "From data mining to knowledge discovery in databases," *AI magazine*, vol. 17, no. 3, p. 37, 1996.

[16] E. Alpaydin, *Introduction to machine learning*. MIT press, 2004.

[17] R. Schapire, "Theoretical machine learning course, lecture 1." University Lecture, 2008.

[18] O. Z. Maimon and L. Rokach, *Data mining and knowledge discovery handbook*, vol. 1. Springer, 2005.

[19] A. Kao and S. R. Poteet, *Natural language processing and text mining*. Springer, 2007.

[20] J. Allen, "Natural language understanding," 1987.

[21] J. J. Webster and C. Kit, "Tokenization as the initial phase in nlp," in *Proceedings of the 14th conference on Computational linguistics-Volume 4*, pp. 1106–1110, Association for Computational Linguistics, 1992.

[22] M. Choy, "Effective listings of function stop words for twitter," *arXiv preprint arXiv:1205.6396*, 2012.

[23] F. Li, "The information content of forward-looking statements in corporate filings—a naïve bayesian machine learning approach," *Journal of Accounting Research*, vol. 48, no. 5, pp. 1049–1102, 2010.

[24] J. B. Lovins, *Development of a stemming algorithm*. MIT Information Processing Group, Electronic Systems Laboratory, 1968.

[25] P. Willett, "The porter stemming algorithm: then and now," *Program: electronic library and information systems*, vol. 40, no. 3, pp. 219–223, 2006.

[26] EnglishClub, "Parts-of-speech in english." http://www.englishclub.com/grammar/parts-of-speech_1.htm, 2013. [Online; accessed October-2013].

[27] H. Schmid, "Treetagger Website." http://www.ims.uni-stuttgart.de/forschung/ressourcen/werkzeuge/treetagger.html, 2013. [Online; accessed October-2013].

[28] H. Tang, S. Tan, and X. Cheng, "A survey on sentiment detection of reviews," *Expert Systems with Applications*, vol. 36, no. 7, pp. 10760–10773, 2009.

[29] J. M. Wiebe, "Tracking point of view in narrative," *Computational Linguistics*, vol. 20, no. 2, pp. 233–287, 1994.

[30] K. Dave, S. Lawrence, and D. M. Pennock, "Mining the peanut gallery: Opinion extraction and semantic classification of product reviews," in *Proceedings of the 12th international conference on World Wide Web*, pp. 519–528, ACM, 2003.

[31] B. Pang, L. Lee, and S. Vaithyanathan, "Thumbs up?: sentiment classification using machine learning techniques," in *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pp. 79–86, Association for Computational Linguistics, 2002.

[32] J. G. Shanahan, Y. Qu, and J. Wiebe, *Computing attitude and affect in text: theory and applications*, vol. 20. Springer, 2006.

[33] B. Pang, L. Lee, and S. Vaithyanathan, "Thumbs up? sentiment classification using machine learning techniques," in *Proceedings of EMNLP*, pp. 79–86, 2002.

[34] M. Koppel and J. Schler, "The importance of neutral examples for learning sentiment," *Computational Intelligence*, vol. 22, no. 2, pp. 100–109, 2006.

[35] S. R. Das and M. Y. Chen, "Yahoo! for amazon: Sentiment parsing from small talk on the web," in *EFA 2001 Barcelona Meetings*, 2001.

[36] M. Taboada, M. A. Gillies, and P. McFetridge, "Sentiment classification techniques for tracking literary reputation," in *LREC workshop: towards computational models of literary analysis*, pp. 36–43, 2006.

[37] S. Morinaga, K. Yamanishi, K. Tateishi, and T. Fukushima, "Mining product reputations on the web," in *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 341–349, ACM, 2002.

[38] M. Hu and B. Liu, "Mining and summarizing customer reviews," in *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 168–177, ACM, 2004.

[39] A. Java, X. Song, T. Finin, and B. Tseng, "Why we twitter: understanding microblogging usage and communities," in *Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 workshop on Web mining and social network analysis*, pp. 56–65, ACM, 2007.

[40] K. Wickre, "Twitter #7 anniversary." https://blog.twitter.com/2013/celebrating-twitter7, 2013. [Online; accessed October-2013].

[41] J. Read, "Using emoticons to reduce dependency in machine learning techniques for sentiment classification," in *Proceedings of the ACL Student Research Workshop*, pp. 43–48, Association for Computational Linguistics, 2005.

[42] A. Go, R. Bhayani, and L. Huang, "Twitter sentiment classification using distant supervision," *CS224N Project Report, Stanford*, pp. 1–12, 2009.

[43] D. Davidov, O. Tsur, and A. Rappoport, "Enhanced sentiment learning using twitter hashtags and smileys," in *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pp. 241–249, Association for Computational Linguistics, 2010.

[44] R. Parikh and M. Movassate, "Sentiment analysis of user-generated twitter updates using various classification techniques," *CS224N Final Report*, pp. 1–18, 2009.

[45] N. Kwak and C.-H. Choi, "Input feature selection by mutual information based on parzen window," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 24, no. 12, pp. 1667–1671, 2002.

[46] D. D. Lewis, "Naive (bayes) at forty: The independence assumption in information retrieval," in *Machine learning: ECML-98*, pp. 4–15, Springer, 1998.

[47] P. Domingos and M. Pazzani, "On the optimality of the simple bayesian classifier under zero-one loss," *Machine learning*, vol. 29, no. 2-3, pp. 103–130, 1997.

[48] E. T. Jaynes, "On the rationale of maximum-entropy methods," *Proceedings of the IEEE*, vol. 70, no. 9, pp. 939–952, 1982.

[49] D. Beeferman and A. Berger, "Agglomerative clustering of a search engine query log," in *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 407–416, ACM, 2000.

[50] M. Junker, R. Hoch, and A. Dengel, "On the evaluation of document analysis components by recall, precision, and accuracy," in *Document Analysis and Recognition, 1999. ICDAR'99. Proceedings of the Fifth International Conference on*, pp. 713–716, IEEE, 1999.

[51] A. Abbasi, H. Chen, and A. Salem, "Sentiment analysis in multiple languages: Feature selection for opinion classification in web forums," *ACM Transactions on Information Systems (TOIS)*, vol. 26, no. 3, p. 12, 2008.

[52] M. Gamon, A. Aue, S. Corston-Oliver, and E. Ringger, "Pulse: Mining customer opinions from free text," in *Advances in Intelligent Data Analysis VI*, pp. 121–132, Springer, 2005.

[53] M. Gamon, "Sentiment classification on customer feedback data: noisy data, large feature vectors, and the role of linguistic analysis," in *Proceedings of the 20th international conference on Computational Linguistics*, p. 841, Association for Computational Linguistics, 2004.

[54] J. Yi, T. Nasukawa, R. Bunescu, and W. Niblack, "Sentiment analyzer: Extracting sentiments about a given topic using natural language processing techniques," in *Data Mining, 2003. ICDM 2003. Third IEEE International Conference on*, pp. 427–434, IEEE, 2003.

[55] L. B. Batista and S. Ratte, "A multi-classifier system for sentiment analysis and opinion mining," in *Proceedings of the 2012 International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2012)*, pp. 96–100, IEEE Computer Society, 2012.

[56] T. Wilson, J. Wiebe, and P. Hoffmann, "Recognizing contextual polarity in phrase-level sentiment analysis," in *Proceedings of the conference on human language technology and empirical methods in natural language processing*, pp. 347–354, Association for Computational Linguistics, 2005.

[57] M. Sharifi and W. Cohen, "Finding domain specific polar words for sentiment classification," in *Language Technologies Institute Student Research Symposium*, 2008.

[58] C. Whitelaw, N. Garg, and S. Argamon, "Using appraisal groups for sentiment analysis," in *Proceedings of the 14th ACM international conference on Information and knowledge management*, pp. 625–631, ACM, 2005.

[59] T. Mullen and N. Collier, "Sentiment analysis using support vector machines with diverse information sources.," in *EMNLP*, vol. 4, pp. 412–418, 2004.

[60] P. D. Turney, "Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews," in *Proceedings of the 40th annual meeting on association for computational linguistics*, pp. 417–424, Association for Computational Linguistics, 2002.

[61] M. Efron, "Cultural orientation: Classifying subjective documents by cociation analysis," in *AAAI Fall Symposium on Style and Meaning in Language, Art, and Music*, 2004.

[62] N. Agarwal and H. Liu, "Modeling and data mining in blogosphere," *Synthesis lectures on data mining and knowledge discovery*, vol. 1, no. 1, pp. 1–109, 2009.

[63] J. Wiebe, T. Wilson, R. Bruce, M. Bell, and M. Martin, "Learning subjective language," *Computational linguistics*, vol. 30, no. 3, pp. 277–308, 2004.

[64] "Stopwords en español." http://snowball.tartarus.org/algorithms/spanish/stop.txt, 2013. [Online; accessed October-2013].