PONTIFICIA UNIVERSIDAD CATÓLICA DE VALPARAÍSO
FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA INFORMÁTICA

ANÁLISIS DE OPINIÓN A NIVEL NACIONAL DE EMPRESA DE CONFITERÍA CHILENA, USANDO SENTIMENT ANALYSIS

Mauricio Sebastián Galeas Moreno

Profesor Guía: **Silvana Roncagliolo de la Horra**
Profesor co-referente**: Rodrigo Alfaro Arancibia**

INFORME FINAL DEL PROYECTO
PARA OPTAR AL TÍTULO PROFESIONAL
DE INGENIERO CIVIL EN INFORMÁTICA

Carrera: **Ingeniería Civil Informática**
Diciembre de 2017

Resumen

En Twitter constantemente se publican opiniones respecto a diversas temáticas, alcanzando un alto volumen de datos. Analizarlas de manera individual es muy complejo. Es aquí donde entra en juego el uso de Sentiment Analysis, teniendo como objetivo extraer grandes cantidades de opiniones (en este caso tweets), procesarlas y así determinar la opinión actual respecto a una empresa. Dado lo anterior, se propone la utilización Sentiment Anlysis y NLP basado en Machine Learning, para desarrollar una solución que permita clasificar automáticamente opiniones respecto a las empresas. Como caso de estudio, se considera la empresa Fruna que durante el primer semestre del año 2017 estuvo en la palestra mediática, debido a incidentes que menoscabaron su imagen, generándose así una gran cantidad de datos respecto a ella.

Abstract

In Twitter thousands of opinions are posted about certain topics, reaching a large volume of data. Analyze this manually is very complex. In response to that problematic, emerges Sentiment Analysis. Its main goal is to process large amount of opinion data (in this specific case, tweets) and process it to determine the actual opinion about a specific company. During the decade of 2010, this area of knowledge reached its peak. It is proposed the use of Sentiment Analysis and NLP based in Machine Learning, to develop a solution for automatically classify opinions about the company. The selected case of study was the company Fruna. This company was in the eye of the storm during the 1st semester of 2017, because of incidents that harm it very deeply. Those incidents generated a lot of data about them in the social networks.

Glosario

- **Dataset:** Set de datos
- **EEUU:** Estados Unidos de Norteamérica
- **ML:** Machine Learning
- **NLP:** Natural Language Processing
- **OM**: Opinion Mining
- **SA:** Sentiment Analysis / análisis de sentimientos

Contenido

Lista de Figuras

Lista de tablas

1. Introducción

En las últimas dos décadas, las tecnologías de la información han experimentado un enorme aumento en su disponibilidad y facilidad de uso. Esto ha provocado que en la sociedad actual cada vez sean más utilizadas, convirtiéndose en una parte esencial del día a día. Su uso, antes limitado solo a gente de áreas del saber especializado, se ha visto enormemente democratizado por el uso las redes sociales y la web 2.0. Debido a lo anterior, en la web actual, no solo se comparte información como artículos especializados y noticias, sino las vivencias de cada persona y sus opiniones respecto a los acontecimientos que diariamente ocurren en el mundo. En estas opiniones y vivencias, la mayoría de las veces se puede identificar en ellas una emoción predominante que las subyace.

Durante los años 90, el estudio de la expresión de las emociones en la web no era considerado un tema de interés, sin embargo en la década del 2010, estos estudios han comenzado a ser valorados debido a que permite conocer la opinión general de la sociedad respecto a un tópico o entidad con gran exactitud. Lo anterior, sumado a la gran cantidad de la información que circula en la web, ha despertado un interés considerable desde el punto de vista académico y comercial, pues se busca saber qué siente la gente de manera general o colectiva respecto a hechos de actualidad, personajes públicos, productos de grandes empresas, entre otros. Es así como ha surgido una nueva área de la informática llamada *Sentiment Analysis*. Esta nueva rama tiene como propósito determinar las emociones expresadas en cualquier forma de comunicación escrita en la web. Esta área ha experimentado un desarrollo muy acelerado y exponencial desde la llegada de las redes sociales (Facebook, Twitter, Youtube, Instagram, entre otros). Asimismo, se ha observado que su aplicación, uso y exposición al público en general ha estado limitado a algunos fines particulares, como por ejemplo en tiempos de elecciones políticas para conocer la opinión respecto a algún candidato en especial, dejando afuera otras áreas que podrían resultar de interés.

Si bien, en países anglosajones este tema ha sido ampliamente estudiado, en países hispanoparlantes su investigación y utilización ha sido considerablemente menor. Acotándolo a nuestro país, el tema ha sido poco tratado públicamente y su uso se ha restringido, principalmente, al ámbito académico. A su vez, aún no se ha visto en la práctica de manera efectiva el uso de Sentiment Analysis como en los países del primer mundo, como por ejemplo EEUU en donde se han hecho una diversidad de estudios sobre las elecciones presidenciales del 2016, como por ejemplo, *Analyzing Twitter Sentiment of the 2016 Presidential Candidates.* Es por eso que en este trabajo se propone el uso práctico de Sentiment Analysis para la determinación de los sentimientos alrededor de productos asociados a una marca comercial o hechos de actualidad, usando como base los datos obtenidos de la red social con mejor y mayor acceso y disponibilidad: Twitter. Dentro de esta red social, su unidad básica de comunicación es denominada “tweet”, el cual corresponde a un mensaje de un tamaño máximo de 140 caracteres. Para efectos de esta investigación, cabe mencionar que se elegirá un tópico de alta relevancia y con un alto grado de polémica. Esto es debido que para estos casos se cuenta con una mayor cantidad de datos disponible y los

tópicos con un alto grado de polémica, generan emociones colectivas, las cuales son exacerbadas, fervientes y se repiten de manera general entre aquellos que las emiten.

Las condiciones ya propuestas se cumplen idóneamente para la marca nacional de confites Fruna. Durante el primer semestre del 2017, la conocida empresa fue mencionada altamente en las redes sociales. Las razones de esto fueron positivas en un comienzo y posteriormente se volvieron notablemente negativas. Su alta popularidad se debió en gran medida a una campaña publicitaria realizada por la empresa que reforzó la imagen positiva de ser una industria cercana al ciudadano promedio. Sin embargo, a mediados del mismo periodo, la imagen de la empresa decayó enormemente debido a la difusión de noticias que revelaron las pésimas condiciones laborales de sus trabajadores. En este tiempo, se publicaron una gran cantidad de tweets en los que, por una parte, se observaron mensajes de admiración a la empresa y, por otra parte, una crítica férrea y profunda. La gran cantidad de estos mensajes, positivos o negativos, permiten que se pueda desarrollar una propuesta para determinar la opinión actual de la ciudadanía con respecto a la empresa utilizando Sentiment Analysis.

Para desarrollar la propuesta se utilizará Sentiment Analysis con su enfoque más moderno el cual está basado en las tecnologías Machine Learning (ML) y Natural Language Processing (NLP). ML es una rama de las ciencias de la computación que permite al software hacer inferencias y mejorarse a sí mismo. El desarrollo de la solución está compuesto de las siguientes fases: Extracción de los tweets, limpieza y selección de características relevantes de estos, entrenamiento del modelo, llevar a cabo la etapa de pruebas y finalmente obtención de métricas de eficiencia del modelo.

2. Problemática y solución

2.1. Problema

Actualmente en la Web se comparten miles de artículos, mensajes y hechos que se valoran a través de la opinión por cada persona de manera distinta. Dentro de estos miles de mensajes, hay algunos que son valorados de la misma manera por parte de la gente. Esto permite distinguir y determinar con claridad si existe una valoración sentimental general respecto a un tópico, la cual es compartida de manera homogénea por la mayoría de las personas que opinaron al respecto.

Desde los inicios de la comunicación escrita, filósofos, sociólogos y psicólogos han hecho uso de su conocimiento para determinar los sentimientos expresados en la intención comunicativa de los escritores. En nuestra sociedad actual, existe un interés creciente en determinar que sentimientos subyacen en los mensajes compartidos en las redes sociales. Este interés puede nacer desde el receptor del mensaje o del emisor de este, buscando provocar una emoción determinada a quienes son sus receptores. Quizás para alguien que comparte su día a día en las redes sociales no sea algo muy relevante, pero para personas o entidades que buscan influir en la gente para utilizarlo en su beneficio propio, como lo son empresas o personajes públicos, es un tema de suma importancia.

La tarea de determinar las emociones expresadas por el emisor y percibidas por el receptor del mensaje, antes era trabajo solo de las ciencias sociales. Realizar este análisis en la Web actual, el cual consiste en determinar cómo es valorizado sentimentalmente una entidad o personaje en base a las miles o millones de opiniones compartidas respecto a él; es una tarea altamente compleja de realizar manualmente. Es en este momento, en que a través de la informática se puede hacer un análisis con base en estos miles de mensajes y datos que poseen valoración sentimental. Sin embargo, la lógica computacional no es capaz de acotar y analizar adecuadamente las emociones humanas. El área humanista e informática por si solas no son suficientes para realizar una valoración sentimental adecuada en base a todas las opiniones compartidas en la Web. Es por esto que ha surgido una nueva área de la informática denominada Sentiment Analysis. Esta rama de las ciencias computacionales permite determinar la valoración sentimental de un tópico, entidad o personaje, teniendo como su unidad más básica un simple mensaje de opinión respecto a este. Esta área combina el poder de procesamiento de datos de la computación, junto con el análisis matemático-estadístico para analizar lingüísticamente miles mensajes con un alto contenido sentimental. Además, permite a la máquina el aprendizaje de patrones comunes para poder realizar una predicción aproximada de una valoración sentimental dado mensajes nuevos.

La llegada de la Web 2.0 y la masificación de las redes sociales ha permitido un gran desarrollo de esta área en el último tiempo, especialmente en los países desarrollados. En estos ha sido ocupado principalmente para estudiar la opinión de la gente respecto a políticos durante elecciones. En menor medida ha sido utilizada para estudiar la opinión de la gente respecto a empresas importante como aerolíneas. En nuestro país el uso de SA. ha sido principalmente en el ámbito académico, con un escaso uso práctico conocido. Su utilización

ha estado enfocada en el estudio y análisis opiniones de figuras públicas y políticos importantes. Se ha observado en muy pocas ocasiones un uso de este al servicio de las empresas nacionales, pudiendo ser de gran ayuda para estas debido a que les permitiría saber de manera más fácil la opinión que tiene la ciudadanía respecto a ellos. Si se han realizado estudios al respecto, no han salido a la luz pública y han sido tratado netamente como un tema de empresa.

2.2. Solución propuesta

Debido a lo anterior es que se propone como solución para el problema detectado, el uso de Sentiment Analysis para la determinación de la opinión o valoración sentimental de la gente respecto a empresas nacionales. El análisis sentimental tendrá como base el uso de técnicas de NLP para un correcto entendimiento de la información textual ingresada y Machine Learning para determinar la valoración sentimental de manera eficiente. La empresa elegida para hacer uso de lo anterior es una que ha estado en el ojo de huracán durante el primer semestre del año 2017: **Fruna**. Los datos usados para realizar este análisis provienen de las redes sociales, donde la gente en la actualidad expresa sus opiniones frecuentemente y de manera completamente sincera. La fuente de estos datos corresponde a una de las redes sociales con mayor uso en la década actual: Twitter. Esta se refiere específicamente a los mensajes compartidos en esta red social, los cuales son denominados tweets y poseen una longitud máxima de 140 caracteres. Cabe mencionar que los datos obtenidos son los de más fácil acceso y obtención de todas las redes sociales, y su alta contenido sentimental los hace perfectos para esta investigación.

Los datos extraídos corresponden a tweets que hagan alusión a la empresa Fruna y que hayan sido publicados entre el 1 de Mayo de 2017 y 30 de Julio de 2017. Fue seleccionado este rango de fechas debido a que durante esa época la opinión respecto a la empresa cambio radicalmente, para después volver a un estado similar a la inicial. Durante los primeros meses del 2017 la empresa contaba con una gran popularidad debido a los bajos precios de sus productos, ser un referente de la cultura popular y [1] principalmente por la campaña en pos de apoyar esta imagen de la empresa iniciado por el medio alternativo de internet *NoEsNaLaFeria*. Pero la imagen positiva de la empresa cambiaría radicalmente. A finales de abril empezaron a surgir rumores del supuesto suicidio de uno de los trabajadores de la empresa. [2] Estos rumores finalmente se verificaron cuando el medio nacional The Clinic, publicó un artículo donde se informaba el suicidio de 2 trabajadores de la empresa Fruna, debido a un gran estrés y malas condiciones laborales. Durante el transcurso del mes de Mayo se publicaron más noticias respecto a la empresa que hablaban de las muy malas condiciones laborales de los empleados. El conjunto de estas noticias dejó en claro que la empresa presentaba turnos agotadores, malos tratos a los trabajadores y muy malas condiciones laborales; afirmando el hecho que el suicidio de ambos trabajadores había sido culpa de lo mismo. Las noticias sumado a declaraciones negativas del mismismo dueño de la empresa, dejaron la opinión de la empresa por los suelos durante Mayo y Junio. Esto se vio expresado en miles de tweets que expresaban el descontento por la situación,

observándose una clara expresión de emociones colectivas negativas en estos. Finalmente, durante Julio los hechos ocurridos parecieron ser olvidados por la mayoría de las personas. Durante ese periodo se pudo observar que la mayoría de las opiniones respecto a la empresa son neutras, seguidos por opiniones negativas y finalmente positivas.

3. Objetivos

3.1. Objetivo general

Proponer y desarrollar un software para clasificar las opiniones respecto a la empresa nacional de confites Fruna, usando Sentiment Analysis en conjunción con Machine Learning y Natural Language Processing.

3.2. Objetivos específicos

- Elaborar el marco teórico respecto a la aplicación de Sentiment Analysis.
- Proponer un modelo que integrando ML y NLP permita resolver el caso en estudio.
- Desarrollar la solución propuesta utilizando los algoritmos de clasificación elegidos, adaptándolos al caso actual.
- Evaluar los resultados para determinar así cual algoritmo es más eficiente.

4. Cronología

Dada la naturaleza del proyecto, este se dividirá principalmente en 4 fases las cuales son: Marco Teórico, Análisis y Diseño, Desarrollo y Pruebas.

4.1. Marco Teórico

Tiempo estimado: Marzo - Abril del 2017

- Definición y descripción del problema
- Contexto psicológico y social
- Marco teórico tecnológico
- Propuesta de la solución

4.2. Análisis y Diseño

Tiempo estimado: Mayo - Junio del 2017

- Análisis y especificación de las características del sistema solución
- Diseño de la arquitectura del sistema
- Especificación de tecnologías a usar
- Implementación preliminar
- Definición de métricas de eficiencia

4.3. Desarrollo

Tiempo estimado: Julio - Septiembre del 2017

- Procesamiento de los datos usando Natural Language Processing
- Implementación de la solución
- Pruebas preliminares

4.4. Pruebas

Tiempo estimado: Septiembre - Noviembre del 2017

- Mejoramiento de la implementación
- Ejecución de las pruebas finales del sistema
- Análisis de las pruebas y sus resultados.
- Conclusiones

5. Estado del arte

5.1. Teoría

5.1.1. Las emociones individuales

[4]El origen de la palabra emoción proviene del latín "*emovere*" cuyo significado es remover, agitar o excitar. Según esta definición, se da a entender que las emociones son acciones que producen cambios de estado en los seres humanos.

El poder distinguir una emoción de otra quizás tiene que ver con las manifestaciones y reflejos del cuerpo humano, es decir, la valoración de los cambios fisiológicos como indicativos de la gestación emocional. Esto permite llevar a cabo el proceso de entendimiento y clasificación de las emociones.

Si bien, los signos vitales son referentes para evaluar las emociones no son considerados como las únicas discriminantes para el reconocimiento de las mismas. Existen otras funciones de los seres humanos las cuales les permiten discernir si es efectivamente una emoción o no lo es, llamadas procesos cognitivos, los cuales operan estableciendo relaciones sobre la memoria y la percepción. Entre los procesos cognitivos se pueden encontrar el pensar, razonar y la constitución de la consciencia. Estos hechos apuntan sobre la participación de la cognición sobre el qué y el por qué detrás de ciertos comportamientos, lo que permite diagnosticar de forma evidente las emociones.

Lo que da origen a una emoción es una acción asociada a un evento, noticia o hecho que impactan directamente sobre los individuos los cuales se les denominan "estímulos". Dichos estímulos se dividen en dos tipos, externos e internos Los primeros se refieren a cómo afecta la opinión de otras personas la percepción propia, en particular cómo influye la sociedad, la cultura y la religión en ellos. Por otra parte, los internos se refieren a la visión y valoración propia sobre diferentes acontecimientos.

Es indudable que de cara a la gestación de emociones se requiere la participación de un conjunto de funciones de los seres humanos que no se pueden generalizar ya que están condicionadas por las vivencias y experiencias de cada individuo, por lo que difieren de individuo a individuo, debido a las características cognitivas de cada uno. Este hecho, pone en evidencia que el reconocimiento de emociones colectivas no representa el resumen de los estados emocionales individuales de los miembros de un colectivo.

5.1.2. Emociones colectivas

Dado la gran disparidad entre una emoción de un individuo y la otro, estudiar las emociones de manera individual, no será de ayuda para el análisis sentimental. Es por eso que se deben usar las emociones colectivas. Las emociones colectivas se entienden como un fenómeno que precisa de la participación de colectivos y eventos para que se puedan suscitar, y que serán registradas y evaluadas por más de una persona bajo su criterio cognitivo. Las

emociones colectivas sólo se manifiestan cognitivamente. Las manifestaciones biológicas o fisiológicas se desprecian para realizar el reconocimiento de emociones colectivas, debido a la dificultad de cuantificar variables biológicas o fisiológicas de un gran número de personas.

El proceso para el reconocimiento de emociones individuales y colectivas, presenta ciertas semejanzas al igual que diferencias. En el primer caso el proceso se basa en el uso de funciones fisiológicas y cognitivas para este fin, mientras en el segundo caso se priorizan las componentes de las funciones netamente cognitivas tales como la actitud, el comportamiento, el lenguaje, entre otras, dado que es difícil o imposible reconocer las manifestaciones fisiológicas del colectivo.

Otras características importantes en la gestación de expresiones colectivas son: la tónica pluralista es decir pasa del yo al nosotros. Además, del uso de representaciones de expresión mínimas con capacidad intensificadora, que se refieren a ciertas manifestaciones que aparecen de forma rápida, ocasionando revuelo y euforia en tiempo cortos sobre las personas de un colectivo, como por ejemplo los cánticos en las protestas.

5.2. Algoritmos y métodos

5.2.1. Sentiment Analysis

5.2.1.1. Historia y definición

"El saber lo que la gente piensa", ha sido un tema importante desde siempre. En los 90's, antes de la gran masificación de internet ya había intereses por parte de políticos del saber que opinaban de ellos o por parte de pequeñas empresas que opinaba la gente de sus productos. Todo esto obtenido de datos públicos los cuales fueron aumentando con la masificación de internet. La explosión de la Web 2.0 permitió descubrir a las compañías el enorme potencial de las opiniones de los consumidores emitidas en blogs y comentarios en foros.

Esto dio paso a la generación de algoritmos y herramientas probabilísticas para su estudio. Todos estos casos presentan claramente un elemento en común que es la capacidad de poder reconocer expresiones afectivas y de opiniones de las personas por medio de la aplicación de técnicas de Opinion Mining (OM) y Sentiment Analysis. Lo expuesto anteriormente establece un marco conocido como análisis subjetivo que comprende el estudio del OM y SA.

En el primer caso, se encuentra en OM. una técnica para el reconocimiento de opiniones, que pertenece a un conjunto de técnicas de Data Mining, cuyo uso está destinado a la recuperación de información de un conjunto de documentos, textos o Webs. O.M. se usa generalmente para hallar y evaluar atributos de un objeto, el cual permite categorizarlo de forma positiva, neutral o negativa.

Por otro lado, los casos en los cuales se busca develar la presencia de una valoración sentimental dentro de una colección de datos requieren el uso de técnicas más específicas. Para esto, existen varias aproximaciones acordes a las problemáticas a investigar.

Sentiment Analysis es una disciplina de la computación cuyo fin es la extracción, identificación o caracterización de una emoción en una unidad de texto. Para aquello se usan una diversidad de técnicas entre las que se encuentran Machine Learning, técnicas estadísticas y Natural Language Processing. En otras palabras, Sentiment Analysis se refiere al método general para extraer la subjetividad y la polaridad de un texto.

5.2.1.2. Tipos de evaluación sentimental

***Evaluación cuantitativa o polar*:** Este sistema de valoración sentimental evalúa cuantitativamente un elemento, normalmente en una escala numérica del 1 al 10 o usando porcentajes. Los valores varían desde un extremo a otro, desde muy bueno a muy malo o desde muy positivo a muy negativo. Justo al medio de estos se encontrará una valoración neutral la cual representa que el elemento no expresaba una emoción clara. Los valores entre neutral y muy positivo o neutral y muy negativo, representarán valores intermedios entre los extremos. Esto se puede observar en el siguiente ejemplo gráfico de una valoración sentimental cuantitativa.

Evaluación cualitativa o de etiquetas: Dada la naturaleza subjetiva de las opiniones y los sentimientos, el sistema anterior muchas veces no parece ser el apropiado. Es por esto que se recurre a un sistema de valoración cualitativo donde se ocupan etiquetas las cuales describen la emoción u opinión transmitida en el texto analizado. En la mayoría de los casos las etiquetas se refieren derechamente a emociones como felicidad, tristeza, enojo. Un alcance alternativo a lo anteriormente mencionado es el uso de “emojis” como etiquetas en vez de la emoción como tal. También existe el caso que las etiquetas usadas no sean emociones sino opiniones como bonito, feo, agradable, desagradable.

5.2.1.3. Proceso de estimación de la valorización sentimental

En la actualidad existe una diversidad de técnicas de valoración sentimental las cuales han ido mejorando a través de la historia y especializándose. Estas técnicas se pueden dividir según el procesamiento que se hace al llevar el texto a una representación entendible para la máquina y al clasificar este mismo. En la figura 5.1 se puede observar de manera general los diversos enfoques que posee Sentiment Analysis.

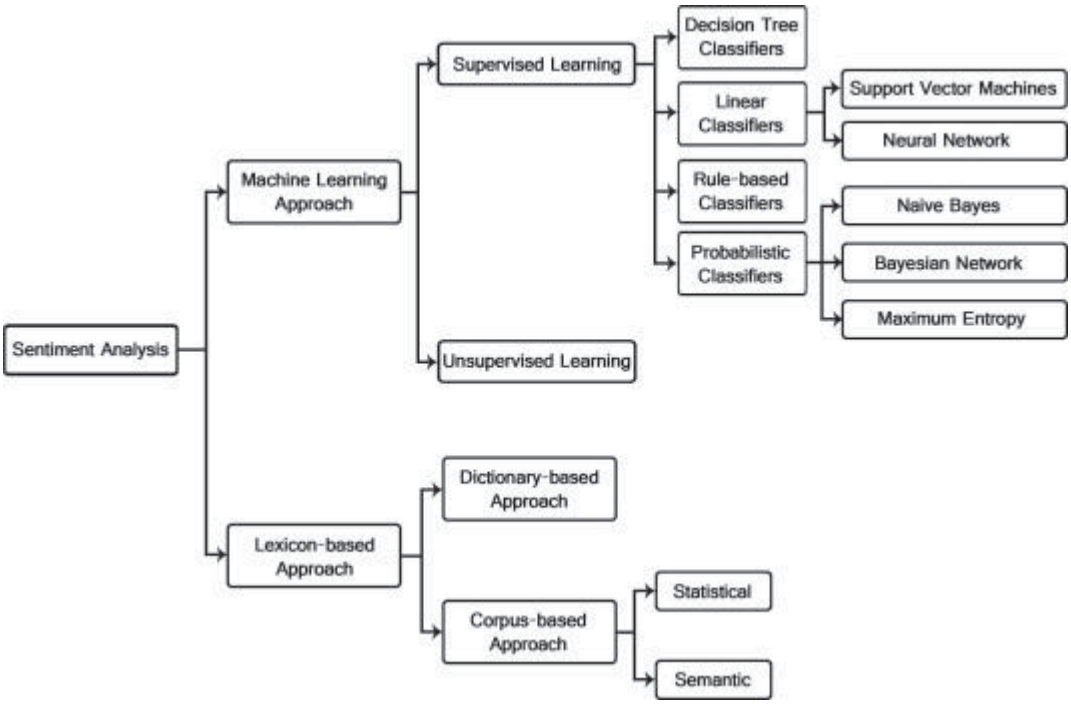

Figura 5.1: Tipo de clasificadores usados en Sentiment Analysis.

5.2.1.4. Enfoque basado en Lexicón

El enfoque basado en Lexicón implica el cálculo de la orientación o polaridad de un documento basándose en la orientación semántica de palabras o frases en el documento. El enfoque Lexicón usa diccionarios como base para hacer sus estimaciones. Los diccionarios para este enfoque pueden ser creados manualmente o automáticamente, usando palabras semilla para expandir la lista total de palabras. Gran parte de la investigación basada en Lexicón, se ha centrado en el uso de adjetivos como indicadores de la orientación semántica del texto. Los otros tipos de palabras que más aportan en el uso de este enfoque son los verbos y pronombres, siendo de especial importancia aquellas palabras que contribuyen a la negación.

El proceso general de la valorización sentimental basada en lexicón consiste generalmente en los siguientes pasos. En primer lugar, la lista de palabras relevantes para la clasificación como las mencionadas anteriormente y sus correspondientes valores de orientación semántica son compilados en un diccionario. Entonces, para cualquier texto dado, todas las palabras relevantes son extraídas y anotadas con su valor de orientación semántica, usando las puntuaciones del diccionario. Los puntajes de orientación semántica son a su vez agregados a la sola puntuación principal que corresponde a la valorización de todo el texto analizado.

El principal problema del enfoque basado en lexicón es que no poseen la capacidad de detectar el contexto adecuadamente y el cómo se relacionan las palabras entre ellas. Asimismo, no logra determinar la importancia de una palabra y como esta pueda ir cambiando, respecto al texto. Lo anterior hace que actualmente el uso este enfoque sea

considerado ineficiente para su uso en la mayoría de los tipos de textos analizados haciendo uso de Sentiment Analysis.

5.2.2. Machine Learning (Aprendizaje de máquinas)

Es el programar computadores para optimizar un criterio de rendimiento, utilizando datos de ejemplo o experiencia pasada. Se tiene un modelo definido sobre algunos parámetros, y el aprendizaje es la ejecución de un software para optimizar los parámetros del modelo usando los datos de entrenamiento o experiencia pasada.

El modelo puede ser predictivo para hacer predicciones en el futuro, o descriptivo para obtener conocimiento de los datos, o ambos. Machine Learning utiliza la teoría estadística en la construcción de modelos matemáticos, porque la tarea principal es hacer una inferencia a partir de una muestra.

La introducción del papel de las ciencias de la computación es doble: En primer lugar, en el entrenamiento, se necesitan algoritmos eficientes para resolver el problema de optimización, así como para almacenar y procesar la gran cantidad de datos que generalmente se tienen. En segundo lugar, una vez que un modelo se aprende, su representación y solución algorítmica para la inferencia debe ser eficiente también. En ciertas aplicaciones, la eficiencia del algoritmo de aprendizaje o de inferencia, es decir, su complejidad espacial y temporal, puede ser tan importante como su precisión predictiva.

Como se mencionaba anteriormente hay 2 enfoques principales dentro de ML la obtención de conocimientos y predicción. El sistema de la solución busca la etiquetación sentimental de textos, cuyo caso corresponde al uso de Machine Learning para la clasificación de textos.

Todo modelo de Machine Learning independiente del uso que se le quiera dar tienen elementos comunes y básicos los cuales son:

- Modelo de representación: La totalidad de los elementos ingresados a un sistema de Machine Learning tienen que ser llevados a una representación entendible y relevante para el sistema. La totalidad de los datos ingresados, son llevados a un modelo de representación el cual puede variar según la función de valorización elegida, los tipos de datos ingresados y finalmente el problema a resolver. El modelo de representación es el campo donde entrarán a jugar las variables.
- Función de evaluación: [13] Para cualquier tipo de problema de aprendizaje de máquina, se debe saber cómo evaluar nuestros resultados, o cuál es la métrica u objetivo de evaluación. La función de evaluación lleva a una representación matemática cada dato ingresado, indicando la importancia de este respecto a los demás.
- Entrenamiento: [14] El entrenamiento es el proceso de hacer que el sistema pueda aprender lo deseado. Es el proceso de la generalización de comportamientos a

partir de una información suministrada en forma de ejemplos. Es, por lo tanto, un proceso de inducción del conocimiento para su uso posterior para un determinado fin del sistema.

- Pruebas: Es el proceso donde se probará si el conocimiento obtenido es el correcto, comparando los datos obtenidos en la fase de entrenamiento con la fase de prueba. En esta fase también se entregará un resultado de error en la predicción y variará según los objetivos del sistema y el tipo de aprendizaje elegido.

Estos son los elementos en común dentro de todo el mundo de Machine Learning. Pero el funcionamiento u objetivo que persigue un sistema de Machine Learning hará variar los componentes usados en este. Lo anterior hace que existan diversos criterios para clasificar los sistemas de Machine Learning. El criterio más general es respecto al tipo de aprendizaje que usará el sistema. Estos pueden ser:

- Aprendizaje no supervisado
- Aprendizaje semi-supervisado
- Aprendizaje supervisado: el cual será usado en la solución, usándolo específicamente para la clasificación.

5.2.2.1. Aprendizaje supervisado

[15][16]En el Aprendizaje Supervisado se cuenta con un conjunto de ejemplos de los cuáles se conoce la respuesta. Lo que se busca es formular algún tipo de regla o correspondencia que nos permita entregar una respuesta aproximada para todos los objetos que sean ingresados en el sistema. En el aprendizaje supervisado, cada ejemplo es un par que consiste en un objeto de entrada (típicamente un vector) y un valor de salida deseado (también llamado la señal de supervisión). En la figura 5.2 se puede observar el proceso de aprendizaje supervisado.

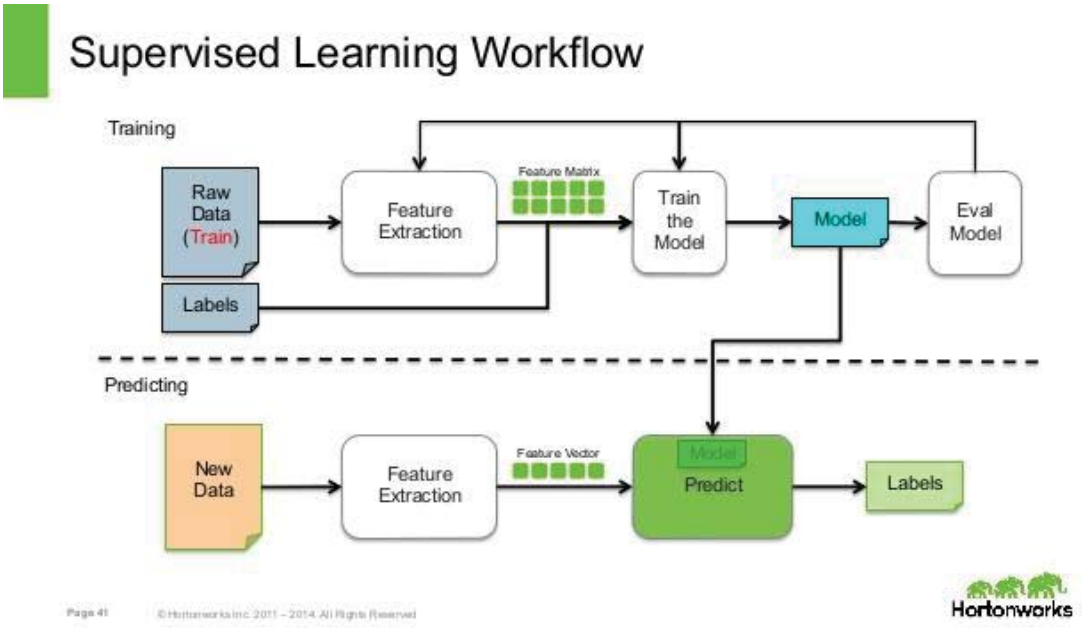

Figura 5.2: Proceso de aprendizaje supervisado.

Cada objeto está compuesto por una serie de *predictores o características*, también conocidos como *features.* Se tiene:

- $x^{(i)} \in \mathbb{R}^n$: un objeto se representa como un vector de n dimensiones, donde n es el número de features de dicho objeto.
- $x^{(i)}$: denota el objeto i
- x_i: para denotar el *feature* i del objeto.

Un algoritmo de aprendizaje supervisado analiza los datos de entrenamiento y produce una función inferida. Esto formalmente se ve expresado de la siguiente manera, siendo:

X– conjunto de objetos.

Y– conjunto de etiquetas (o respuestas).

$y : X \to Y$– función de dependencia entre los objetos y etiquetas.

$y_i = y(x^{(i)}), i = 1, \ldots, l$ valores **conocidos** de la función.

Se desea encontrar $h : X \to Y$, una función h, que permita *aproximar* el conjunto de objetos al de respuestas. Esta función se conoce como **hipótesis.** Para este propósito, se utiliza un algoritmo de aprendizaje, que permite *aproximar* esta función.

El tipo de función obtenida varía según el problema a resolver los cuales en el aprendizaje supervisado puede ser de 2 tipos.

En primer lugar puede ser de regresión donde $Y \in \mathbb{R}$, es decir, la respuesta es una variable numérica con una infinidad de posibles resultados. La resolución de un problema de regresión produce como hipótesis la función de regresión, la cual da una salida o respuesta del tipo continua

En segundo lugar está el problema de clasificación el cual es precisamente el problema a resolver en la investigación: clasificación sentimental en base de tweets. Al resolver un problema de clasificación se busca la asignación de una categoría perteneciente a un conjunto bien definido (y finito) a los objetos de entrada. La función de hipótesis entregada en Clasificación se conoce Clasificador, los cuales entregan una salida del tipo discreta. Originalmente los clasificadores fueron plantados solamente como entidades matemáticas teóricas, siendo muchos olvidados por años. Pero con el surgimiento y auge del Machine Learning, estos han sido rescatados e incorporados con usos prácticos en el área.

5.2.3. Algoritmos de clasificación basados en Latent Semantic Analysis

5.2.3.1. Multinomial Naive Bayes

Es una versión mejorada del algoritmo Naive Bayes, utilizado especialmente para la clasificación de textos.

[17] Naive Bayes se basa en la aplicación de la Regla de Bayes para predecir la probabilidad condicional de que un documento pertenezca a una clase P (ci|dj), a partir de la probabilidad de los documentos dada la clase P (dj|ci) y la probabilidad a priori de la clase en el conjunto de entrenamiento P (ci). Se observa esto en la siguiente fórmula:

$$P(c_i|d_j) = \frac{P(c_i)P(d_j|c_i)}{P(d_j)}$$

Dado que la probabilidad de cada documento P (dj) no aporta información para la clasificación, el término suele omitirse. La probabilidad de un documento dada la clase suele

asumirse como la probabilidad conjunta de los términos que aparecen en dichos documentos dada la clase y se calculan como:

$$P(d_j|c_i) = \Pi_{t=1}^{|V|} P(w_t|c_i)$$

Adicionalmente, el modelo Naive Bayes Multinomial considera la frecuencia de aparición de cada término en los documentos x_t, en vez de una ocurrencia binaria como lo sería en el modelo tradicional:

$$P(d_j|c_i) = \Pi_{t=1}^{|V|} P(w_t|c_i)^{x_t}$$

El término $P(w_t|c_i)$ se calcula a partir del número de apariciones de cada término w_t en una clase c_i . Para evitar el problema de la ocurrencia probabilidades 0 se usa la *estimación o suavizado de Laplace.* Todo la anterior se observa en la fórmula:

$$P(w_t|c_i) = \frac{1 + n(w_t, c_i)}{|V| + n(c_i)}$$

Donde $n(w_t, c_i)$ es el número de ocurrencias de w_t en c_i, $|V|$ es el tamaño del vocabulario y $n(c_i)$ es el conteo total de palabras en c_i. De este modo, la clasificación se hace buscando el argumento que maximiza la función:

$$c^*(d) = argmax_{ci} p(c_i) \Pi_{t=1}^{|V|} P(w_t|c_i)^{x_t}$$

5.2.3.2. Multi-Class Support Vector Classifier

[21] El algoritmo SVM (Support Vector Machine) es un algoritmo de clasificación / regresión lineal. Intenta encontrar un hiperplano que separe los datos en dos clases lo más óptimamente posible. Lo más óptimamente posible significa que los puntos de la etiqueta A desde un lado del hipeplano, deben estar separados lo más posible de los puntos de la etiqueta B desde el otro lado, mientras se maximiza la distancia de todos los puntos a este hiperplano.

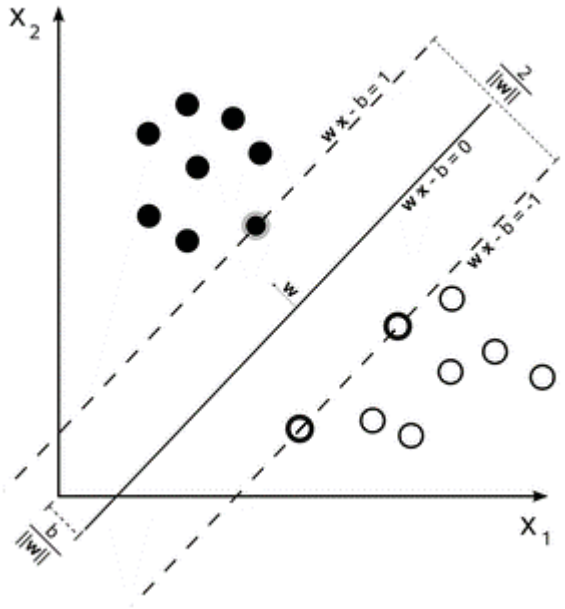

Figura 5.3. Representación gráfica de SVM

En la figura 5.3 se puede ver esto ilustrado para el ejemplo de puntos trazados en el espacio 2D. Los conjuntos de puntos se etiquetan con dos categorías (ilustradas aquí con puntos en blanco y negro) y SVM elige el hiperplano que maximiza el margen entre las dos clases. Este hiperplano está dado por la ecuación:

$$\langle \vec{w} \cdot \vec{x} \rangle + b = \sum_i y_i \alpha_i \langle \vec{x_i} \cdot \vec{x} \rangle + b = 0$$

Donde $\vec{x_i} = (x_{i1}, x_{i2}, .., x_{in})$ es un vector de entrada n-dimensional, y_i es su valor de salida, $\vec{w} = (w_1, w_2, .., w_n)$ es el vector de pesos (el vector normal) que define el hiperplano y los términos Alpha i son los multiplicadores de Lagrange.

Una vez que se construye el hiperplano (el vector $\vec{w}$ se define) con un conjunto de entrenamiento, la clase de cualquier otro vector de entrada $\vec{x_i}$ se puede determinar: Si $\vec{w} \cdot \vec{x_i} + b \geq 0$ entonces pertenece a la clase positiva de lo contrario pertenece a la clase negativa que es la única otra clase disponible.

Cuando el problema a resolver es demasiado grande o no es linealmente separable utiliza una función de kernel para mapear los datos a un espacio dimensional superior en el que es linealmente separable.

El sistema SVM clásico es un clasificador binario, lo que significa que sólo se puede separar el conjunto de datos en dos clases. Para tratar con conjuntos de datos con más de dos clases, el dataset se reduce a un conjunto de datos de clase binaria con el que el SVM puede funcionar. Existen dos enfoques para descomponer un problema de clasificación multiclase en un problema de clasificación binaria: el enfoque de uno contra todos y uno frente a uno.

En el enfoque de uno-contra-todos uno clasificador SVM se construye por clase. Este Clasificador toma una clase como la clase positiva y el resto de las clases como la clase negativa. Un datapoint se clasifica sólo dentro de una clase específica si es aceptado por esa clase clasificadora, y rechazado por todos los demás clasificadores. Aunque esto puede llevar a resultados precisos (si el conjunto de datos está agrupado), muchos de los puntos de datos también pueden dejarse sin clasificar (si el conjunto de datos no está agrupado).

En el enfoque uno contra uno, se crea un clasificador SVM por cada par de clases. Puesto que hay 0,5 N (N-1) posibles combinaciones de pares para un conjunto de N clases, esto significa que tienen que construir más clasificadores. Los datapoints se clasifican en la clase para la que han recibido más puntos.

5.2.3.3. Árbol de decisión clasificador

[25]Árbol de decisión clasificador es una técnica de clasificación simple y ampliamente utilizada. Se plantea una serie de preguntas cuidadosamente elaboradas sobre los atributos del registro de prueba. Cada vez que se recibe una respuesta, se realiza una pregunta de seguimiento hasta que se llega a una conclusión sobre la etiqueta o clase del registro.

[25]Los árboles de decisión organizan una serie de preguntas y condiciones de prueba en una estructura de árbol. La figura 5.4 muestra un árbol de decisión de ejemplo para predecir si una persona hace trampa o no. En el árbol de decisiones, los nodos raíz e internos contienen atributos de condiciones de prueba para separar los registros que tienen diferentes características. Todo el nodo terminal se le asigna una clase que en este caso puede ser: yes o no.

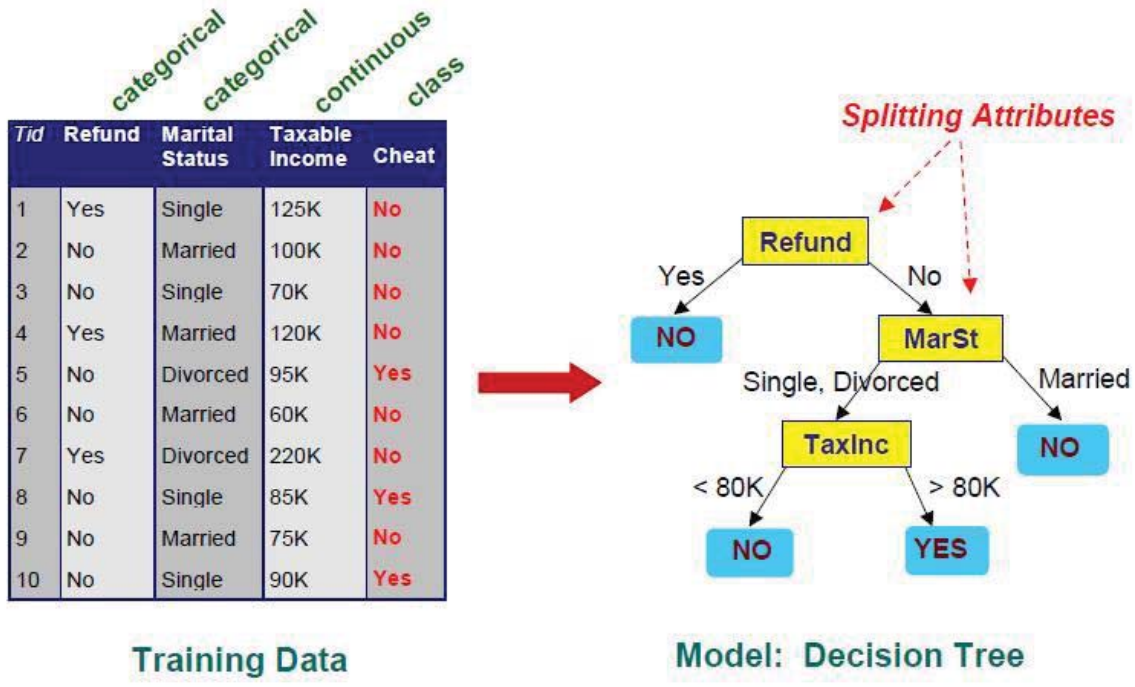

Figura 5.4: [25] Fase de pruebas en un árbol de decisión.

[25]Una vez que se ha construido el árbol de decisiones, clasificar un registro de prueba es sencillo. Comenzando desde el nodo raíz, aplicamos la condición de prueba al registro y se sigue la rama apropiada en función del resultado de la prueba. Luego nos lleva a otro nodo interno, para el cual se aplica una nueva condición de prueba, o a un nodo hoja. Cuando se llega al nodo hoja, la etiqueta de clase asociada con el nodo hoja entonces es asignada al registro. Como se muestra en la figura 5.5, se rastrea la ruta en el árbol de decisión para predecir la etiqueta de clase del registro de prueba, y el camino termina en un nodo hoja etiquetado con la clase NO.

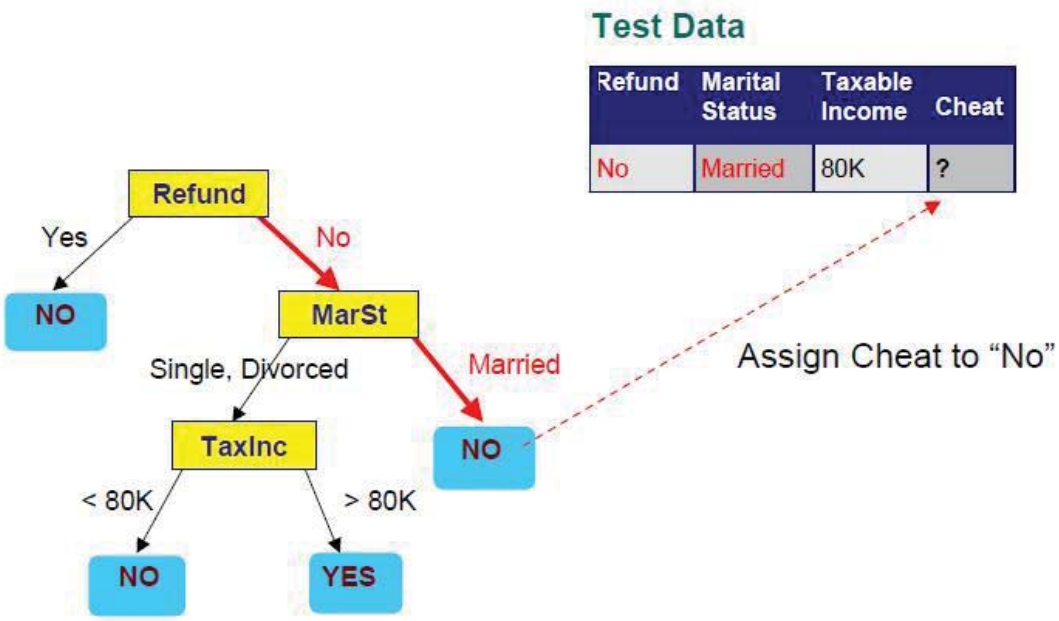

Figura 5.5: [25] Clasificación de data prueba usando árbol de decisión.

[25]Construir un árbol de decisión óptimo es un problema clave en el clasificador árbol de decisión clasificador. En general, los árboles de decisión pueden construirse a partir de un conjunto dado de atributos. Si bien algunos de los árboles son más precisos que otros, encontrar el árbol óptimo no es posible computacionalmente, debido al tamaño exponencial del espacio de búsqueda. Sin embargo, se han desarrollado varios algoritmos eficientes para construir arboles de decisión razonablemente precisos, aunque menos óptimos, en un período de tiempo razonable. Por lo general, estos algoritmos emplean una estrategia ambiciosa que genera un árbol de decisiones al hacer una serie de decisiones localmente óptimas sobre qué atributo usar para particionar los datos. Por ejemplo, el algoritmo de Hunt, ID3, C4.Los algoritmos de árboles de decisión más conocidos son el ID3 y el C4.x. Ambos fueron propuestos en la década de los noventas por Ross Quinlan.

Algoritmo ID3

[26]El algoritmo se inicia respondiendo a la interrogante: qué atributo ingresado debe ser la raíz del árbol. Para ello se evalúa cada atributo usando un test estadístico para determinar cuán bien clasifica esos ejemplos (en realidad se determina el más representativo o el que mejor describe ese conjunto).

[27]La primera métrica que define este el algoritmo es la Entropía. Esta mide el grado de incertidumbre o de desorden, y es usado para ayudar a decidir qué atributo debe ser el siguiente en seleccionarse. En general, un atributo que puede ayudar a discriminar más objetos tiende a reducir más la entropía, y por tal motivo, debe ser seleccionado como un nodo de prueba o de selección para la siguiente subdivisión.

Para calcular la entropía de *n* clases se utiliza la fórmula:

$$\text{Entropia}(S) = \sum_{i=1}^{n} -p_i \log_2 p_i$$

Donde:

- *S*: es una colección de objetos
- P_i : es la probabilidad de los posibles valores
- *i*: las posibles respuestas de los objetos

[27]Para una muestra homogénea la entropía es igual a 0 y ,por el contrario, si la distribución de objetos que pertenecen a una clase es la misma, entonces la entropía es igual a 1, o sea es la máxima incertidumbre. Si en el cálculo de la entropía se presentara una proporción 0.0, entonces se tomará la expresión 0 $log_2 (0)$ como igual a 0.

[28]En segundo lugar se encuentra el concepto de ganancia de información. Se define como la función que permite seleccionar la prueba que debe etiquetar el nodo actual. Define la ganancia para una prueba *T* y una posición *p*.

$$\boldsymbol{Gain(p,T) = Entropie(p) - \sum_{j=1}^{n} (p_j \times Entropie(p_j))}$$

[28] Donde (pj) es el conjunto de todos los valores posibles para el atributo T. Esta medida finalmente para clasificar atributos y construir el árbol de decisión donde en cada nodo se ubique el atributo con la mayor ganancia de información entre los atributos aún no considerados en la ruta desde la raíz.

Algoritmo C4.X

Una limitación de ID3 es que es demasiado sensible a atributos con grandes cantidades de valores. Para resolver esto se utiliza el algoritmo C4.5, una extensión ID3. C4.5 usa "Ganancia de información". Este cálculo no produce en sí mismo nada nuevo. Sin embargo, permite medir una nueva métrica llamada “Ratio de Ganancia”. Se define como:

$$\boldsymbol{GainRatio(p,T) = \frac{Gain(p,T)}{SplitInfo(p,T)}}$$

$$SplitInfo(p, test) = -\sum_{j=1}^{n} P'\left(\frac{j}{p}\right) \times log\left(P'\left(\frac{j}{p}\right)\right)$$

Donde:

P '(j / p) es la proporción de elementos presentes en la posición p, tomando el valor de j-ésima prueba. Tenga en cuenta que, a diferencia de la entropía, la definición anterior es independiente de la distribución de ejemplos dentro de las diferentes clases. Al igual que ID3, los datos se ordenan en cada nodo del árbol para determinar el mejor atributo de división. Utiliza el método de cálculo de impureza de ratio de ganancia para evaluar el atributo de división.

5.2.3.4. Random Forest Classifier

[29] Random forest es un método que combina una cantidad grande de árboles de decisión independientes probados sobre conjuntos de datos aleatorios con igual distribución. La fase de aprendizaje consiste en crear muchos árboles de decisión independientes, construyéndolos a partir de datos de entrada ligeramente distintos. Se altera, por tanto, el conjunto inicial de partida, haciendo lo siguiente:

- Se **selecciona aleatoriamente con reemplazamiento** un porcentaje de datos de la muestra total. Es habitual incluir un segundo nivel aleatoriedad, esta vez afectando los atributos.
- En cada nodo, al seleccionar la **partición óptima**, tenemos en cuenta sólo una porción de los atributos, elegidos al azar en cada ocasión.

Una vez que tenemos muchos árboles, 1000 por ejemplo, la fase de clasificación se lleva a cabo evaluando cada árbol de forma independiente. La predicción del *bosque,* será la media de los 1000 árboles. La proporción de árboles que toman una misma respuesta se interpreta como la probabilidad de la misma. El método de random forest aporta principalmente **estabilidad**. La definición de la solución propuesta por los *random forests* se desajusta de la muestra de datos utilizada para generar el algoritmo, en comparación con el ajuste de un árbol individual simple.

Cada uno de los árboles que forman parte del bosque ha sido diseñado automáticamente para ajustarse a un escenario diferente. Esto aporta coherencia al random forest y capacidad de ajustarse adecuadamente a nuevos escenarios desconocidos. Los árboles de decisión de manera individual son altamente ajustables, pero este problema se reduce con el uso de este algoritmo.

5.2.3.5. K-Nearest Neighbor

La idea básica de K-Nearest Neighbors se ejemplifica mejor con el siguiente dicho: "Si camina como un pato, grazna como un pato y parece un pato, entonces probablemente

sea un pato". Explica la idea de que la clase de una instancia de prueba está determinada por el tipo de clase de sus vecinos más cercanos.

El clasificador K-Nearest Neighbors representa cada ejemplo como un punto de datos en un espacio d-dimensional, donde d es el número de atributos. Dado un ejemplo de prueba, se calcula su proximidad al resto de los datapoints en el conjunto de entrenamiento, usando una métrica de medición de proximidad apropiada. Los k vecinos más cercanos dado un *vecino z*, se refieren a los k-puntos más cercanos al *punto z*. Luego, el datapoint z se clasifica en función de las etiquetas de clase de sus vecinos. Si los vecinos tienen más de una etiqueta, el datapoint, se asigna a la clase mayoritaria que pertenecen sus vecinos más cercanos.

En el ejemplo de la figura 5.6 se tienen elemento de 2 clases (+ y -). En el momento que se quiere clasificar un nuevo elemento el algoritmo KNN propone considerar los elementos más cercanos a este nuevo elemento, en el caso del ejemplo, se representa con un punto.

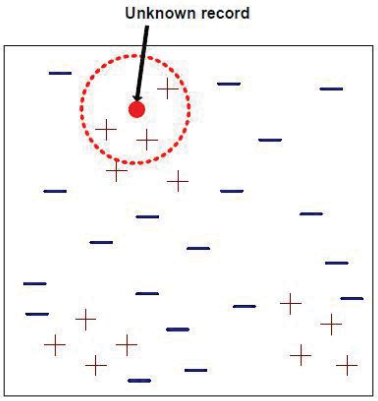

Figura 5.6: Representación de KNN.

En general, en un conjunto de datos de entrenamiento, se conoce la etiqueta de clase de otros objetos de datos. También se da el número de K antes de la clasificación. Los principales pasos para realizar la clasificación K-Nearest Neighbors son:

- Calcular la distancia desde el registro desconocido a otros registros de entrenamiento.
- Identificar los k vecinos más cercanos según la métrica de distancia
- Usar las etiquetas de clase de los vecinos más cercanos para determinar la etiqueta de clase de registro desconocido. Si hay una etiqueta de clase entre los vecinos, entonces el registro desconocido se asigna a la misma etiqueta de clase. Si los vecinos tienen más de una etiqueta de clase, entonces la etiqueta de clase de registro desconocido se determina tomando el voto de la mayoría.

Elegir un valor K correcto es importante para el rendimiento del clasificador K-Nearest Neighbor. Si el valor de k es demasiado grande, como se muestra en la figura

siguiente, el algoritmo puede clasificar incorrectamente la instancia de tes porque su lista de vecinos más cercanos puede incluir puntos de datos que se encuentran lejos de su vecindario. Por otro lado, si el valor k es demasiado pequeño, entonces el clasificador puede ser susceptible de ajuste debido al ruido en el conjunto de datos de entrenamiento.

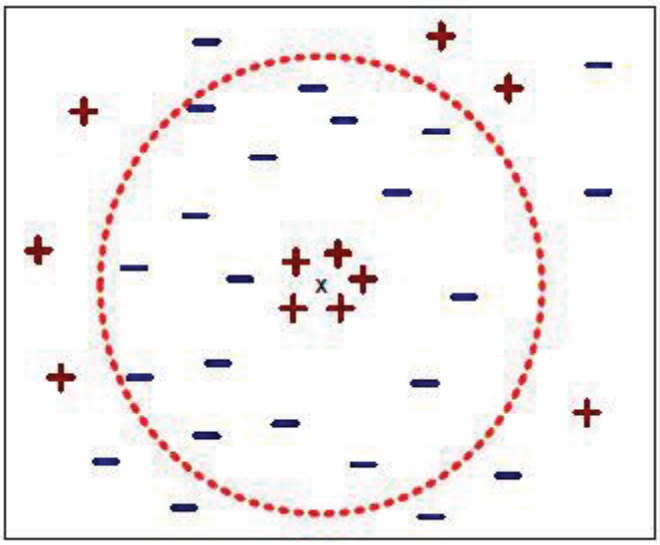

Figura 5.7: Amplificación de la vecindad

5.2.4. Algoritmos de clasificación basados en Redes Neuronales

5.2.4.1. Red neuronal secuencialmente construida

[22] Corresponde a un tipo de Red Neuronal de la Clase secuencial que es básicamente una pila lineal de capas. Para los problemas de clasificación, la salida corresponderá a los datos clasificados. Los elementos básicos que definen la red neuronal son:

- Tipo de capa:
  - Densa: capa totalmente conectada y el tipo más común de capa utilizada en los modelos de perceptrón multicapa.
  - Dropout: Aplicar dropout al modelo, estableciendo una fracción de entradas a cero en un esfuerzo por reducir el ajuste excesivo.
  - Combinada: Combinar las entradas de varios modelos en un solo modelo.
- Inicialización de peso: El tipo de inicialización utilizado para una capa se especifica en el argumento inicial. Algunos tipos comunes de inicialización de capas incluyen:
  - "Uniforme": Los pesos se inicializan a pequeños valores uniformemente aleatorios entre 0 y 0,05.

- "Normal": los pesos se inicializan a pequeños valores aleatorios gaussianos (media cero y desviación estándar de 0,05).
- "Cero": Todos los pesos se establecen en valores cero.

- Función de activación: La función de activación calcula la activación de la unidad en función de la entrada total y la activación previa, aunque en la mayor parte de los casos es simplemente una función no decreciente de la entrada total. Los tipos de función más empleados son: la función escalón, función lineal y la función sigmoidal.

5.2.5. Natural Language Processing (NLP)

[4] El Procesamiento del Lenguaje Natural es una disciplina de la Inteligencia Artificial que se ocupa de la formulación e investigación de mecanismos computacionales para la comunicación entre personas y maquinas mediante el uso de Lenguajes Naturales. Los Lenguajes Naturales son los utilizados en la comunicación humana, ya sean escritos, hablados o con signos. [18] El objetivo de NLP es ser capaz de diseñar algoritmos para permitir que los ordenadores "comprender" el lenguaje natural con el fin de realizar alguna tarea.

Se analiza la estructura del lenguaje a cuatro niveles:

- Análisis morfológico: El análisis de las palabras para extraer raíces, rasgos flexivos, unidades léxicas compuestas y otros fenómenos.
- Análisis sintáctico. El análisis de la estructura sintáctica de la frase mediante una gramática de la lengua en cuestión.
- Análisis semántico. La extracción del significado (o posibles significados) de la frase.
- Análisis pragmático. El análisis de los significados más allá de los límites de la frase, por ejemplo, para determinar los antecedentes referenciales de los pronombres.

Las distintas fases y problemáticas del análisis del lenguaje se afrontan principalmente con las siguientes técnicas:

- Técnicas lingüísticas formales: Se basan en el desarrollo de reglas estructurales que se aplican en las fases de análisis del lenguaje
- Técnicas probabilísticas: Se basan en el estudio en base a un conjunto de textos de referencia (*corpus*) de características de tipo probabilístico asociadas a las distintas fases de análisis del lenguaje.

Los modelos principales para el procesamiento del lenguaje natural son los Lógicos (gramáticas) y los Probabilísticos (basados en *corpus*). Para la resolución del problema propuesto (el cual corresponde al tipo “Clasificación”), se hará uso de técnicas probabilísticas, las cuales son usadas ampliamente en la actualidad. Esto es debido a que logran resultados más eficientes en comparación a las técnicas lingüísticas formales.

La finalidad principal del uso de NLP en los problemas de clasificación y análisis de sentimientos el texto del set de datos. La simplificación de este set de datos sirve

principalmente para aumentar el rendimiento del clasificador. Esto es debido a que los términos ingresados al sistema son más simples. Los procesos más ocupados para simplificar los textos son: eliminación de stop-words, stemming o radicación y lematización.

La eliminación de stopwords consiste en eliminar todas las palabras que poseyendo harta frecuencia son completamente irrelevantes para el sistema. Estas palabras son principalmente son conector, artículos, proposiciones, entre otros.

[31][32]Los procesos de Stemming y Lematización persiguen el mismo fin el cual es reducir formas de inflexión y, en ocasiones, formas derivadas de una palabra a una base común. Pero estos se diferencian en la complejidad del proceso realizado y en los resultados entregados. Stemming o radicación usualmente se refiere a un proceso heurístico crudo de cortar los extremos de las palabras con la esperanza de reducir la palabra a la manera más pequeña posible y, a menudo, incluye la eliminación de los afijos derivacionales. La Lematización tiene como objetivo reducir una palabra a su raíz o lema. El lema de una palabra comprende su forma básica más sus formas declinadas, la cual corresponde a su representación base encontrada en un diccionario. Por ejemplo, "informa" podría ser el lema de "información", "informaciones", e "informar". Este proceso de se lleva a cabo con el uso de un vocabulario y un análisis morfológico de las palabras. Dicho de otra manera, la lematización es el mejoramiento del proceso de radicación logrando su objetivo adecuadamente. Para ambos existen diferentes algoritmos entre los que se destacan: Porter Stemmer, Snowball Stemmer, WordNet Lemmatizer, entre otros. En la tabla 5.1 se observan palabras en inglés en su forma radicalizada y en forma de lema.

Palabra	Stemming	Lemmatisation
features	featur	feature
destruction	destruc	destroy
distribution	distri	distribute
ponies	poni	pony
verification	verifi	verify

Tabla 5.1: Ejemplo de palabras en su forma radicalizada y de lema.

5.2.5.1. Word embeddings y Vector Space Model

[19] Word Embeddings, dicho en palabras simples son los textos convertidos en números pudiendo haber diferentes representaciones numéricas del mismo texto. Para realizar esto, las palabras son llevadas a números basándose en el Modelo de Espacio Vectorial o Vector Space Model. [10] VSM es un modelo algebraico que representa la información textual como un vector, las componentes de este vector podrían representar la importancia de un término (tf-idf) o incluso la ausencia o presencia (tf) de él en un documento. Es importante señalar que el VSM clásico propuesto por Salton incorpora parámetros/información locales y globales (en el sentido de que utiliza tanto el término aislado siendo analizado, como toda la colección de documentos). VSM, interpretado en un lato sensu, es un espacio donde el texto se representa como un vector de números en lugar de

su representación textual original. En este modelo, se representa las características extraídas del documento.

Explicándolo de manera práctica se muestra el paso a paso para crear una representación VSM. El primer paso para modelar el documento en un espacio vectorial es crear un diccionario de términos presentes en los documentos. Para ello, puede seleccionar todos los términos del documento y convertirlos a una dimensión en el espacio vectorial o seleccionar todos los documentos exceptuando las palabras que no tienen relevancia para el análisis. Como ejemplo, dados las cadenas de entrenamiento:

Train Document Set:

- d1: The sky is blue.
- d2: The sun is bright.

Test Document Set:

- d3: The sun in the sky is bright.
- d4: We can see the shining sun, the bright sun.

Se crea el diccionario:

$$\mathrm{E}(t) = \begin{cases} 1, & \text{if } t \text{ is “blue”} \\ 2, & \text{if } t \text{ is “sun”} \\ 3, & \text{if } t \text{ is “bright”} \\ 4, & \text{if } t \text{ is “sky”} \end{cases}$$

Posteriormente, se elige un modelo para representar numéricamente la importancia del uso cada uso de cada palabra del diccionario en los strings de entrenamiento. Dado ambas cosas VSM entregará esta representación vectorial de los strings de entrenamiento:

$$\vec{v_{d_n}} = (\mathrm{tf}(t_1, d_n), \mathrm{tf}(t_2, d_n), \mathrm{tf}(t_3, d_n), \ldots, \mathrm{tf}(t_n, d_n))$$

Donde vdn es la representación vectorial del documento (en este caso un string) n. Cada casilla del vector contiene el valor dado por la función de valoración de la palabra respecto al documento.

Ya que se tiene una colección de documentos, ahora representados por vectores, se puede representar matricialmente como el producto cruz entre **| D |** y **|F |.** |D | es la cardinalidad del espacio del documento, o cuántos documentos tenemos y el F es el número de características, en nuestro caso representado por el tamaño del vocabulario total.

$$\begin{array}{c} \\ term\ 1 \\ term\ 2 \\ \\ term\ 300,000 \end{array} \begin{array}{c} \begin{array}{cccccccc} d_1 & d_2 & d_3 & d_4 & d_5 & d_6 & & d_{3,000,000,000} \end{array} \\ \begin{pmatrix} 0 & 0 & 1 & 0 & 1 & 0 & \cdots & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & 1 & 0 & \cdots & 0 \end{pmatrix}, \end{array}$$

[19] Las maneras de calcular la representación numérica en los componentes de un Word Embedding pueden pertenecer a 2 categorías distintas las cuales son Frequency based Embedding y Vectores Basados en predicción (Prediction based Vector).

5.2.5.2. Frequency based Embedding

Se determina el valor de numérico de una cadena de texto a través de una fórmula matemática**Term Frequency, Inverse Document Frequency (tf- idf)**

[10] Tf-Idf es uno de los modelos valoración más usadas en Machine Learning para el procesamiento de texto. La fórmula corresponde el producto cruz entre la representación matricial VSM (en este caso) usando *Term Frequency* (frecuencia de términos) por la representación matricial VSM usando *Inverse Document Frequency* (Frecuencia Inversa de documentos).

$$\text{tf-idf}(t) = \text{tf}(t, d) \times \text{idf}(t)$$

[12] Term Frequency simplemente cuenta la ocurrencia de cada palabra del diccionario en un documento. Dado el ejemplo

- d4: We can see the shining sun, the bright sun.

La representación VSM ocupando tf del documento anterior es:

$$\vec{v_{d_4}} = (0, 2, 1, 0)$$

La fórmula de Inverse Document Frequency corresponde al logaritmo de la división entre el número total de documentos del corpus y la cantidad de documentos en el cual aparece el termino *t* más 1, es decir:

$$\text{idf}(t) = \log \frac{|D|}{1 + |\{d : t \in d\}|}$$

Dado los ejemplos de los documentos de entrenamiento visto en la explicación de VSM:

- d1: The sky is blue.
- d2: The sun is bright.

El desarrollo para llegar a la representación VSM utilizando tf-idf, sería:

$$\begin{bmatrix} \mathrm{tf}(t_1,d_1) & \mathrm{tf}(t_2,d_1) & \mathrm{tf}(t_3,d_1) & \mathrm{tf}(t_4,d_1) \\ \mathrm{tf}(t_1,d_2) & \mathrm{tf}(t_2,d_2) & \mathrm{tf}(t_3,d_2) & \mathrm{tf}(t_4,d_2) \end{bmatrix} \times \begin{bmatrix} \mathrm{idf}(t_1) & 0 & 0 & 0 \\ 0 & \mathrm{idf}(t_2) & 0 & 0 \\ 0 & 0 & \mathrm{idf}(t_3) & 0 \\ 0 & 0 & 0 & \mathrm{idf}(t_4) \end{bmatrix}$$
$$= \begin{bmatrix} \mathrm{tf}(t_1,d_1) \times \mathrm{idf}(t_1) & \mathrm{tf}(t_2,d_1) \times \mathrm{idf}(t_2) & \mathrm{tf}(t_3,d_1) \times \mathrm{idf}(t_3) & \mathrm{tf}(t_4,d_1) \times \mathrm{idf}(t_4) \\ \mathrm{tf}(t_1,d_2) \times \mathrm{idf}(t_1) & \mathrm{tf}(t_2,d_2) \times \mathrm{idf}(t_2) & \mathrm{tf}(t_3,d_2) \times \mathrm{idf}(t_3) & \mathrm{tf}(t_4,d_2) \times \mathrm{idf}(t_4) \end{bmatrix}$$

$$M_{tf\text{-}idf} = M_{train} \times M_{idf} =$$
$$\begin{bmatrix} 0 & 1 & 1 & 1 \\ 0 & 2 & 1 & 0 \end{bmatrix} \times \begin{bmatrix} 0.69314718 & 0 & 0 & 0 \\ 0 & -0.40546511 & 0 & 0 \\ 0 & 0 & -0.40546511 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$
$$= \begin{bmatrix} 0 & -0.40546511 & -0.40546511 & 0 \\ 0 & -0.81093022 & -0.40546511 & 0 \end{bmatrix}$$

Finalmente, a la matriz resultante se le aplica el proceso de normalización l2 el cuya formula es:

$$M_{tf\text{-}idf} = \frac{M_{tf\text{-}idf}}{\|M_{tf\text{-}idf}\|_2}$$

5.2.5.3. Prediction Based Vector

[19] Hasta ahora, se han visto métodos deterministas para determinar vectores de palabras. Pero estos métodos demostraron ser limitados en sus representaciones de la palabra, hasta que Tomás Mitolov de Google introdujo Word2Vec a la comunidad NLP.

Estos métodos fueron basados predicción, en el sentido de que proporcionaron probabilidades a las palabras y demostraron ser el estado del arte para tareas como analogías de palabras y similitudes de palabras. También así que fueron capaces de lograr tareas como el relacionar [Rey, hombre, mujer] para lograr como resultado la palabra Reina, lo cual se consideraba un resultado casi mágico.

[20] Los modelos predictivos tratan directamente de predecir una palabra desde sus vecinos en términos de pequeños y densos vectores embebidos aprendidos (considerando parámetros del modelo). Word2Vec es un modelo predictivo particularmente computacionalmente eficiente para aprender word embeddings a partir de texto en bruto. Viene en dos sabores, el Continuous Bag-of-Words model (CBOW) y el Skip-Gram model. Algorítmicamente, estos modelos son similares, excepto que CBOW predice las palabras objetivo (por ejemplo, 'mat') desde las palabras de contexto del origen('el gato se sienta en'), mientras que el skip-gram hace la inversa y predice el contexto desde las palabras objetivo. Esta inversión podría parecer una elección arbitraria, pero estadísticamente tiene el efecto de que CBOW suaviza mucho la información distributiva (al tratar un contexto completo como

una observación). En su mayor parte, esto resulta ser algo útil para conjuntos de datos más pequeños. Sin embargo, skip-gram trata cada par de contexto-objetivo como una nueva observación, y esto tiende a hacerlo mejor cuando se tienen conjuntos de datos más grandes.

5.3. Investigaciones actuales

5.3.1. La realidad mundial

La mayoría de las investigaciones sobre Sentiment Analysis se concentran en la última década mayoritariamente, con la entrada de las redes sociales en todos los ámbitos de la sociedad. Gracias a ellas, la manera de estudiar la opinión de la gente respecto a los hechos importantes de la sociedad cambió, centrándose en estas mismas. Se observa que los datos son obtenidos casi completamente desde Twitter usando herramientas de Data Mining y Big Data debido a la gran cantidad de información disponible.

A nivel general, los estudios prácticos de opinión haciendo uso de Sentiment Analysis han estado centrado principalmente en la política y en las grandes compañías. El principal fin de estos estudios es determinar la opinión generalizada de la gente y predecir comportamientos. Los que más destacan y han sido más difundidos son los estudios respecto a la política.

Haciendo un análisis simple en los sitios especializados en documentos cientificos Google Scholar y IEEE Xplore, se puede observar un aproximado de 50 artículos respecto al estudio de elecciones políticas haciendo uso de sentiment analysis. Los artículos que tratan al tema han sido principalmente de las elecciones presidenciales del 2016. Cabe mencionar 2 artículos llamativos de libre acceso. El primero [33]“And the Winner is ...: Bayesian Twitter-based Prediction on 2016 U.S. Presidential Elections” y [34]“Analyzing Twitter Sentimentof the 2016 Presidential Candidates”. [33]El primero buscaba predecir el ganador de las elecciones primarias en EE.UU y las elecciones presidenciales, en base a tweets etiquetados positivio, negativo y neutro; y haciendo uso de un clasificador Bayesiano Ingenuo. Los resultados reportaron como ganadores de las primarias a Ted Cruz y Bernie Sanders y, finalmente, reporta como futuro presidente de EE.UU a Bernie Sander. [34] El segundo hace un estudio de las emociones expresadas acerca los candidatos, segmentados por los estados del país. Este último estudio busca mostrar que emoción es expresada mayoritariamente respecto a los candidatos presidenciales en los principales estas de USA. En el siguiente mapa (figura 5.8) se pueden observar las emociones expresadas respecto al candidato Donald Trump, simbolizada con emojis. Se observa la emoción predominante por cada estado.

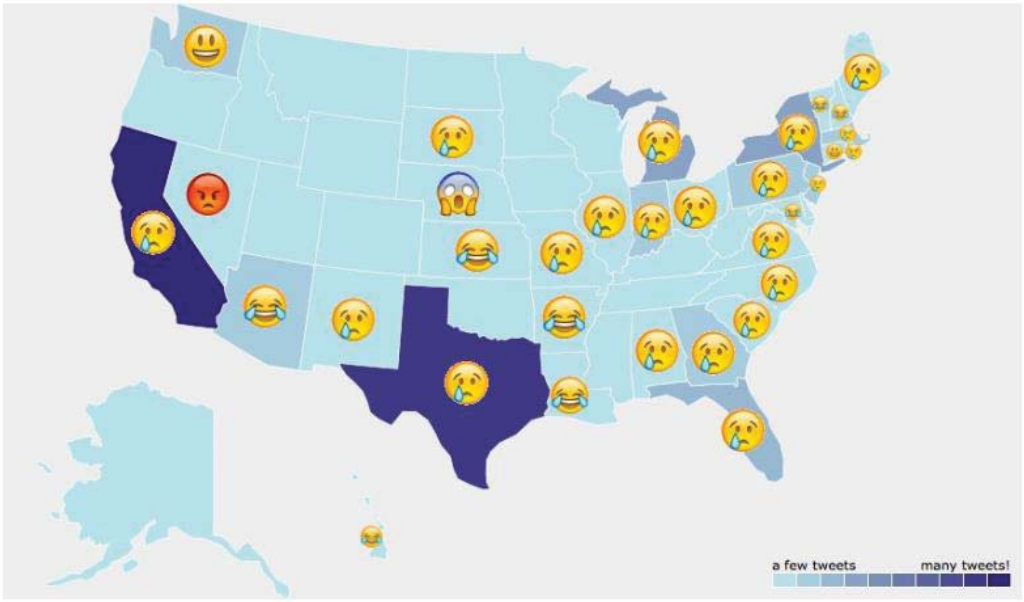

Figura 5.8 [34] Mapas de las emociones expresadas hacia Donald Trump

Sin destacar en la opinión pública, existen una buena cantidad de estudios de opinión haciendo uso de S.A. hechos a medianas y grandes compañías. Su difusión por parte de la opinión pública ha sido menor debido a la falta de interés de la opinión pública, porque estos trabajos estén restringidos para visualización bajo licencias comerciales o porque estos trabajos no han sido publicados por las empresas solicitantes. Analizando los resultados obtenidos en IEEE Xplorer se observa que la aplicación de S.A por parte de las empresas son principalmente para predecir resultados en la bolsa de valores, predecir precios, predecir comportamiento de los clientes y conocer la opinión de la gente respecto a la compañía. Entre las investigaciones de libre acceso llamativas se destaca las hechas sobre la opinión de las aerolíneas estadounidenses. [35]El gatillante de esto fueron incidentes específicos ocurridos durante Abril de 2017, dónde pasajeros fueron removidos forzosamente de los vuelos de la aerolínea estadounidense United Airlines sin explicación alguna. Durante este proceso los pasajeros removidos resultaron con heridas leves. Este polémico acontecimiento, sumado a otros originó una serie de tweets negativos en contra de la empresa. Aquellos hechos hicieron que pasados unos meses del incidente se realizarán estudios de opinión respecto a esta aerolínea y otras. [36]El estudio de “US Airlines sentiment analysis using twitter data” haciendo uso de Sentiment Analysis y Machine Learning, determinó la opinión existente de las principales aerolíneas en EE. UU. En la figura 5.9 se puede observar la distribución de opiniones de las aerolíneas mencionadas. Las opiniones pueden ser positivas, negativas o neutras.

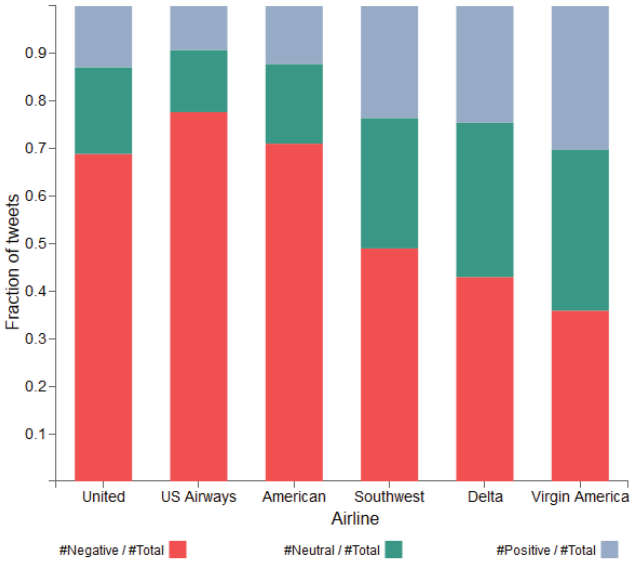

Figura 5.9: [36] Opiniones de las principales aerolíneas estadounidenses

Otra investigación llamativa corresponde a [37]“Location Based Association of Customers’ Sentiments and Retail Sales”. Este estudio sacó por conclusión que hay una relación directa posible entre una alta volumen de ventas en los locales de Walmart y una opinión positiva respecto a ellos. En la figura 5.10 se puede visualizar la situación descrita la cual muestra la relación entre cantidades de ventas y porcentaje de opiniones positivas de distintitos locales de Walmart alrededor de EEUU. Los círculos grises representan el porcentaje de opiniones positivas y los naranjos cantidades de ventas.

Figura 5.10: [37]: Relación entre ventas y opiniones de locales Walmart.

5.4. La realidad chilena

En Chile existen una cantidad pequeña de artículos de investigación de acceso público respecto al análisis de sentimientos. Aunque esta cantidad va en aumento cada año. En base

a lo observado al repositorio de documentos de la Universidad de Chile, a simple vista, la mayoría de los artículos proponen herramientas generales o la utilización de distintos enfoques para el mejoramiento de sentiment analysis. Ente los papers que proponen la utilización práctica de Sentiment Analysis se destaca [38]“ANÁLISIS DE SENTIMIENTOS Y PREDICCIÓN DE EVENTOS EN TWITTER”. Lo expresado en este documento propone un método que usando un diccionario de palabras valoradas sentimentalmente y machine learning, busca predecir el resultado de las elecciones presidenciales primarias realizadas en la Alianza por Chile el año 2013 entre los candidatos Andrés Allamand de Renovación Nacional (RN) y Pablo Longueira del partido Unión Demócrata Independiente (UDI). [38]El algoritmo propuesto determina como ganador a Pablo Longueira (UDI) por sobre Andrés Allamand (RN) con un 53% de preferencia mientras que en las elecciones en urnas realizadas en Julio de 2013 en Chile el resultado fue de un 51% sobre 49% a favor de Longueira, lo cual da un error de un 2%, lo que implica que el análisis realizado fue capaz de predecir, con un cierto margen de error, lo que sucedió en las elecciones efectivamente.

Respecto al análisis de sentimientos para determinar la opinión de empresas chilenas, siendo el principal objetivo a perseguir en esa investigación, se destaca hecho dentro de la escuela de informática de la PUCV denominado [39] “Clasificación Automática del Sentido de los Mensajes en Twitter: Comparando Entrenamiento específico y contextual”. Este trabajo plantea 4 hipótesis y están buscan la mejor forma que se pueden utilizar los datos para obtener buenos resultados en la clasificación automática sentimental. La primera hipótesis es que si se clasifican marcas de un determinado rubro, como el retail, sirve para clasificar marcas que se encuentren dentro del mismo rubro. La segunda hipótesis consiste en que la clasificación es mejor que la anterior si se realiza para cada marca independientemente. La tercera hipótesis habla sobre la clasificación de una marca por sentido serviría para clasificar otra marca dentro de la misma industria. Finalmente, la última hipótesis hace relación sobre la representación textual a utilizar, lo cual busca ver si ésta influye en la precisión de la clasificación. Para este último caso se ocupan diferentes métodos de representación textual para la hipotesis 1, 2 y 3. Las representaciones textuales propuestas corresponden a unigrama, bigrama, TF-IDF y TF-RFL. Los clasificadores usados fueron: Naive Bayes, SVM y árboles de decisión. En la tabla 5.2 se observan la totalidad de los resultados obtenidos para la métrica F1:

	F1 / HIPOTESIS 1			F1 / HIPOTESIS 2						F1 / HIPOTESIS 3						
	NB	SVM	J48	NB	SVM	J48	NB	SVM	J48	NB	SVM	J48	NB	SVM	J48	PROMEDIO
UNIGRAMAS	71,6	82,6	69	71,9	77,9	68,1	75,7	81,3	73,8	63,6	61,3	63,5	66	71	57,1	70,29
BIGRAMAS	69,1	80,9	62,3	70,7	75,5	63,9	74,7	75	76,1	56	63,4	52,4	62,8	66,5	62,1	67,43
TRIGRAMAS	70,9	78,7	56,1	68	68,7	55,4	68,4	82,2	70,4	60,5	63,6	51,4	63,3	66,7	59,9	65,61
TF-IDF	71,6	82,6	68,6	71,9	77,9	68,1	75,8	81,2	73,8	63,5	61,3	63,5	65,9	71	57,1	70,25
TF-RFL	94,8	91,9	98	95,6	92,1	95,1	96,1	89,2	99,5	57,2	65,5	57,3	72,5	65,8	70,8	82,76

Tabla 5.2: Resultados F1 de las pruebas.

La investigación concluyo es la representación textual es un factor determinante, que mejora el éxito de clasificador.

5.5. Herramientas disponibles

La mayoría de las soluciones que ocupan Sentiment Analysis se centran en áreas específicas. Aunque, existen herramientas generalizadas desde el inicio de desarrollo de esta área que han sido de gran ayuda para los trabajos de investigación. Se destacan:

- SentiWordNet: [40]SENTIWORD NET 3.0 es un recurso léxico diseñado explícitamente para apoyar la clasificación del sentimiento y las aplicaciones de minería de opinión. SENTIWORDNET 3.0 es una versión mejorada de SENTIWORDNET 1.0, un recurso léxico disponible públicamente para fines de investigación, en la actualidad cuenta con más de 300 grupos de investigación y se utiliza en una variedad de proyectos de investigación en todo el mundo. Tanto SENTIWORDNET 1.0 como 3.0 son el resultado de anotar automáticamente todos los sintonizadores de WORDNET según sus grados de positividad, negatividad y neutralidad. SENTIWORDNET 1.0 y 3.0 difieren (a) en las versiones de WORDNET que anotan (WORDNET2.0 y 3.0, respectivamente), (b) en el algoritmo utilizado para anotar automáticamente a WORDNET, que ahora incluye (además del paso de aprendizaje semi-supervisado anterior)) al paso de caminar al azar para refinar los puntajes. Su uso más avanzado es de pago
- SentiStrength: Según sus creadores, esta herramienta estima la fuerza del sentimiento positivo y negativo en textos breves, incluso para el lenguaje informal. Tiene precisión a nivel humano para textos cortos de redes sociales en inglés, excepto textos políticos. SentiStrength informa dos fortalezas de opinión negativas (de -1 a -5) y positivas de (+1 a +5). SentiStrength también puede informar resultados binarios (positivos / negativos), trinarios (positivos / negativos / neutros) y de una sola escala (-4 a +4). SentiStrength fue desarrollado originalmente para inglés y optimizado para textos generales de redes sociales cortas, pero se puede configurar para otros lenguajes y contextos.
- Sentiment140: Es una avanzada aplicación web que permite saber polarmente (negativo, positivo, neutro) la valoración sentimental respecto a una marca comercial. Muestra graficas e información estadística al respecto. Es de código abierto y está disponible para hacer análisis también en español

6. Desarrollo de la Solución

Dada la naturaleza del trabajo, se usarán las técnicas contemporáneas para el Análisis de Sentimientos o Sentiment Analysis, las cuales están basadas en Machine Learning con enfoque estadístico. Esto será apoyado por el uso de NLP. La implementación del sistema estará basado en el uso de bibliotecas y herramientas propias de *Data Science*.

Para la solución de la problemática descrita se propone el uso del análisis de opinión sobre mensajes publicados en la red social de Twitter, que cuenten con características de las emociones colectivas, un modelo de valoración y de estimación.

6.1. Descripción de datos a utilizar en este trabajo

El set de datos corresponde a 2500 tweets que expresan la opinión respecto a la empresa Fruna o sobre sus productos. Estos fueron clasificados manualmente por un grupo de 10 personas entre 20 y 30 años las cuales se encuentran trabajando o realizando estudios universitarios. El conjunto de datos es dividido para primeramente realizar la fase de entrenamiento y luego usando la porción restante, se utiliza para la fase de pruebas. En la tabla 6.1 se observa un extracto del conjunto de datos.

emotion	id	date	user	text
1	8,59E+17	30-04-2017 21:30	_despreocupado	Fruna porque tan mala clase con sus trabajadores ? Ojalá la cierren
1	8,59E+17	30-04-2017 21:39	Josmica	Ay, no te creo lo del Tío Fruna
1	8,59E+17	30-04-2017 23:15	Fenomen0ide	Evolution versión fruna #PaybackCL
1	8,59E+17	30-04-2017 23:58	garci1973	@cathybarriga alcaldesa, la semana pasada en alimentos Fruna hubieron dos hechos lamentables de muerte y nadie dijo nada puede ud averiguar?
1	8,59E+17	01-05-2017 0:26	joakyty	Porque no salio en las noticias que ayer murieron 2 trabajadores de Fruna !!!?
1	8,59E+17	01-05-2017 10:01	carothiare28	Ministra @akraussvalle hagase presente con el trabajador muerto en fruna se suicida por hostigamiento laboral !!!!
1	8,59E+17	01-05-2017 10:03	carothiare28	Así es !!! Pero existen muchos empleadores explotadores ejemplo el trabajador que se suicidó en la empresa fruna por hostigamiento laboral
1	8,59E+17	01-05-2017 12:48	CNNChile	Bárbara Figueroa: "Nuestras condolencias profundas a las familias de los dos trabajadores muertos este fin de semana en la empresa Fruna "
1	8,59E+17	01-05-2017 13:07	HectorLigarius	@FrunaChile Lamento mucho lo sucedido en Maipu con 2 de sus trabajadores,si fue hostigamiento laboral,nunca + consumo productos #fruna
1	8,59E+17	02-05-2017 8:00	MadreSinEstadio	Últimamente Fruna la gente quiere demostrar que comen Fruna para que se sientan pueblo. Obviamente sus dueños felices y se cuelgan de eso.
1	8,59E+17	02-05-2017 10:58	SataFLASH	Fruna irá a sacar un nuevo producto? Tabletones "Finadito" o algo así?
2	8,59E+17	02-05-2017 11:37	ellocomustafa	un yupi es muy poca cosa te invito una inkacola y una selva negra de fruna Lo mejor para mi reina
1	8,59E+17	02-05-2017 11:54	FeloSalazar22	Lo único bacán de Fruna son sus productos, porque como la empleadores son asquerosos.
1	8,59E+17	02-05-2017 12:00	santiheroe	Y Fruna se pronunció sobre la acusación de hostigamiento laboral denunciada por familia de uno de sus trabajadores fallecidos en la fábrica?
0	8,60E+17	02-05-2017 20:19	MarcoPaolo	Que pasó por que odian a fruna
1	8,60E+17	02-05-2017 21:55	N3S3P4S	Noesnalaferia sacando una carta sobre la muerte de los trabajadores de Fruna , después de vivir haciendo publicidad gratis.
1	8,60E+17	02-05-2017 22:13	don_digger	Lustros que no comía tableton y debo decir una cosa: fruna te estay cagando entero con el chocolate
1	8,60E+17	02-05-2017 22:13	ffuentes_	@rstuven Richard Sandoval y Fruna juntos debe ser alto riesgo de cáncer

1	8,60E+17	02-05-2017 22:15	hueonsnob	no tengo simpatía por fruna pero noesnalaferia/gamba es más malo, period.
0	8,60E+17	02-05-2017 22:17	ampiciliina	Toda la gente que le chupaba el pico al loco de fruna anda escondida jajajajaj

Tabla 6.1: Ejemplos de tweets clasificados en el set de datos

Los tweets pueden ser clasificados de la siguiente manera:

- Neutro (0): No existe una opinión positiva o negativa respecto a la empresa. En este caso normalmente se hace mención a la empresa porque se compró un producto de ella recientemente, sin emitir un juicio de valor respecto a ella. Se comparan hechos positivos y negativos respecto a ella, sin llegar a una conclusión valorativa.
- Negativo (1): Se observa una opinión mala respecto a la empresa. En opiniones de esta categoría se observa que algún producto o varios de la empresa son malos, muchas veces en comparación a otro criterio positivo como los bajos precios de esta. También que las condiciones laborales son malas, pero se disminuye el efecto de esta afirmación diciendo que es un común en la industria de la confitería de este país.
- Positivo (2): Se observa una opinión positiva respecto a la empresa y sus productos. Se dice que estos son baratos y/o sabrosos. En esta clase de opiniones también se observan las que hacen defensa a esta, respecto a las noticias negativas que hay respecto a ella.

6.2. Funcionamiento del algoritmo

El funcionamiento del sistema cuenta con las siguientes etapas, las que se muestran gráficamente en la figura 6.1. Estas etapas se muestran a continuación:

- La extracción de los tweets desde la Base de Datos de Twitter.
- Pre-procesamiento. Se realiza la purificación de los datos, de tal manera que la información extraída sea representativa, es decir, aporte contenido al proceso de análisis semántico y así no se generen errores.
- Clasificación manual. Los datos extraídos son clasificados manualmente en una categoría adecuada según el criterio correspondiente. Este proceso es realizado por un grupo de personas.
- Procesamiento de Lenguaje Natural. Los datos se procesan usando NLP para obtener mejores resultados. Las palabras extraídas del texto se simplifican a través del proceso de lemmatization o stemming Finalmente, los datos se llevan a una representación relevante para el sistema. Esta fase dará como salida el conjunto de datos de entrenamiento y pruebas.
- Entrenamiento del modelo. Se ingresan los datos de entrenamiento para que sean procesados y considerados por el sistema. Se utilizan 3 clasificadores distintos para esto: Naive Bayes Multi-class, Random Forest y K-Nearest Neighbor.

- Prueba del modelo. Se ingresa el set de pruebas. Se usan los clasificadores obtenidos en la fase de entrenamiento, dando como resultado una clasificación, la cual se compara con la etiqueta asignada, obteniendo así en las métricas correspondientes. Se entregan estadísticas del funcionamiento de cada uno.

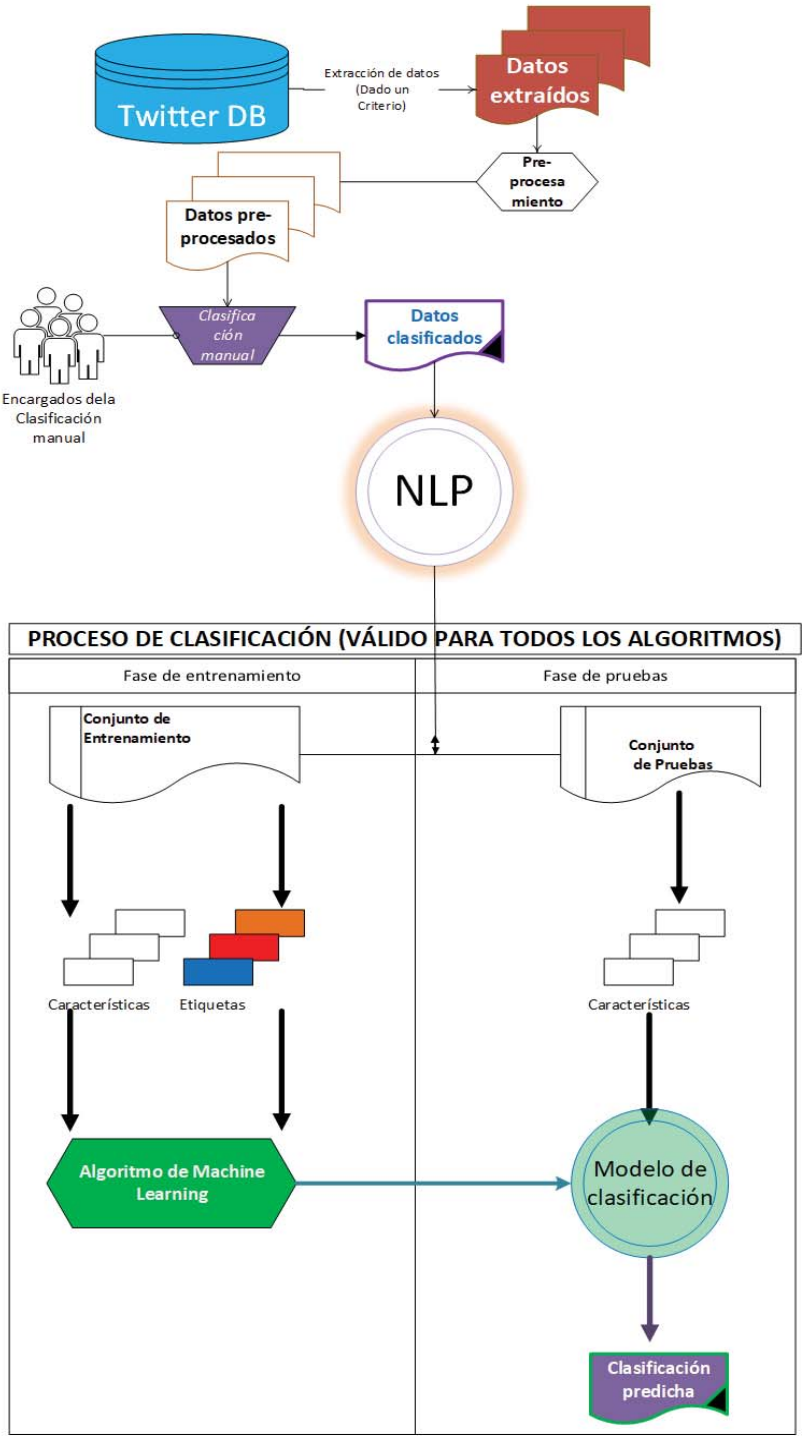

Figura 6.1. Diagrama de funcionamiento del sistema

6.2.1. Extracción de tweets

La extracción de datos es la primera etapa del funcionamiento del sistema. Los datos corresponden directamente a los tweets devueltos que coinciden con ciertos criterios de búsqueda, los cuales en este caso son palabra clave "Fruna" y un rango de fechas determinado. La mayoría de las opciones disponibles está condicionada al uso de Interfaces de Programación de Aplicaciones (API). Estas interfaces dan acceso a un conjunto de subrutinas, funcionalidades o métodos abstraídos en forma de bibliotecas que permite la distribución y su uso por distintos programas.

Por otra parte, existen bibliotecas desarrolladas que trabajan sobre la API de Twitter y usan funciones adicionales que incluso mejoran la funcionalidad de esta. Es el caso de GET-OLD Tweets. La razón para seleccionar esta biblioteca es debido a que permite extraer una alta cantidad de tweets dado un rango de fechas. Esto representa una gran ventaja respecto a su competidor Tweepy, la cual solo puede extraer tweets desde hace 2 semanas desde la fecha actual.

6.2.2. Sanitización automática de los datos

Corresponde a la segunda etapa. Haciendo un breve análisis, se observó una gran cantidad de registros con contenido poco relevante que interferían con su procesamiento y la obtención de resultados. Se detectaron patrones comunes entre estos y en base a aquello se hizo un filtrado automático de registros irrelevantes.

Se suprimen los hipervínculos de los tweets que poseen imágenes y videos. Estos no aportan ninguna información sobre el proceso de reconocimiento emocional en el texto, ya que dicha información se encuentra codificada haciéndola ilegible para el sistema. En segundo lugar, se suprimieron tweets en otros idiomas los cuales no se referían a la empresa bajo ninguna circunstancia. Finalmente se suprimieron tweets que estaban repetidos de manera exactamente igual. Esta etapa disminuyó la cantidad de tweets aproximadamente a la mitad y estos fueron almacenados en un nuevo archivo csv.

6.2.3. Clasificación y filtrado manual de datos

En esta etapa un grupo de personas clasificó los tweets según la opinión que se expresara respecto a la empresa. Este grupo consta de 10 personas entre 20 y 30 años que estén estudiando en la universidad o se encuentren trabajando. Los tweets son clasificados de acuerdo a los criterios descritos anteriormente.

Mientras se clasifican los tweets, se hace en conjunto una eliminación manual de tweets que no son relevantes para la investigación o que derechamente no se relacionan en nada con esta. Entre estos se cuentan principalmente:

- Tweets donde la valoración emocional expresada es poco clara o se contradice constantemente.

- Tweets donde se pregunta qué pasa con la empresa, sin emitir juicio de valor alguno.
- Tweets donde se menciona la empresa solamente para un hecho determinado, sin expresar la opinión de esta.
- Tweets en otros idiomas.
- Tweets donde se usa la palabra “Fruna”, pero en realidad se está refiriendo a otra cosa que no es la empresa en sí misma. Esto se da en otros países de habla hispana.

6.2.4. Procesamiento de Lenguaje Natural

Esta etapa agrupa el conjunto de procesos que abarca el post-procesamiento de los datos, eliminar palabras y caracteres que no son relevantes para el sistema y llevar los datos a una representación apropiada. La figura 6.2 muestra todas las fases que involucra esta etapa.

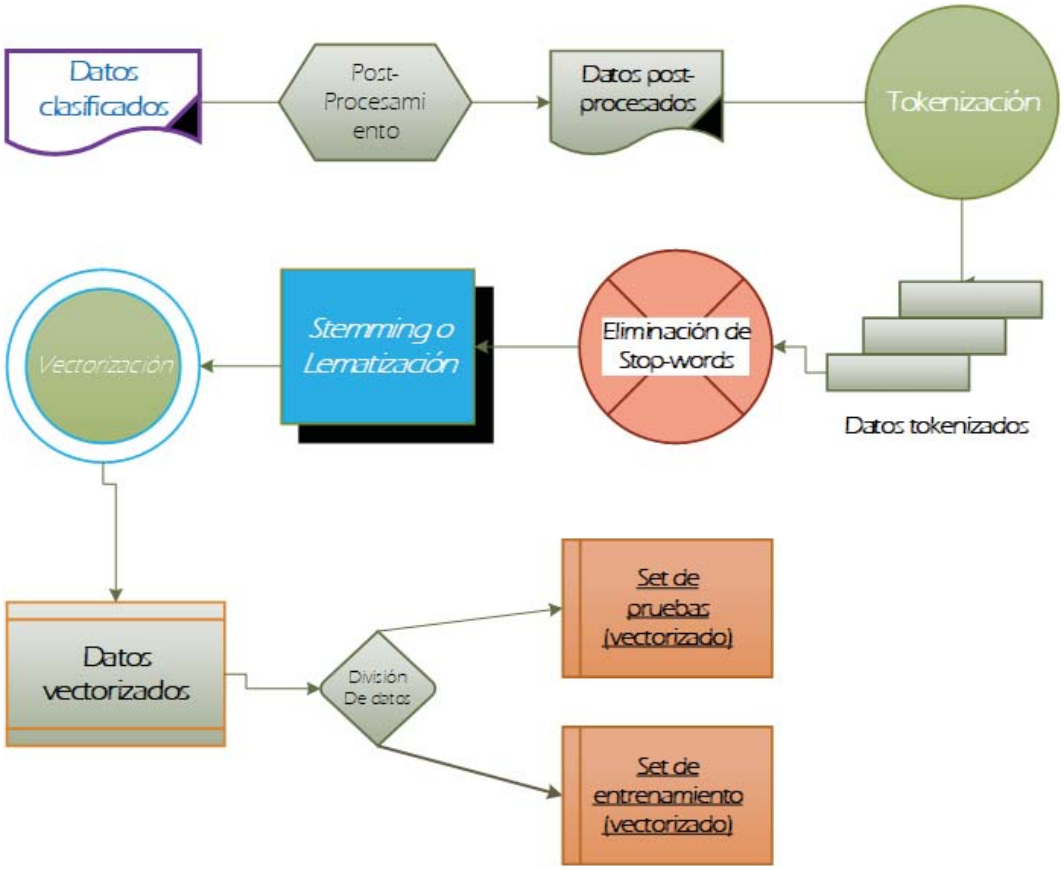

Figura 6.2. Diagrama de “Procesamiento de Lenguaje Natural”

6.2.4.1. Post-Procesamiento y Tokenización

El set de datos clasificados es ingresado al sistema. Se seleccionan las columnas relevantes para este desde el archivo csv. Estas son la columna *"text"*, la cual es la que tiene el texto del tweet y la columna *"emotion"* que tiene la clase/etiqueta de este. Sobre la columna text es donde se hará el postprocesamiento y todo el trabajo de NLP. Posteriormente se le vuelve a hacer un filtro automático a esta, eliminando los tweets que contengan hashtags, menciones y enlaces que no hayan sido filtrado antes. El texto se tokeniza, es decir, cada tweet se transforma en una matriz que contiene cada palabra de este.

6.2.4.2. Eliminación de Stop-words

Las stop-words son palabras que no son importantes para el sistema y que tienen una alta frecuencia de aparición en el texto a ser procesado. Su inclusión afecta el rendimiento del clasificador, entregando resultados erróneos. Ejemplos de stop-words en español: "a", "te", "me", "tu". En esta fase se eliminan estas optimizando el procesamiento del texto por parte del sistema

6.2.4.3. Stemming y Lemmatization

Las palabras del texto tokenizado restantes son simplificadas a través de un proceso de Stemming o Lemmatization. Estos procesos buscan llevar las palabras tokenizadas a su forma más simple posible a través de 2 maneras distintas. Para la ocasión ha sido elegido el *Snowball Stemmer* de la librería NLTK. En la tabla 6.2 se puede observar el antes y después del texto tokenizado al aplicarle Stemming.

Texto tokenizado	**Texto radicalizado (stemmed text)**
[Fruna, tan, mala, clase, trabajadores, Ojalá,...	[frun, porqu, tan, mal, clas, con, sus, trabaj...
[Ay, creo, Tío, Fruna]	[ay, no, te, cre, lo, del, tio, frun]
[Evolution, versión, fruna, PaybackCL]	[evolution, version, frun, paybackcl]
[alcaldesa, semana, pasada, alimentos, Fruna, ...	[alcaldes, la, seman, pas, en, aliment, frun, ...
[Porque, salio, noticias, ayer, murieron, trab...	[porqu, no, sali, en, las, notici, que, ayer, ...
[Ministra, hagase, presente, trabajador, muert...	[ministr, hag, present, con, el, trabaj, muert...
[Así, Pero, existen, empleadores, explotadores...	[asi, es, per, exist, much, empleador, explot,...

Tabla 6.2: Texto tokenizado vs Stemmed text

6.2.4.4. Vectorización

La matriz de palabras por sí sola no representa nada para la máquina, por lo cual debe llevarse a una representación numérica. Es por esto que se convierte a una representación numérica basada en el modelo de espacio vectorial.

En esta etapa el conjunto de palabras es llevado a una matriz vectorial. La matriz esta compuesta por filas las cuales representan el tweet convertido. La fila es un vector dividido en casillas, donde cada casilla representa el valor numérico de una palabra del texto. Esta conversión numérica se realiza utilizando una determinada función de valorización la cual, en este caso, es TF-IDF. Ahora la totalidad del set de datos esta convertido en una representación relevante para el sistema. En la figura 6.3 se ve la representación numérica del set de datos gracias a la función TF-IDF.

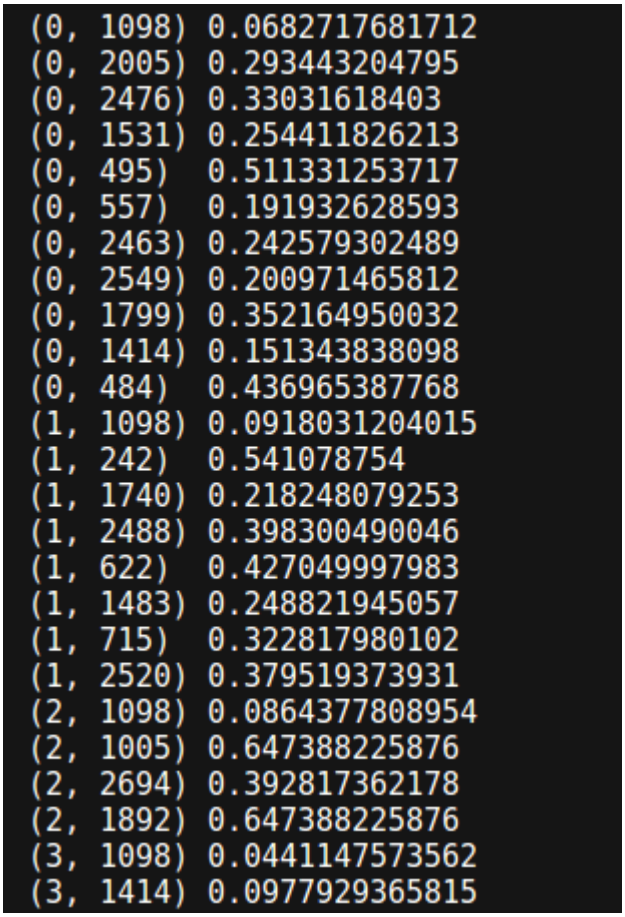

Figura 6.3: Representación TF-IDF

Finalmente, la totalidad del set de datos se divide entre el set de datos de entrenamiento y el de prueba.

6.2.5. Etapa de entrenamiento

Se extraen las características relevantes para el sistema, las cuales en este caso corresponden al texto del tweet y la clase a la cual pertenece cada uno. Ambas son ingresadas a la máquina. Usando un algoritmo determinado y después de un número finito de iteraciones, la máquina queda entrenada y se genera un modelo de clasificación. En este caso, se generan múltiples modelos de clasificación los cuales serán usados posteriormente. En la figura 6.4 se observa la etapa de entrenamiento.

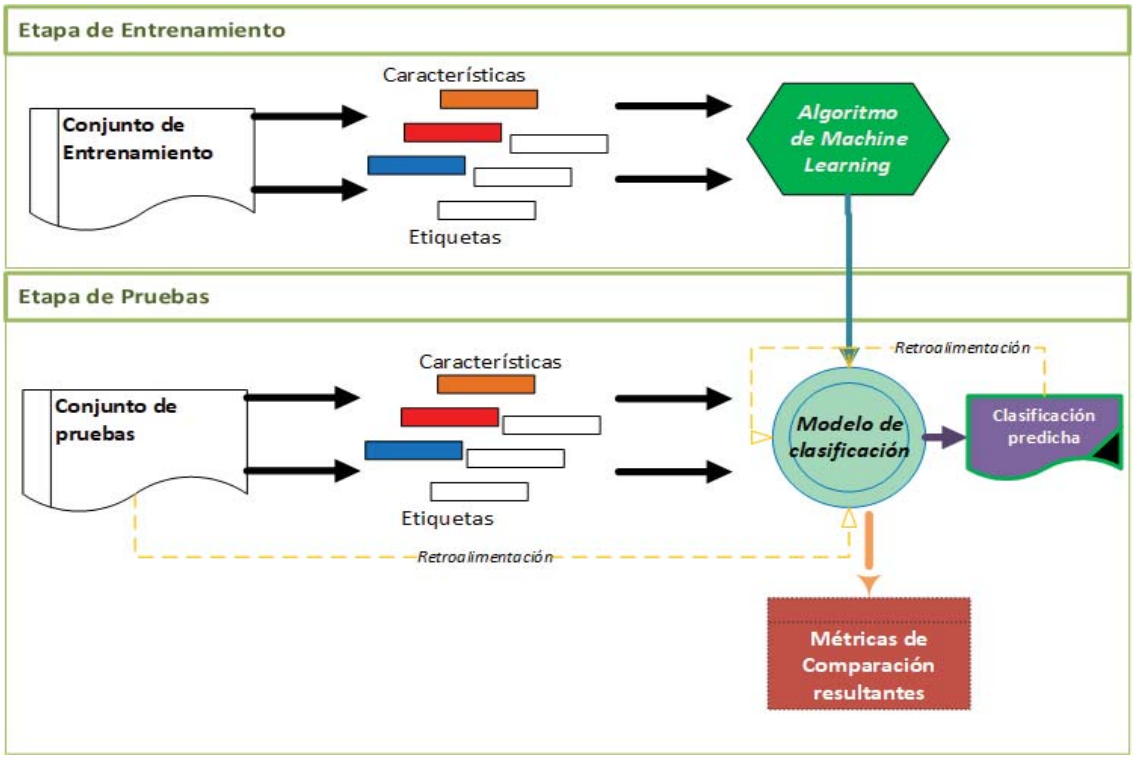

Figura 6.4: Diagrama de las etapas de entrenamiento y pruebas

6.2.6. Etapa de pruebas

Corresponde a la última etapa de la solución propuesta para resolver el problema. En primer lugar, se ingresa el set de pruebas. Se hace una predicción clasificando emocionalmente los tweets ingresados, haciendo uso del modelo generado en la anterior fase. Esta predicción se compara con la etiqueta la cual ha sido asignada previamente. Esta comparación genera métricas al respecto mostrando que tan eficiente y exacta fue la predicción hecha por el algoritmo clasificador. En la figura 6.4 se observa la etapa de pruebas

Dado que en la solución propuesta hay varios modelos clasificadores, se lleva a cabo este proceso para cada uno de ellos. Los modelos presentados ya anteriormente se describen en la tabla 6.3.

Modelo	Principales características	Aspectos de su configuración
Multinomial Naive Bayes	Versión mejorada del "Ciego Bayesiano", especialmente diseñada para problemas de clasificación multiclase.	Suavizado de parametros alpha=1. Posee un parámetro llamado “Prioridad de clase”, el cual se puede ajustar para darle mayor importancia a una clase sobre otra. Este es especialmente útil para set de datos desequilibrados. Si la cantidad de datos de una clase es baja, se le da mayor prioridad para que sea tomado en cuenta mejor.
Árbol de Decisión	Se plantea una serie de preguntas sobre los atributos del registro de prueba, las cuales están jerárquicamente organizados en un árbol. Cada vez que se recibe una respuesta, se realiza una pregunta de seguimiento hasta que se llega a una conclusión sobre la etiqueta del registro.	Se usa un árbol de tipo ID3 que utiliza el criterio de entropía para la medición de ganancia de información
Random Forest	Se ocupa una serie de árboles de decisión generados aleatoriamente. Es seleccionado automáticamente el árbol y camino que entregue una clasificación lo más precisa posible	Posee 403 árboles posibles a generar y recorrer. El número de procesos en paralelo para que funcione equivale al número de núcleos del procesador. Siendo estos equivalente a 4.
K-Nearest Neighbor	Este es un método de clasificación no paramétrico que estima el valor de la f la probabilidad a posteriori de que un elemento pertenezca a una clase. Esto lo logra usando el algoritmo k-nn basado en un entrenamiento mediante ejemplos cercanos en el espacio de los elementos	Se usan 5 vecinos. El algoritmo más eficiente para computar esos vecinos más cercanos se elige automáticamente. Minowsky es la métrica de distancia entre cada vecino, con el "número de poder" = 2.

Tabla 6.3: Clasificadores elegidos

6.3. Métricas de resultados usadas

Para el cálculo de cada métrica se utilizan las siguientes variables:

- Vp: Verdaderos Positivos
- Vn: Verdaderos Negativos
- Fp: Falsos Positivos
- Fn: Falsos Negativos

6.3.1. Exactitud (accuracy)

La exactitud hace referencia a cuán cerca del valor real se encuentra el valor efectivamente medido. En términos estadísticos, la exactitud está relacionada con el sesgo de una estimación. Cuanto menor es el sesgo, más exacta es una estimación. Cuando se expresa la exactitud de un resultado, se expresa mediante el error absoluto que es la diferencia entre el valor experimental y el valor verdadero. En el caso de un problema de clasificación indica de porcentaje de registros que fueron correctamente clasificados del total de registros. Matemáticamente hablando:

$$\mathbf{exactitud} = \frac{\mathbf{VP} + \mathbf{VN}}{\mathbf{VP} + \mathbf{FP} + \mathbf{FN} + \mathbf{VN}}$$

6:1: Fórmula de exactitud

6.3.2. Precisión

La precisión es la cantidad de verdaderos positivos dividida por la cantidad de verdaderos positivos y falsos positivos. Dicho de otra forma, es el número de predicciones positivas dividido por el número total de valores de clase positivos predichos. También se llama Valor Predictivo Positivo (VPP). La precisión se puede considerar como una medida de la certitud de un clasificador. Una baja precisión también puede indicar una gran cantidad de falsos positivos. Matemáticamente hablando se tiene:

$$Precision = \frac{Vp}{Vp + Fp}$$

6.3.3. Exhaustividad (Recall)

La exhaustividad es el número de Verdaderos Positivos dividido por el número de Verdaderos Positivos y el número de Falsos Negativos. Dicho de otra forma, es el número de predicciones positivas dividido por el número de valores de clase positivos en los datos de prueba. También se llama Sensibilidad o Tasa positiva real. La exhaustividad puede considerarse como una medida de la completitud de los clasificadores. Una exhaustividad baja indica muchos falsos negativos. Matemáticamente hablando se tiene:

$$Recall = \frac{Vp}{Vp + Fn}$$

6.3.4. Puntaje F1 (F1 Score)

El Puntaje F1 es el promedio armónico entre la precisión y la exhaustividad y transmite el equilibrio entre ambos elementos. Su mejor valor es 1 y el peor 0. Matemáticamente hablando se tiene:

$$F1 - Score = \frac{2 * (Precision + Recall)}{(Precision + Recall)}$$

7. Implementación

La implementación fue hecha completamente en Python en su versión 2.7. Se descubrió que esa versión es más amigable con el desarrollo ocupando Machine Learning. Python en su base es un lenguaje simple y fácil de aprender, agregándole las bibliotecas adecuadas, este lenguaje se convierte en una herramienta tremendamente potente al servicio de las ciencias de la computación, teniendo a su disposición muchas bibliotecas gratis con diversas funcionalidades. A continuación, se detalla la implementación del sistema.

7.1. Librerías usadas

- **Get Old Tweets**: es un proyecto el cual está publicado en la plataforma GitHub y es de libre acceso y uso. Tiene la capacidad de extraer tweets de cualquier fecha y lugar sin límite alguno. Esto en contraste a su principal competidor Tweepy que usa la API oficial de Twitter, la cual limita la obtención de Tweets emitidos a un plazo no más allá de una semana. Cabe mencionar que los datos extraídos desde GetOldTweets son exactamente los necesitados para el desarrollo de la solución. Este proyecto cuenta con una biblioteca principal generada por desarrollador llamada got, la cual es la columna vertebral de este Software.
- **UnicodeCSV:** Esta librería permita que los tweets a medida que se obtienen son decodificados en UTF-8 y son guardados en un archivo *CSV*.
- **Pandas**: Pandas es una biblioteca de código abierto, con licencia BSD que proporciona estructuras de datos de alto rendimiento y herramientas de análisis de datos. En el caso de estudio proporciona la funcionalidad para trabajar.
- **Sklearn o scikit-learn:** Es una biblioteca con licencia BSD que cuenta con herramientas de todo tipo para su uso en las ciencias de la computación. Entrega principalmente algoritmos clásicos de M.L. Esta se encuentra basada en las bibliotecas SciPy y NumPy.
- **TensorFlow™:** Es una biblioteca para Python desarrollada por Google para su uso en Machine Learning, especialmente enfocado para su uso en los modelos probabilísticos neuronales. Posee un alto rendimiento y puede ser usado para set de datos enormes y la resolución de problemas complejos. Cuenta con la versión clásica que solo hace uso de la CPU del pc para trabajar y una versión que hace uso de la GPU del pc para un mejor rendimiento y problemas de alta envergadura. Funciona mejor en Ubuntu ya que en este S.O. Tensorflow GPU puede trabajar con Python 2.x y 3.x; en cambio en Windows solo trabaja sobre Python 3.x
- **GenSim:** Basada en Tensorflow. Proporciona tecnologías avanzadas de NLP como por ejemplo el uso Word2Vec y Doc2Vec.
- **NLTK:** Es una biblioteca que proporciona funcionalidades clásicas de NLP.

7.2. Desarrollo

7.2.1. Extracción de datos

Usando la librería GetOldTweets se obtienen un total aproximado de 7000 tweets que contienen la palabra “Fruna”, durante el periodo de 1 de Mayo del 2017 y 30 de Julio de 2017. Los tweets puros son guardados en un archivo csv, sin ser procesados. En la figura 7.1 se puede observar la parte principal del código para esta etapa.

```
max_tweets = 10000
def extractor (fecha_inicio,fecha_final):
    tweetCriteria = \
    got.manager.TweetCriteria().setSince(fecha_inicio)\
        .setUntil(fecha_final).setQuerySearch("fruna")\
        .setMaxTweets(max_tweets)
    csvFile = open('dataset_tweets_fruna_10k.csv', 'w')
    csvWriter = csv.writer(csvFile, delimiter='|')
    tweet = got.manager.TweetManager.getTweets(tweetCriteria)
    for i in range(max_tweets):

        print i
        csvWriter.writerow(['X',
                            tweet[i].id, tweet[i].date, tweet[i].username,
                            tweet[i].text])

def main():
    extractor('2017-05-01', '2017-07-30')
```

Figura 7.1: Código de extracción de datos

7.2.2. Filtrado automático de datos

Usando bibliotecas básicas de Python, se analiza cada tweet revisando si este contiene elementos que puedan interferir gravemente en la clasificación sentimental. Si se cumple esto, el tweet es removido. Entre los elementos prohibidos se encuentran los enlaces e imágenes. Se genera un archivo csv con los tweets filtrados. En la figura 7.2 se observa el código que realiza esto:

```
def word_finder(domain_to_be_removed_list, csv_string):
    flag=0
    for word_to_remove in domain_to_be_removed_list:
        # print word_to_remove
        # print csv_domain
        # print csv_domain.split()

        if word_to_remove in csv_string.split():
            flag=1
            return flag
    return flag

CSV_file = 'dataset_tweets_fruna_10k.csv'
OUT_file = 'dataset_tweets_fruna_10k_filtro1.csv'
```

Figura 7.2: Código de filtrado automático de datos

7.2.3. Ingreso de datos al sistema y procesamiento de datos

En esta etapa los datos son ingresados al sistema para ser procesados. Se leen desde el set de datos solo las columnas text y emotion, las cuales contienen el texto a ser procesado y la clasificación respectivamente. En la figura 7.3 se puede observar parte del procedimiento para realizar esto.

```
class TwitterData_Initialize():
    data = []
    processed_data = []
    wordlist = []

    data_model = None
    data_labels = None
    is_testing = False

    def initialize(self, csv_file, is_testing_set=False, from_cached=None):
        if from_cached is not None:
            self.data_model = pd.read_csv(from_cached)
            return

        self.is_testing = is_testing_set

        if not is_testing_set:
            self.data = pd.read_csv(csv_file, header=0, names=["emotion","id","date","user","text"],sep=";")
            self.data = self.data[self.data["emotion"].isin(["0","1","2"])]
        else:
            self.data = pd.read_csv(csv_file, header=0, names=["id", "text"], dtype={"id": "int64", "text": "str"},
                                    nrows=1400)
            not_null_text = 1 ^ pd.isnull(self.data["text"])
            not_null_id = 1 ^ pd.isnull(self.data["id"])
            self.data = self.data.loc[not_null_id & not_null_text, :]

        self.processed_data = self.data
        self.wordlist = []
        self.data_model = None
        self.data_labels = None
```

Figura 7.3: Código para realizar el ingreso al sistema del set de datos

7.2.4. Post-procesamiento

En esta etapa se eliminan todos los caracteres que no tengan relevancia para el sistema. En la figura 7.4 se observar el método para lograrlo y sus sub-etapas.

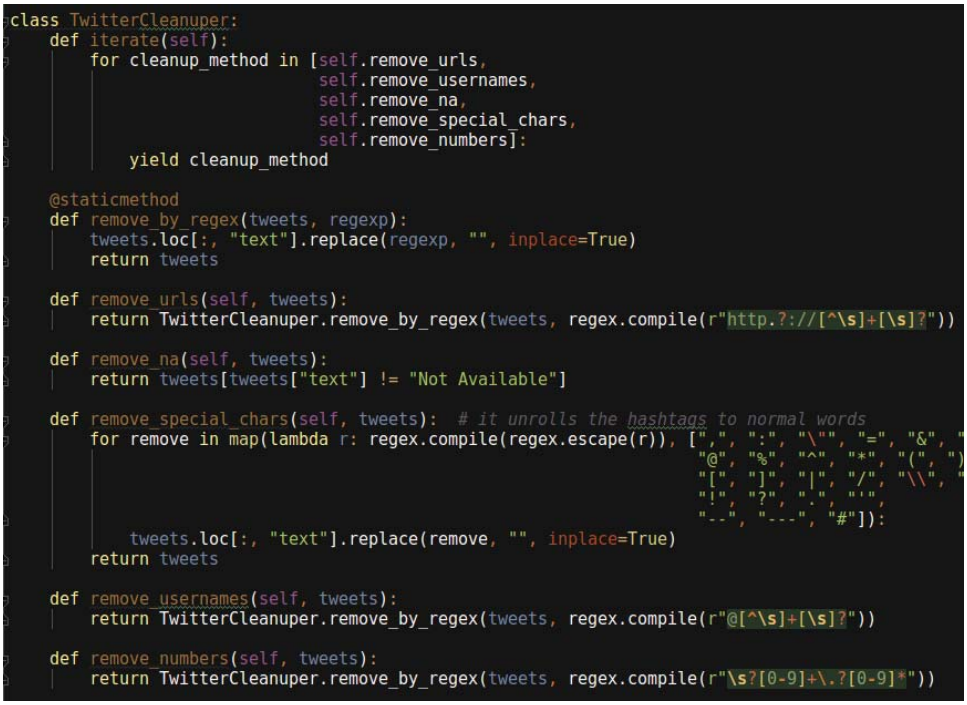

Figura 7.4: Método y funciones encargadas del post-procesamiento

En totalidad se eliminan del texto procesado:

- Los links e imágenes.
- Los tweets sin textos.
- Los caracteres especiales y hashtags.
- Las menciones y los nombres de usuario.
- Los números.

7.2.5. Tokenización y eliminación de stop words

En esta fase, cada tweet del set de datos se deja de la forma tokenizada y se le quitan las stopwords. La función de la biblioteca NLTK *tokenizer* es la encargada de tokenizar el texto. Para que una palabra sea considerada stopword, esta debe estar en la *base de datos de stopwords en español* de la biblioteca NLTK. Una vez hecho esto para cada tweet del set de datos, se finaliza esta fase. En la figura 7.5 se muestra el extracto de código encargado de lo anterior.

```
def tokenize(self, tokenizer=nltk.word_tokenize):
    def tokenize_row(row):
        row["text"] = tokenizer(row["text"].decode('utf-8'))

        row["tokenized_text"] = [] + row["text"]

        row["tokenized_text"]= [w for w in row["tokenized_text"] if w not in self.stopwords]

        return row
```

Figura 7.5: Tokenización y eliminación de Stopwords

Finalmente cabe mencionar ya que el texto procesado está en español, este poseerá caracteres del tipo de codificación utf-8. Para que esto sea correctamente entendido por NLTK, debe de ser decodificado a el mismo.

7.2.6. Stemming o lemmatization

Para realizar el proceso de stemming sobre el set de Fruna se debe usar el módulo de NLTK llamado *"Snowball Stemmer*". Este módulo cuenta con una serie de funcionalidades para una diversidad de idiomas que no son inglés. En este caso se hace uso de la funcionalidad para el idioma español, como se observa en la figura 7.6

```
def __init__(self, previous, stopwords=None, punct=None, lower=True, strip=True):
    self.processed_data = previous.processed_data
    self.lower      = lower
    self.strip      = strip
    self.stopwords  = set(stopwords) if stopwords else set(sw.words('spanish'))
    self.punct      = set(punct) if punct else set(string.punctuation)

def stem(self, stemmer=nltk.SnowballStemmer("spanish")):
    def stem_and_join(row):
        row["stemmed_text"] = list(map(lambda str: stemmer.stem(str.lower()), row["text"]))
        return row

    self.processed_data = self.processed_data.apply(stem_and_join, axis=1)
```

Figura 7.6: Implementación del uso de Snowball Stemmer.

Para realizar lematización se recurre al módulo *Word Net Lemmatizer* de NTLK, el cual está sólo disponible en inglés. En la figura 7.7 se muestra el uso de este. Para el caso de Fruna, no se ocupa este módulo.

```
def lemmatize (self, lemmatizer=nltk.WordNetLemmatizer()):
    def lemmatize_and_join(row):
        row["lemmatized_text"] = list(map(lambda str: lemmatizer.lemmatize(str.lower()), row["text"]))
        return row

    self.processed_data = self.processed_data.apply(lemmatize_and_join, axis=1)
```

Figura 7.7: Implementación del uso de Word Net Lemmatizer

7.2.7. Transformación a TF-IDF

En la figura 7.8 se muestra el código encargado de transformar el texto a su representación TF-IDF. De aquello se destaca lo siguiente:

- Con *countvectorizer* la matriz de vectores de texto tokenizado y stemizado se transforma a representación Term-frequency.
- Con TF-IDFTransformer y fit_transform la matriz vectorial es transformado a TF-IDF finalmente. Se normaliza con la función l2.

```
def vectorize(data_wordlist):
    #print data_wordlist.processed_data["tokenized_text"]

    count_vect = CountVectorizer(tokenizer=lambda doc: doc, lowercase=False,encoding='utf-8')
    X_train_counts = count_vect.fit_transform(data_wordlist.processed_data["stemmed_text"])
    print "bow: \n"
    #print X_train_counts

    tf_transformer = TfidfTransformer(use_idf=True,norm='l2').fit(X_train_counts )
    X_train_tf_idf = tf_transformer.fit_transform(X_train_counts )
   # print "TF-IDF: \n"
    #print X_train_tf
    return X_train_tf_idf
```

Figura 7.8: Método para realizar transformación a tf-idf

7.2.8. Entrenamiento y pruebas

Desde la sección de Código mostrada en la figura 7.9, las funciones necesarias son llamadas y se setean los parámetros necesarios para el entrenamiento y pruebas.

```
seed = 42
random.seed(seed)

X_train, X_test, y_train, y_test = train_test_split(data_tf_idf, data.processed_data["emotion"],
                                                    test_size=0.25, stratify=data.processed_data["emotion"],random_state=seed
                                                    )
precision, recall, accuracy, f1 = test_classifier(X_train, y_train, X_test, y_test, MultinomialNB(class_prior=[0.35, 0.3,0.35], alpha=1.0))

nb_acc = cv(BernoulliNB(), data_tf_idf,data.processed_data["emotion"])


precision, recall, accuracy, f1 = test_classifier(X_train, y_train, X_test, y_test, RandomForestClassifier(n_estimators=403,n_jobs=-1, randc
rf_acc = cv(RandomForestClassifier(n_estimators=403,n_jobs=-1,random_state=seed),data_tf_idf,data.processed_data["emotion"])

precision, recall, accuracy, f1 = test_classifier(X_train, y_train, X_test, y_test, KNeighborsClassifier(n_neighbors=5, p=2,
                                                  metric='minkowski'))

knn_acc = cv(KNeighborsClassifier(n_neighbors=10, p=2,
                                  metric='minkowski'), data_tf_idf,data.processed_data["emotion"])
```

Figura 7.9: Implementación de entrenamiento y pruebas.

Con la función train_test_split se logra en primer lugar la división del set de datos en entrenamiento y pruebas. Las secciones del set de datos que son tomados para entrenamiento y pruebas, son seleccionadas aleatoriamente gracias al parámetro random_state, pero siempre sin toparse y cumpliendo con los porcentajes asignados previamente. La modificación del parámetro anterior afecta la eficiencia de cada algoritmo. En la mayoría de los experimentos de Sentiment Analysis se le da el valor de 42. Los procesos hechos son asignados a variables. En segundo lugar son procesadas las etiquetas. Finalmente, todo lo anterior se asigna a variables estas las cuales se puede resumir en :

- X_train: set de entrenamiento

- Y_train: etiquetas de entrenamiento
- X_test: set de pruebas.
- Y_test: etiquetas del set de pruebas.

En la figura 7.10 se puede ver el método para realizar el entrenamiento y pruebas. Esta recibe como parámetro las etiquetas, los sets de datos y el clasificador ya configurado. Esta entrega como resultado la eficiencia de la clasificación a través de las métricas: Exactitud, precisión, exhaustividad y puntaje F1.

```
def test_classifier(X_train, y_train, X_test, y_test, classifier):
    log("")
    log("=============================================")
    classifier_name = str(type(classifier).__name__)
    log("MODIFIED Testing " + classifier_name)
    now = time()
    list_of_labels = sorted(list(set(y_train)))
    model = classifier.fit(X_train, y_train)
    log("Learning time {0}s".format(time() - now))
    now = time()
    predictions = model.predict(X_test)
    log("Predicting time {0}s".format(time() - now))

    precision = precision_score(y_test, predictions, average=None, pos_label=None, labels=list_of_labels)
    recall = recall_score(y_test, predictions, average=None, pos_label=None, labels=list_of_labels)
    accuracy = accuracy_score(y_test, predictions)
    f1 = f1_score(y_test, predictions, average=None, pos_label=None, labels=list_of_labels)
    log("=================== Results ===================")
    log("            pNeutral      Negative    Positive")
    log("F1       " + str(f1))
    log("Precision" + str(precision))
    log("Recall   " + str(recall))
    log("Accuracy " + str(accuracy))
    log("=============================================")
    print classification_report(y_test, predictions, list_of_labels)
    return precision, recall, accuracy, f1
```

Figura 7.10: Implementación de función principal de entrenamiento y pruebas

8. Análisis de resultados obtenidos

8.1. Resultados de experimentos preliminares

8.1.1. Prueba preliminar 1 y 2: Sentiment analysis sobre tweets clasificados en inglés

Para la primera prueba preliminar se utiliza un set de datos en inglés extraído del sitio ZABLO.NET. Desde aquel sitio fue utilizado el código para las implementación y pruebas, el cual fue modificado para realizar el funcionamiento descrito anteriormente, logrando así un mejor funcionamiento. En el sitio se reportan los resultados de las pruebas hechas por ellos, por lo cual se esperaba un resultado similar con el código modificado. En la tabla 8.1 se pueden observar los resultados utilizando Stemming sobre el set de datos y en la tabla 8.2 los resultados usando Lemmatization sobre el set de datos.

Clasificador	Exactitud	Precisión	Recall	F1 Score
Naive Bayes	**58,33%**	58,00%	**58,00%**	**57,00%**
Decision Tree	49,19%	49,00%	49,00%	49,00%
Random Forest	56,56%	**59,00%**	57,00%	52,00%
K-N-Neighbors	53,10%	52,00%	53,00%	52,00%

Tabla 8.1: Resultados usando Stemming

Clasificador	Exactitud	Precisión	Recall	F1 Score
Naive Bayes	56,56%	56,00%	57,00%	**55,00%**
Decision Tree	48,53%	49,00%	49,00%	49,00%
Random Forest	**57,15%**	**60,00%**	**57,00%**	53,00%
K-N-Neighbors	52,58%	51,00%	53,00%	52,00%

Tabla 8.2: Resultados promedios usando Lemmatization

Al utilizar Naive Bayes, se le utilizaron los siguientes porcentajes de prioridad de clase:

- Neutro: 35%
- Negativo: 40%
- Positivo: 25%

8.1.2. Prueba preliminar 3: Sentiment analysis sobre tweets de multitiendas chilenas

Se hizo una tercera prueba, la cual usa un set de datos distintos en español. Este set contiene una lista de tweets respecto a las multi-tiendas chilenas Falabella y Ripley clasificados como neutro, negativo o positivo. Los resultados observados se presentan en la tabla 8.3.

Clasificador	Exactitud	Precisión	Recall	F1 Score
Naive Bayes	76,93%	77,00%	77,00%	77,00%
Decision Tree	70,27%	70,00%	70,00%	70,00%
Random Forest	**78,40%**	**79,00%**	**78,00%**	**78,00%**
K-N-Neighbors	70,13%	70,00%	70,00%	70,00%

Tabla 8.3: Resultados usando Stemming para casode multitiendas

En el uso del algoritmo Naive Bayes, los parámetros de porcentaje de prioridad de clase fueron asignados de manera automática por el mismo programa.

8.2. Resultados de la investigación

Dado los resultados con los casos anteriores, se decidió hacer uso de los mismos algoritmos para el tema de esta investigación, exceptuando el Árbol de Decisión por sus resultados promedios insuficientes. Se hicieron una serie de experimentos donde, en cada uno, se hizo variar los factores de los algoritmos de hasta entregar resultados satisfactorios. Entre estos parámetros, se encuentran el Random State y los porcentajes del set de datos usados para entrenamiento y pruebas.

Se hizo una serie de experimentos, donde los primeros 6 fueron hechos con la totalidad del set de datos. Este set de datos estaba altamente desproporcionado poseyendo una gran cantidad de tweets negativos y una cantidad pequeña de tweets positivos. Lo anteriormente mencionado trajo como resultado métricas promedio bajas y con valores extraños, en el caso de F1 y Recall, siendo aún peores para las clases neutra y positiva individualmente. El experimento que mejores resultados dio usando este set de datos, posee un *Random State* igual a 42 y se usó el 65% de los datos para entrenamiento y 35% para pruebas. En la tabla 8.4 se muestran los resultados para todos los algoritmos usados.

Clasificador	Exactitud	Precisión	Recall	F1 Score
Naive Bayes	63,918%	47,311%	43,658%	**44,272%**
Random Forest	**67,354%**	**75,379%**	39,968%	38,632%
K-N-Neighbors	65,865%	60,253%	**60,253%**	42,717%

Tabla 8.4: Resultados del mejor experimento, utilizando set de datos original

El algoritmo que dio resultados mejores y más equilibrados en este caso fue K-Nearest Neighbors. En la tabla 8.5 se muestra la matriz de confusión para aquel caso.

Clase/Clase a la cual se clasificó	Neutro	Negativo	Positivo
Neutro	77	150	0
Negativo	58	493	3
Positivo	24	63	5

Tabla 8.5: Matriz de confusión usando K-N-Neighbors

Observando la tabla 8.5 se puede ver que el algoritmo K-Nearest Neighbors ha clasificado de manera correcta la gran mayoría de los tweets negativos, siendo la clase que más aporta a las métricas promedio de clasificación. Pero para las otras 2 clases, ha hecho un mal trabajo, especialmente para la clase positiva, donde solo ha clasificado el 5,435% de manera correcta. Analizando las anteriores tablas 8.4 y 8.5 se observa que para este experimento los resultados fueron solo buenos para la clase negativa, no cumpliendo el objetivo de la investigación. Dado el resultado de este experimento y los otros, se decidió revisar el set de datos esperando obtener mejores resultados, dando como resultado un set de datos con una cantidad de datos similar para cada clase. A este nuevo set de datos se le llamará "set de datos equilibrado".

Con el set de datos equilibrado, se realizaron otros 6 experimentos más. Este set de datos posee una cantidad de datos similar para cada clase, ahora hay menos tweets negativos con una cantidad ligeramente superior a los tweets positivos y neutros. El mejor experimento usando este set de datos, fue realizado utilizando un 25% de los datos para la fase de pruebas y con un *Random State* igual a 42. En la tabla 8.6 se muestran los resultados para los algoritmos utilizando los parámetros mencionados.

Clasificador	Exactitud	Precisión	Recall	F1 Score
Naive Bayes	**63,424%**	**63,849%**	**60,566%**	**60,569%**
Random Forest	59,533%	59,136%	56,627%	56,739%
K-N-Neighbors	58,366%	59,809%	56,894%	57,457%

Tabla 8.6: Resultados del mejor experimento, utilizando set de datos equilibrado.

Se observa claramente que los mejores resultados fueron obtenidos utilizando el algoritmo Naive Bayes Multinomial. En la tabla 8.7, se muestra la matriz de confusión para aquel algoritmo.

Clase/Clase a la cual se clasificó	**Neutro**	**Negativo**	**Positivo**
Neutro	43	29	16
Negativo	10	90	3
Positivo	10	26	30

Tabla 8.7: Matriz de confusión, utilizando set de datos equilibrado.

Observando la tabla anterior, se aprecia una mayor cantidad de tweets correctamente clasificados para las clases positivas y neutras. Analizando los resultados de las tablas 8.6 y 8.7, se observa una menor exactitud y precisión, pero un mayor Recall y con un mayor puntaje F1. Estas últimas 2 métricas indican que bajaron de manera considerable los tweets clasificados erróneamente, siendo esto especialmente importante para un problema de clasificación con clases múltiples. Esta situación se puede observar en las siguientes 2 gráficas. En la primera figura se observa una comparación de la métrica Recall para los

experimentos usando el set de datos sin equilibrar y el equilibrado, y en la segunda para la métrica F1.

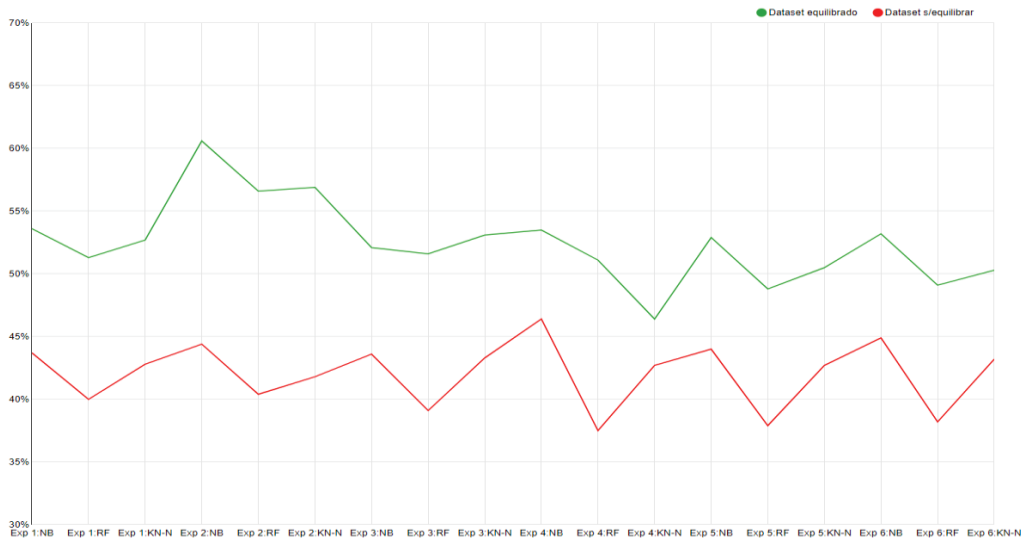

Figura 8.1: Comparación de la métrica Recall

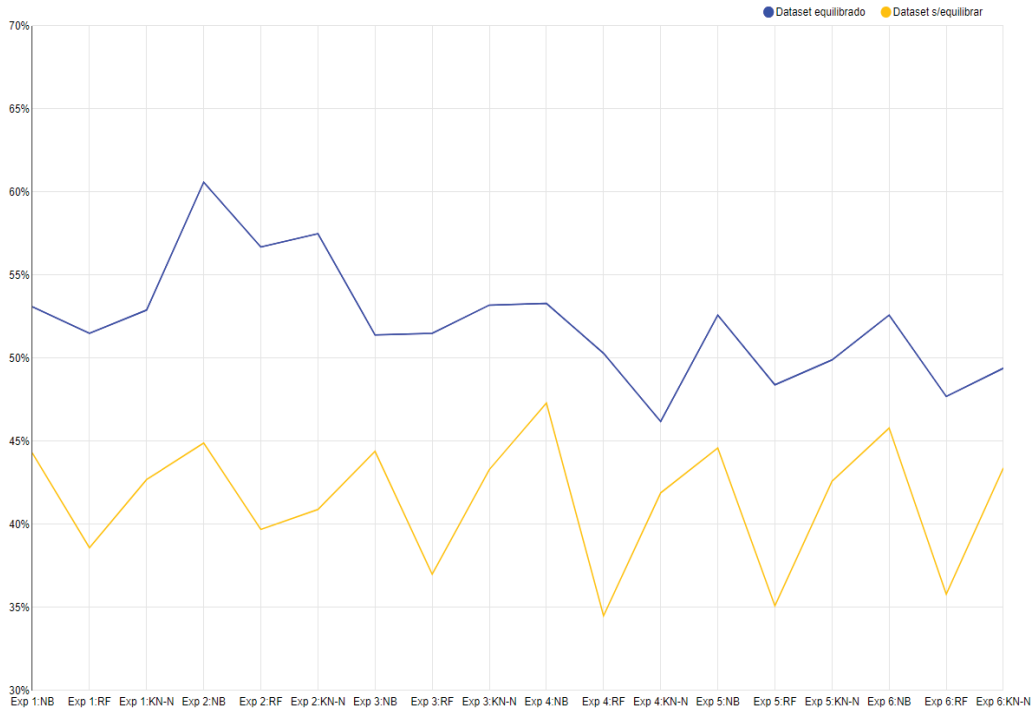

Figura 8.2: Comparación de la métrica F1

Es por esto que se propone como mejor algoritmo clasificador Naive Bayes Multinomial, utilizando como parámetros un 75% y 25% del *set datos equilibrado* para la fase de entrenamiento y pruebas, respectivamente, y con un Random State de valor 42.

9. Conclusiones

Sentiment Analysis constituye una herramienta la cual permite a las empresas saber, analizar y predecir las opiniones de sus clientes. El análisis de resultados de lo anterior puede hacer que la empresa se promocione de manera más efectiva, mejore su servicio y su imagen ante el público. La conjunción de todos estos factores logrará que una empresa sea más cercana a sus clientes y se adapte mejor al mercado; todo esto teniendo como consecuencia final el aumento de sus utilidades. Sin embargo, las herramientas estándar para Sentiment Analysis no constituyen una solución adecuada para una empresa, debido a que los clientes y el mercado de una empresa varían mucho respecto a otra. Esto se ve reflejado de manera clara en la opinión escrita expresada por estos clientes. Los factores que influyen sobre la opinión expresada son el rango etario de sus consumidores, la clase social de estos y el lugar donde sus locales se encuentran. Por lo tanto, la forma textual que los clientes expresan su opinión variará mucho de una empresa a otra. Es por esto que para analizar adecuadamente la opinión de una empresa usando Sentiment Analysis lo mejor es usar datos donde se mencione específica y directamente a ella; usando un clasificador en específico para cada organización, así reduciendo el error en la determinación de la opinión.

Se ha determinado que un modelo eficiente para determinar la opinión de una empresa que tiene como mercado gran parte de la población Chile y sin distinción de clases sociales, como lo es Fruna, tiene que cumplir los siguientes factores. En primer un set de datos clasificados que hagan mención directa a la empresa, existiendo para cada clase una cantidad equitativa de datos sin que los de una clase sobrepase en gran cantidad a las otras. El tamaño del conjunto de pruebas no debe ser mayor a un 35% del total de los datos. En segundo lugar, que estos se procesen usando stemming y una representación TF-IDF. Y finalmente, que el conocimiento obtenido de estos datos sea abstraído a través algoritmos de clasificación de mediana complejidad, los cuales sean conocidos por brindar buenos resultados en problemas de Sentiment Analysis. Esta investigación determinó que el algoritmo que mejor resultado dio para resolver la problemática propuesta fue Naive Bayes multinomial, seguido estrechamente por Random Forest.

Así se ha establecido una propuesta para determinar la opinión actual de una empresa chilena de alcance nacional. Esta sin duda puede ser mejorada obteniendo más datos previamente clasificados, depurados y que aporten de manera equitativa a cada clase del set de datos. Lo anterior, sumado a nuevos ajustes al modelo para abstraer conocimientos estos, podrán entregar resultados altamente precisos de predicción de opiniones para las empresas.

10.Referencias

[1] El Desconcierto. “Noesnalaferia se disculpa por apología a Fruna y sus productos: ‘Nos hemos equivocado’”. http://www.eldesconcierto.cl/2017/05/30/noesnalaferia-se-disculpa-por-apologia-a-fruna-y-sus-productos-nos-hemos-equivocado/ . Mayo 2017

[2] The Clinic. “Los muertos de Fruna: El prontuario laboral de la popular empresa de confites”. http://www.theclinic.cl/2017/05/28/los-muertos-fruna-prontuario-laboral-la-popular-empresa-confites/ . 28 de mayo de 2017.

[3] Christine Day: The Importance of Sentiment Analysis in Social Media Analysis. https://www.linkedin.com/pulse/importance-sentiment-analysis-social-media-christine-day. Marzo 2015

[4] Saccá Espinoza, Gary. Estrategia para el Reconocimiento de Emociones Colectivas en Entornos Sociales basados en TIC.

[5] Bo Pang and Lillian Lee: Opinion mining and sentiment analysis. http://www.cs.cornell.edu/home/llee/omsa/omsa.pdf . 2008

[6] F. J. Martín Mateos, J. L. Ruiz Reina: Procesamiento del lenguaje natural. https://www.cs.us.es/cursos/ia2/temas/tema-06.pdf.

[7] Tony Mullen: Introduction to Sentiment Analysis. https://lct-master.org/files/MullenSentimentCourseSlides.pdf.

[8] Richard Socher, Alex Perelygin, Jean Y. Wu, Jason Chuang,Christopher D. Manning, Andrew Y. Ng and Christopher Potts. Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. Stanford University, Stanford 2013.

[9] Introduction to Machine Learning Second Edition. The MIT Press Cambridge, Massachusetts London, England.2010.

[10] Walaa Medhat. Sentiment analysis algorithms and applications: A survey. http://www.sciencedirect.com/science/article/pii/S2090447914000550. 2014

[11] GetOldTweets-python. Jefferson Henriquez. GitHub. https://github.com/Jefferson-Henrique/GetOldTweets-python

[12] Machine Learning : Text feature extraction. Christian S. Perone, Machine Learning Engineer,Software Engineer. Montreal, QC, Canada. http://blog.christianperone.com/2011/09/machine-learning-text-feature-extraction-tf-idf-part-i/ , http://blog.christianperone.com/2011/10/machine-learning-text-feature-extraction-tf-idf-part-ii/ .

[13] Peter Stefek. Getting Sentimental.Jul 14, 2016. http://peterstefek.me/sentiment/analysis/logistic/regression/n-gram/2016/07/14/sentiment.html

[14] http://classes.ischool.syr.edu/ist664/NLPFall2014/SciKitLearn.pdf

[15] Approaching (Almost) Any Machine Learning Problem. Abhishek Thakur. Kaggle.com. http://blog.kaggle.com/2016/07/21/approaching-almost-any-machine-learning-problem-abhishek-thakur/. Julio 2016.

[16] Aprendizaje automático. Wikipedia. https://es.wikipedia.org/wiki/Aprendizaje_autom%C3%A1tico

[17] Aprendizaje Supervisado. Alejandro Cassis. Octubre de 2015. https://inteligenciaartificial101.wordpress.com/author/alejandrocassis/
[18] Supervised Learning. Anonymous. Noviembre de 2011. https://www.saylor.org/site/wp-content/uploads/2011/11/CS405-6.2.1.2-WIKIPEDIA.pdf
[19] Naive Bayes Multinomial para Clasificación de Texto Usando un Esquema de Pesado por Clases. Emmanuel Anguiano-Hernández. Abril 2009. http://ccc.inaoep.mx/~esucar/Clases-mgp/Proyectos/MGP_RepProy_Abr_29.pdf
[20] Deep Learning for NLP Lecture Notes: Part I. Francois Chaubard, Rohit Mundra, Richard Socher. Stanford University. Primavera de 2016. https://cs224d.stanford.edu/lecture_notes/notes1.pdf
[21] An Intuitive Understanding of Word Embeddings: From Count Vectors to Word2Vec. Neeraj Singh Sarwan. Analytics Vidhya, Gurgaon, India. Junio 2017. https://www.analyticsvidhya.com/blog/2017/06/word-embeddings-count-word2veec/
[22] Tutorial: Vector Representations of Words. TensorFlow™. Agosto 2017. https://www.tensorflow.org/tutorials/word2vec
[23] Text Classification and Sentiment Analysis. Ahmet Taspinar.Noviembre 2015. http://ataspinar.com/2015/11/16/text-classification-and-sentiment-analysis/
[24] How To Build Multi-Layer Perceptron Neural Network Models with Keras. Jason Brownlee. Mayo, 2016. https://machinelearningmastery.com/build-multi-layer-perceptron-neural-network-models-keras/
[25] Young Joseph Bakos. Decision Tree Classifier. http://mines.humanoriented.com/classes/2010/fall/csci568/portfolio_exports/lguo/decisionTree.html . 20 de Septiembre de 2010.
[26] Constantino Malagón. Universidad Nebrija. https://www.nebrija.es/~cmalagon/inco/apuntes_mios/arboles_de_decision.pdf.
[27] Varios autores. Wikipedia. https://es.wikipedia.org/wiki/%C3%81rbol_de_decisi%C3%B3n_(modelo_de_clasificaci%C3%B3n_ID3)
[28] A comparative study of decision tree ID3 and C4.5. Badr HSSINA, Abdelkarim MERBOUHA, Hanane EZZIKOURI, Mohammed ERRITALITIAD laboratory,Computer Sciences Department, Faculty of sciences and techniques, Sultan Moulay Slimane University. http://saiconference.com/Downloads/SpecialIssueNo10/Paper_3-A_comparative_study_of_decision_tree_ID3_and_C4.5.pdf
[29] Random forest vs Simple tree. QuantDare. https://quantdare.com/random-forest-many-are-better-than-one/. Febrero, 2017.
[30] Young Joseph Bakos. K-Nearest Neighbors. http://mines.humanoriented.com/classes/2010/fall/csci568/portfolio_exports/lguo/knn.html.

[31] Christopher D. Manning, Prabhakar Raghavan, Hinrich Schütze. An Introduction to information retrieval. Cambridge University. https://nlp.stanford.edu/IR-book/pdf/irbookonlinereading.pdf . Abril 2009.

[32] Mari Vallez y Rafael Pedraza-Jimenez. El Procesamiento del Lenguaje Natural en la Recuperación de Información Textual y áreas afines. Universitat Pompeu Fabra. https://www.upf.edu/hipertextnet/numero-5/pln.html. Mayo 2007.

[33] Tunggawan, E., & Soelistio, Y. E. (2016, October). And the winner is…: Bayesian Twitter-based prediction on 2016 US presidential election. In *Computer, Control, Informatics and its Applications (IC3INA), 2016 International Conference on* (pp. 33-37). IEEE.

[34] Chin, D., Zappone, A., & Zhao, J. (2016). Analyzing Twitter Sentiment of the 2016 Presidential Candidates.

[35] FOX NEWS (Julio 2017). United Airlines controversies: From revoking toddler's seat to David Dao's removal. . http://www.foxnews.com/travel/2017/07/06/united-airline-controversies-from-revoking-toddlers-seat-to-david-dao-s-removal.html.

[36] Vivek Yadav. US Airlines sentiment analysis using twitter data. http://vxy10.github.io/p6_vis_versions/scroll_Submission1/index.html.

[37] Asish Satpathy, School of Business and Administration, University of California, Riverside, CA; Tanvi Kode, Goutam Chakraborty, Spears School of Business, Oklahoma State University, Stillwater, OK. Location Based Association of Customers' Sentiments and Retail Sales. https://support.sas.com/resources/papers/proceedings15/3342-2015.pdf

[38] ELÉCTRICO, I. C., & GARCÍA, L. M. (2014). ANÁLISIS DE SENTIMIENTOS Y PREDICCIÓN DE EVENTOS EN TWITTER.

[39] Felipe Oliva and Rodrigo Alfaro. Escuela de informática, Pontifica Universidad Católica de Valparaíso. Clasificación Automática del Sentido de los Mensajes en Twitter: Comparando Entrenamiento específico y contextual.

[40] Baccianella, S., Esuli, A., & Sebastiani, F. (2010, May). SentiWordNet 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining. In *LREC* (Vol. 10, pp. 2200-2204).

ANEXOS

Se realizaron un total de 12 experimentos, 6 usando el set de datos sin equilibrar y 6 usando el set de datos equilibrado. Los experimentos usando el primer set de datos, usaron los mismos parámetros y los mismos algoritmos que los experimentos usando el segundo set de datos.

1. ANEXO A: Resultados de los experimentos utilizando set de datos sin equilibrar.

1.1. Experimento 1

Random state: 42

Porcentaje del set de datos usado para pruebas: 35%

Clasificador	Exactitud	Precisión	Recall	F1 Score
Naive Bayes	63,918%	47,311%	43,658%	44,272%
Random Forest	67,354%	75,379%	39,968%	38,632%
K-N-Neighbors	65,865%	60,253%	42,782%	42,717%

1.2. Experimento 2

Random state: 42

Porcentaje del set de datos usado para pruebas: 25%

Clasificador	Exactitud	Precisión	Recall	F1 Score
Naive Bayes	63,242%	46,879%	44,382%	44,863%
Random Forest	66,934%	73,998%	40,449%	39,750%
K-N-Neighbors	64,205%	54,903%	41,758%	40,941%

1.3. Experimento 3

Random state: 42

Porcentaje del set de datos usado para pruebas: 40%

Clasificador	Exactitud	Precisión	Recall	F1 Score
Naive Bayes	63,390%	47,942%	43,564%	44,357%
Random Forest	67,001%	72,460%	39,131%	37,031%
K-N-Neighbors	65,998%	68,101%	43,263%	43,283%

1.4. Experimento 4

Random state: 666

Porcentaje del set de datos usado para pruebas: 25%

Clasificador	Exactitud	Precisión	Recall	F1 Score
Naive Bayes	65,650%	50,331%	46,433%	47,269%
Random Forest	65,490%	61,922%	37,513%	34,480%
K-N-Neighbors	65,971%	72,920%	42,686%	41,930%

1.5. Experimento 5

Random state: 666

Porcentaje del set de datos usado para pruebas: 35%

Clasificador	Exactitud	Precisión	Recall	F1 Score
Naive Bayes	64,719%	48,925%	44,037%	44,644%
Random Forest	65,865%	76,327%	37,886%	35,145%
K-N-Neighbors	65,292%	59,501%	42,741%	42,592%

1.6. Experimento 6

Random state: 666

Porcentaje del set de datos usado para pruebas: 40%

Clasificador	Exactitud	Precisión	Recall	F1 Score
Naive Bayes	65,597%	50,887%	44,884%	45,769%
Random Forest	65,697%	70,943%	38,168%	35,758%
K-N-Neighbors	65,296%	64,686%	43,232%	43,423%

2. ANEXO B: Resultados de los experimentos utilizando set de datos equilibrado.

2.1. Experimento 7

Random state: 42

Porcentaje del set de datos usado para pruebas: 35%

Clasificador	Exactitud	Precisión	Recall	F1 Score
Naive Bayes	56,667%	55,875%	53,575%	53,062%
Random Forest	53,889%	54,487%	51,297%	51,546%
K-N-Neighbors	55,000%	54,863%	52,671%	52,882%

2.2. Experimento 8

Random state: 42

Porcentaje del set de datos usado para pruebas: 25%

Clasificador	Exactitud	Precisión	Recall	F1 Score
Naive Bayes	63,424%	63,849%	60,566%	60,569%
Random Forest	59,533%	59,136%	56,627%	56,739%
K-N-Neighbors	58,366%	59,809%	56,894%	57,457%

2.3. Experimento 9

Random state: 42

Porcentaje del set de datos usado para pruebas: 40%

Clasificador	Exactitud	Precisión	Recall	F1 Score
Naive Bayes	55,718%	55,606%	52,093%	51,421%
Random Forest	54,501%	55,961%	51,584%	51,465%
K-N-Neighbors	55,474%	56,114%	53,052%	53,171%

2.4. Experimento 10

Random state: 666

Porcentaje del set de datos usado para pruebas: 25%

Clasificador	Exactitud	Precisión	Recall	F1 Score
Naive Bayes	56,420%	54,000%	53,541%	53,263%
Random Forest	54,864%	57,209%	51,055%	50,271%
K-N-Neighbors	48,638%	47,015%	46,422%	46,193%

2.5. Experimento 11

Random state: 666

Porcentaje del set de datos usado para pruebas: 35%

Clasificador	Exactitud	Precisión	Recall	F1 Score
Naive Bayes	56,111%	54,008%	52,925%	52,560%
Random Forest	52,222%	54,210%	48,824%	48,406%
K-N-Neighbors	53,056%	49,335%	50,519%	49,910%

2.6. Experimento 12

Random state: 666

Porcentaje del set de datos usado para pruebas: 40%

Clasificador	Exactitud	Precisión	Recall	F1 Score
Naive Bayes	56,691%	54,677%	53,159%	52,559%
Random Forest	53,041%	55,661%	49,141%	47,691%
K-N-Neighbors	53,528%	51,601%	50,302%	49,424%