Universidad Autónoma de Ciudad Juárez

Instituto de Ingeniería y Tecnología

Departamento de Ingeniería Eléctrica y Computación

Maestría en Cómputo Aplicado

“Detección de sugerencias en tuits de usuarios y seguidores de aerolíneas con minería de sugerencias”

Trabajo recepcional para obtener el grado de

Maestro en Cómputo Aplicado

Rafael Jiménez Castro

“Becado por el Consejo Nacional de Ciencia y Tecnología”

Bajo la Dirección de el

Dr. Rogelio Florencia Juárez

Y la Codirección de el

Dr. Vicente García Jiménez

Ciudad Juárez, Chihuahua 8 de diciembre de 2021

AGRADECIMIENTOS

Al Consejo Nacional de Ciencia y Tecnología (CONACyT) por la beca nacional recibida durante los dos años de maestría, así como el apoyo para estancia de investigación nacional.

CONTENIDO

1. DEFINICIÓN DEL PROBLEMA 1

1.1. Introducción 1

1.2. Planteamiento del problema 3

1.3. Objetivos 5

1.4. Justificación 6

1.5. Alcances y limitaciones 7

2. ANTECEDENTES 8

2.1. Procesamiento de Lenguaje Natural 8

2.2. Minería de Sugerencias 10

2.2.1. Limpieza de la información 10
2.2.2 Valores faltantes 11
2.2.3 Eliminación de ruido 11
2.2.4 Integración de la información 11
2.2.5 Transformación 11
2.2.6 Extracción de características 12
2.2.7 Clasificación 12
2.2.8 Evaluación del clasificador 14
2.2.9 Validación 15

2.3. Trabajos relacionados 16

3. PROPUESTA DE SOLUCIÓN 22

3.1. Metodología basada en prototipos 22

3.2. Creación del producto 23

3.2.1. Recolección de comentarios 23
3.2.2. Preprocesamiento de comentarios 25
3.2.3. Creación de un diccionario de sugerencias 27
3.2.4. Creación de un vector de características 29
3.2.5 Clasificación de comentarios 30

3.3. Configuración de experimentos 31

4. RESULTADOS 32

5. CONCLUSIONES Y TRABAJOS FUTUROS 37

REFERENCIAS 39

APÉNDICES..43

LISTA DE TABLAS

Tabla 1: Ejemplo de una matriz de confusión para una clasificación de dos clases..... 15
Tabla 2: Validación cruzada utilizando una variación de 5 iteraciones. 16
Tabla 3: Descripción breve de la base de datos utilizada en los experimentos..... 31
Tabla 4: Resultados de exactitud global por clasificador. 32
Tabla 5: Resultados de precisión por clasificador. 32
Tabla 6: Resultados de recall por clasificador. 32
Tabla 7: Matriz de confusión para el clasificador Bagging. 32
Tabla 8: Matriz de confusión para el clasificador Naive Bayes. 33
Tabla 9: Matriz de confusión del clasificador Naive Bayes y una submuestra de la clase mayoritaria. 33

LISTA DE FIGURAS

Figura 1: Diagrama de la metodología de desarrollo guiada por pruebas de software. . 22
Figura 2: Imagen de la aplicación Twitter Archiver donde se muestran las cuentas monitoreadas para la búsqueda de tuits. 24
Figura 3: Ejemplo de los subconjuntos separados para utilizar el etiquetado multitudinario. 25
Figura 4: Diseño bidimensional del conjunto de datos utilizando tSNE. 34
Figura 5: Mapa de calor de los datos contenidos en el conjunto de datos. 35
Figura 6: Segundo cuadrante del mapa de calor. 36

1. Definición del problema

En este capítulo se expondrán los antecedentes de Twitter y sus usuarios, se planteará el actual problema que existe para detectar sugerencias y se hará una breve explicación sobre una propuesta de solución a este problema, así como sus objetivos y justificación.

1.1. Introducción

Actualmente el nivel de interacción que tienen las personas en redes sociales ha provocado que las campañas de mercadotecnia tengan un cambio en su modelo de negocios (Saravanakumar & SuganthaLakshmi, 2012), pasando de crear anuncios en medios de comunicación impresos, la radio y la televisión, con los cuales se podía llegar a un gran número de personas fueran clientes o no, a medios digitales en busca de aumentar el alcance a clientes potenciales.

Las redes sociales han pasado de sitios en línea solo para interactuar con las amistades, a una plataforma en la cual se comunican artistas, marcas y hasta presidentes, con multitudes de personas diariamente. Las redes digitales son utilizadas por los usuarios para compartir sus gustos e ideas, estar en contacto con sus amistades, hacer publicidad y para estar informados de lo que acontece en su localidad y el mundo. Al realizar publicidad a negocios, las redes sociales promueven la interacción entre usuarios y gratifican con mayor exposición al contenido que se vuelve tendencia, siendo la interacción entre el emisor del mensaje con publicidad y los usuarios la cual crea valor para la marca y el producto (Ahmed, 2017).

- Una de las cinco plataformas digitales más utilizadas en el 2020 fue Twitter según un estudio publicado por Adobe (Adobe, 2020). Asimismo, la empresa Hootsuite (Kemp, 2019) menciona que Twitter es la séptima página de internet más accedida en el mundo, con una generación diaria de 500 millones de tuits diariamente (Stricker, 2017). En México, Twitter cuenta con 7.22 millones de usuarios (Kemp, 2019), cifra que va en aumento, y que se muestra en un incremento del 3.7% en el último cuarto del 2018.

Twitter es una red social la cual se destaca de las demás redes por ser un medio popular, por el cual, se pueden comunicar noticias de última hora de forma rápida y masiva, además ser el centro de actividad en línea para la comunicación de usuarios y empresas para dar y recibir atención al cliente.

Las aerolíneas mexicanas utilizan Twitter para emitir noticias de vuelos[1], dar a conocer promociones[2], destinos nacionales e internacionales, información general sobre vuelos[3] y protocolos de seguridad[4].

Los usuarios al recibir los tuits de las aerolíneas cuentan con tres opciones para interactuar con ellas:

1. Compartir el tuit a sus seguidores por medio de un retuit,
2. Darle "*me gusta*" cuando el tuit de la aerolínea es de su agrado, y
3. Responder al tuit emitido por la aerolínea por medio de comentarios, donde el usuario expresa su opinión acerca del tuit al que responde, en el cual podrá o no incluir una sugerencia para mejorar el servicio de la compañía.

Dentro de los diversos tuits que redactan los usuarios a las aerolíneas, se encuentran sugerencias de cómo mejorar los servicios o procesos según las experiencias que han tenido los usuarios al utilizar alguna de las aerolíneas. Las sugerencias son importantes ya que el éxito de muchas de las empresas está basado en dar al cliente lo que quiere o necesita (Brun & Hagege, 2013), y estas necesidades pueden cambiar con el tiempo. Un ejemplo de ello es la compañía *DeWalt*, que da a conocer a sus clientes la posibilidad de crear en conjunto

---

[1] Aeroméxico [@Aeromexico]. (2021, 27 de noviembre). *¿Quieres que tu siguiente aventura sea en Estados Unidos? Conoce aquí los nuevos requisitos de ingreso y comienza a* [Imagen adjunta]. [Tweet]. Twitter. https://twitter.com/Aeromexico/status/1464776130708131840

[2] Aeroméxico [@Aeromexico]. (2021, 28 de noviembre). *¡Los Ángeles te espera!* ☆ *Prepara tus lentes de sol* 😎 *y descúbrelo en tu siguiente viaje.* [Imagen adjunta]. [Tweet]. Twitter. https://twitter.com/Aeromexico/status/1465047621379182605

[3] Aeroméxico [@Aeromexico]. (2021, 29 de noviembre). *¿Ya conoces los aviones más nuevos de nuestra flota? Bienvenido a bordo de nuestros 737 MAX.* 🙂🛫 [Imagen adjunta]. [Tweet]. Twitter. https://twitter.com/Aeromexico/status/1465332136832118793

[4] Aeroméxico [@Aeromexico]. (2021, 30 de noviembre). El mejor viaje es el que se hace seguro, es por eso que continuamos cuidándote con los protocolos de higiene [Miniatura con enlace adjunta] [Imagen adjunta]. [Tweet]. Twitter. https://twitter.com/Aeromexico/status/1465848166297788416

nuevos productos a base de las sugerencias de sus consumidores (DeWalt, 2010). Esto ha ayudado a la compañía ha mantenerse en la cima de las empresas fabricantes de herramientas de construcción y ganar varios premios con sus productos (DeWalt, 2019). Otro ejemplo es el de la zapatería *Archibald London*, que en el 2019 creó una encuesta en el foro de moda *StyleForum* para que los usuarios diseñarán por consenso un par de tenis (Arhibald London, 2020). Esta colaboración con los clientes ha durado aún después de haber lanzado el producto, ya que ahora los usuarios del foro han dado a conocer los problemas que han tenido con el producto y cómo mejorarlos. Este tipo de innovación es llamada innovación guida por el cliente (*customer-driven innovation*) (Penisi & Kim, 2003), y requiere que una organización esté enfocada a entregar productos con atributos que el consumidor pueda valorar positivamente.

La propuesta de solución pretende brindar atención a este nicho de oportunidad de detección de sugerencias a través de un prototipo de procesamiento de lenguaje natural y aprendizaje automático que permita extraer las características de los tuits emitidos por clientes de aerolíneas y clasificarlos en sugerencias y no sugerencias.

1.2. Planteamiento del problema

Desde el 2015 cuando se firmó el acuerdo entre la Secretaría de Comunicaciones y Transporte de México y el Departamento de Transporte de Estados Unidos (Rico, 2018), las aerolíneas mexicanas han pasado por una etapa de crecimiento en el mercado internacional donde se ha incrementado la cuota de vuelos de empresas mexicanas en vuelos internacionales. Esto sumado al incremento del turismo en vuelos nacionales ha creado un gran número de usuarios de aerolíneas que, de acuerdo con sus experiencias de viaje, emiten comentarios hacia las aerolíneas en Twitter. Este incremento de comentarios es aún más notable durante las épocas vacacionales, lo cual haría que una búsqueda manual de sugerencias sea aún más costosa en tiempo y esfuerzo.

La detección de una sugerencia entre comentarios es difícil aun para una persona, ya que las sugerencias requieren un alto nivel de conocimiento ligústico (Negi S. , 2016) para poder realizar la distinción entre una sugerencia directa, una sugerencia inferida, una orden,

y un deseo, las cuales son definidas en la Tabla 1. Por ejemplo, el tuit "*@VivaAerobus Si contestarán el teléfono todo sería más fácil …*"[5] implica una connotación negativa sobre el servicio al cliente y de forma inferida sugiere a la aerolínea que debería mejorar su servicio, por otra parte, el tuit "*un lunch en la espera no cae mal*" está estructurado como un deseo, donde el usuario le sugiere a la aerolínea que ofrezca algo de comer mientras esperan cuando los vuelos se retrasan.

Tabla 1: Definición de los tipos de comentarios encontrados en tuits.

Tipo de comentario	**Definición**
Sugerencia directa	Son sugerencias explicitas donde se propone, recomienda o aconseja la realización de una acción. Ej. @VivaAerobus sugiero que en su servicio telefónico los asistentes sean más pacientes para explicar sus procedimientos a adultos mayores sobre todo lo que se refiere a su página web.. se desesperan y hablan golpeado.. gracias
Sugerencia inferida o indirecta	Son sugerencias donde se proporciona información de la cual una sugerencia puede ser deducida. Ej. @VivaAerobus porque no contestan el puto teléfono? Llevo 5+ horas tratando de hablar con alguien.
Ordenes	Son comentarios con un comando, dirección o instrucción que denota la solicitud de una acción. Ej. @interjet contesten los teléfonos!
Deseos	Son comentarios donde se hace una solicitud con la esperanza de un cambio en un producto o servicio. Estas solicitudes pueden variar desde muy generales a muy específicas. Ej. @interjet algún teléfono de atención al cliente que si contesten?
No sugerencia	Son comentarios que no contienen una sugerencia, orden, o deseo. Ej. @VivaAerobus Acá los esperamos 🌴

A partir de una recolección de tuits que se realizó durante el periodo del 09 de agosto al 27 de octubre del 2019 de las aerolíneas Aeroméxico con 4,177 tuits, Interjet con 13,814

[5] Sharloth [@Sharloth_01]. (2019,2 de agosto). *@VivaAerobus Si contestarán el teléfono todo sería más fácil…* [Tweet]. Twitter. https://twitter.com/Sharloth_01/status/1157317291702231040

tuits, Volaris con 40 tuits, Viva Aerobús con 3,310 tuits, AEROMAR y TAR con solo 2 tuits, se pudo detectar que el proceso de detección manual podría representar un reto debido a que, la cantidad de tuits enviados con sugerencias es de un 0.1% del total de tuits.

Las sugerencias emitidas podrían ser aprovechadas por las empresas para ofrecer un producto enfocado al usuario, buscando incrementar el número de clientes satisfechos. Asimismo, se reduciría la pérdida de clientes, al mismo tiempo que se puede evaluar la efectividad de las campañas publicitarias que se están realizando. Por tal motivo, contar con un sistema de clasificación capaz de identificar automáticamente las sugerencias entre todos los tuits puede ser de gran ayuda para las empresas. Cabe hacer mención que implementaciones de este tipo existen para el idioma inglés y son detallados en la sección 2.3.

El reto tecnológico de desarrollar una solución cognitiva radica en formular un diccionario de palabras claves en español mexicano que, utilizando técnicas de procesamiento de lenguaje natural, permita a un algoritmo de clasificación identificar las sugerencias. El énfasis que se le da a este diccionario consiste en que, según el contenido de palabras y el orden de estas, será utilizado para formar un conjunto de entrenamiento al cual se le podrán aplicar varios algoritmos de clasificación.

1.3. Objetivos

En esta sección se presenta el objetivo general de este trabajo de investigación, así como los objetivos específicos.

Objetivo general

Implementar un modelo de clasificación de tuits dirigidos a aerolíneas mexicanas que contengan sugerencias en español, emitidos por usuarios de Twitter, utilizando minería de sugerencias.

Objetivos específicos

1. Crear un repositorio de tuits dirigidos a aerolíneas mediante la descarga automática utilizando un API disponible en la literatura.
2. Crear un conjunto de datos a partir del repositorio utilizando preprocesamiento basado en técnicas de procesamiento de lenguaje natural, como la tokenización, la limpieza, la normalización y la eliminación de ruido de los tuits contenidos en el repositorio.
3. Construir un diccionario de características basado en el conjunto de datos.
4. Entrenar y evaluar un modelo de clasificación supervisado.

1.4. Justificación

Actualmente, para que una empresa pueda explotar las áreas de oportunidad expresadas explícitamente por los usuarios a través de sugerencias en redes sociales, implica que una persona de la empresa esté revisando personalmente las redes sociales para detectar dichas sugerencias. Esto podría traer consigo principalmente las siguientes situaciones:

- El dedicar una gran cantidad de tiempo en la revisión de todos los tuits, lo cual le podría implicar no realizar otras actividades de la empresa.
- El no identificar todas las sugerencias de los clientes podría implicar el dejar escapar una oportunidad de mejora para la empresa.

Como es de imaginarse, la primera opción podría impactar en los costos de la empresa, mientras que la segunda, podría impactar en sus ganancias.

La implementación de un prototipo para la minería de sugerencias en comentarios realizados mediante la plataforma Twitter será utilizando técnicas probadas en la literatura con implementaciones con idioma inglés las cuales es necesario trasladar a una implementación en español donde aún no se es tan prevalente, además de que tendrá efectos favorables para las aerolíneas que utilicen redes sociales, tales como:

- Clasificar de forma automática los tuits en sugerencias y no sugerencias.
- Mejorar la atención al cliente y al mismo tiempo reforzar la imagen de la empresa al detectar las sugerencias más populares, con lo cual se le puede dar retroalimentación al usuario de que la sugerencia está siendo atendida o está en proceso de implementación si la empresa cree que sea conveniente.

1.5. Alcances y limitaciones

Dentro de los alcances de este trabajo indican que se espera alcanzar durante este trabajo, mientras las limitaciones indican que aspectos quedan fuera de la cobertura. Ambas se describen a continuación.

Alcances

- Se implementó un prototipo de clasificación que apoye en la detección de sugerencias en tuits de usuarios y seguidores de aerolíneas en Twitter, expresados en el idioma español.
- Se utilizaron los tuits en las cuentas de las aerolíneas mexicanas: Aeroméxico, Interjet, TAR, Viva Aerobús y Volaris.
- Se integró un clasificador supervisado para identificar los tuits que contengan sugerencias y los que no contengan sugerencias.

Limitaciones

- El prototipo no tiene conexión en tiempo real a ninguna fuente de información en redes sociales para extraer los comentarios de los usuarios.
- Se necesita revisar de forma manual los comentarios a clasificar.
- Los comentarios utilizados tienen un tamaño máximo de 280 caracteres que es el máximo número de caracteres permitidos por Twitter.

2. Antecedentes

En este capítulo se define el procesamiento de lenguaje natural, la minería de sugerencias y los recursos relacionados con la realización del proyecto. Asimismo, se detallan los trabajos relacionados en el área de la minería de sugerencias.

2.1. Procesamiento de Lenguaje Natural

Microsoft establece que el Procesamiento de Lenguaje Natural (PLN) tiene como propósito el diseñar y construir software que pueda analizar, entender y generar frases lingüísticas que los humanos puedan utilizar de forma natural, para que eventualmente puedan comunicarse con una computadora como si lo estuvieran haciendo con otra persona (Microsoft, 2016) . Sus aplicaciones incluyen (Research hubs, 2015):

- La traducción mecánica de un lenguaje humano a otro.
- La generación de texto en lenguaje humano.
- El desarrollo de interfaces con otros sistemas, como bases de datos, para habilitar el uso de lenguaje humano al realizar comandos y búsquedas.
- La comprensión de texto en lenguaje humano para realizar un resumen.

El rol del PLN en la minería de sugerencias es brindar la información lingüística necesaria para la extracción de información (Liu F. , Wang, Zhu, & Wang, 2019). Esta información se puede obtener a través del etiquetado gramatical de las palabras de una frase, analizando los resultados y asignando límites a las frases para que las lea el extractor de información. Un sistema de PLN realiza las siguientes tareas:

- Analiza una frase para determinar su sintaxis.
- Determina la semántica de una frase.
- Analiza el contexto de un texto para determinar su significado al compararse con otros textos.

El PLN se divide en dos ramas, las cuales son (Ovchinnikova, 2012):

1. Entendimiento de lenguaje natural: Se ocupa de la comprensión de lectura con la finalidad de interpretar un fragmento de texto o frase. El proceso de interpretación es la traducción de lenguaje natural a una representación ambigua en lenguaje formal.

2. Generación de lenguaje natural: Se encarga de la generación de nuevo lenguaje natural y de la traducción de un lenguaje a otro.

El PLN es una herramienta con grandes beneficios para el procesamiento y análisis de texto (Negi, Daudert, & Buitelaar, 2019), pero también tiene hasta el momento grandes limitaciones, como lo son:

- Expresiones figurativas: El uso de expresiones en sentido figurado y con sarcasmo son comunes en el habla coloquial por lo cual es necesario adquirir el significado según el contexto. Por ejemplo, el tuit "*@interjet Sería más útil su aviso si sirvieran las líneas. Increíble el interés que tienen por sus clientes.*"[6], podría ser interpretado como un comentario positivo sin sugerencia ya que aprecia el interés que tienen por los clientes. Pero también se puede interpretar como un comentario negativo donde sarcásticamente indica que no tienen interés por los clientes y una sugerencia a que activen las líneas (telefónicas).
- Ambigüedad: El lenguaje humano es ambiguo por naturaleza, por lo cual su forma escrita también es ambigua. La ambigüedad se puede clasificar en:
    - Léxica - Donde una palabra puede ser utilizada como verbo, sustantivo o adjetivo.
    - Semántica – Se da cuando una frase o comentario se puede analizar y entender de varias formas, como por ejemplo "*Arreglo el problema rápido*". Esto se puede interpretar como que, había más de un problema y lo solucionó de la forma más sencilla o también como que arregló el problema con rapidez.
    - Sintáctica – Esta ambigüedad también llamada estructural se presenta cuando existen dos o más significados posibles dentro de una frase o secuencia de palabras. El significado de esta frase puede, o no, ser determinado por su contexto. Ejemplo, la frase "*Cómprame un boleto*", puede interpretarse como que una persona está ofreciendo a la venta un boleto y otro significado podría ser que una persona le está pidiendo a otra persona que le compre un boleto.

[6] Bulnes,Ramon [@RamonBulnesN]. (2019, 30 de julio). *@interjet Sería más útil su aviso si sirvieran las líneas. Increíble el interés que tienen por sus clientes*. [Tweet]. Twitter. https://twitter.com/RamonBulnesN/status/1156402640927858690

2.2. Minería de Sugerencias

La minería de sugerencias es la extracción automática de sentimientos de texto no estructurado, donde una sugerencia se define como la expresión de un consejo o recomendación. Esta disciplina tiene su base en la minería de sentimientos, la cual estudia la extracción de sentimiento utilizando la Inteligencia Artificial (IA) y técnicas de PLN (Elyasir & Anbananthen, 2013). Aún cuando las sugerencias pueden poseer una polaridad de sentimientos positiva, negativa o neutra, la detección de sugerencias es algo relativamente nuevo que está llamando la atención por la riqueza del contenido que se puede extraer (Negi S. , 2016).

Para llevar a cabo la minería de sugerencias son necesarios varios pasos, durante los cuales se aplican varias técnicas y metodologías como son: recolección de información, limpieza de la información, extracción de características y clasificación.

Uno de los principales retos con los cuales se puede enfrentar al realizar minería de sugerencias son los problemas causados por un conjunto de datos no estructurados y altamente desequilibrados (Liu F. , Wang, Zhu, & Wang, 2019). Esto quiere decir que los comentarios, las reseñas o los tuits no han sido redactados de una forma estandarizada y que al momento de clasificarlos manualmente para construir el conjunto de entrenamiento existe una mayor cantidad de ejemplos de un tipo de clase que de otro. En el caso de las sugerencias, estas aparecen de una forma muy escaza en comparación de tuits sin sugerencias, lo que hace que el clasificador tenga una tendencia a clasificar la mayoría de los tuits como no sugerencias.

2.2.1. Limpieza de la información

Consiste en operaciones básicas como normalización, eliminación de ruido y el manejo de la pérdida de datos, datos faltantes o datos atípicos (Akora et al., 2011). Para poder utilizar la información es necesario pasarla por varias rutinas descritas a continuación.

2.2.2 Valores faltantes

Para el manejo de valores faltantes se pueden utilizar los siguientes métodos (Akora et al., 2011):

- Ignorar la *tupla*: No es muy efectivo y sólo debe utilizarse cuando la *tupla* contenga varios valores faltantes.
- Asignación manual: Sólo es viable cuando no existen muchas *tuplas* con valores faltantes, ya el trabajo puede llegar a ser costoso en horas hombre.
- Usar un valor constante: Se utiliza el valor que más se repita en los valores de las otras *tuplas* de la misma clase.
- Utilizar el promedio de los valores de la misma clase para asignarlo a la *tupla*.

2.2.3 Eliminación de ruido

El ruido es información corrupta, sin sentido o que se sale del rango establecido (Akora et al., 2011). En la minería de sugerencias dado el análisis de texto que conlleva, el ruido se manifiesta principalmente en dos formas:

1. En la estructura ambigua del lenguaje natural, ya que léxicamente una palabra puede tener varios significados dependiendo del contexto en el cual se esté utilizando.
2. En la ambigüedad de los datos contenidos en el lenguaje natural ya que, es difícil de identificar la ironía de forma textual, aun para los seres humanos.

El proceso realizado para la eliminación de ruido durante este proyecto es descrito en la Sección **3.2.2.,** donde se detalla el preprocesamiento realizado al texto proveniente de comentarios realizados en forma de tuits.

2.2.4 Integración de la información

La integración de información según Akora et al. (2011), se realiza al combinar y consolidar la información de múltiples fuentes en una sola base de datos coherente.

2.2.5 Transformación

La transformación de la información involucra el consolidar la información en una forma apropiada y puede incluir (Akora et al., 2011):

- Normalización: El atributo es escalado a un rango específico. Ejemplo: pasar un valor en un rango de 0 a 100 a un rango de 0 a 1.
- Agregación: Es el método de almacenar y representar datos en una forma resumida para ser analizados. Esta agregación se puede realizar calculando todos los datos durante un tiempo específico de dos formas, utilizando todos los puntos para un solo recurso y utilizando todos los puntos de un grupo de recursos.
- Generalización: La generalización de la información es la que sustituye valores de bajo nivel en una jerarquía por valores de alto nivel en dicha jerarquía.

2.2.6 Extracción de características

La extracción de características es el proceso de seleccionar un subconjunto de palabras que ocurren en un conjunto de entrenamiento C. Este subconjunto de palabras describe C, por lo tanto, se utilizan como características para la clasificación de texto dentro del dominio del conjunto C (Manning, Raghavan, & Schütze, 2008). La selección de características tiene dos propósitos generales:

1. Mejorar la clasificación eliminando las características con ruido, donde, el ruido, al ser agregada al subconjunto de palabras C, empeora la exactitud del clasificador.
2. Reducir el costo computacional en la clasificación al disminuir el número de las características utilizadas para describir el conjunto C.

2.2.7 Clasificación

La clasificación es el proceso de formar un modelo creado a partir de las características de un conjunto de datos para predecir la etiqueta desconocida de nuevos objetos (Akora et al., 2011).

Para un trabajo de minería de sugerencias se utiliza una clasificación supervisada de dos clases (Jakkampudi, 1999), donde el conjunto cuenta con etiquetas de clase asignadas previamente por un experto o grupo de expertos.

Dos de los métodos de clasificación existentes son los estadísticos y los árboles de decisión.

Algoritmos Estadísticos

Los algoritmos estadísticos utilizan funciones matemáticas aplicadas a información extraída de conjuntos de datos (IBM, 2020).

Uno de los algoritmos estadísticos más utilizados es *Naive Bayes*, el cual está basado en el teorema de Bayes y asume de forma ingenua que la ocurrencia de una característica es independiente de la ocurrencia de otras características. El teorema de Bayes en el que está basado también es conocido como probabilidad condicional y este dado por la función:

$$P(A/B) = \frac{P(B/A)\,P(A)}{P(B)}$$

Donde,

- $P(A/B)$ Es la probabilidad posterior o la probabilidad de que la hipótesis A ocurra durante el evento B.
- $P(B/A)$ Es la probabilidad condicional.
- $P(A)$ Es la probabilidad a priori o de la hipótesis antes de observar la evidencia.
- $P(B)$ Es la probabilidad de la evidencia.

Árboles de decisión

Un árbol de decisión es un algoritmo de aprendizaje supervisado y su meta principal es la de entrenar un modelo que pueda predecir la clase o valor utilizando reglas de decisión inferidas por datos de entrenamiento previamente etiquetados.

Partes de un árbol de decisión,

- Raíz: Representa el atributo más importante.
- Ramificaciones: Indica un posible resultado.

- Nodo de Probabilidad: Muestra múltiples resultados inciertos.
- Nodo de decisión: Indica una decisión a tomar.
- Hoja o nodo terminal: Indica un resultado definitivo.
- Poda: Eliminación de ramificaciones.

Un ejemplo de los árboles de decisión es *RepTree*, el cual es un algoritmo que crea un árbol utilizando la información de qué atributo es el más importante. Asimismo, poda el árbol totalmente creado empezando por las hojas. Cada vez revisa si el árbol podado tiene mejores resultados que el árbol anterior, antes de podar el último nodo o ramificación. Este proceso continúa hasta que la poda empieza a interferir con una alta exactitud (Khalaf, 2016).

2.2.8 Evaluación del clasificador

Para evaluar el rendimiento de cada clasificador se utilizó la plataforma de software WEKA (Witten et al., 2016) la cual es una coleccion de algoritmos de aprendizaje automatico y herramientas de preprocesamiento de datos que ayudan a realizar proyectos de mineria de datos y por consecuente de mineria de sugerencias. Haciendo uso de *WEKA* se genera automáticamente una matriz de confusión de 2×2 durante el proceso de prueba. Un ejemplo de una matriz de confusión se presenta en la Tabla 2, donde se pueden ver los cuatro resultados de la clasificación binaria: TP, FP, FN y TN. Estos cuatro resultados se pueden explicar cómo:

- TP (Positivos verdaderos): número de instancias etiquetadas como sugerencias que en realidad son sugerencias.
- FP (Falsos positivos): número de instancias etiquetadas como sugerencias que en realidad son no-sugerencias.
- FN (Falsos negativos): número de instancias etiquetadas como no-sugerencias que en realidad son sugerencias
- TN (Negativos verdaderos): número de instancias etiquetadas como no-sugerencias que en realidad son no-sugerencias.

Tabla 2: Ejemplo de una matriz de confusión para una clasificación de dos clases

		Etiqueta real	
		Sugerencia	**No-sugerencia**
Etiqueta prevista	**Sugerencia**	TP	FP
	No sugerencia	FN	TN

La exactitud de un clasificador son las predicciones correctas que se realizaron y se obtiene de la siguiente forma:

$$exactitud = \frac{TP + TN}{TP + FP + FN + TN}$$

La precisión de un modelo de clasificación es la proporción de identificaciones positivas (sugerencias) que fueron predichas correctamente por el clasificador, de todo lo que dijo que era positivo. Esto viene dado de la siguiente manera:

$$precisión = \frac{TP}{TP + FP}$$

Otra forma de evaluar un modelo de clasificación es por medio del *recall*, en donde se determina que proporción de instancias positivas reales (sugerencias) fueron predichas correctamente.

$$recall = \frac{TP}{TP + FN}$$

2.2.9 Validación

Para obtener una estimación de la precisión de los clasificadores seleccionados, se aplicó un método de validación cruzada tanto a *Bagging* como a *Naive Bayes*. La validación es un método utilizado para estimar la habilidad de predicción de un clasificador en datos nuevos, la idea de la validación cruzada usando cinco particiones es dividir el conjunto de datos en cinco particiones iguales usando cuatro de ellos como datos de entrenamiento y uno de ellos como datos de entrenamiento. La Tabla 3 muestra la rotación de las particiones de datos de prueba y de entrenamiento, donde durante cada una de las cinco iteraciones se utiliza una partición diferente para realizar las pruebas.

Tabla 3: Validación cruzada utilizando una variación de 5 iteraciones.

Iteración	1	2	3	4	5
Conjunto de datos	**Pruebas**	Entrenamiento	Entrenamiento	Entrenamiento	Entrenamiento
	Entrenamiento	**Pruebas**	Entrenamiento	Entrenamiento	Entrenamiento
	Entrenamiento	Entrenamiento	**Pruebas**	Entrenamiento	Entrenamiento
	Entrenamiento	Entrenamiento	Entrenamiento	**Pruebas**	Entrenamiento
	Entrenamiento	Entrenamiento	Entrenamiento	Entrenamiento	**Pruebas**

2.3. Trabajos relacionados

Actualmente existe un número limitado de aplicaciones enfocadas en la minería de sugerencias, la cual trata de identificar los comentarios que contengan sugerencias de manera implícita o explícita, en comparación con la cantidad de proyectos creados para realizar la minería de opiniones. La minería de sugerencias se ha aplicado a reseñas con dominio abierto (Nagi & Buitelaar, 2015), en redes sociales (Nagi & Buitelaar, 2015) (Negi, Asooja, Mehrotra, & Buitelaar, 2016) y en productos (Brun & Hagege, 2013).

Unas de las principales limitaciones de las aplicaciones de minería de sugerencias es el contexto en el que se manejan los comentarios a clasificar, ya que regularmente utilizan comentarios en inglés.

Una sugerencia, según la Real Academia, es una insinuación, inspiración, idea que se insinúa (Real Academia Española, 2019). En general, esta definición de sugerencia hace posible la distinción de las sugerencias de otro tipo de comentarios.

Los primeros trabajos relacionados a la detección de necesidades del cliente estaban más enfocados a los deseos de los clientes (Goldberg, et al., 2009) (Ramanand, Bhavsar, & Pedanekar, 2010), donde trataban de detectar qué quería el cliente, en forma de características que deseaban de un producto o de sugerencias para la decisión de comprar un producto. Estos sistemas son de contexto libre y utilizan un diccionario para detectar los deseos tanto de productos como de año nuevo.

El concepto de la minería de sugerencias se puede decir que es reciente, el primer caso relacionado es del 2013 cuando Brun y Hagège (2013) notaron las crecientes fuentes de información en línea como sitios de reseñas y blogs personales. Desarrollaron un sistema de extracción automático de sentimientos para tener un mejor entendimiento de las opiniones y subjetividad de las personas. Al estar desarrollando el sistema se dieron cuenta que dentro de las opiniones y los sentimientos que estaban siendo expresados, se encontraba información interesante por parte de los clientes. Recopilar este tipo de información no suele estar contemplado dentro de las aplicaciones de minería de sentimientos. No obstante, el conocer el grado de satisfacción de los clientes, permitiría idear mejoras ya sea para incluir características o componentes que el cliente desea, o para indicar cuando un cliente lamenta la ausencia de una característica o componente.

Para detectar sugerencias el sistema utilizó cuatro componentes principales, los cuales también se detallan en la patente que adquirieron en el 2014 (Brun & Hagege, 2013). Estos componentes fueron:

- Una terminología estructurada: Consiste en un vocabulario de productos de los cuales se quiere conocer información, por ejemplo: *Printer, Copy machine, Scanner y Product*.
- Un diccionario de palabras: Contiene las palabras utilizadas con frecuencia en comentarios que denotan sugerencias, como: *I would like, I suggest, missing, I miss*.
- Un analizador lingüístico: Contiene las características morfológicas de las palabras y utiliza los componentes de terminología estructurada y el diccionario de palabras para detectar la sugerencia y determinar a qué producto se aplica.
- Un extractor de sugerencias: Extrae las sugerencias utilizando un conjunto de patrones semánticos y sintácticos. los cuales trata de emparejar con reseñas previamente analizadas.

El corpus utilizado en este proyecto fue construido a partir de 3500 reseñas de impresoras de la página "*Epinion*". Para probar el sistema creado se utilizaron 3440 reseñas de entrenamiento y 60 reseñas al azar como casos de prueba. Los resultados que se describen son de una exactitud del 77%.

Los siguientes trabajos de minería de sugerencias fueron presentados por Negi y Buitelar en el 2015, donde empezaron a identificar que en las reseñas utilizadas para la minería de sentimientos podían encontrar un "humor subjetivo" (Sepna & Buitelaar, 2015), el cual es un deseo que se describe como la descripción de una acción que aún no ocurre. Al solo tratar de identificar deseos y sugerencias, formaron un corpus de frases obtenidas de páginas y foros de gramática.

En el 2015, se presentó una propuesta para la detección de sugerencias de cliente a cliente, (Nagi & Buitelaar, 2015). Este proyecto se enfocó en extraer las sugerencias de comentarios de hoteles en *TripAdvisor* y de comentarios sobre aparatos eléctricos en *Yelp*. El objetivo fue además detectar a quién estaba dirigida la sugerencia, qué se sugiere y si la sugerencia está implícita o explícita en el comentario.

Para desarrollar este sistema, se recabaron 8050 comentarios de hoteles y 3782 comentarios de aparatos eléctricos. Los comentarios se etiquetaron de forma manual según su sentimiento positivo, negativo o neutro. Este etiquetado se hizo a base de *crowdsourcing* utilizando una plataforma llamada *Crowdflower*, la cual anota un puntaje de confianza de acuerdo con el porcentaje de etiquetamiento que se le ha dado. Para poder validar un comentario, el nivel de confianza mínimo del etiquetado para cada comentario era de 0.6.

La detección de sugerencias se basó en la identificación de palabras claves de un diccionario obtenido de *WordNet* y del etiquetado gramático de las palabras de un comentario. La clasificación se hizo a partir de una máquina de soporte vectorial en *WEKA* y para estimar el error se utilizó la validación cruzada con diez particiones. Los autores concluyeron en su investigación que las sugerencias no pudieron ser clasificadas de forma correcta ya que, las características sintácticas y léxicas eran inefectivas en la mayoría de los casos.

La minería de sugerencias también se aplicó en las redes sociales, el primero de ellos fue presentado por Pitchayaviwat (2016) al implementar un sistema que recababa información en Facebook y Twitter de los clientes de servicios de seguros, donde se detectaban las quejas

de los usuarios y se les daba prioridad para mantener un estándar de calidad establecido por la compañía de seguros.

Para este sistema se utilizaron 800 comentarios, los cuales, se representaron por medio de un vector de características, empleando para ello las palabras que se repetían más de cinco veces en el corpus. Para la agrupación se utilizó un modelo *K-means* con el cual se crearon tres grupos: el grupo uno contenía preguntas del usuario sobre las pólizas de seguro, el grupo dos era retroalimentación acerca de servicios de reclamo y el grupo tres abarcaba las preguntas de información general.

De forma similar, las sugerencias se han utilizado para encontrar las áreas de oportunidad en los comentarios de alumnos hacían a sus profesores en el proyecto desarrollado por Gottipati (2018). Este trabajo trataba con las encuestas de cada semestre de varias instituciones, que estaban conformadas por dos partes, una cuantitativa donde se evalúa en una escala numérica y otra cualitativa, donde el alumno hace comentarios relacionados al método de enseñanza del profesor, el contenido de la clase y el aprovechamiento.

En este proyecto se construyó una base de datos de encuestas, las cuales, fueron procesadas de dos formas: 1) la generación de un promedio de las calificaciones obtenidas para la parte cuantitativa y 2) la identificación de sugerencias utilizando los métodos de procesamiento de texto de Negi y Buitelar (2016) (2017) y un clasificador de árbol de decisión.

Dado que el proyecto trata con encuestas con dos tipos de características uno cuantitativo derivado del promedio y uno cualitativo, donde se pueden encontrar sugerencias dentro de los comentarios de los alumnos, los autores le asignaron la carga cuantitativa derivada del promedio de cada encuesta a su comentario asociado, esto con el fin de denotar la prioridad en la que se deben atender las sugerencias.

La base de datos utilizada para realizar el proyecto incluyó 5342 instancias de una escuela en Singapur y todas fueron manualmente etiquetadas como positivas, negativas, con

sugerencia o sin sugerencia. Los autores mencionan que la eliminación de *stopwords*, la cual es una técnica utilizada para mejorar el rendimiento, redujo la exactitud del clasificador de un 0.78 a 0.18, por lo cual omitieron dicho paso de eliminación.

Durante el 2019 se llevó a cabo la competencia SemEval-2019 como parte del *International Workshop on Semantic Evaluation*. La tarea 9 de dicha competencia estaba enfocada a la minería de sugerencias de reseñas en línea y de foros. Al ser un evento patrocinado por Microsoft, el conjunto de datos que se empleó para esta competencia contenía en el conjunto de entrenamiento 1428 sugerencias y 4356 no sugerencias, en el caso del conjunto de validación 296 sugerencias y 296 no sugerencias.

Durante esta competencia trabajaron con BERT (*Bidirectional Encoder Representations from Transformers*), el cual es un modelo previamente entrenado que produce un vector de una frase (Liu, Wang, & Sun, 2019). Asimismo, se obtuvieron buenos resultados mediante el uso de *Masked Language Modeling* (Masked LM) para crear un modelo que se entrena sin etiquetas y con supervisión propia (Devlin, Chang, Lee, & Toutanova, 2019).

En cuanto al análisis de la satisfacción del cliente también se han realizado proyectos utilizando modelos de regresión empotrados (García, et al., 2019) con lo que se busca predecir qué tan satisfecho estará un cliente, pero no extrae o detecta sugerencias como es lo que se busca en este proyecto.

Hasta el momento no se ha encontrado alguna una herramienta que ayude a realizar una minería de sugerencias sin muchas complicaciones ni en idioma inglés, ni en español. Existen algunos libros (Pedrycz, et al., 2021) y (Pazos-Rangel, et al., 2021) donde se presentan proyectos de resolucion de problemas o tutoriales en internet donde se muestra paso a paso cómo realizar la minería de sugerencias utilizando comentarios en inglés previamente etiquetados. El principal problema de estos tutoriales es que los comentarios que se utilizan para entrenar el modelo de clasificación no requieren preprocesamiento, los comentarios ya están clasificados y el corpus de entrenamiento es pequeño y sin mucha

ambigüedad, ya que los comentarios fueron previamente seleccionados para que los clasificadores den un mejor resultado, por lo cual no representan un ejemplo del mundo real.

3. Propuesta de solución

La propuesta de solución se detalla durante esta sección y se divide en la metodología de desarrollo, donde se expone de qué forma se realizó la solución, así como la creación de la solución cognitiva donde se muestran los pasos que se siguieron durante el proyecto.

3.1. Metodología basada en prototipos

Para el desarrollo del software de clasificación de sugerencias se utilizó una metodología basada en prototipos o en prototipado. En esta metodología cíclica se construye un producto funcional que se aproxima al software final que se prueba y se retrabaja según los resultados (Ganzabal-Garcia, 2015). Usualmente esta metodología está compuesta por cuatro etapas: la etapa de planificación, la etapa de construcción, la etapa de pruebas y la etapa de despliegue. Estas 4 etapas se pueden observar en la Figura 1 y las actividades que se llevaron a cabo en cada etapa consistieron en:

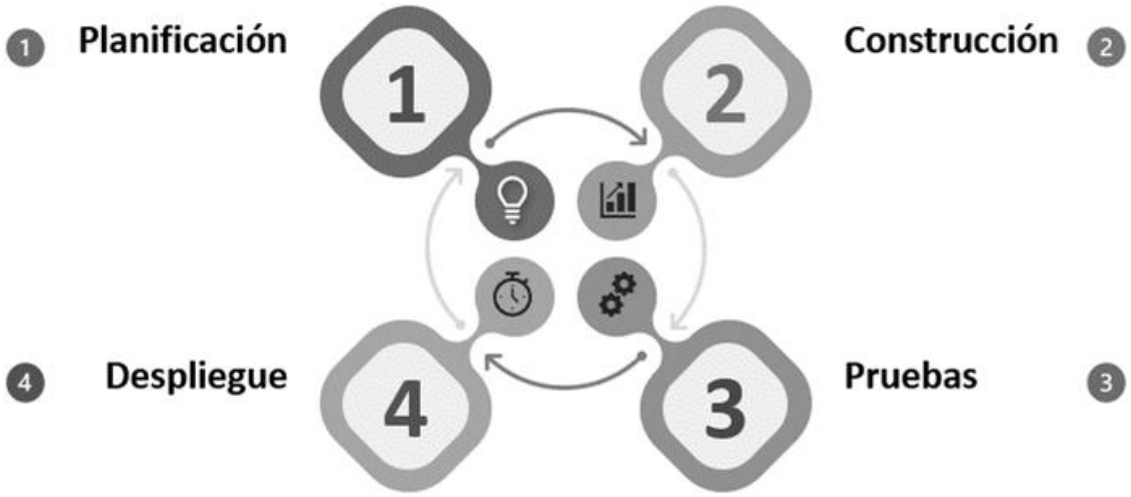

Figura 1: Diagrama de la metodología de desarrollo por prototipos.

1. Etapa de planificación: Durante esta etapa se realiza la determinación de los requerimientos, el diseño del producto y se definen los objetivos y las metas.
2. Etapa de construcción: Se desarrolla el prototipo según los requerimientos.
3. Etapa de pruebas: Se aplican los casos de prueba de acuerdo con los requerimientos establecidos para el prototipo.

4. Etapa de despliegue: Se recibe la retroalimentación de las pruebas y se toman las decisiones sobre el prototipo. Se puede mejorar o desechar y diseñar otro nuevo.

3.2. Creación del producto

Los pasos requeridos durante la minería de sugerencias siguen las acciones indicadas en el análisis de sentimientos, por lo cual se tomó como guía el desarrollo de análisis de texto con opiniones desarrollado por Negi y Buitelar (2015) , en donde se realizaron los siguientes pasos:

1. La recolección de tuits creados por usuarios de aerolíneas.
2. El preprocesamiento de los comentarios.
3. La creación de un vector de características.
4. La clasificación de los comentarios.

Las acciones realizadas durante estas etapas se describirán brevemente en las siguientes secciones.

3.2.1. Recolección de comentarios

Para la recolección de tuits se utilizó la aplicación *Twitter Archiver* (Digital Inspiration, 2020), la cual trabaja en la nube con *Google Sheets*. Durante 79 días, entre el 09 de agosto y el 27 de octubre del 2019, se recolectaron 21161 tuits de las cuentas de Twitter de diversas aerolíneas mexicanas, tales como: @AeroMexico, @interjet, @flyvolaris, @VivaAerobus, @AeromarMx, @Magnicharters y @vuelaTAR correspondientes a las empresas Aeroméxico, Interjet, Volaris, Viva Aerobús, AEROMAR y TAR respectivamente. En la Figura 2 se puede observar una pantalla de Twitter Archiver para la creación y modificación de reglas de búsqueda.

Durante el periodo de recolección se realizó de forma automática una búsqueda de tuits nuevos cada 15 minutos. Estos se escribían en una hoja de cálculo.

Debido a que los comentarios en Twitter carecen de un etiquetado que los identifique como sugerencias, se realizó un etiquetado multitudinario (Chai, Fan, Li, Wang, & Zheng, 2018), el cual consiste en utilizar personas ajenas al proyecto, para que etiqueten los

comentarios y así facilitar la identificación de las sugerencias dentro del conjunto de comentarios.

El etiquetado se llevó a cabo con la ayuda de estudiantes de tercer y quinto semestre de las carreras de Ingeniería de Software e Ingeniería en Sistemas Computacionales de la División Multidisciplinaria en Ciudad Universitaria de la UACJ.

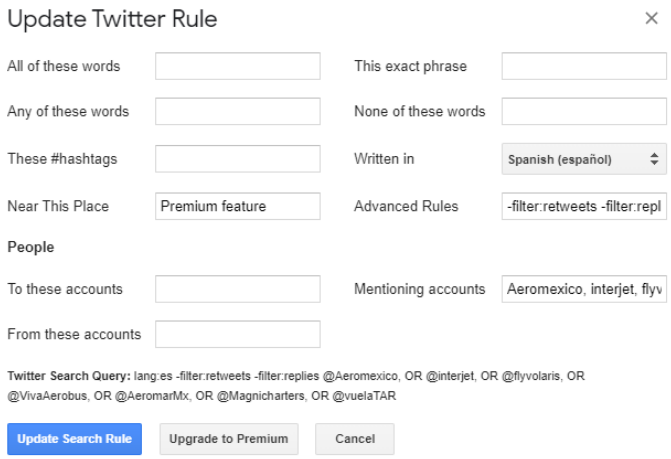

Figura 2: Imagen de la aplicación Twitter Archiver donde se muestran las cuentas monitoreadas para la búsqueda de tuits.

El conjunto de datos de 21161 instancias se redujo a 3657 removiendo todos aquellos tuits que:

1. Adjuntaran una imagen sin comentario.
2. Escribieran *hashtags* sin comentario.
3. El comentario fuera menor a tres palabras.

Una vez removidos los tuits sin comentarios útiles se partió el conjunto de datos en 74 subconjuntos, donde cada subconjunto contenía 50 comentarios. Ejemplo de cómo se guardan los subconjuntos etiquetados en Google drive se puede apreciar en la Figura 3. Cada subconjunto se etiquetó dos veces por estudiantes de diferentes grupos y se evaluaron aquellos comentarios cuyas etiquetas generaran discrepancia, para llegar a un consenso sobre la etiqueta que debería tener el comentario.

Por último, se volvieron a integrar todos los subconjuntos ya etiquetados en un solo conjunto de datos. De este conjunto se volvió a partir en dos subconjuntos, uno de comentarios con sugerencias y otro de comentarios sin sugerencias esto con la finalidad de llevar un mejor control de las instancias de comentarios según su etiqueta de clase.

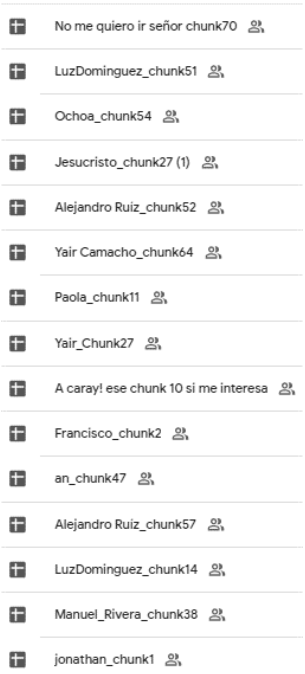

Figura 3: Ejemplo de los subconjuntos separados para utilizar el etiquetado multitudinario.

3.2.2. Preprocesamiento de comentarios

Dado el uso del lenguaje informal por parte de los usuarios de Twitter, el preprocesamiento de texto es un paso importante, ya que ayuda a reducir el ruido, la variación entre las palabras en el conjunto de comentarios y facilita el análisis del texto.

Antes de crear el vector de características de comentarios, se realizó un preprocesamiento manual, que consistió en la validación del contenido del comentario y sus etiquetas, así como el preprocesamiento automático para la normalización del texto utilizando la biblioteca NLTK de herramientas disponibles en Python 3.6, que consistía en:

- Tokenización de cada cadena de comentarios en una lista de palabras de las cuales se compone un comentario. La tokenización es una forma útil de preparar un comentario

para la automatización, ya que al estar en una cadena es posible iterar fácilmente cada palabra de un comentario y aplicar las funciones de normalización. Ejemplo de tokenización en Python 3 usando *tokenize* de NLTK se puede ver en el Apéndice A.

- El cambio de mayúsculas a minúsculas en todos los comentarios permitirá minimizar los posibles errores al comparar palabras. Un ejemplo del cambio de mayúsculas a minúsculas en un comentario se puede ver en el Apéndice B.
- Eliminación de acentos, ya que los usuarios no tienden a acentuar correctamente las palabras en sus tuits ya sea por omisión o ignorancia, lo que puede causar desajustes en las palabras. En el Apéndice C se encuentra un ejemplo de eliminación de acentos mediante el reemplazo de letras en Python 3. La eliminación de los acentos también puede ser realizada utilizando la librería unidecode incluida en Python 3, mediante el código del Apéndice D.
- Eliminación de emoticonos, comillas inglesas y guiones bajos. Este tipo de caracteres, como los emoticonos, no aportan información significativa cuando se buscan sugerencias, ya que son típicamente utilizados para expresar la emoción del emisor, no una idea. Mediante el código del Apéndice E, se pueden eliminar los caracteres especiales en Python 3.
- Eliminación de *Stopwords*, también conocidas como palabras vacías, en español usando la biblioteca Python NLTK. Las palabras de detención son términos comunes como artículos y determinantes, entre otros. No contienen información significativa o su contenido semántico es insignificante. Otras palabras que deben eliminarse son aquellas que comienzan con '@' o '#', ya que están destinadas a etiquetar a otros usuarios en tuits o crear una etiqueta tipo *hashtag*. Por lo tanto, no agregan información relevante que ayude a determinar una idea de producto. Una descripción detallada de la eliminación de *Stopwords* se encuentra en el Apéndice F.
- Eliminación de caracteres numéricos ya que no contienen una sugerencia dentro de sí mismos, por lo tanto, se pueden descartar de forma segura de los comentarios (vea Apéndice G).
- Validación de las palabras en los comentarios contra una lista de palabras existentes en el CREA, o Corpus de Referencia Actual en Español de la Real Academia Española (RAE) (Real Academia Española, 2015), la cual es una lista de palabras aprobadas por

la RAE como relevantes en conversaciones y publicaciones en línea. Este paso tiene como objetivo verificar que cada palabra en un comentario existe en el idioma español. Esta operación es necesaria ya que todos los comentarios en el repositorio provienen de una red social, donde los tuits se escriben de manera informal y donde comúnmente se alargan palabras para dar énfasis escribiendo una letra varias veces, (por ejemplo, "*graciaaaas*"), o sustituyendo palabras enteras por letras simples (por ejemplo, "*está bien* → "*k*"). Por simplicidad y para evitar ingresar a otra área de PNL, una palabra marcada como inválida, o no en la lista CREA, no puede considerarse una palabra real y por lo tanto es eliminada del comentario.

- Extracción de la raíz de cada una de las palabras en los comentarios para llegar a su forma básica usando la librería *SnowballStemmer* de la biblioteca Python NLTK.

3.2.3. Creación de un diccionario de sugerencias

Un diccionario para el análisis de texto es como un mapa que guía la conversión de texto no estructurado en datos estructurados (vectores). Al trabajar en el análisis de sentimientos, es probable que haya un diccionario disponible en la web o que use nltk.download ('opinion_lexicon') si el proyecto se está desarrollando en Python. Dado que la minería de sugerencias es relativamente nueva y no hay diccionarios disponibles para un dominio que se ocupe de sugerencias para tuits de aerolíneas, se deberá construir un diccionario específico para esta solución.

El momento correcto para crear un diccionario de características es después de realizar el procesamiento previo en un corpus de texto. Este procesamiento previo se mencionó en la Sección 3.2.2 tanto en los subconjuntos de sugerencias como en los comentarios que no son sugerencias, y se puede hacer con la ayuda de una distribución de frecuencia de palabras en cada uno de los subconjuntos.

Al usar *Gutenberg Corpus*, parte de NLTK, se pueden cargar fácilmente los repositorios con comentarios que contienen sugerencias para aplicar una distribución de frecuencia. Para crear un corpus NLTK se utiliza el siguiente comando,

corpus = nltk.corpus.gutenberg.words ("Comentarios con sugerencias.txt"),

donde "Comentarios con sugerencias.txt" es el repositorio que contiene comentarios con sugerencias. Asimismo, se deberá crear otro corpus con un nombre diferente para el repositorio que contenga comentarios sin sugerencias.

Después de cargar el repositorio como un corpus, se debe aplicar una función de distribución de frecuencia y tabular.

freqdist = nltk.FreqDist (corpus)
freqdist.tabulate ()

Esta distribución de frecuencias crea una lista de tuplas que consiste en cada una de las palabras en el corpus y el número de veces que una palabra apareció. Tener una gran cantidad de ocurrencias en un subconjunto puede describir un subconjunto al revelar sus contenidos más relevantes. Como esta distribución de frecuencia no contiene palabras de detención, se puede decir que las palabras más frecuentes son las más importantes para describir su subconjunto y que las menos frecuentes son las menos importantes. Después de que la distribución de frecuencias de un subconjunto fue construida, se calcularon las palabras que menos se repiten y se eliminaron aquellas que aparecieron menos de cinco veces a lo largo del subgrupo, puesto a que estas se definen como poco frecuentes y son las menos probables a volver a aparecer en un nuevo tuit a clasificar en el futuro.

Ejemplo de un diccionario con palabras que aparecen más de cinco veces en todo el corpus:

```
SuggestionDictionary = list (filter (lambda x: x [1]> 4 y (len (x [0])> 1), freqdist.items ()))
```

El número de corte utilizado para eliminar las palabras frecuentes es un tema difícil, ya que se debe tomar en consideración cuántas palabras hay en la distribución o si hay pocas palabras, de tal manera que, si eliminan esas palabras que solo aparecieron dos veces y el

resultado reduce un tercio de su distribución, tal vez no sean tan poco frecuentes y solo sea posible eliminar las que solo aparecen una sola vez. Encontrar un equilibrio entre lo común y lo poco frecuente solo se puede hacer probando los resultados de la eliminación y los ajustes.

Lo último que se necesita para crear un diccionario es crear una lista de palabras de la lista de tuplas *SuggestionDictionary*. Esto se puede hacer copiando todas las palabras en la lista de tuplas mediante la siguiente línea de código:

```
SuggestionDictionaryList = [palabra para (palabra, _) en SuggestionDictionary]
```

Con una lista de palabras utilizadas en los comentarios de sugerencias, almacenadas en SuggestionDictionaryList, se necesita agregar este conjunto de palabras a la lista de palabras utilizadas en los comentarios que no son sugerencias. Esto con el objetivo de conformar el diccionario final. Durante el proceso de adjuntar una lista a la otra es se eliminan también cualquiera de las palabras que puedan resultar duplicadas.

En la lista final el número de palabras utilizadas para formar el diccionario de sugerencias, después de eliminar palabras poco frecuentes, fue un total de 556 palabras.

3.2.4. Creación de un vector de características

Un vector de características es un vector compuesto de n elementos o atributos que describen numérica o simbólicamente las propiedades de un objeto. En este proyecto el objeto es un comentario hecho en Twitter y el vector de características representará matemáticamente un comentario de texto. Los vectores de características se usan ampliamente en aprendizaje automático debido a su capacidad de representar objetos de una manera numérica que ayudan al análisis del objeto por un clasificador.

Para crear un vector de características a partir de un comentario, se necesitan tomar los atributos previamente definidos en un diccionario de sugerencias y con la ayuda de una lista nueva se crea un vector con la capacidad del tamaño del diccionario.

Los valores del vector de características serán 0 o 1, en donde, 0 significa ausencia de la palabra y 1 si la palabra existe en el comentario.

El Algoritmo 1 es un ejemplo de la creación de un vector de características del tamaño del número de palabras contenidas en el diccionario, su inicialización, y cómo se les asignan valores a sus características de acuerdo con las palabras extraídas de un comentario.

Algoritmo 1: Algoritmo para la creación de un vector de características

```
  featureVector = [0] * (len(dictionary)) //crea un vector de
tamaño diccionario

  for w in tweetComment:
      w = spanishstemmer.stem(w)
      featureIndex = find_element_in_list(w, dictionary)
      if featureIndex > -1:
         featureVector [featureIndex] = 1

  stringVector = ", ". join (map (str, featureVector))
fileWrite.write(stringVector + "\n")
```

Para representar el conjunto de datos en su totalidad se necesita crear un vector de características para cada uno de los comentarios en el conjunto de datos. Después de crear todos los vectores, existe la posibilidad de crear vectores duplicados los cuales tienen que ser eliminados. El resultado final fue de 3330 instancias en el conjunto de datos para los comentarios que contienen sugerencias y comentarios sin sugerencias.

3.2.5 Clasificación de comentarios

Para detectar las sugerencias se utilizaron dos algoritmos de clasificación: *Bagging* y *Naive Bayes.*

Bagging

Es un método combinado de clasificación que utilizan N algoritmos de aprendizaje iguales utilizando distintos subconjuntos de entrenamiento creados a partir de un remuestreo con reemplazamiento. Para clasificar una nueva instancia se realiza una predicción en los N algoritmos y se realiza un voto mayoritario para determinar la clasificación de una nueva instancia.

Naive Bayes

Los clasificadores probabilísticos hacen suposiciones sobre la independencia de las características en las clases.

3.3. Configuración de experimentos

Todas las pruebas se realizaron con el software WEKA y una validación cruzada de cinco particiones. En la Tabla 4 se describe brevemente la base de datos empleada. Los comentarios contenidos en la base de datos pasaron por un preprocesamiento descrito en la Sección 3.2., donde finalmente solo quedaron 88 sugerencias y 3407 comentarios de no-sugerencias.

Para los clasificadores Bagging y Naive *Bayes*, se utilizaron las configuraciones por defecto que utiliza WEKA, donde el algoritmo de *Bagging* utiliza un algoritmo de clasificación *Reduced Error Pruning Tree (RepTree)* el cual es un algoritmo de árbol de decisión.

Tabla 4: Descripción breve de la base de datos utilizada en los experimentos.

Número de instancias			**Número de atributos**
Sugerencias	No Sugerencias	Total	Total
88	3407	3495	556

4. Resultados

Para comparar el rendimiento de los clasificadores *Bagging* y *Naive Bayes*, se muestran los resultados de exactitud general de los clasificadores en la Tabla 5, donde se puede observar que el algoritmo *Bagging* obtuvo los mejores resultados, pero, como se observa en la Tabla 7, todas las sugerencias se clasificaron incorrectamente utilizando el clasificador *Bagging*.

Tabla 5: Resultados de exactitud global por clasificador.

	Bagging	***Naive Bayes***
Exactitud global	0.9748	0.9387

Los resultados de alta exactitud y la baja precisión, la cual se muestra en la Tabla 6, son una indicación del alto grado de desbalance que existe entre las clases por el hecho de que las sugerencias no son dadas comúnmente por los usuarios tal como se ha mencionado en la literatura por Negi (2016),

Tabla 6: Resultados de precisión por clasificador.

	Bagging	***Naive Bayes***
Precisión	0	0.215

Como se puede apreciar en la Tabla 7, el *recall* de cada algoritmo es muy bajo, esto quiere decir que se predijeron correctamente muy pocas de las sugerencias, el número exacto de las cuales se puede ver en la matriz de confusión de ambos clasificadores en las Tabla 8 y Tabla 9.

Tabla 7: Resultados de recall por clasificador.

	Bagging	***Naive Bayes***
Recall	0	0.115

Tabla 8: Matriz de confusión para el clasificador Bagging.

	Bagging	
	Sugerencias	**No Sugerencias**
Sugerencias	0	88
No Sugerencias	0	3407

Para lograr un mejor *recall* para las sugerencias, se aplicó un submuestreo de la clase mayoritaria utilizando el filtro *SpreadSubsample* incluido en WEKA con una distribución de reducción de 3.0 con esto se trata de eliminar o minimizar el problema de desbalance (Rivera, et al., 2020) entre clases que existe. Esto da una submuestra de la clase mayoritaria de tres veces el tamaño de la clase minoritaria, en este caso, crea una submuestra de 264 instancias de no sugerencias ya que la clase minoritaria de sugerencias está compuesta de solo 88 instancias.

Tabla 9: Matriz de confusión para el clasificador Naive Bayes.

	Naive Bayes	
	Sugerencias	**No Sugerencias**
Sugerencias	19	69
No Sugerencias	145	3262

Una clasificación que utiliza submuestreo y *Naive Bayes* reduce la exactitud, pero aumenta el puntaje de *recall* de 0.115 a 0.511 al aumentar el número de sugerencias clasificadas correctamente como se muestra en la Tabla 10.

Tabla 10: Matriz de confusión del clasificador Naive Bayes y una submuestra de la clase mayoritaria.

	Naive Bayes	
	Sugerencias	**No Sugerencias**
Sugerencias	45	43
No Sugerencias	49	213

Dada los bajos valores de precisión y *recall* en ambos clasificadores, es necesario ver cómo el conjunto de datos se visualiza en un espacio bidimensional. Primero se utilizó *tSNE* para crear un mapa en dos dimensiones que muestra cómo se organizan las instancias. El método *tSNE* es un proceso para reducir la dimensionalidad mediante el uso de relaciones locales entre puntos para crear un mapa de baja dimensión del conjunto de datos (van der Maaten, 2020). Cuando se usó *tSNE* para representar la base de datos original de 3330 instancias en dos dimensiones, se obtuvo un mapeo que se muestra en la Figura 4, donde se exponen las instancias marcadas como sugerencias en azul y no sugerencias en rojo. Esta visualización puede dar la idea de cómo se agrupan las clases y, como se observa, las sugerencias se

superponen a las no sugerencias. Estas agrupaciones que se crean indican la formación de pequeños disyuntos, los cuales se caracterizan por tener una alta tasa de error de clasificación.

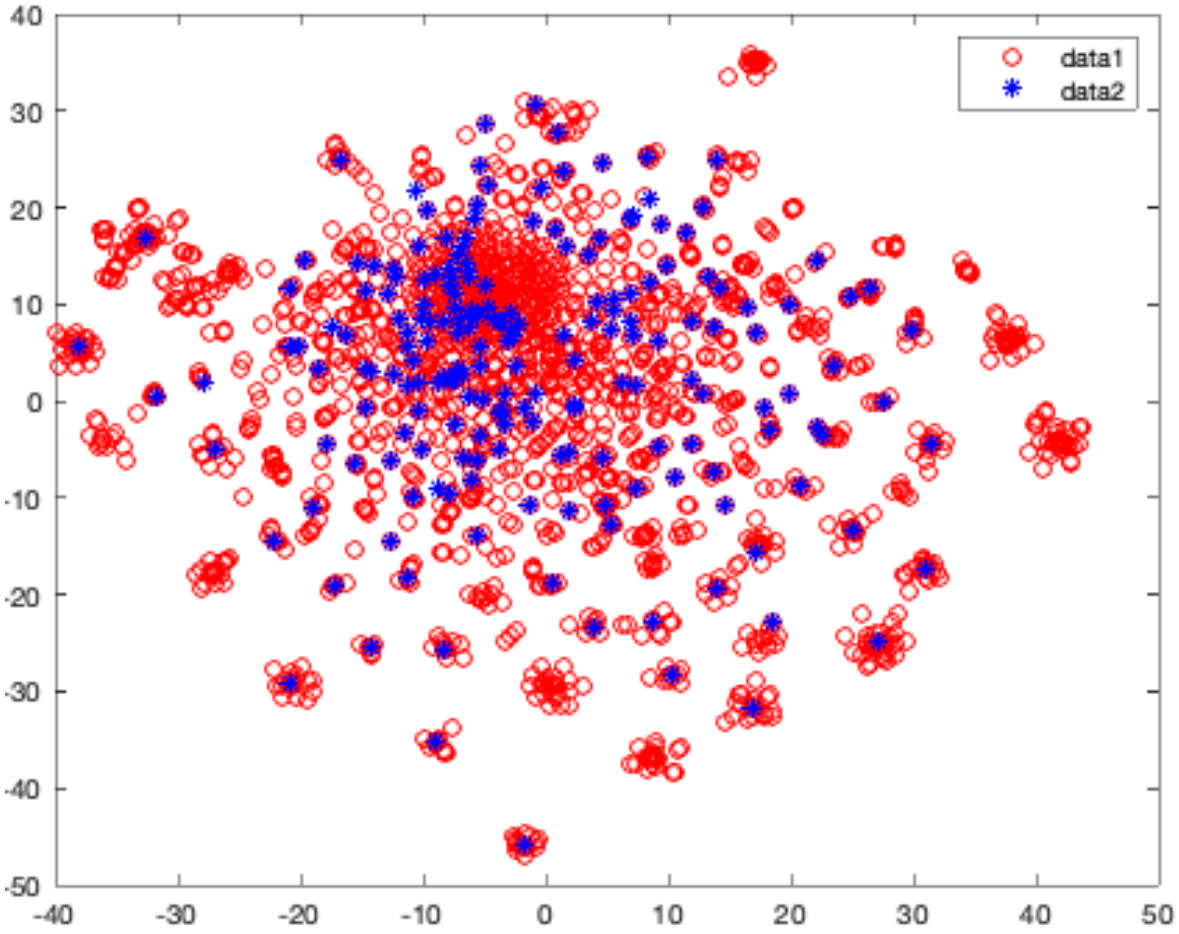

Figura 4: Diseño bidimensional del conjunto de datos utilizando *tSNE*.

Como se puede ver, la formación de pequeños disyuntos en el mapa *tSNE*, se creó un mapa de calor en MATLAB para visualizar la matriz de datos en dos dimensiones. La imagen completa se muestra en la Figura 5. El mapa de calor ayuda a visualizar la existencia de una característica en alguna parte de los vectores de instancias, dado que estos vectores fueron creados a partir del diccionario de características, en el cual se ordenaron primero las características de las sugerencias y después las de no sugerencias, lo que significa que podemos ver en el mapa si los vectores con sugerencias poseen características de las no sugerencias y viceversa.

Para ilustrar mejor las características exhibidas por el mapa de calor del conjunto de vectores de características en la Figura 5, el segundo cuadrante A1 de la imagen se aprecia en la Figura 6 donde las regiones mostradas en B1 y B2 contienen las instancias de sugerencia y B3 y B4 contienen las instancias de no sugerencia. Como se puede ver, hay pequeños puntos en las cuatro regiones, estos pequeños puntos representan la presencia de una característica en una instancia. En el mejor de los casos, solo se debería notar la presencia de características en las regiones B1 y B4 indicando que existe una clara división en las características que describen un tuit con sugerencias y a uno sin sugerencias. Dado que el conjunto de datos utilizado muestra la presencia de características en B2 y B3, se puede deducir que las características utilizadas para describir una sugerencia también están presentes en los comentarios que no son de sugerencia, y viceversa.

Figura 5: Mapa de calor de los datos contenidos en el conjunto de datos.

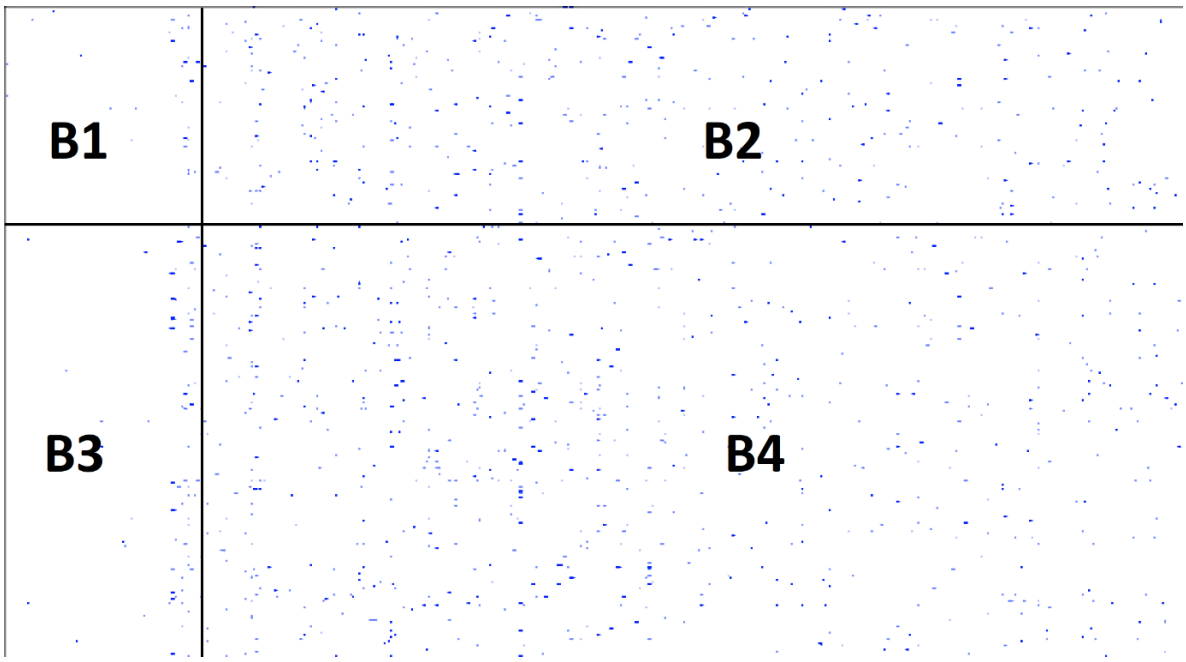

Figura 6: Segundo cuadrante del mapa de calor.

Dado que las características extraídas de los tuits con sugerencias contienen palabras que no son exclusivas solo a esa clase, estas palabras son comunes a un contexto de aerolíneas, como las palabras: "*vuelo*" que se repite 1260 veces, "*horas*" que se repite 312 veces y "*servicio*" la cual se encontró 316 veces. La existencia de palabras comunes en ambas puede llevar a la creación de pequeños disyuntos, los cuales tienen un mayor efecto en la clasificación errónea de tutis que el desbalance existente entre clases (Jo & Japkowicz, 2004).

La solución propuesta obtuvo la mejor precisión con 0.215 utilizando el algoritmo de clasificación *Naive Bayes*.

Los bajos resultados de clasificación arrojados causados por la aparición de pequeños disyuntos en la solución propuesta ya se han mostrado con anterioridad como lo dice Van der Bosch, et al. (1997). La aparición de este problema es dada a que los datos que han sido extraídos del lenguaje contienen en si pequeños conjuntos de instancias clasificadas de la misma forma, las cuales pueden ser de un tamaño de tres a cien instancias.

5. Conclusiones y trabajos futuros

Dado a que las características extraídas de los comentarios de Twitter con sugerencias no son exclusivas de esa clase, crea pequeñas disyunciones, estas representan características comunes y tienen más sensibilidad para el clasificador a medida que hace su predicción sobre nuevos datos.

Para intentar solucionar esto, se trató de modificar el conjunto de datos al:

- Utilizar solo se comentarios con sugerencias directas, las cuales son las sugerencias que explícitamente indican una sugerencia y tienen las palabras "*sugiero*", "*sugeriría*", "*deberían*" y "*necesitan*", entre otras. Al hacer esto no se encontraron mejoras a la detección de sugerencias.
- Etiquetar los comentarios con sugerencias indirectas o implícitas como no sugerencias. Al hacerlo la exactitud mejoró, pero a expensas de perder casi un sexto de las sugerencias, lo cual no fue útil ya que la meta de la solución trataba de detectar sugerencias.
- Crear bigramas y trigramas, los cuales son subsecuencias de dos y tres palabras encontradas en los tuits. No se mejoró en la predicción y se empeoró en el tiempo necesario para entrenar el modelo ya que los vectores llegaban a tener 29000 características.
- Utilizaron comentarios sin sugerencias de domino diferente donde si se logró una mejora en la detección de sugerencias.
- Utilizaron comentarios sin sugerencias de domino diferente utilizando bigramas y trigramas y también se notó una mejora en la detección de sugerencias.

La utilización de comentarios con dominio diferente se hizo principalmente para poder probar si las sugerencias eran posibles de detectar aun cuando las palabras sean de dominio distinto, con esto se llegó a la conclusión a que las palabras repetidas en ambas clases tienen mucha importancia e influyen en la clasificación de instancias de la clase minoritaria como pertenecientes a la clase mayoritaria.

Como trabajo futuro se propone solucionar el problema de pequeños disyuntos que surgió para poder mejorar la detección de sugerencias, esto puede presentar un gran reto ya que según algunos autores, la utilización de un árbol de decisión (Ting, 1994) ayuda a mitigar los errores de clasificación cuando existen pequeños disyuntos, mientras otros autores mencionan que se deben utilizar algoritmos como KNN (Holte, Acker, & Porter , 1989), pero también causan problemas con grupos de instancias grandes, por lo que no son una solución para todos los problemas de pequeños disyuntos.

Referencias

Adobe. (12 de 11 de 2020). *Adobe Spark*. Obtenido de The 7 Top Social Media Sites You Need to Care About in 2020: https://www.adobe.com/express/learn/blog/top-social-media-sites

Ahmed, R. I. (2017). The influence of perceived social media marketing activities on brand loyalty: The mediation effect of brand and value consciousness. *Asia Pacific Journal of Marketing and Logistics, 29*(1), 129-144.

Akora, A., Malhotra, P., Marwah, S., Bhardwaj, A., & Dahiya, S. (2011). *Data Mining Techniques and Tools for Knowledge Discovery in Agricultural Datasets.* New Delhi: Aegis of Education Division, ICAR.

Arhibald London. (1 de 5 de 2020). *styleforum x ARCHIBALD*. Obtenido de Arhibald London: https://www.archibaldlondon.com/us/survey/styleforum-x-archibald

Brun, C., & Hagege, C. (2013). Suggestion Mining: Detecting Suggestions for Improvement in Users' Comments. *Research in Computing Science*(70), 199-209.

Chai, C., Fan, J., Li, G., Wang, J., & Zheng, Y. (Agosto de 2018). Crowd-Powered Data Mining. *Conference on Knowledge Discovery and Data Mining* (págs. 19-24). London, UK: Asociation for Computing Machinery.

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). *Bert: Pre-training of deep bidirectional transformers for language understanding.* arXiv preprint arXiv:1810.04805.

DeWalt. (09 de 03 de 2010). *Facebook*. (Facebook, Inc.) Recuperado el 19 de 03 de 2019, de https://www.facebook.com/DEWALT/posts/356194591449?__xts__[0]=68.ARA90EfkJWygxl-Rb3wE--eWxFSxCwcfkjiCC1iW6mkp9KQGy0lTeTJskoRvNRStHCdZdjSx6PwbtdqKFl_wF4T12QaDtt_JMzGLBAqEep5pmMhuCVQrPBRYwUVmRQ7WsrXL-mgxxYLCkXpYUvE7Vg-f4lFsusE8EqgT6JsXUtv08izL6-BG3I92oGPHNO

DeWalt. (2019). *Best of 2018*. (DeWalt) Recuperado el 12 de 03 de 2019, de https://www.dewalt.com/company-info/dewalt-awards

Digital Inspiration. (27 de 03 de 2020). *Save Tweets in Google Sheets*. Obtenido de Digital Inspiration: https://digitalinspiration.com/product/twitter-archiver

Elyasir , A. H., & Anbananthen, K. S. (2013). Evolution of Opinion Mining. *Australian Journal of Basic and Applied Sciences, 6*(7), 359-370.

Ganzabal-Garcia, X. (2015). - *Desarrollo y reutilización de componentes software y multimedia mediante lenguajes de guión.* Madrid: Ediciones Paraninfo.

García, V., Florencia-Juárez, R., Sánchez-Solís, J., Rivera-Zarate, G., & Contreras-Masse, R. (07 de 10 de 2019). Predicting Airline Customer Satisfaction using k-nn Ensemble Regression Models. *Research in Computing Science* , págs. 205-215.

Goldberg, A. B., Fillmore, N., Andrzejewski, D., Xu, Z., Gibson, B., & Zhu, X. (2009). May All Your Wishes Come True: A Study of Wishes and How to Recognize Them. *The Annual Conference of the North American Chapter of the ACL.* Boulder, Colorado, USA.

Gottipati, S., Shankararaman, V., & Rongsheng Lin, J. (2018). Text analytics approach to extract course improvement suggestions from students' feedback. *Research and Practice in Technology Enhanced Learning*, 6.

Holte, R. C., Acker, L. E., & Porter , B. W. (1989). Concept Learning and the Problem with Small Disjuncts. *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence* (págs. 813-818). San Mateo: Morgan Kaufmann Publishers.

IBM. (07 de 05 de 2020). *Modelos estadísticos*. Obtenido de SPSS Modeler: https://www.ibm.com/docs/es/spss-modeler/SaaS?topic=nodes-statistical-models

Jakkampudi, C. (1999). *Robust Bayesian Classifier.* Recuperado el 2 de 10 de 2016, de www.eecs.wsu.edu/~cook/dm/presentations/chandra.ppt

Jo, D., & Japkowicz, N. (01 de 06 de 2004). Class imbalances versus small disjuncts. *SIGKDD Explorations*, págs. 40-49.

Kemp, S. (2019). *Digital 2019: essential insights into how people around the world use the internet, mobile devices, social media, and e-commerce.* Vancouver, Canada: Hootsuite.

Khalaf, A. (2016). Selection of Best Decision Tree Algorithm for Prediction and Classification of Students' Action. *American International Journal of Research in Science, Technology, Engineering & Mathematics*, 26-32.

Liu, F., Wang, L., Zhu, X., & Wang, D. (2019). Suggestion Mining from Online Reviews usingRandom Multimodel Deep Learning. *18th IEEE International Conference On Machine Learning And Applications* (págs. 667 - 672). Boca Raton, FL, USA: IEEE.

Liu, J., Wang, S., & Sun, Y. (2019). OleNet at SemEval-2019 Task 9: BERT based Multi-Perspective Models for Suggestion Mining. *International Workshop on Semantic Evaluation* (págs. 1231–1236). Minneapolis, Minnesota, USA: Association for Computational Linguistics.

Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Feature selection.* Cambridge, United Kingdom: Cambridge University Press. Obtenido de https://nlp.stanford.edu/IR-book/html/htmledition/feature-selection-1.html

Microsoft. (27 de 06 de 2016). *Natural Language Processing*. (Microsoft Research) Recuperado el 30 de 10 de 2016, de https://www.microsoft.com/en-us/research/group/natural-language-processing/

Nagi, S., & Buitelaar, P. (2015). Towards the Extraction of Customer-to-Customer Suggestions from Reviews. *Conference on Empirical Methods in Natural Language Processing*, (págs. 2159-2167). Lisbon, Portugal.

Negi, S. (2016). Suggestion Mining from Opinionated Text. *Association for Computational Linguistics*, 119–125.

Negi, S., & Buitelaar, P. (2017). Inducing Distant Supervision in Suggestion Mining through Part-of-Speech Embeddings. *arXiv preprint arXiv:1709.07403*, 1-9.

Negi, S., Asooja, K., Mehrotra, S., & Buitelaar, P. (2016). A Study of Suggestions in Opinionated Texts and their Automatic Detection. *Conference on Lexical and Computational Semantics.* Berlin, Germany.

Negi, S., Daudert, T., & Buitelaar, P. (2019). SemEval-2019 Task 9: Suggestion Mining from Online Reviews and Forums. *International Workshop on Semantic*

Evaluation (SemEval-2019) (págs. 877–887). Minneapolis, Minnesota, USA: Association for Computational Linguistics.
Ovchinnikova, E. (2012). *Integration of World Knowledge for Natural Language Understanding.* Atlantis Press.
Pazos-Rangel, R. A., Florencia-Juarez, R., Paredes-Valverde, M. A., & Rivera, G. (2021). *Handbook of Research on Natural Language Processing and Smart Service Systems.* IGI Global. doi:https://doi.org/10.4018/978-1-7998-4730-4
Pedrycz, W., Martínez, L., Espin-Andrade, R. A., Rivera, G., & Gómez, J. M. (2021). *Computational Intelligence for Business Analytics.* Springer. doi:https://doi.org/10.1007/978-3-030-73819-8
Penisi, R., & Kim, G. (2003). Customer Driven Innovation Process. *IEEE/CPMT/SEMI International Electronics Manufacturing Technology (IEMT) Symposium.* San Jose, California, USA.
Pitchayaviwat, T. (2016). A Study on Clustering Customer Suggestion on Online Social Media about Insurance Services by Using Text Mining Techniques. *Management and Innovation Technology International Conference.* Bang Saen, Thailand.
Ramanand, J., Bhavsar, K., & Pedanekar, N. (2010). Wishful Thinking: Finding suggestions and 'buy' wishes from product reviews. *Workshop on Computational Approaches to Analysis and Generation of Emotion in Text.* Los Angeles, California, USA.
Real Academia Española. (noviembre de 2015). *CREA*. Obtenido de RAE: Banco de Datos: https://www.rae.es/recursos/banco-de-datos/crea
Real Academia Española. (2019). *Diccionario de la lengua española* . (Real Academia Española) Recuperado el 18 de 03 de 2019, de https://dle.rae.es/?id=YfTKl7q
Research hubs. (2015). *What is Artificial Intelligence*. (Research hubs) Recuperado el 02 de 11 de 2016, de http://researchhubs.com/post/ai/fundamentals/what-is-artificial-intelligence.html
Rico, O. (2018). *El aerotransporte comercial entre México y los EE. UU. en el contexto del nuevo acuerdo bilateral.* Sanfandila, Qro: Instituto Mexicano del Transporte.
Rivera, G., Florencia, R., García, V., Ruiz, A., & Sánchez-Solís, J. P. (18 de 10 de 2020). News classification for identifying traffic incident points in a Spanish-speaking country: A real-world case study of class imbalance learning. *Applied Sciences*, pág. 6253.
Saravanakumar, M., & SuganthaLakshmi, T. (2012). Social Media Marketing. *Life Science Journal, 9*(4), 4444 - 4451.
Sepna, N., & Buitelaar, P. (2015). Curse or Boon? Presence of Subjunctive Mood in Opinionated Text. *International Conference on Computational Semantics*, 101-106.
Stricker, G. (10 de Diciembre de 2017). *The 2014 #YearOnTwitter*. (Twitter) Recuperado el 02 de Julio de 2019, de https://blog.twitter.com/official/en_us/a/2014/the-2014-yearontwitter.html
Ting, K. (1994). The problem of small disjuncts: its remedy in decision trees. *Proceedings of the Tenth Canadian Conference on Artificial Intelligence* (págs. 91–97). Kingston: SpringerLink.

Van den Bosch, A., Weijter, T., Van den Her, H. J., & Daelemans, W. (1997). When small disjuncts abound, try lazy learning: A case Study. *BENELEARN-97: proceedings of the seventh Belgian-Dutch Conference on Machine Learning* (págs. 109-118). Tilburg University: Tilburg University.

van der Maaten, L. (05 de 01 de 2020). *github*. Obtenido de t-SNE: https://lvdmaaten.github.io/tsne/

Witten, I. H., Frank, E., Hall, M. A., & Pal, C. J. (2016). *Data Mining: Practical Machine Learning Tools and Techniques.* Morgan Kaufmann, Fourth Edition.

Apéndices

Apéndice A:

```
from nltk.tokenize import word_tokenize
str = “hello world!”
strList = word_tokenize(str)
print(strList)

Output: [‘hello’, ‘world!’]
```

Apéndice B:

```
str = “Hello WORLD”
str = str.lower()
print(str)

Output: hello world
```

Apéndice C:

```
def normalize(str):
    replacements = (
        ("á", "a"),
        ("é", "e"),
        ("í", "i"),
        ("ó", "o"),
        ("ú", "u"),
    )
    for a, b in replacements:
        str = str.replace(a, b)

return str
```

Apéndice D:

```
import unidecode
str = “Acción”
strUni = unicode(somestring, "utf-8")
str = unidecode.unidecode(strUni)

print(str)

Output: Acción
```

Apéndice E:

```
str = “Hello World ❤”
str = str.encode('ascii', 'ignore').decode('ascii')
print(str)
inputString.encode('ascii', 'ignore').decode('ascii')s
Output: Hello World
```

Apéndice F:

```
from nltk.corpus import stopwords

stop_words = set(stopwords.words('spanish'))
str = “Michael likes to play inside, not outside”
word_list = str.split() //splits the string into a list of strings
clean_list = [] //a list to store the list without stop words

for x in word_list:
        if x not in stop_words:
                clean_list.append(x)

str = ' '.join(clean_list)
print(str)
```

Output: Michael likes play inside outside

Apéndice G:

```
str = “abc123def”
str = ''.join([x for x in str if not x.isdigit()])
print(str)

Output = abcdef
```