Capítulo 2

Conceptos de la tesis

2.1. Minería de datos

La minería de datos es la tecnología de la cual se desprende la minería de textos, por lo que es importante definirla y comprenderla.

Hoy, en el mundo, se generan inmensas cantidades de información diariamente gracias, en gran parte, a la Web, a las computadoras personales omnipresentes y a los aparatos electrónicos que permiten guardar nuestros documentos, nuestras fotografías, nuestras decisiones, nuestros hábitos de consumo, entre otros tipos de información digital. La Web nos permite acceder a toda esta información, al mismo tiempo que todos los quehaceres personales (comercio, encuestas, juegos, sitios sociales, etc.) son almacenados. La brecha que existe entre la generación de información y la comprensión de esa información crece vertiginosamente y conforme aumenta el volumen de datos, la cantidad de personas que lo entienden disminuye de forma alarmante [20].

La minería de datos se define como el proceso de descubrir patrones en grandes cantidades de datos. Este descubrimiento debe ser de forma automática o semi-automática y los patrones encontrados deben ser de alguna utilidad, ya sea de utilidad económica, de utilidad científica, que demuestren la existencia de fenómenos no encontrados o no estudiados con anterioridad, que sirvan para realizar sugerencias de acuerdo a los datos analizados o incluso para identificar posibles

amenazas sociales[1], entre otros tipos de utilidad.

Un sistema de minería de datos tiene generalmente una entrada y una salida. La entrada es un conjunto de ejemplos del cual se pretende generalizar nuevos ejemplos. La salida es una descripción que clasifica ejemplos desconocidos. Por ejemplo, si se cuenta con ejemplos de transacciones bancarias fraudulentas, sería de gran interés clasificar las nuevas transacciones en dos categorías, transacciones legítimas y transacciones fraudulentas.

Los métodos de minería de datos procesan información numérica estructurada, obteniendo medidas de cada ejemplo del conjunto de entrada y entregando una predicción, basada en los ejemplos de entrada, acerca de algún ejemplo nuevo y desconocido.

2.2. Minería de textos

Los textos son considerados como información sin estructura, por lo que se podría pensar que los métodos de minería de datos no se aplican a ellos [21]. Sin embargo los textos pueden convertirse a valores medidos: ya sea la presencia de palabras, la frecuencia con la que aparecen o alguna otra métrica existente. Si se tienen estos valores entonces se pueden aplicar los mismos métodos de minería de datos a los textos, aunque estos deben ser implementados con consideraciones, por ejemplo, a la alta dimensionalidad, ya que los textos contienen miles de palabras que los definen y existen miles de documentos. La alta dimensionalidad representa un reto importante, ya que el desempeño de los algoritmos usados en la minería de datos depende generalmente del número de rasgos que definen a un objeto, por lo que es necesario realizar optimizaciones en estos algoritmos con el fin de reducir el uso de los recursos computacionales (tales como el tiempo de procesamiento y el espacio en memoria).

Los beneficios de la minería de textos han resultado en innovaciones tecnológicas que ayudan a la gente a entender mejor y a usar la información disponible en repositorios de documentos [22]. Estas tecnologías, como detección de contenidos,

[1]Como el sistema ADVISE desarrollado por el Departamento de Seguridad Nacional de los Estados Unidos: http://en.wikipedia.org/wiki/ADVISE

rastreo y obtención de tendencias, son usadas hoy en día en una gran cantidad de ámbitos, ya sea en bancos y en finanzas, en la industria, en comercios, entre otros.

Como su nombre lo indica, la minería de textos hace uso de documentos y el conjunto de textos estructurados es llamado corpus. Generalmente estos conjuntos contienen grandes cantidades de textos, por lo que los algoritmos que los procesen deben ser escalables, independientes del lenguaje y confiables [22].

Computacionalmente, los métodos para analizar los corpus se dividen en tres categorías principales: los basados en métodos estadísticos, los basados en métodos simbólicos y los híbridos.

- Métodos Estadísticos: son aquellos que no toman en cuenta la información semántica ni las propiedades lingüísticas de un texto. Cada documento de un corpus es representado por un vector que contiene la frecuencia, o alguna otra métrica estadística, de cada palabra que aparece en el documento. Este modelo es llamado bolsa de palabras (*bag-of-words*). Luego los vectores (que representan a cada documento del corpus) unidos, forman una matriz que representa al modelo de espacio vectorial. Esta representación, a pesar de no contar con información semántica, entrega resultados extremadamente buenos para una variedad de aplicaciones [22].

- Métodos Simbólicos: los métodos simbólicos (a veces mal llamados "lingüísticos"), generalmente basados en técnicas de procesamiento de lenguaje natural (PLN o NLP por sus siglas en inglés, *Natural Language Processing*), utilizan modelos de lenguaje para extraer y representar relaciones y significados expresados en el mismo. Pueden obtener representaciones profundas acerca de la estructura del texto. Sin embargo, estos modelos resultan difíciles de construir, mantener y depurar debido a la gran cantidad de reglas y presuposiciones que los componen.

- Métodos híbridos: combinan técnicas de los dos descritos anteriormente.

A lo largo de este trabajo de tesis se usan métodos estadísticos y simbólicos (acercamiento híbrido), que analizan el texto de forma numérica pero conservando algún tipo de información semántica, como se verá más adelante.

2.2.1. Etapas de un sistema de minería de textos

En general, los sistemas de minería de textos cuentan con cuatro etapas, las cuales son las mostradas en la figura 2.1.

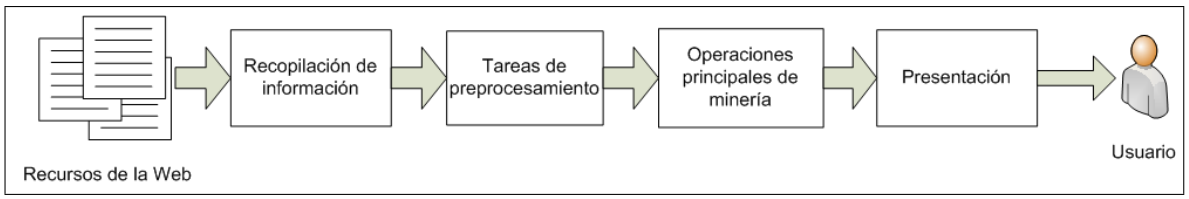

Figura 2.1: Etapas de un sistema de minería de textos. - La figura muestra las cuatro etapas más comunes en un sistema de minería de textos.

2.2.1.1. Etapa I: Recopilación de documentos

El primer paso para analizar texto es recolectar la información, en este caso los documentos relevantes. Es importante notar que en la minería de textos un documento puede ser un texto de miles de líneas o de solo una línea. En muchos casos esta información ya está disponible y solo basta con asegurarse de su calidad para poder analizarlos.

En otros casos, se necesita una colección de documentos; es decir, un corpus. Para extraer documentos, por ejemplo, de la Web, se ocupan herramientas (por ej., un Web crawler) cuyo propósito es recolectar documentos. Un Web crawler es una aplicación de software que recorre la Web automáticamente descargando las páginas de interés de acuerdo a su objetivo. En la minería de textos es usado para formar grandes colecciones de texto desde la Web, que no están disponibles de ninguna otra forma.

2.2.1.2. Etapa II: Tareas de preprocesamiento

Las técnicas usadas en esta etapa se desprenden del PLN. De acuerdo a [23], el procesamiento del lenguaje natural es el área de la ciencias de la computación que busca que las máquinas o computadoras lleven a cabo tareas útiles que impliquen el uso del lenguaje humano. Tareas tales como permitir la comunicación humano-máquina, mejorar la comunicación humano-humano, o realizar procesamiento de

texto o de voz que resulte útil de alguna forma. Para lograr estas tareas, el PLN requiere varios tipos de conocimiento acerca del lenguaje:

- Fonética y fonología: conocimiento acerca de sonidos lingüísticos.
- Morfología: conocimiento de los componentes significativos de las palabras.
- Sintaxis: conocimiento de las relaciones estructurales entre palabras.
- Semántica: conocimiento del significado.
- Pragmática: conocimiento de la relación del significado con los objetivos e intenciones del hablante.
- Discurso: conocimiento sobre unidades lingüísticas más largas que una simple declaración.

El PLN, al intentar extraer un representación más completa del significado del texto, ayuda a la minería de textos a descubrir conocimiento interesante y útil de texto no estructurado.

Las tareas de preprocesamiento descritas a continuación están basadas en técnicas del PLN.

2.2.1.2.1. Estandarización de los documentos

Una vez obtenidos los documentos que conformarán el corpus, se deben almacenar de manera uniforme que permita manipularlos, leerlos y escribirlos fácilmente.

Para este fin se ha adoptado, en la comunidad de procesamiento de texto, el lenguaje XML (*Extensible Markup Language*) [21]. Este formato estándar permite insertar etiquetas dentro del texto para identificar sus partes. Estas etiquetas forzosamente deben existir en pares de inicio y de finalización. Dentro de cada etiqueta pueden existir más etiquetas, permitiendo especificar aun más cada parte. Los nombres de las etiquetas son arbitrarios, sin embargo existen ya ciertos patrones que se siguen para estandarizar las colecciones de texto. Para trabajar como usuario con este formato, hoy en día existen muchos editores de texto y procesadores de palabras que permiten leer y guardar archivos en este formato.

Como desarrollador de aplicaciones, la mayoría de los lenguajes de programación modernos pueden, por medio de librerías y de interfaces de programación de aplicaciones (APIs), procesar información XML.

El objetivo de estandarizar el texto en un formato como XML es poder utilizar las herramientas de minería de textos con cualquier documento sin importar como fue generado ni su formato original [21].

2.2.1.2.2. Segmentación

Cuando ya se tienen los documentos estandarizados, el siguiente paso es encontrar rasgos que caractericen al texto almacenado. Por lo que es necesario definir fronteras y separar el texto en partes más simples.

Dos tipos de segmentaciones resultan relevantes para esta tesis: la segmentación de enunciados y la segmentación de palabras gráficas o *tokens*.

La segmentación de enunciados es un paso crucial en el procesamiento del texto [24]. Pretende dividir el texto en las oraciones individuales que lo componen. Antes de segmentar en palabras o tokenizar, es necesario segmentar el texto en enunciados. En inglés (la lengua usada en los experimentos de este trabajo), el punto, los signos de exclamación y de interrogación son delimitadores razonables. Sin embargo el punto es ambiguo, ya que existe también en las abreviaturas que lo utilizan y que pueden o no indicar la finalización de un enunciado.

Generalmente los algoritmos de segmentación de oraciones trabajan construyendo un clasificador binario (basado en secuencias de reglas o aprendizaje de máquinas) el cual decide si un signo de puntuación es parte de una palabra o es un delimitador de un enunciado [24].

La segmentación de palabras (tokenización) puede resultar complicada en lenguajes que no tengan una representación visual de las fronteras entre cada palabra [25]. En inglés y en otras lenguas, el espacio en blanco es considerado una frontera para delimitar palabras. También lo son los signos de puntuación, tales como los caracteres ()¡¿!?.^{en}tre otros [21]. Sin embargo, hay situaciones en las que los signos de puntuación están dentro de las palabras, como en Ph.D, AT&T, reddit.com y 444,444 (cuando se consideren los números como tokens). Para segmentar palabras se pueden definir distintas reglas que abarquen cada caso, dependiendo de

qué caracter precede qué caracter o si el caracter es letra mayúscula o no. Sin embargo, estas reglas se pueden volver complicadas de entender y de mantener, por lo que una opción práctica es usar expresiones regulares para definir las fronteras entre palabras y separar el texto por esas fronteras designadas.

Cuando se han identificado los tokens, se pueden encontrar los tipos. Un tipo es un conjunto de tokens; esto es, un token es una instancia de un tipo, por lo que suele haber un número de tokens por cada tipo. Si se tiene la oración "La gata es de la señora", en esta oración existen dos tokens "la", y estos dos tokens son una instancia del tipo "la". O sea un tipo que tiene dos tokens.

2.2.1.2.3. Lematización

El siguiente paso es convertir los tokens a una forma estándar. Esto puede ser útil dependiendo de la aplicación del sistema. Lematizar, en este trabajo de tesis, implica llevar un token a su raíz, removiendo las flexiones de las palabras. Con la raíz, se puede generar una conflación, que significa tratar como sinónimos varias palabras con la misma raíz. Existen diversos métodos para obtener las raíces de los tokens. El usado en este trabajo es el algoritmo Porter, descrito en [23].

Un ejemplo de lematización es el siguiente: para los tokens en inglés *exciting*, *excited* y *excitation*, la raíz encontrada por el algoritmo Porter es *excit*. Esta raíz *excit* aglomera las tres palabras anteriores en una sola palabra. Cuando aparezca en el texto alguna de esas tres palabras, se tratarán como sinónimos y se representarán por la raíz encontrada.

2.2.1.2.4. Generación del vector de rasgos

Para caracterizar un texto es necesario definir un conjunto de rasgos. Estos rasgos pueden ser simplemente las palabras que aparecen en cada texto o las palabras más frecuentes de un corpus. Una vez identificados estos rasgos se procede a formar una tabla o matriz basada en el concepto del modelo de espacio vectorial, es decir, representar cada texto en el corpus por medio de un vector donde cada dimensión de este es el valor del rasgo elegido. Posteriormente, el conjunto de vectores forma la matriz que contiene los rasgos en las columnas, los textos en los renglones y cada celda toma el valor que le corresponde a ese rasgo en ese texto. En la minería de textos, esta matriz tiene la característica de ser de alta

dimensionalidad, debido a que, como ya se explicó, un conjunto de textos puede significar una gran cantidad de palabras. Además, la mayoría de las entradas en la matriz serán cero, dado que los textos generalmente no comparten las mismas palabras. Por lo anterior, un tratamiento especial se aplica a este tipo de matrices llamadas dispersas. Se deben utilizar estructuras de datos y algoritmos diseñados explícitamente para almacenar este tipo de matrices y realizar operaciones con estas.

Para reducir el tamaño del vector se puede utilizar una lista de palabras de paro (*stoplist*), para filtrar las palabras funcionales que la mayoría del tiempo no tienen capacidades predictivas de interés en la minería [21], tales como "el", "la", "a", entre otras. Estas palabras se pueden remover del vector de rasgos para hacer más pequeño dicho vector. También se pueden utilizar técnicas de selección de rasgos que intentan elegir un subconjunto de palabras que puedan tener un mayor potencial para la predicción; aunque generalmente no se utilizan y se confía en la frecuencia de las palabras para colocarlas o no en el vector de rasgos. Lematizar las palabras también ocasiona que el vector se reduzca. Si se almacena solo la raíz de cada palabra podemos aglomerar en esa única raíz todas las variantes presentes en el texto.

2.2.1.2.5. N-gramas

En esta tesis, los n-gramas se definen como secuencias de n palabras consecutivas. Estas secuencias pueden ser de una sola palabra, cuando n es igual a uno, llamados unigramas, de dos palabras, bigramas, de tres palabras, trigramas y así sucesivamente. Estos n-gramas se pueden utilizar como rasgos. De hecho, cuando se usan palabras únicas se podría decir que se usan unigramas. También se pueden utilizar bigramas o trigramas para formar el vector de rasgos.

2.2.1.2.6. Etiquetado PoS (*Part of Speech tagging*)

Como se dijo antes, el objetivo del PLN es analizar y entender el lenguaje. Al estar aún lejos de alcanzar esa meta, el PLN se ha enfocado en tareas intermedias que buscan encontrar sentido de alguna parte de la estructura inherente del lenguaje sin requerir un entendimiento completo de él. El etiquetado PoS (*Part of Speech Tagging*) es una de estas tareas [25].

Etiquetar es asignar a cada palabra de una oración una categoría gramatical basada en el papel que cumple dentro de la oración. Las etiquetas proveen información acerca del contenido semántico de la palabra [26].

La mayoría de las gramáticas en inglés deberían tener como mínimo el sustantivo, el verbo, el adjetivo, el adverbio, la preposición y la conjunción.

El conjunto de etiquetas *Penn Treebank* [27], construido del corpus del periódico *Wall Street Journal*, contiene 36 categorías. Este conjunto de etiquetas es el utilizado en este trabajo.

De acuerdo a [23], la entrada para un algoritmo de etiquetado es una cadena de palabras y un conjunto de etiquetas. La salida es la mejor etiqueta encontrada para cada palabra, tal como se aprecia en la figura 2.2.

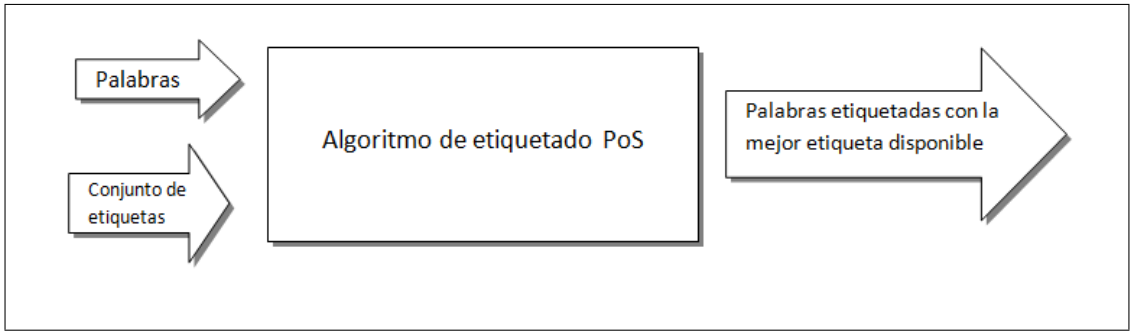

Figura 2.2: Diagrama de un etiquetador PoS - La figura muestra, a grandes rasgos, las entradas y la salida de un etiquetador PoS.

Un ejemplo en inglés de un etiquetado POS es el siguiente:
Oración original:
"Deliver me from Swedish furniture"
Oración etiquetada:
Deliver/VB (Verbo)
me/PRP (Pronombre personal)
from/IN (Preposición)
Swedish/JJ (Adjetivo)
furniture/NN (Sustantivo)

Después de aplicar las tareas de procesamiento del texto, se tiene un corpus estandarizado y etiquetado al cual se le aplicará la siguiente etapa.

2.2.1.3. Etapa III: Operaciones principales de minería

En esta etapa se aplican los algoritmos de minería de datos a los textos preparados para obtener la información deseada por el sistema. Aquí se encontrarían los métodos necesarios para clasificar texto, resumirlo, recuperar información, entre otras tareas.

2.2.1.4. Etapa IV: Presentación

Incluye la Interfaz Gráfica de Usuario (*GUI*) del sistema, herramientas de visualización para los resultados obtenidos, editores de consultas, entre otros instrumentos que faciliten al usuario final la interpretación y la manipulación de la información entregada por el sistema.

2.3. Aprendizaje de Máquinas (*Machine Learning*)

El aprendizaje de máquinas es un área de la inteligencia artificial cuyo objetivo es desarrollar algoritmos y técnicas que permitan a las computadoras resolver problemas mediante datos de ejemplo o experiencias obtenidas con anterioridad. Tiene una amplia gama de aplicaciones tales como procesamiento del lenguaje natural, motores de búsqueda, diagnósticos médicos, bioinformática, análisis de mercados y reconocimiento de patrones, entre otras.

El aprendizaje de máquinas se aplica cuando tenemos un problema y no se cuenta con un algoritmo que lo pueda solucionar. Por ejemplo, para separar correos electrónicos legítimos de correos basura, tenemos una entrada (un correo electrónico) y sabemos cuál debe ser la salida: correo basura o correo legítimo. En este contexto, la pregunta es cómo transformar la entrada en la salida [28]. La idea general del aprendizaje de máquinas es compensar la falta de conocimiento (el algoritmo) con la información disponible. Fácilmente se pueden juntar miles de correos electrónicos de los cuales conocemos su clase, legítimos o basura, y a partir de ellos hacer que la computadora aprenda que es lo que hace al correo

basura diferente del correo legítimo, esto con base en los ejemplos (experiencia) obtenidos.

El aprendizaje de máquinas utiliza la estadística para construir modelos matemáticos[2]. Estos modelos están definidos con ciertos parámetros, y el aprendizaje consiste en programar y ejecutar un programa de computadora que optimice los parámetros de dicho modelo, usando datos de entrenamiento (experiencia pasada). El modelo generalmente es predictivo, es decir, puede hacer predicciones en el futuro (clasificaciones)[28].

2.3.1. Clasificación[3]

Una de las aplicaciones más importantes del aprendizaje de máquinas es la clasificación, que es cuando tenemos dos o más clases a las que debemos asignar un caso no visto antes. La experiencia anterior nos ayuda a entrenar un sistema que encuentre automáticamente la salida (la etiqueta que designa la clase) dada una entrada (un nuevo caso). En cuanto a la clasificación se refiere, la experiencia anterior consiste en un dominio o conjunto de objetos, donde cada uno de ellos pertenece a una clase conocida.

Por ejemplo, en el reconocimiento de rostros, la entrada es la imagen de un rostro y las clases son las personas a ser reconocidas. Los datos de entrenamiento pueden ser miles de imágenes de rostros que son usados para entrenar el sistema. La salida es la asociación de cada rostro con una identidad [28].

Otro ejemplo es la clasificación de textos, donde los documentos pueden ser organizados por los temas que tratan.

Para realizar la clasificación de opiniones, en este trabajo se ha optado por usar el algoritmo de Bayes ingenuo, el cual se explica a continuación.

2.3.1.1. Método de Bayes ingenuo (*naïve Bayes*)

El método de Bayes ingenuo es importante dentro de los métodos usados para la clasificación por diversas razones, entre ellas está que es fácil de construir, fácil

[2]Un modelo matemático es la descripción de un sistema utilizando lenguaje matemático.

[3]También conocido como aprendizaje supervisado. Supervisado porque requiere de datos de entrenamiento etiquetados

de interpretar y a pesar de que asume la independencia condicional de las variables utilizadas (de ahí su nombre de ingenuo), se desempeña sorprendentemente bien. Podrá no ser el clasificador más robusto pero sus resultados suelen ser bastante confiables [29].

El proceso de aprendizaje bayesiano es muy eficiente. Analiza los datos de entrenamiento solo una vez para estimar todas las probabilidades requeridas para la clasificación [30].

A continuación se explicará el algoritmo orientado a la clasificación de textos.

Según [31], la clasificación con Bayes Ingenuo es vista como la estimación de la probabilidad *a posteriori* de una clase c dado un texto d:

$$P\left(c|d\right) \propto P\left(c\right) \prod_{1 \leq k \leq n_d} P\left(t_k|c\right)$$

Donde $P\left(t_k|c\right)$ es la probabilidad condicional del término t_k ocurriendo en un documento de la clase c[4]. $P\left(t_k|c\right)$ es una medida de qué tanto la evidencia t_k contribuye a que c sea la clase correcta. $P\left(c\right)$ es la probabilidad *a priori* de que un documento pertenezca a la clase c. Por otra parte, $(t_1, t_2, t_3, \ldots, t_{nd})$ son los tokens o palabras en d de donde se infiere el vocabulario usado (los tipos) para la clasificación y n_d es el número de tokens en d.

La clase elegida es la mejor clase posible, que es la que tiene una probabilidad mayor a las demás clases. Por lo que para predecir la clase c de un documento d es necesario calcular:

$$c = \arg\max_{c \in \mathbb{C}} P(c|d) = \arg\max_{c \in \mathbb{C}} P(c) \prod_{1 \leq k \leq n_d} P(t_k|c) \tag{2.1}$$

donde $\mathbb{C}$ es el conjunto de las clases posibles: $c_1, c_2, \ldots, c_{|\mathbb{C}|}$.

[4]En [31] se explica porque $P(c|d)$ es proporcional ($\propto$) y no igual al elemento de la derecha.

Se observa que cada parámetro condicional $P(t_k|c)$ es un peso que indica qué tan bueno es t_k como un indicador para la clase c. Así como $P(c_i)$ es un indicador de la frecuencia relativa con la que aparecen las clases. Las clases que cuentan con una mayor proporción en C son las que tienen mayor posibilidad de ser las correctas. Entonces la multiplicación de los indicadores es una medida de qué tanto pertenece un documento a una clase.

Una vez conocida la ecuación, es necesario estimar los parámetros para entrenar el clasificador.

$P(c_i)$ se estima simplemente dividiendo el número de documentos que pertenecen a esa clase, N_c, entre el número de documentos de entrenamiento disponibles, N.

$$P(c) = \frac{N_c}{N} \tag{2.2}$$

La probabilidad condicional $P(t_k|c)$ se calcula como la frecuencia relativa del término k en los documentos que pertenecen a la clase c:

$$P(t|c) = \frac{T_{ct}}{\sum_{t' \in V} T_{ct'}} \tag{2.3}$$

Donde T_{ct} es el número de ocurrencias del término t en los documentos de entrenamiento de la clase c. V es el conjunto de palabras distintas o vocabulario en los documentos de entrenamiento para la clase c. El denominador es simplemente el número de palabras que ocurren en los datos de entrenamiento para esa clase. Existen dos problemas que hay que resolver al aplicar Bayes ingenuo:

1. En la ecuación 2.1, muchas probabilidades condicionales son multiplicadas, una por cada posición $1 \leq k \leq n_d$. Estas probabilidades son valores menores a la unidad. Por lo que al multiplicarse sucesivamente, los valores tienden a

cero, posiblemente causando un *floating point underflow*[5]. Por lo tanto, se utiliza en lugar de la multiplicación, la suma de logaritmos. La clase con el logaritmo de la probabilidad más grande será todavía la clase más probable.

Dado que:

$$\log(xy) = \log(x) + \log(y)$$

Entonces la predicción de la clase se calcula, con suma de logaritmos, de la siguiente forma:

$$c = \arg\max_{c \in \mathbb{C}} [\log P(c) + \sum_{1 \leq k \leq n_d} P(t_k|c)]$$

2. Si se desea clasificar un documento que contiene alguna palabra que no apareció durante el entrenamiento, se obtendría una probabilidad de cero para esa palabra en esa clase, lo cual provocaría que la probabilidad se hiciera cero o se volviera indeterminada al sumar logaritmos naturales de cero. La solución consiste en suavizar las probabilidades. La forma estándar de hacerlo es aumentando la cuenta de cada palabra distinta con una pequeña cantidad $\lambda\,(0 \leq \lambda <= 1)$, de tal forma que cada palabra tendrá al menos una muy pequeña probabilidad de ocurrencia. Esto es llamado suavizado de Lidstone. Cuando $\lambda = 1$, el suavizado es conocido como suavizado de Laplace [30]. Por lo tanto, la ecuación para obtener la probabilidad condicional de un término dada una clase queda de la siguiente forma:

$$P(t|c) = \frac{T_{ct} + \lambda}{\sum_{t' \in V} T_{ct'} + \lambda |V|}$$

[5]El agotamiento de punto flotante o *floating point underflow* es una condición en un programa de computadora que ocurre cuando el verdadero resultado de una operación es menor en magnitud al mínimo valor representable como punto flotante normal en el tipo de dato usado. El resultado es la transformación automática del valor a cero, alterando completamente el resultado de la operación.

donde $|V|$ es el número de términos en el vocabulario.

Cuando se entrena el sistema se almacenan los valores de probabilidad condicional para cada palabra encontrada en los documentos de entrenamiento. Cuando se clasifica se busca la probabilidad de cada palabra del documento nuevo. Si no se encuentra la palabra, se le asigna el valor determinado por el suavizado.

Dado que solo se analiza una vez la información durante el entrenamiento, el algoritmo es lineal al número de ejemplos de entrenamiento, haciéndolo extremadamente eficiente, siendo esta una de sus grandes fortalezas [30].

2.3.2. Evaluación del clasificador

Después de construir el clasificador es necesario conocer la calidad de los resultados entregados por el sistema. Este paso consiste en evaluar el desempeño de la solución propuesta al problema planteado. ¿Es el sistema mejor que predecir aleatoriamente las clases? ¿Alcanza un desempeño que haga que su futura aplicación valga la pena? [32].

Para hacer esta evaluación es necesario realizar pruebas sobre la muestra de datos con la que se cuenta. Esta muestra contiene ejemplos con sus clases ya conocidas *a priori*. Como se dijo antes, el objetivo del clasificador es generalizar, a partir de esta información, nuevos ejemplos que aparecerán en el futuro.

Ahora, cuando se entrena y se prueba con el mismo conjunto de casos, seguramente se obtendrían resultados muy buenos. Sin embargo, estos resultados no se generalizarían para casos nuevos [32]. A esto se le conoce como *overfitting* y se busca evitar por medio de métodos que separan la muestra total en dos conjuntos separados: conjunto de entrenamiento y conjunto de pruebas. De hecho, la técnica más sencilla usada para la evaluación consiste en separar la muestra en dos, una parte para entrenar y otra parte para probar.

2.3.2.1. Remuestreo (*resampling*)

En lugar de simplemente separar los casos en dos muestras (entrenamiento y pruebas), se podrían elegir aleatoriamente varios de los casos para entrenar y el resto para realizar las pruebas; después, repetir este proceso varias veces. Esto es

el remuestreo y pretende reducir los errores en las estimaciones cuando se pruebe el clasificador con nuevos casos [32].

2.3.2.1.1. Validación cruzada con k pliegues (k - *fold cross validation*)

Este método utiliza toda la información disponible tanto para entrenar como para probar.

Se divide aleatoriamente el conjunto total de muestras en k partes del mismo tamaño. Se generan varios pares de dos elementos: un conjunto de entrenamiento y uno de pruebas. Para generar cada par se deja una de las k partes fuera como conjunto de pruebas y se combinan las $k-1$ partes restantes para el conjunto de entrenamiento. Haciendo esto k veces, cada vez dejando fuera otra de las k partes, se obtienen k pares de conjunto de entrenamiento y conjunto de pruebas[6].

Por ejemplo, si se desea realizar una validación con $k = 3$, con una muestra de 999 casos de la clase A y 999 casos de la clase B, se divide el conjunto total en tres partes: parte 1, parte 2 y parte 3, cada una con 333 casos de la clase A y 333 casos de la clase B. Luego se entrenan tres clasificadores:

- El primero se entrena con las partes 1 y 2 y se prueba con la parte 3.
- El segundo se entrena con las partes 1 y 3 y se prueba con la parte 2.
- El tercero se entrena con las partes 2 y 3 y se prueba con la parte 1.

En cada ocasión se usan 666 casos para entrenar y 333 casos para probar.

Típicamente k es igual a 10 ó 30 [28]. Conforme k crece, se obtienen estimaciones más robustas, sin embargo el tamaño del conjunto de prueba disminuye para una misma muestra, si bien aumenta el conjunto de entrenamiento. Esto es, el costo en tiempo y recursos para el entrenamiento depende también de k: Se debe entrenar un clasificador k veces, y el tamaño del conjunto de entrenamiento crece también con k.

[6]Un caso especial de la validación cruzada es la validación cruzada deja uno afuera (*take one out*). En este método k es igual al número de ejemplos disponibles (no se tiene la opción de cambiar k, sino que se asume que el número de casos es la unidad primordial). Es usado normalmente cuando el número de casos es pequeño, pero obviamente no es eficiente para un conjunto de gran tamaño, ya que se deben entrenar k clasificadores [30].

2.3.3. Medición del desempeño del clasificador

Para un clasificador binario, se genera una matriz de dimensión 2 x 2, la cual contiene todos los posibles resultados de la clasificación.

	Clasificado como positivo	Clasificado como negativo	Total
Realmente positivo	tp	fn	p
Realmente negativo	fp	tn	n
Total	p'	n'	N

Tabla 2.1: Matriz con los únicos posibles resultados de una clasificación binaria.

Esta matriz es llamada matriz de confusión (ver tabla 2.1) y contiene cuatro valores que representan el desempeño del clasificador que se explican en la siguiente sección.

2.3.3.1. Medidas de desempeño

Tomando como referencia los valores de la tabla 2.1, para un caso realmente positivo, si la predicción es también positiva, se le denomina positivo verdadero, **tp** (*true positive*). Si la predicción es negativa, para el mismo caso positivo, se le llama falso negativo, **fn** (*false negative*). Si el caso es realmente negativo y se predice como positivo, es un falso positivo, **fp** (*false positive*). Si con el mismo caso negativo, se predice negativo, se obtiene un negativo verdadero, **tn** (*true negative*).

Además, **p** es el número de casos positivos reales, **n** es el número de casos negativos reales, **p'** es el número de casos clasificados como positivos y **n'** es el número de casos clasificados como negativos.

En la diagonal principal de la matriz se encuentran las clasificaciones correctas (tp y tn), mientras que los otros dos valores (fn y fp) representan los dos tipos de errores.

Por ejemplo, si se tiene una aplicación cuyo fin es autenticar usuarios de un estacionamiento por su huella digital, el **error falso negativo** representaría la ocasión en que un usuario válido quisiera entrar al estacionamiento y fuera

Nombre	Fórmula
error	$(fp + fn)/N$
exactitud	$(tp + tn)/N = 1 - error$
razón-tp	tp/p
razón-fp	fp/n
precisión	tp/p'
exhaustividad	$tp/p =$ razón-tp
sensitividad	$tp/p =$ razón-tp
especificidad	$tn/n = 1-$ razón-fp

Tabla 2.2: Medidas de desempeño para clasificadores

rechazado por la aplicación. El **error falso positivo** sería cuando a un usuario no autorizado se le da acceso erróneamente. Se puede ver que los dos tipos de errores no son igual de graves, el falso positivo es más grave en una aplicación de este tipo. Aunque, dependiendo de la aplicación, estos dos errores pueden variar su grado de importancia.

Las medidas más comunes para medir el desempeño de los clasificadores binarios son las mostradas en la tabla 2.2. Nótese que la razón-tp sirve para estimar tanto la exhaustividad como la sensitividad.

La **exactitud** (*accuracy*) es utilizada frecuentemente para determinar el desempeño de un clasificador. Sin embargo, no es muy útil cuando se está interesado sólo en la clase minoritaria (la clase menos representada en la muestra), ya que, retomando el ejemplo del estacionamiento, asumiendo que el 90 % de los usuarios son auténticos, un clasificador sólo tiene que predecir cada caso como positivo (aceptado) para obtener una exactitud del 90 %, aunque haya existido un solo caso, el más relevante, que fue clasificado como válido cuando en realidad era un usuario no autorizado, es decir, un caso negativo clasificado como positivo. El **error** es el inverso de la exactitud. La mayoría del tiempo nuestro interés no se enfoca en medidas generales como estas dos, sino en los tipos de errores antes mencionados.

En cuanto a la precisión (*precision*) y la exhaustividad (*recall*), estas son también medidas adecuadas para determinar qué tan completa y precisa fue la predicción en la clase positiva. La **precisión** es el número de ejemplos correctamente clasificados como positivos dividido entre el número total de ejemplos

clasificados como positivos. La **exhaustividad** es el número de ejemplos correctamente clasificados como positivos dividido entre el número total de ejemplos positivos reales en el conjunto de pruebas [30].

En aplicaciones de recuperación de información la medida F (F-measure), es comúnmente utilizada como medida única de desempeño. La **medida F** es la media armónica de la precisión y la exhaustividad, y está definida por:

$$Medida\ F = \frac{2}{\frac{1}{precisión} + \frac{1}{exhaustividad}}$$

La exactitud es por mucho la métrica de desempeño más utilizada. Sin embargo, se ha demostrado que las razones para usar la exactitud como medida única son muy cuestionables, siendo el análisis ROC (acrónimo de *Receiver Operating Characteristic*, esto es, característica operativa del receptor), que se describe abajo, una alternativa no tan simple de obtener como la exactitud pero que permite realizar conclusiones firmes y generales [33].

2.3.3.1.1. Espacio ROC

Una gráfica ROC es una técnica para visualizar, organizar y seleccionar clasificadores basados en su desempeño. El espacio ROC se originó en la teoría de detección de señales y tiene por $eje - x$ la razón-fp y por $eje - y$ la razón-tp. El análisis ROC se ha extendido recientemente al aprendizaje de máquinas y ahora es usado con mayor frecuencia en la evaluación y comparación de algoritmos.

Para trazar un punto o una curva en el espacio ROC, se necesita definir el tipo de clasificador que se tiene. Existen generalmente dos tipos de clasificadores por la forma en la que designan la clase predicha:

1. Clasificadores discretos: Determinan la clase de un caso de forma binaria, *sí o no* se pertenece a la clase. Esos clasificadores entregan un solo punto, el cual es trazado en el espacio ROC.

2. Clasificadores probabilísticos: Entregan un valor numérico que representa el grado de pertenencia de un caso a una clase. Si se determina un umbral, y la salida del clasificador es mayor a ese umbral, entonces ese caso se considera

positivo, de lo contrario, el caso es clasificado como negativo. Si se varía ese umbral, se obtendrá un punto para cada valor diferente. Uniendo estos puntos se obtendría una curva ROC.

Para ubicar un punto en el espacio ROC basta con conocer los valores de razón-tp y de razón-fp del clasificador a representar. Estos dos valores se usan como coordenadas en el plano XY y con ellas se traza el punto.

Para trazar la curva se requiere de un procedimiento más complejo. Como se dijo antes, es necesario tener un clasificador probabilístico y contar con un umbral, de tal forma que si este se supera, el clasificador asigna la clase positiva, de lo contrario asinga la clase negativa. Al variar este umbral se obtienen puntos que se grafican en el espacio ROC. Toda curva ROC generada de un conjunto finito de casos producirá una función escalón que se aproxima a una verdadera curva al mismo tiempo que el número de casos tiende a infinito.

En el espacio ROC, entre más arriba y a la izquierda se encuentre un punto o la cresta de la curva, mejor es el desempeño del clasificador. Idealmente un clasificador debería estar en la coordenada $(0, 1)$ donde se tiene la máxima razón-tp y la mínima razón-fp. La línea diagonal $y = x$ corresponde a usar un clasificador aleatorio, por lo que es poco común observar un punto o curva por debajo de esta línea, ya que si se negara el resultado obtenido por ese clasificador, se obtendría automáticamente un mejor desempeño. Un clasificador que se encuentra en la diagonal no ofrece información acerca de la pertenencia a las clases [34]. En la figura 2.3 se ilustra el espacio ROC.

Las curvas y puntos en el espacio ROC representan herramientas útiles para visualizar y evaluar clasificadores. Como se dijo, son capaces de ofrecer información más detallada acerca del desempeño de un clasificador en comparación con medidas como la exactitud o el error (ya que estas medidas dependen directamente de la distribución de las clases). La curva ROC sirve también para conocer el *trade-off*[7] de un clasificador y poderlo ajustar dependiendo de lo que se deseé clasificar con él.

[7]El trade-off se refiere a perder un tipo de calidad, pero ganando otro tipo de calidad. Es decir, se elige entre clasificar mejor los casos realmente positivos a costa de clasificar con mayor frecuencia casos realmente negativos como positivos.

En este trabajo, el espacio ROC (aunado a las medidas convencionales de error y exactitud) es utilizado para determinar el método que ofrece el mejor desempeño.

2.3.4. Agrupamiento[8] (*Clustering*)

Cuando no se conocen las clases en las que queremos separar los datos, no se cuenta con datos de entrenamiento y/o solo se tiene la información de entrada, se aplican las técnicas de agrupamiento para organizar los datos automáticamente.

Todos los problemas de agrupamiento son problemas de optimización. El objetivo es elegir la mejor agrupación de objetos posible de acuerdo a cierta función de calidad.

Un buen resultado de un agrupamiento debería reunir objetos similares y separar los distintos. Por lo tanto la función de calidad se expresa en términos de una función de similitud entre objetos. Una función de similitud toma dos objetos y produce un valor real, el cual indica la proximidad que existe entre esos objetos. Este valor se obtiene con base en los rasgos que representan a cada objeto.

Como se dijo antes, el concepto de representar un objeto como vectores de rasgos multidimensionales es llamado modelo de espacio vectorial. En este modelo, la función de similitud está usualmente basada en la distancia entre vectores de acuerdo a alguna métrica [26].

La función de similitud más popular es la distancia Euclidiana:

$$D(x_i, x_j) = \sqrt{\sum_k (x_{ik} - x_{jk})^2}$$

donde x_i y x_j son dos objetos y k es la dimensión de su vector de rasgos.

La distancia Euclidiana es un caso particular de la métrica de Minkowski, cuando $p = 2$:

$$D_p(x_i, x_j) = \left(\sum_k (x_{ik} - x_{jk})^p \right)^{\frac{1}{p}}$$

[8]También conocido como aprendizaje no supervisado. No requiere o no se cuentan con datos de entrenamiento etiquetados.

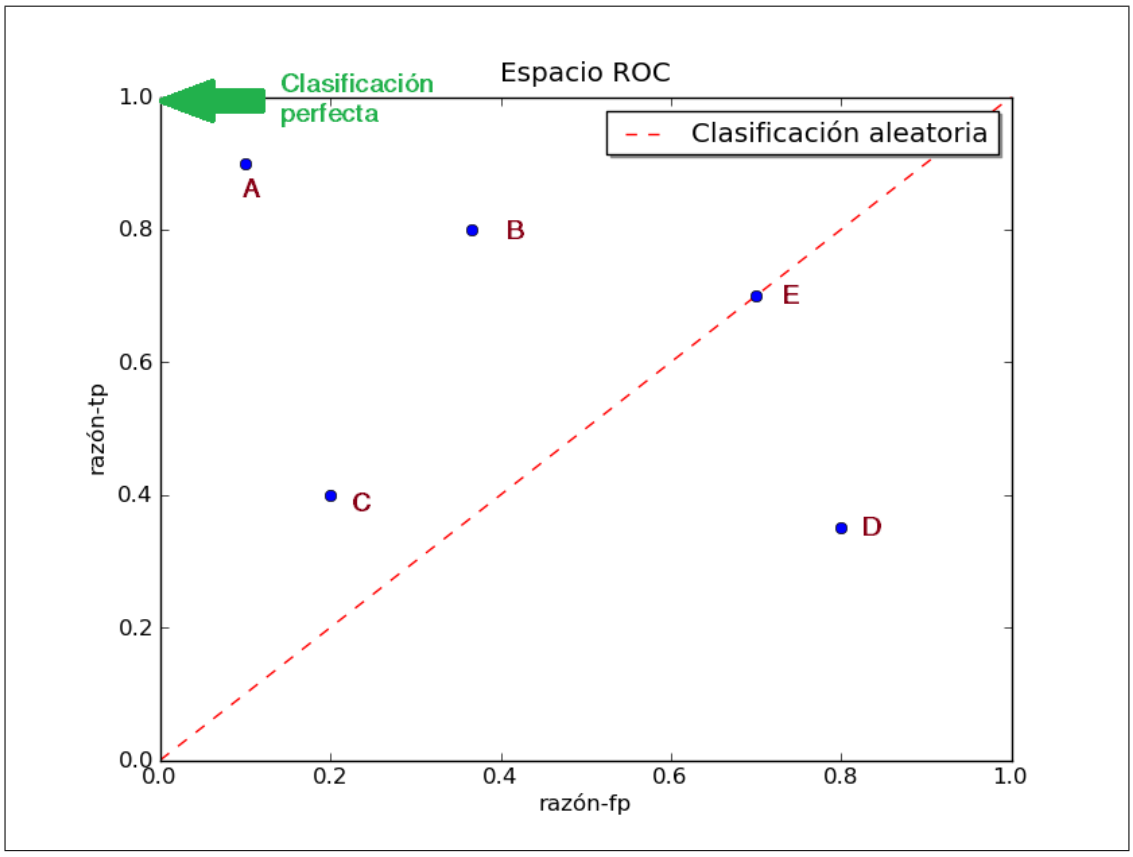

Figura 2.3: Espacio ROC - La figura muestra varios puntos en el espacio ROC. Entre más arriba y a la izquierda se encuentre el punto, el desempeño del clasificador es mejor. La línea diagonal $y = x$ representa una predicción aleatoria. El punto (0.0,1.0) representa el resultado de una clasificación perfecta: sin errores falsos positivos y todos verdaderos positivos. En este ejemplo, el punto A representa al mejor clasificador, el punto E representa a un clasificador con desempeño igual al de un clasificador aleatorio. El punto C representa a un clasificador con un desempeño muy pobre, muy cercano al desempeño de un clasificador aleatorio. Finalmente, el punto B representa el mismo clasificador que el que representa el punto D pero negado.

Existe un gran número de medidas de similitud disponibles, cada una sirve a un propósito particular [26].

2.3.4.1. Tipos de algoritmos de agrupamiento

Los algoritmos de agrupamiento se pueden clasificar en algunos tipos básicos. Por la estructura que producen, existen dos tipos: algoritmos jerárquicos y algoritmos particionales o planos.

Un **algoritmo jerárquico**, como su nombre lo indica, entrega como resultado una estructura de datos de tipo árbol, jerárquica, donde cada nodo representa una subclase de su nodo padre. Las hojas del árbol son los objetos individuales del conjunto agrupado. Cada nodo representa el grupo que contiene a todos los objetos de sus descendientes. Un ejemplo de los resultados de este tipo de algoritmo se observa en la figura 2.4

Un **algoritmo particional** entrega un cierto número de grupos. La relación entre los grupos es pocas veces determinada. La mayoría de estos algoritmos son iterativos: inician con un conjunto inicial de grupos y los reubican en cada iteración, mejorando su distribución. En la figura 2.5 se muestra un ejemplo de este tipo de agrupamiento.

Otra diferencia importante de los algoritmos de clasificación depende de la membresía de cada objeto. La membresía de un objeto indica a qué grupo pertenece ese objeto. Si cada objeto es asignado a uno y solo un grupo, entonces es **agrupamiento duro**. Por otro lado, el **agrupamiento suave** permite varios grados de membresía y permite tener membresía en múltiples grupos.

Existen ciertas técnicas de agrupamiento estándar, tales como árboles de decisión, bosques de árboles de decisión, máquinas de vectores de soporte, el algoritmo de k - medias, el algoritmo de esperanza - maximización, entre otras [36]. Estas técnicas se desempeñan bien con conjuntos de datos comunes, sin embargo, para conjuntos de datos complejos, son usadas técnicas de análisis más complejas.

La **descomposición de matrices**, descrita a continuación, puede ofrecer un análisis más completo, o puede también producir datos más limpios para después ser usados por los métodos estándar.

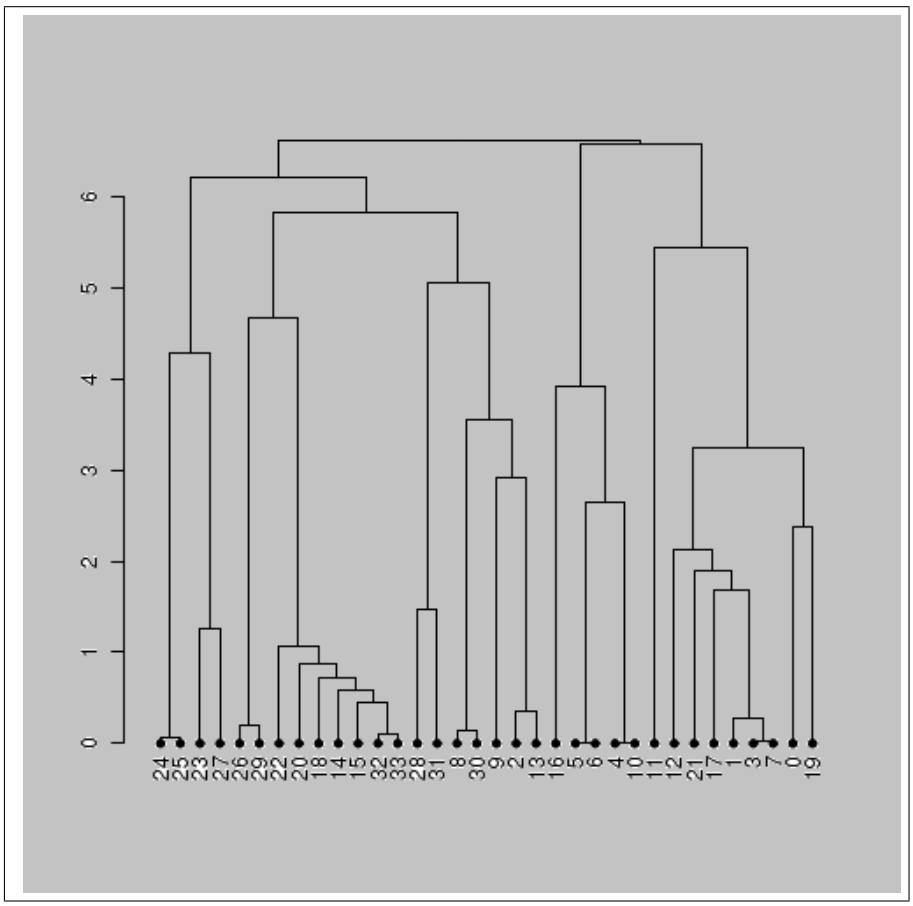

Figura 2.4: Ejemplo de un agrupamiento jerárquico - La figura muestra los resultados de la aplicación de un agrupamiento jerárquico. Este tipo de diagramas es llamado dendrograma y representa el ordenamiento de los grupos producidos por este método de agrupamiento. Esta imagen fue tomada de [35].

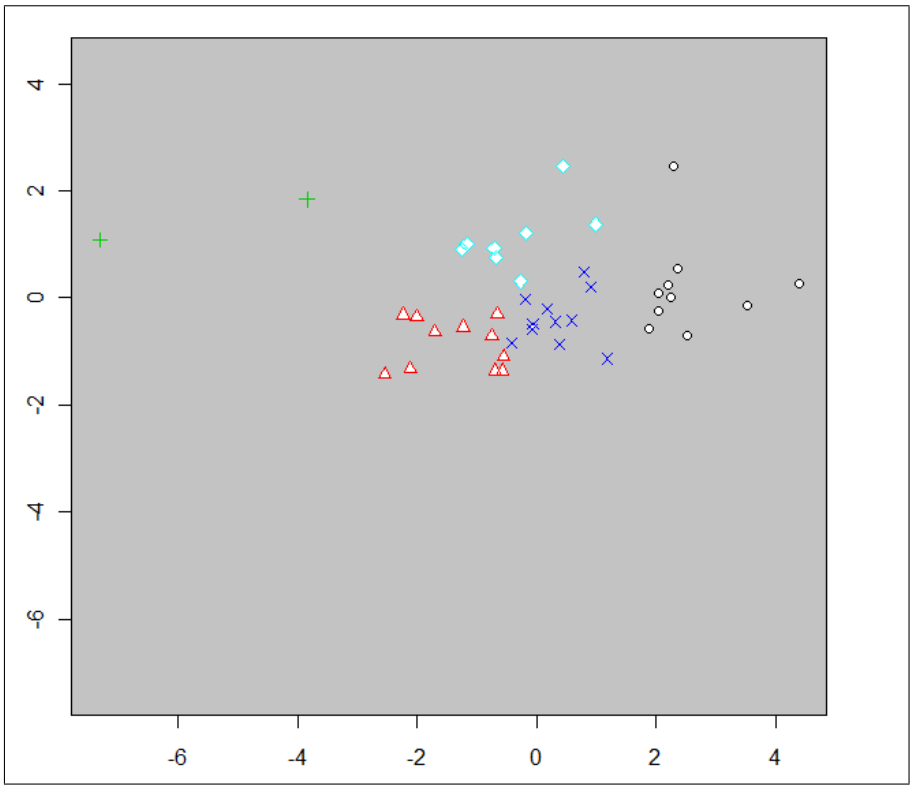

Figura 2.5: Ejemplo de un agrupamiento particional - La figura muestra los resultados de la aplicación de un agrupamiento particional. Cada objeto es miembro de un grupo encontrado.

2.3.4.2. Descomposición de matrices

De acuerdo a [36], la descomposición de matrices es usada principalmente para dos tareas en el análisis de datos:

- Es capaz de separar los datos basura, producto de procesos no controlados o de errores, de los datos disponibles. Esto es útil al aplicar técnicas convencionales ya que de esta forma producirán resultados mejores. Esta tarea podría llamarse limpieza de datos.
- Agrupa objetos de un conjunto de datos, ya sea usando alguna técnica estándar o interpretando el resultado de la descomposición.

Los algoritmos más comunes de descomposición de matrices son:

- Descomposición en Valores Singulares (*Singular Value Decomposition SVD*) y Análisis de Componentes Principales (*Principal Component Analysis PCA*);
- Descomposición semidiscreta (*SemiDiscrete Decomposition SDD*);
- Análisis de Componentes Independientes (*Independent Component Analysis ICA*);
- Factorización no negativa de matrices (*Non-Negative Matrix Factorizarion NMF*).

Descripción

Si consideramos un conjunto de datos como una matriz, con n renglones, cada uno representando a un objeto y m columnas, cada una representando un rasgo, entonces la celda ij representa el valor del rasgo j para el objeto i. En los algoritmos de descomposición de matrices, se busca expresar una matriz de datos, A, como el producto de un conjunto de nuevas matrices que sacan a la luz las estructuras y relaciones implícitas en A.

Formalmente, una descomposición de matrices puede ser descrita por una ecuación de la forma:

$$A = WCH \tag{2.4}$$

donde las dimensiones de las matrices son:

- La dimensión de A es $n \times m$ con $n \gg m$ (esto es, n es mucho mayor que m)
- W es $m \times r$ para una r que es usualmente menor a m
- C es $r \times r$
- H es $r \times m$

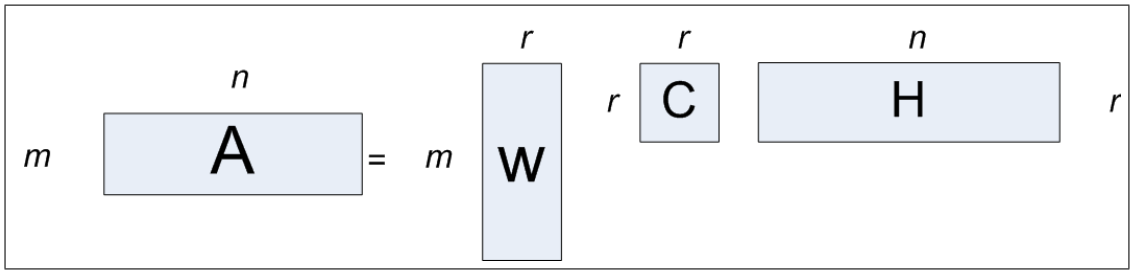

Figura 2.6: Descomposición de la matriz de datos A en las matrices W, C y H - La figura muestra las matrices y sus dimensiones después de una descomposición de matrices.

Específicamente, de la ecuación 2.4 se observa que un elemento de A, por ejemplo el elemento a_{11} es generado por la multiplicación del primer renglón de W, el primer elemento de C, y la primera columna de H. Por lo que cada valor en A es una combinación de partes de F, combinados de ciertas formas descritas por W y C.

La matriz W tiene el mismo número de renglones que A. El $i-ésimo$ renglón de W proporciona r piezas de información que juntas dan una nueva interpretación al $i-ésimo$ objeto; mientras que A provee m piezas de información acerca del $i-ésimo$ objeto.

La matriz H tiene siempre el mismo número de columnas que A. Cada columna de H da una nueva interpretación del atributo descrito por la columna correspondiente de A, en términos de r piezas de información.

El papel de r es el de forzar una representación más compacta con respecto de la forma original. Se asume que una matriz más pequeña (que represente a A), capturará las regularidades latentes que puedan existir dentro de A. Generalmente el valor de r es más pequeño que el de m [36].

La matriz C contiene valores que reflejan las relaciones entre los factores latentes: el $ij-ésimo$ valor (c_{ij}) ofrece la conexión que existe entre el factor latente capturado por la $i-ésima$ columna de W y el factor latente capturado por el $j-ésimo$ renglón de H. Algunas descomposiciones no generan esta matriz intermedia, tal es el caso de la factorización no negativa.

La mayoría de los métodos para descomponer matrices pueden ser expresados como problemas de optimización limitada.

En esta tesis se ocupó el algoritmo de factorización no negativa de matrices, el cual se explica a continuación.

2.3.4.2.1. Factorización no negativa de matrices

La factorización no negativa de matrices (FNM) representa una alternativa al análisis de componentes principales. Este método es particional duro y está diseñado para conjuntos de datos cuyos valores de sus atributos nunca son negativos. Asimismo, los elementos de las matrices resultantes no contienen valores negativos tampoco. Por ejemplo, los documentos con texto no pueden contener frecuencias negativas de palabras así como las imágenes no pueden tener cantidades negativas de colores.

La factorización no negativa de matrices produce solo dos matrices, como se observa en la figura 2.7, la matriz W y la matriz H. Por lo que la definición de la factorización no negativa es:

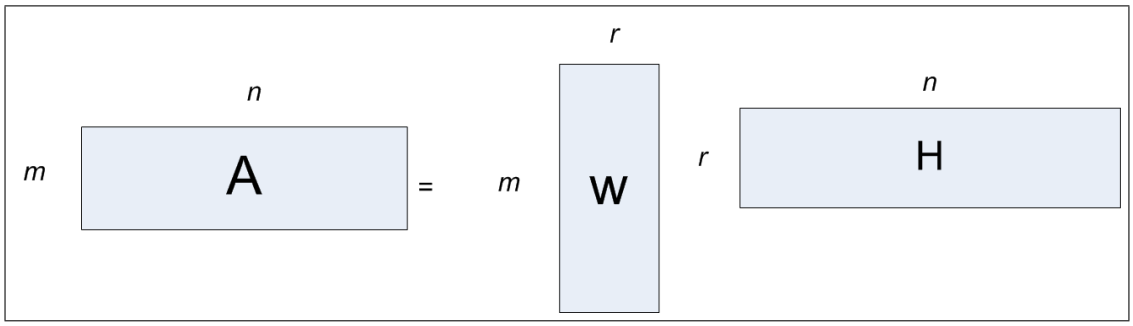

Figura 2.7: Matrices obtenidas después de la factorizacion con FNM. - La figura muestra los resultados obtenidos al aplicar la factorización no negativa de matrices.

$$A \approx WH$$

donde:

- A es $m \times n$
- W es $m \times r$
- H es $r \times n$
- $r \leq m$

Ambas matrices, W y H, deben contener solo valores no negativos. W es la matriz de factores y H es la matriz de mezcla.

El enfoque convencional para encontrar W y H es minimizar la distancia entre A y el producto WH:

$$\min_{W,H} f(W,H) \equiv \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{m} (A_{ij} - (WH)_{ij})^2$$

$$\text{sujeto a } W_{ia} \geq 0, H_{bj} \geq 0, \forall i, a, b, j.$$

Esta descomposición no es única, depende del método para inicializar W y H, del algoritmo con el que se generen finalmente las mismas matrices y de la métrica de error usada para comprobar la convergencia. Algunos de los algoritmos de optimización usados para FNM son:

- Actualización multiplicativa, descrita por [37].
- Codificación dispersa, descrita por [38].
- Descenso de gradientes con mínimos cuadrados limitados, descrito por [39].
- Mínimos cuadrados alternantes, descrito en [40].
- Mínimos cuadrados alternantes usando gradientes proyectados, descrito por [41].

Los algoritmos usados durante el desarrollo de este trabajo, se explican a continuación.

Actualización multiplicativa (*Multiplicative Method MM*)

Usando la norma de Frobenius[9], la función objetivo (o problema de minimización) se define como:

$$\min_{W,H} ||V - WH||_F^2$$

con W y H siempre no negativas.

El algoritmo, usando la norma de Frobenius, se describe como:

1. Inicializar W y H con valores aleatorios no negativos.
2. Iterar por cada c, j, i (índices de cada una de las matrices) hasta que el error de convergencia sea mínimo o después de n iteraciones:

   a) $H_{cj} \leftarrow H_{cj} \frac{(W^T V)_{cj}}{(W^T W H)_{cj} + \epsilon}$

   b) $W_{ic} \leftarrow W_{ic} \frac{(V H^T)_{ic}}{(W H H^T)_{ic} + \epsilon}$

   c) $W \leftarrow |W|, H \leftarrow |H|$

Epsilon (ϵ) se agrega al denominador para evitar divisiones entre cero, su valor es muy pequeño (10^{-9}).

Mínimos cuadrados alternantes usando gradientes proyectados (*Alternating Least Squares using Projected Gradients*)

Además del método de actualización multiplicativa, en esta tesis (como se explicará más adelante, en el capítulo tres de metodología) también se utilizó el algoritmo de mínimos cuadrados alternantes usando gradientes proyectados, el cual difiere de MM en la forma de darle solución al problema de optimización (esto es, lo realiza con mínimos cuadrados alternantes). Se ha demostrado que este método converge más rápido (requiriendo menos iteraciones) [41].

[9]También llamada norma de Hilbert - Schmidt, se define como: $\|A\|_F = \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} |a_{ij}|^2}$

El problema de mínimos cuadrados alternantes se expresa con las siguientes reglas, en sustitución de las reglas del paso 2 del algoritmo descrito arriba:

$$W^{k+1} = \arg\min_{W \geq 0} f(W, H^k),$$

$$H^{k+1} = \arg\min_{H \geq 0} f(W^{k+1}, H)$$

Por la necesidad de brevedad, en esta tesis no se ahondará en la explicación de la aplicación de gradientes proyectados para resolver estas reglas. Véase [41] para una explicación más detallada. De hecho, la implementación en Python de este método utilizada en esta tesis se debe a A. Di Franco, cuyo código puede consultarse en [42].

Obtenidas las matrices W y H, estas se pueden utilizar para realizar el agrupamiento: el vector a_j es asignado al grupo i si h_{ij} es el elemento más grande en la columna j de H.

2.4. Minería de Opiniones

2.4.1. Introducción

Conocer las opiniones de otras personas sobre algún tema en específico siempre ha sido importante para formar un juicio propio. El área de minería de opiniones se encarga del tratamiento computacional de la opinión, sentimiento y subjetividad en el texto. A continuación se esbozan algunas características de este campo de investigación (ver [22]).

La información contenida en textos se puede dividir, a grandes rasgos, en dos categorías: descripciones y opiniones. Las descripciones son expresiones objetivas acerca de cosas, eventos, etc., y de sus propiedades. Las opiniones son usualmente expresiones subjetivas que describen los sentimientos, aprecio o juicio acerca de un objeto, un evento, etc., y de sus propiedades [4].

Dado que el concepto de opinión, y por ende de sentimientos, es muy amplio, en este trabajo se trabajará solamente con opiniones negativas y positivas. Entendiendo como opinión negativa a aquella que habla mal acerca de algo y como opinión positiva a aquella que habla bien acerca de algo.

En las últimas dos décadas, gracias al desarrollo de la Web, se han generado cientos de sitios que permiten a millones de usuarios expresar sus opiniones acerca de casi cualquier tema. Estas opiniones cada vez se vuelven más importantes a la hora de tomar decisiones, ya sea al realizar la compra de un producto o servicio, o en la elección de candidatos políticos. Al existir una cantidad tan grande de opiniones, es necesario construir sistemas de acceso a la información con la finalidad de ayudar a los consumidores a tomar decisiones, ya que la información puede presentarse de forma tramposa, confusa, excesiva o difícil de localizar.

Si se conocen las tendencias de opinión acerca de un producto, es posible, para la empresa que lo produce, ajustar sus campañas publicitarias, el posicionamiento de la marca, y otras posibles estrategias de promoción.

Para realizar un sistema de minería de opiniones se deben resolver los siguientes problemas:

1. Determinar si el usuario desea o no una reseña u opinión acerca de algo. Esto se puede resolver al usar palabras como “opinión” o “reseña” al momento

de realizar una petición y realizando una clasificación dentro de la consulta hecha.

2. Encontrar, dentro de un documento que hable sobre el objeto deseado, las oraciones que contengan material de opinión. Si se sabe de antemano que la información es una opinión (dado que el sitio de donde se extrajo la información está dedicado a ofrecer opiniones), entonces es un problema relativamente fácil de resolver. En cambio, cuando se extrae información de sitios como blogs o páginas personales, se debe identificar la porción de la reseña que contiene algún tipo de opinión.

3. Una vez encontrada la opinión, se debe determinar el parecer o sentido que en ella se expresa, ya sea una opinión negativa o positiva y en ciertos casos una opinión neutral[10].

4. Finalmente, el sistema debe expresar la información encontrada de manera resumida y clara, ya sea con un valor numérico representando la opinión general, o con las frases más representativas de cada tipo de opiniones (positiva o negativa).

El desarrollo de esta tesis se enfoca a resolver los problemas dos y tres.

La investigación sobre minería de opiniones ha crecido rápidamente en la última década gracias a diversos factores:

- El mayor uso de métodos de aprendizaje de máquinas dentro del área de PLN y de recuperación de información.
- La disponibilidad de información, gracias al crecimiento de la Web específicamente de sitios donde se pueden verter opiniones.
- Los retos intelectuales y las aplicaciones comerciales que el área ofrece.

[10]En este trabajo, una opinión neutral se refiere a la clase de enunciados objetivos, es decir, aquellos que no contienen una opinión o juicio

2.4.2. Aplicaciones

Aplicación en sitios orientados a reseñas

Esta aplicación consiste en obtener información acerca de algún tema desde un sitio de la Web enfocado a recolectar reseñas y opiniones de los usuarios. Estas reseñas pueden ser después resumidas automáticamente e incluso se pueden corregir las calificaciones cuando están equivocadas (por ejemplo, cuando un usuario ha dado una calificación baja cuando su reseña es positiva).

Aplicación como una etapa dentro de otro sistema

La minería de opiniones puede ser usada como una etapa de otro sistema de minería de textos:

- En sistemas de recomendaciones, puede evitar que se ofrezcan artículos que tienen muy mala reputación.
- En anuncios en páginas web, convendría presentar solo marcas que tienen una buena impresión en los usuarios e incluso aumentar su frecuencia de aparición cuando se detecte que están siendo calificadas positivamente.
- En la recuperación de información, al eliminar oraciones subjetivas, se podrían mejorar los resultados.
- En los sistemas de pregunta respuesta, cuando se tiene una consulta orientada a una opinión, se le necesita dar un tratamiento especial: una respuesta basada no en hechos factuales sino en información subjetiva, pues es eso lo que el usuario necesita.

Aplicación en los negocios

La aplicación en los negocios es una de las razones más importantes dentro de las empresas para desarrollar este campo.

Cuando se desean conocer las razones del éxito o fracaso de un producto, es de gran utilidad contar con las opiniones que dieron fruto al prestigio actual de ese artículo, ya sea para continuar con su campaña publicitaria o hacer algún cambio de estrategia o en el producto.