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

CLASIFICACIÓN AUTOMÁTICA DE TEXTOS BASADO EN RANKING

FELIPE SEBASTIÁN BRICEÑO SEGOVIA

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

NOVIEMBRE 2011

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

CLASIFICACIÓN AUTOMÁTICA DE TEXTOS BASADO EN RANKING

FELIPE SEBASTIÁN BRICEÑO SEGOVIA

Profesor Guía: **Rodrigo Alfaro Arancibia**

Profesor Co-referente: **Broderick Crawford Labrín**

Carrera: **Ingeniería Civil en Informática**

NOVIEMBRE - 2011

Dedicatoria

A mis Padres por el amor y el apoyo incansable que me han entregado.
A mi Abuelita por guiarme en este largo camino llamado vida.

Índice

Resumen

La clasificación automática de textos mediante técnicas de ranking, consiste en la categorización de textos en base a una consulta, con el objetivo de generar un modelo de ranking que pueda ordenar los textos de acuerdo a sus grados de relevancia, preferencia e importancia, dentro de un conjunto de categorías predefinidas. Los modelos de Ranking se clasifican en tres grandes enfoques: Pointwise, Pairwise y Listwise. En la presente investigación se ha llevado a cabo un análisis de cada uno de éstos, en el cual se presentan las ventajas y desventajas respecto a los componentes de una máquina de aprendizaje tales como: espacio de entrada, espacio de salida, espacio de hipótesis y función de pérdida. Mediante la investigación se determinó, que el enfoque de Listwise es el que más se acerca a la idea de ranking; esto se debe a que relaciona en forma simultánea todos los documentos en base a una consulta, a diferencia de los otros métodos. En una segunda parte, se desarrolla el Modelo de Independencia Binaria (BIM), al cual se le realizan modificaciones para permitir el trabajo con datasets multi-etiqueta. Luego, se realizan pruebas de rendimiento del modelo en base a las medidas de evaluación presentadas. Finalmente, se concluye que el modelo BIM tiene un óptimo rendimiento al trabajar con datasets multi-etiqueta de distintos dominios como texto e imágenes.

Palabras-claves: Categorización de Textos, Aprendizaje de Máquinas, Ranking en IR.

Abstract

The automatic text classification using ranking techniques is text categorization based on a query, with the aim of generating a ranking model that can sort the texts according to their degrees of relevance, preference and importance, into a set of predefined categories. Ranking models fall into three broad approaches: Pointwise, Pairwise and Listwise. In the present investigation was carried out an analysis of each of these, which presents the advantages and disadvantages with respect to the components of a machine learning such as the input space, output space, hypothesis space and loss function. Through the investigation it was determined that the Listwise approach is the closest to the idea of ranking; this is because simultaneously linking all documents based on a query, unlike the other methods. In a second part develops the Binary Independence Model (BIM), in which changes are made for allow working with multi-label datasets. Then, the model was tested based on evaluation measures presented. Finally we conclude that the BIM model has an excellent performance when working with multi-label datasets from different domains such as text and images.

Keywords: Text Categorization, Machine Learning, Ranking in IR.

Lista de Figuras

Lista de Tablas

Glosario de Términos

Datasets: Conjunto de datos.

Feedback: Es el proceso de compartir observaciones, comentarios o sugerencias, con el fin de recabar información para intentar mejorar el funcionamiento de un sistema.

Framework: Es un conjunto estandarizado de conceptos, prácticas y criterios para enfocar un tipo de problemática en particular.

Ground truth: Representa la realidad tal y como es, en el caso de etiquetas ground truth hace alusión al valor real del documento clasificado.

Logit: Transformación logarítmica que permite su uso como una función lineal.

Perceptron: Es un tipo de red neuronal artificial desarrollado por Frank Rosenblatt.

Lista de Abreviaturas o Siglas

AP : Average Precision.

BIM : Binary Independence Model.

DCG : Discounted Cumulative Gain.

DOC : Documento.

H-loss : Hamming Loss.

IR : Information Retrieval.

MAP : Mean Average Precision.

ML : Machine Learning.

MRR : Mean Reciprocal Rank.

QP : Quadratic Programming.

SVM : Support Vector Machine.

TC : Text Categorization.

TERM : Término.

1 Introducción

Con el rápido desarrollo de la Web, cada uno de nosotros está experimentando una avalancha de información. Se estima que hay cerca de 15.97 billones de páginas en Internet hasta la fecha, lo que hace complejo clasificar los datos en forma manual y por otro lado, para los usuarios comunes localizar la información deseada. Este hecho ha motivado diversas investigaciones alrededor de este gran cúmulo de información. La Recuperación de Información, la Minería de Texto, la Extracción de la Información, la Búsqueda de Respuestas y la Categorización de Textos [1] son algunas de tantas líneas de investigación en éste ámbito. Ésta última explica cómo clasificar textos en distintas categorías, su representación y los distintos tipos que existen. En la Figura 1.1 se muestra como fluctúa la cantidad de páginas indexadas en uno de los motores de búsqueda más importante como es Google, el número total de páginas web indexadas en el mundo se calcula a través de la suma de las páginas indexadas en los motores más populares como es Yahoo, Bing, Google y Ask, cabe señalar que las páginas indexadas en varios buscadores se contabilizan una sola vez. La información digital sigue y sigue creciendo, dando paso a muchas investigaciones en éste ámbito, teniendo un gran auge la obtención de conocimiento a partir de esta información, es por eso que la clasificación automática cumple un rol vital en esta línea.

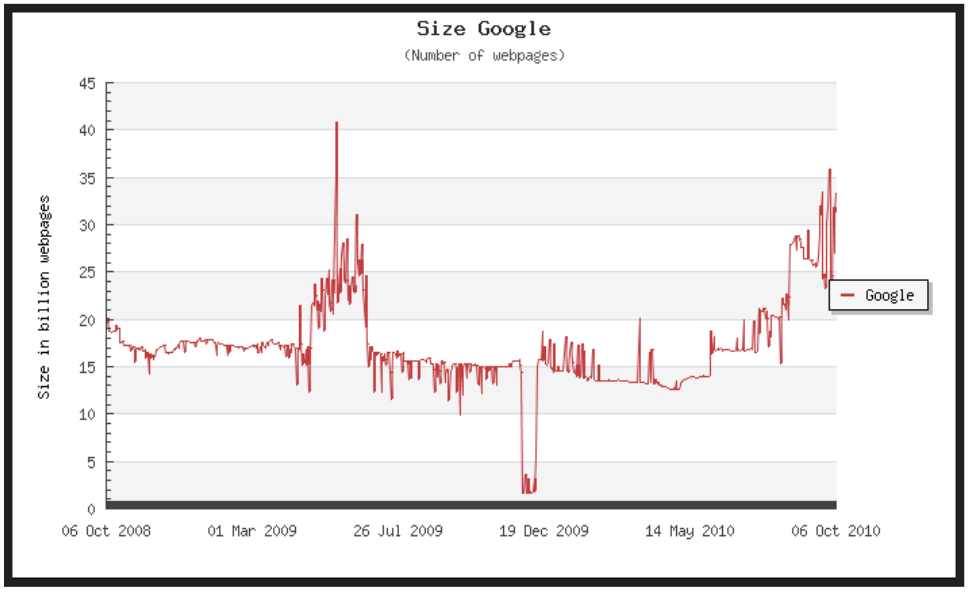


Figura 1.1 Gráfico del N° de páginas indexadas en Google en el último tiempo.

Por otro lado, la clasificación mediante ranking, es un problema central en la Recuperación de Información (IR). Muchos problemas IR son por naturaleza problemas de ranking, tales como la Recuperación de Documentos, Filtrado Colaborativo, Extracción de Términos Claves, Definición de la Búsqueda, Análisis de Sentido son abordados mediante técnicas, que relacionan un conjunto de elementos tales que, para uno o varios criterios, el primero de ellos presenta un valor superior al segundo, este a su vez mayor que el tercero y

así sucesivamente. La idea de ranking se presenta en la Figura 1.2, mostrando las aplicaciones top del momento.

Figura 1.2 Top Ranking de las aplicaciones.

Para generar un modelo de Ranking, se realiza una clasificación supervisada, cuya tarea consiste en generar un modelo a partir de datos de entrada y luego generar una salida. La funcionalidad del modelo generado consiste en predecir los datos de salida en base a nuevos datos de entrada.

El presente trabajo se orienta a la clasificación automática de textos mediante modelos de ranking, para ello se investigaron algoritmos de ranking que relacionen todo el conjunto de documentos frente a una consulta, así generar una lista de ranking más exacta frente a los requerimientos de los usuarios.

En una primera parte, se presentan los conceptos básicos relacionados al contenido de la investigación, partiendo desde la base de la Recuperación de Información, luego se presentan las principales nociones sobre la Clasificación de Textos y la representación de los documentos a procesar, y por último se presentan las Máquinas de Aprendizaje, dándose hincapié en las Máquinas de Soporte Vectorial.

En una segunda parte, se presentan los conceptos de ranking, sus modelos, y las medidas de evaluación que permiten evaluar estos modelos. Luego se presentan los distintos enfoques que existen con sus respectivos algoritmos de ranking, presentando sus ventajas y desventajas de cada enfoque.

En una tercera parte, se presenta la metodología que se llevará a cabo para la creación del prototipo, se presenta el algoritmo seleccionado y los datos que serán utilizados para para entrenar y luego realizar las pruebas.

En una cuarta parte, se presentan los resultados y el análisis del modelo implementado. Estos resultados se comparan con los resultados de otros modelos, así poder establecer ventajas y desventajas del modelo elegido.

Finalmente, se presentan las conclusiones respecto a la investigación realizada y las fortalezas y debilidades del modelo utilizado.

1.1 Motivación

La razón principal que motivo está línea de investigación, tiene directa relación con los motores de búsqueda y los resultados que estos generan en base a una consulta. Conocer las técnicas usadas en indexación, clasificación y búsqueda de documentos, y por otro lado la generación de listas de documentos ordenados en base a distintos criterios son los principales desafíos de este trabajo.

1.2 Objetivos

En esta sección se presenta el objetivo general y los objetivos específicos del proyecto.

1.2.1 Objetivos Generales

- Desarrollar un prototipo que permita la clasificación automática de textos utilizando técnicas de ranking.
- Comparar los resultados obtenidos con otras técnicas existentes.

1.2.2 Objetivos Específicos

- Investigar y analizar las técnicas de ranking para la clasificación automática de textos.
- Seleccionar un algoritmo de ranking para la clasificación automática de textos.
- Implementar un prototipo en base al algoritmo de ranking seleccionado.
- Comparar los resultados obtenidos con otros modelos, y establecer mejoras en el algoritmo.

2 Marco Teórico

A continuación se introducen los conceptos básicos claves para el entendimiento y comprensión del presente proyecto. Primero respecto a la recuperación de información, segundo la categorización de textos y sus distintas clasificaciones. Finalmente, se introduce el tema de aprendizaje de máquinas, para dar paso al modelo de espacio vectorial y sus aplicaciones.

2.1 Recuperación de la Información

Information Retrieval (IR) es la ciencia de la búsqueda de información en documentos, búsqueda de metadatos que describan documentos, o también la búsqueda en bases de datos relacionales, ya sea a través de internet, para textos, imágenes o sonidos, de manera pertinente y relevante. Por otro lado, puede ser descrito como el estudio de ayudar a los usuarios a encontrar la información de acuerdo a sus necesidades de consulta. Una consulta de usuario representa la información que el usuario necesita, y ésta puede ser agrupada de las siguientes formas:

- **Consulta por palabra clave:** el usuario expresa su necesidad de información con una lista de palabras claves con el objetivo de encontrar que el documento contenga algunos o todos los términos de la consulta. Los términos de la lista se supone que están relacionados con una versión suavizada de la lógica AND. El sistema de recuperación luego de encontrar los documentos relevantes y clasificarlos adecuadamente los presenta al usuario. Hay que tener en cuenta que los documentos recuperados no tienen por qué contener todos los términos de la consulta. En algunos sistemas de recuperación de información, el orden de las palabras si es importante y afecta en los resultados de la recuperación.

- **Consultas Booleanas:** El usuario puede utilizar operadores booleanos como AND, OR y NOT para construir consultas más complejas. Por lo tanto, dichas consultas se componen de términos y operadores booleanos. La consulta es devuelta, si la consulta booleana es verdadera.

- **Consultas de frases:** tales consultas consisten en una secuencia de palabras que componen una frase. Cada documento devuelto debe contener al menos una instancia de esa frase.

- **Consultas de Proximidad:** la proximidad de consultas, es una versión diferente de las consultas y las frases, ya que puede ser una combinación entre términos y frases. Consultas de proximidad busca los términos de consulta dentro de la cercanía de uno u otro, por lo que puede ser utilizada como un factor de ranking de los documentos devueltos. La mayoría de los motores de búsqueda tienen en cuenta la proximidad y el orden de los términos en la recuperación de información.

- **Consulta a Documentos Completos:** cuando la consulta es un documento completo, es decir, el usuario desea encontrar otros documentos que son similares al documento de la consulta realizada. Cuando se devuelve el usuario hace clic en el enlace y un conjunto de páginas similares a dicha página son presentadas.

- **Consultas en Lenguaje Natural:** este es el caso más complejo y también el caso ideal. El usuario expresa su necesidad de información como una pregunta en lenguaje natural. Sin embargo, dichas consultas siguen siendo difíciles de entender debido a la dificultad de comprensión del lenguaje natural.

2.2 Categorización de Textos

Text Categorization (TC) [1], tiene sus orígenes en IR y últimamente ha recibido más atención debido al incremento en la cantidad de información disponible en formato electrónico. Es por esta razón, que cada vez es mayor la necesidad de herramientas que ayuden a satisfacer las necesidades del usuario en cuanto a la información que busca, y además encontrar ésta en un tiempo adecuado. El objetivo de la clasificación de texto es categorizar documentos dentro de un número fijo de categorías predefinidas en función de su contenido, por ejemplo, un mismo documento puede pertenecer a una, varias, todas o ninguna de las categorías dadas. Cuando se utiliza aprendizaje automático, el objetivo es aprender a clasificar a partir de ejemplos que permitan hacer la asignación de categoría automáticamente.

Categorización de textos (TC), se define como un valor booleano para cada tupla $< d_j, c_i > \in D \, x \, C$, donde D corresponde al dominio de los documentos y $C = \{c_1, \dots, c_{|c|}\}$ a un conjunto de categorías predefinidas. Un valor verdadero True (T) asignado a la tupla $< d_j, c_i >$ indica la decisión de archivar d_j bajo c_i, es decir, el documento d_j pertenece a la categoría c_i, en caso contrario si el valor es False (F), no se archiva d_j bajo c_i, esta decisión también es conocida como *Hard Categorization*. Más formalmente, se desea aproximar la función objetivo $\breve{\Phi} : \mathrm{D \, x \, C} \rightarrow \{\mathrm{T}, \mathrm{F}\}$ (descripción de cómo son clasificados los textos), a la función $\Phi : \mathrm{D \, x \, C} \rightarrow \{\mathrm{T}, \mathrm{F}\}$ llamado clasificador, modelo o hipótesis, de tal manera que $\breve{\Phi}$ y Φ coincidan lo más posible. El grado de coincidencia entre la función objetivo y el clasificador es llamado *efectividad*.

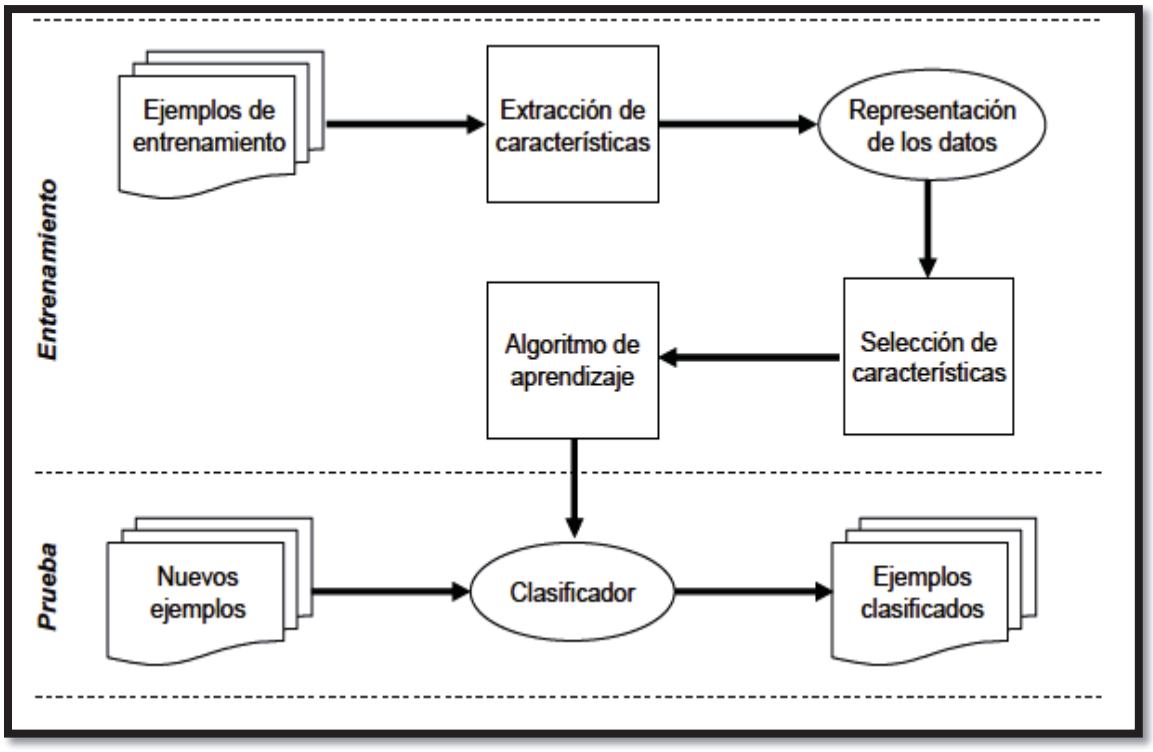


Figura 2.1 Un paradigma de aprendizaje.

En la Figura 2-1 podemos observar el paradigma de aprendizaje inductivo, el cual intenta aprender conceptos a través de ejemplos de éstos. El clasificador construido utiliza los conceptos aprendidos para clasificar los nuevos ejemplos. El aprender de ejemplos es un problema conocido como aprendizaje supervisado, ya que parte de una serie de clases o categorías diseñadas *a priori*, en las cuales hay que distribuir a cada uno de los documentos.

En resumen, la construcción de un clasificador automático de texto comienza con la recopilación y clasificación manual de un conjunto de documentos (documentos de entrenamiento), después se llevan los documentos a una representación adecuada para que finalmente se puedan aplicar distintos algoritmos de clasificación y así obtener el clasificador, más adelante se entra en detalle respecto a estos conceptos.

2.2.1 Categorización de Textos Simple-Etiqueta y Multi-Etiquetas

La categorización de textos Simple-Etiqueta, consiste en la asignación de una sola categoría a cada documento, en cambio, en Multi-Etiqueta se le puede asignar 0 a |C| categorías, a un mismo documento. Un ejemplo de Multi-Etiqueta, es la Categorización Binaria, la cual permite asignar varias categorías a un mismo documento, siempre y cuando las categorías sean independientes la una de la otra.

2.2.2 Categorización de Textos Categoría-Pivote y Documento-Pivote

Esta categorización es relevante al momento de elegir un clasificador, por un lado Categoría-Pivote consiste en que dada una categoría, encontrar todos los documentos que se relacionen con esta. En el caso de Documento-Pivote, consiste en buscar todas las categorías que tengan relación con un documento en específico.

2.2.3 Ranking

A menudo es difícil tomar la decisión binaria sobre si un documento es relevante o no para una consulta determinada. En cambio, los documentos se ordenan de acuerdo a sus grados de relevancia a la consulta, generando un ranking [2] de los documentos. Una forma de calcular ese grado de relevancia es calculando la similitud de la consulta q en cada documento d_j de la colección de documentos D, la clasificación de los documentos se realiza utilizando los valores de similitud. Los documentos destacados se consideran más relevantes para la consulta. La medida de similitud más conocida, es la similitud del coseno, que es el ángulo entre el vector de la consulta q y el vector del documento d_j:

$$\text{Coseno}\,(d_j, \text{q}) = \frac{\langle d_j \cdot q \rangle}{\left|\left|d_j\right|\right| \times ||q||} = \frac{\sum_{i=1}^{|V|} w_{ij} \times w_{iq}}{\sqrt{\sum_{i=1}^{|V|} w_{ij}^2} \times \sqrt{\sum_{i=1}^{|V|} w_{iq}^2}} \tag{2.2.1}$$

El producto escalar de dos vectores es otra medida de similitud,

$$\text{sim}(d_j, \text{q}) = \langle d_j * q \rangle = \sum_{i=1}^{|V|} w_{ij} \times w_{iq} \tag{2.2.2}$$

2.2.4 Aplicaciones de la Categorización de Textos

- Indexación automática de los Sistemas de IR booleanos: a cada documento se le asigna una o más palabras o frases claves describiendo su contenido, donde estas palabras o frases claves pertenecen a un conjunto finito, llamado *Diccionario Controlado*.

- Organización de Documentos: consiste en la organización de documentos, mediante la clasificación de éstos respecto a una categoría predefinida.

- Filtrado de Texto: es una actividad de clasificación a colecciones dinámicas de textos. Puede ser visto como un caso de Simple-Etiqueta.

- Sentido de las Palabras Ambiguas: se refiere a la actividad de encontrar, dada la ocurrencia en el texto, una palabra ambigua e identificar el sentido que tiene esta palabra en el texto.

- Categorización Jerárquica de Páginas Web: clasificación automática de páginas Web, en una o varias categorías ordenadas en forma jerárquica.

2.3 Representación de los Documentos

Para llevar a cabo la clasificación automática de texto, se tiene que representar cada documento de los ejemplos de entrenamiento, de manera que a esa representación se le pueda aplicar el algoritmo de clasificación. La representación más utilizada es el *modelo vectorial* [3], ésta es manejada ampliamente por los sistemas de recuperación de información.

Este modelo consiste en representar la colección de documentos como una matriz de palabras o términos por documentos. Es decir, cada texto o documento d_j es representado por medio de un vector $d_j = \{W_{1j}, \ldots, W_{|\tau|j}\}$ de términos W, donde τ es el conjunto de palabras del vocabulario de colección y W_{ij} representa un valor numérico que expresa en qué grado el documento d_j posee el término t_i. Frecuentemente, el conjunto τ es el resultado de filtrar las palabras del vocabulario con respecto a una lista de palabras vacías, éstas son palabras frecuentes que no contienen información semántica. Ejemplos de palabras vacías (también llamadas palabras funcionales) son las preposiciones, conjunciones, artículos, etc. Otra estrategia es el uso de un lematizador el cual tiene como objetivo eliminar afijos de una palabra de tal manera que aparezca sólo su raíz léxica. Esto se realiza con la finalidad de que las palabras que tienen el mismo significado conceptual sean representadas por su raíz léxica, por ejemplo, caminar, caminará, caminó, caminando se representa por *camin*. Con respecto al peso del término W_{ij} se tiene distintas maneras de calcularlo. A continuación se da una breve descripción de tres tipos de pesado.

- Ponderado booleano: Asigna el peso de 1 si la palabra t_i está en el documento d_i y 0 en caso contrario.

$$W_{ij} \begin{cases} 1 \;\; si\; t_i\; aparece\; en\; d_i \\ \\ 0 \;\; en\; caso\; contrario \end{cases}$$

- Ponderado por frecuencia de término: Asigna el número de veces que el termino i ocurre en el documento d_j, se denota como f_{ij}. Este cálculo se debe a que si un término aparece muchas veces en un documento, se supone que es importante en ese documento. Las deficiencias son que no tiene en cuenta la situación en la que un término aparece en muchos documentos de la colección, en el cual este término puede ser discriminatorio. Es posible aplicar normalización.

$$W_{ij} = f_{ij} \tag{2.3.1}$$

- Ponderado TF-IDF: Asigna el peso de la palabra i en el documento j en proporción al número de ocurrencias de la palabra en el documento y en proporción inversa al número de documentos en la colección para los cuales ocurre la palabra al menos una vez. TF se entiende como la frecuencia del término e IDF como la frecuencia inversa del documento.

$$W_{ij} = f_{ij} * \log \frac{N}{n_i} \quad (2.3.2)$$

2.4 Aprendizaje de Máquinas

Machine Learning (ML), es una rama de la Inteligencia Artificial cuyo objetivo es el desarrollo de técnicas para que las máquinas puedan aprender por si solas. La función principal es construir un clasificador automático de textos, basado en el aprendizaje de las características de un grupo de documentos previamente clasificados.

Las ventajas de este enfoque es que entrega resultados en forma automática y exacta, aliviando los juicios de relevancia que deben realizar los usuarios expertos a cada documento.

Los algoritmos de aprendizaje automático se agrupan en:

- Aprendizaje Supervisado: El algoritmo se genera en función del espacio de entrada y el espacio de salida, produciendo una función que establezca una correspondencia entre las entradas y las salidas deseadas.

- Aprendizaje No Supervisado: El algoritmo se genera sólo en función del espacio de entrada, no se tiene información de cómo debe ser el espacio de salida.

- Aprendizaje por Refuerzo: El algoritmo aprende del entorno que rodea al sistema, su espacio de entrada es el *feedback* o retroalimentación que obtiene del entorno como respuesta a sus acciones.

- Aprendizaje Multi-tarea: El algoritmo usa conocimiento previamente aprendido por el modelo o sistema, así poder enfrentar problemas parecidos a los ya vistos.

2.4.1 Máquina de Soporte Vectorial

Support Vector Machine (SVM), son un conjunto de algoritmos de aprendizaje supervisado, basado en la teoría de minimización de riego estructural. La técnica de clasificación está basada en la construcción de un modelo que predice si un punto en el espacio pertenece a una categoría u otra. En el ámbito de la clasificación de textos, documentos y consultas se representan como vectores en un espacio euclidiano, en el cual el producto escalar de dos vectores puede ser usado para medir sus similitudes.

Dado un conjunto de entrenamiento podemos etiquetar las clases y entrenar un SVM para construir un modelo que prediga la clase de una nueva muestra. Intuitivamente, una SVM es un modelo que representa a los puntos de muestra en el espacio, separando las clases por un espacio lo más amplio posible. Cuando las nuevas muestras se ponen en correspondencia con dicho modelo, en función de su proximidad pueden ser clasificadas en una u otra clase.

Más formalmente, SVM construye un hiperplano o conjunto de hiperplanos en un espacio de dimensionalidad muy alta, que puede ser utilizado en problemas de clasificación o regresión. Una buena separación entre las clases permitirá una clasificación correcta dado un conjunto de puntos y subconjuntos de un conjunto mayor (espacio), en el que cada uno de ellos pertenece a una de dos posibles categorías. Por lo que, un algoritmo basado en SVM construye un modelo capaz de predecir si un punto pertenece a una categoría o a la otra. Como la mayoría de los métodos de clasificación supervisada, los datos de entrada son vistos como un vector característico p-dimensional.

En ese concepto de "separación óptima" es donde reside la característica fundamental de las SVM, este tipo de algoritmos buscan el hiperplano que tenga la máxima distancia (margen) respecto a los puntos más cercanos a él. Por eso también a veces se les conoce a las SVM como *clasificadores de margen máximo*. De esta forma, los puntos del vector etiquetados con una categoría estarán a un lado del hiperplano y los casos que se encuentren en la otra categoría estarán al otro lado. Los algoritmos SVM pertenecen a la familia de los clasificadores lineales y están estrechamente relacionados con las redes neuronales.

La clasificación mediante SVM permite obtener clasificadores lineales y no lineales. En los siguientes puntos se analiza el clasificador lineal para datos separables y no separables, y luego el clasificador no lineal, en donde se introduce el concepto de *Kernel*.

Recapitulando, un SVM primero mapea los puntos de entrada a un espacio de características de una dimensión mayor y encuentra un hiperplano que los separe y maximice el margen *m* entre las clases en este espacio como se aprecia en la Figura 2.2.

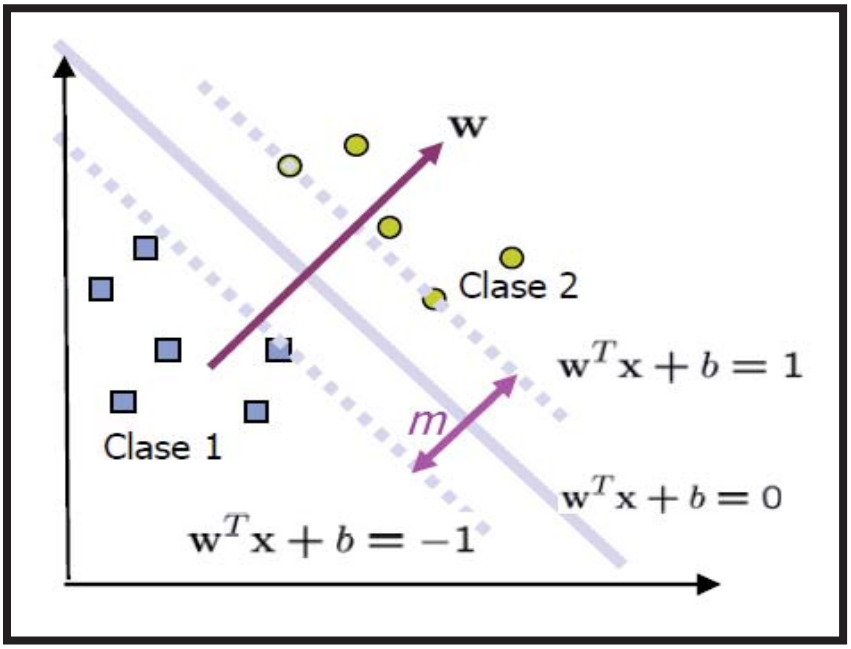


Figura 2.2 Teoría de las Máquinas de Soporte Vectorial.

2.4.1.1 Clasificaciones de SVM

A continuación se presentan los distintos casos en los cuales se aplica SVM en [4].

2.4.1.1.1 Caso linealmente separable

Considere el problema de separar el conjunto S de vectores de entrenamiento $(y_1,x_1),\ldots,(y_i,x_i)\ \epsilon R^n$, como se aprecia en la Figura 2.3.

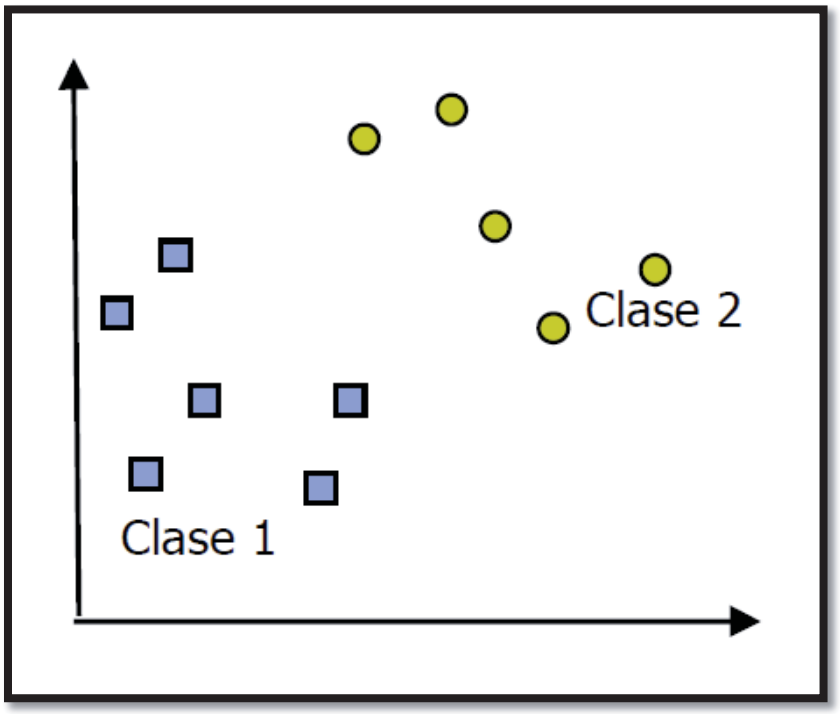


Figura 2.3 Caso linealmente separable.

Cada punto de entrenamiento $x_i \in R^n$ pertenece a alguna de las dos clases y se le ha dado una etiqueta $y_i \in \{-1, 1\}$ para i = -1,..,l. En la mayoría de los casos, la búsqueda de un hiperplano adecuado en un espacio de entrada es demasiado restrictiva para ser de uso práctico. Una solución a esta situación es mapear el espacio de entrada en un espacio de características de una dimensión mayor y buscar el hiperplano óptimo allí. Sea $z = \Phi(x)$ la notación del correspondiente vector en el espacio de características Z. Deseamos encontrar el hiperplano:

$$w \cdot z + b = 0 \tag{2.4.1}$$

Definido por el par (w,b) tal que podamos separar el punto x_i de acuerdo a la función:

$$f(x_i) = \text{signo}(w \cdot z_i + b) = \begin{cases} 1 & y_i = 1 \\ -1 & y_i = -1 \end{cases} \tag{2.4.2}$$

Donde w Є Z y b Є R. Más precisamente, el conjunto S se dice que es linealmente separable si existe (w,b) tal que las inecuaciones:

$$\begin{cases} (w \cdot z_i + b) \geq 1 \ , \ y_i = 1 \\ (w \cdot z_i + b) \leq -1 \ , \ y_i = -1 \end{cases} \quad \text{í = -1,….,l} \qquad (2.4.3)$$

Sean válidas para todos los elementos del conjunto S. para el caso linealmente separable de S, podemos encontrar un único hiperplano optimo, para el cual, el margen entre las proyecciones de los puntos de entrenamiento de dos diferentes clases es maximizado.

2.4.1.1.2 Caso linealmente no separable

Si el conjunto S no es linealmente separable, violaciones a la clasificación deben ser permitidas en la formulación de la SVM.

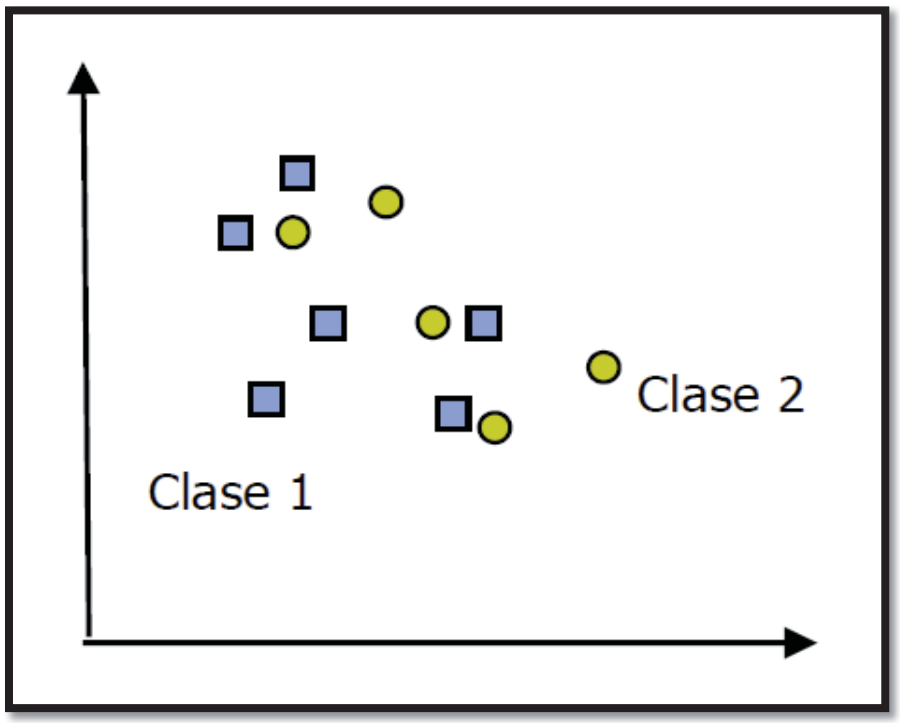


Figura 2.4 Caso linealmente no separable.

Para tratar con datos que no son linealmente separables, el análisis previo puede ser generalizado introduciendo algunas variables no-negativas $\xi_i \geq 0$ de tal modo que la ecuación (2.4.3) es modificada a

$$y_i\,(\mathrm{w}\cdot z_i + \mathrm{b}) \geq 1 - \xi_i\,,\ \ \acute{\imath} = 1,\ldots,\mathrm{l} \tag{2.4.4}$$

Los $\xi_i \neq 0$ en Ecuación 2.4.4 son aquellos para los cuales el punto x_i no satisface Ecuación 2.4.3. Entonces el término $\sum_{i=1}^{l} \xi_i$ puede ser tomado como algún tipo de medida del error en la clasificación.

El problema del hiperplano óptimo es entonces redefinido como la solución al problema:

$$\min\left\{\frac{1}{2}\, w\ \cdot w + C\ \sum_{i=1}^{l} \xi_i\right\} \tag{2.4.5}$$

$$\text{s.a}\quad y_i\,(\mathrm{w}\cdot z_i + \mathrm{b}) \geq 1 - \xi_i\,,\quad \acute{\imath} = 1,\ldots,\mathrm{l}$$

$$\xi_i \geq 0\,,\ \acute{\imath} = 1,\ldots,\mathrm{l}$$

La constante *C*, puede ser definido como un parámetro de regularización, que controla el *trade-off* entre la maximización del margen y la minimización del error de entrenamiento. Así, cuando *C* es pequeño, se permite equivocar (clasificar mal) muchas veces, pero en cambio se obtiene un margen grande. Por otro lado, cuando *C* es grande, no se permite equivocar (se pena altamente por cada error) y se obtiene un margen pequeño, cabe destacar que C es el primer parámetro que debe ser ajustado.

Buscando el hiperplano óptimo en la Ecuación 2.4.5 es un problema de Programación Cuadrática (QP), que puede ser resuelto construyendo un Lagrangiano y transformándolo en el dual.

$$\text{Max } \mathrm{W}(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \alpha_i\ \alpha_j\ y_i\ y_j\ z_i\ \cdot\ z_j \tag{2.4.6}$$

$$\text{s.a}\ \sum_{i=1}^{l} \alpha_i = 0\,,\ 0 \leq \alpha_i \leq \mathrm{C}\,,\ \acute{\imath} = 1,\ldots,\mathrm{l}$$

Donde $\alpha = (\alpha_1,\ldots,\alpha_l)$ es un vector de multiplicadores de Lagrange positivos asociados con las constantes en la Ecuación 2.4.4.

El teorema de Khun-Tucker juega un papel importante en la teoría de las SVM. De acuerdo a este teorema, la solución $\overline{\alpha_i}$ del problema en la Ecuación 2.4.6 satisface:

$$\overline{\alpha_i}\ (\ y_i\ (\ \overline{w}\cdot z_i + \overline{b}\ ) - 1 + \overline{\xi_i}\ ) = 0,\quad \acute{\imath} = 1,\ldots,\mathrm{l} \tag{2.4.7}$$

$$(\mathrm{C} - \overline{\alpha_i}\ )\ \xi_i = 0,\quad \acute{\imath} = 1,\ldots,\mathrm{l} \tag{2.4.8}$$

De esta igualdad se deduce que los únicos valores $\overline{\alpha_i} \neq 0$ en la Ecuación 2.4.8 son aquellos que para las constantes de la Ecuación 2.4.6 son satisfechas con el signo de igualdad. El punto x_i correspondiente con $\overline{\alpha_i} > 0$ es llamado *vector de soporte*. Pero hay dos

tipos de vectores de soporte en un caso no separable. En el caso $0 < \overline{\alpha_i} < C$, el correspondiente vector de soporte x_i satisface las igualdades $y_i (\overline{w} \cdot z_i + \overline{b}) = 1$ y $\xi_i = 0$. En el caso $\overline{\alpha_i} = C$, el correspondiente ξ_i es diferente de cero y el correspondiente vector de soporte x_i no satisface la Ecuación 2.4.3. Nos referimos a estos vectores de soporte como errores. El punto x_i correspondiente con $\overline{\alpha_i} = 0$ es clasificado correctamente y está claramente alejado del margen de decisión mostrado en la Figura 2.5.

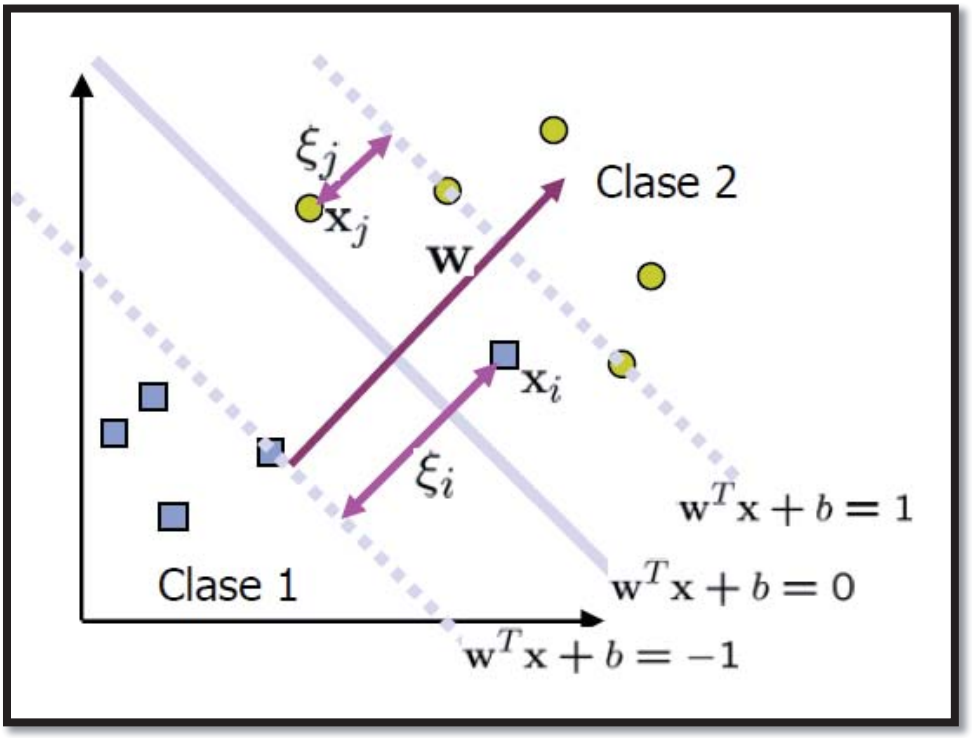


Figura 2.5 Parámetro de error en la clasificación.

Para construir el hiperplano óptimo $\overline{w} \cdot z + \overline{b}$, se utiliza:

$$\overline{w} = \sum_{i=1}^{l} \overline{\alpha_i}\, y_i\, z_i \qquad (2.4.9)$$

Y el escalar *b* puede ser determinado de las condiciones de Kuhn-Tucker mostrado en la Ecuación 2.4.9. La función de designación generalizada de las Ecuaciones 2.4.3 y 2.4.10 es tal que

$$f(x) = \text{signo} (w \cdot z + b) = \text{signo} \left(\sum_{i=1}^{l} \alpha_i\, y_i\, z_i \cdot z + b \right) \qquad (2.4.10)$$

2.4.1.1.3 Caso no lineal

En una SVM, el hiperplano óptimo es determinado para maximizar su habilidad de generalización. Pero, si lo datos de entrenamiento no son linealmente separables, es decir, no se pueden separar las clases dentro del espacio de solución original, el clasificador obtenido puede no tener una alta habilidad de generalización, aun cuando los hiperplanos sean determinados óptimamente. Por este motivo, para poder maximizar el espacio entre clases (hiperplano óptimo), el espacio de entrada original es transformado dentro de un espacio de mayor dimensión llamado "espacio de características", como se muestra en la Figura 2.6. La técnica de SVM puede crear una hipersuperficie de decisión no lineal, capaz de clasificar datos separables no linealmente. Generalmente, para patrones de entrada n-dimensionales, en lugar de una curva no lineal, SVM creará una hipersuperficie de separación no lineal.

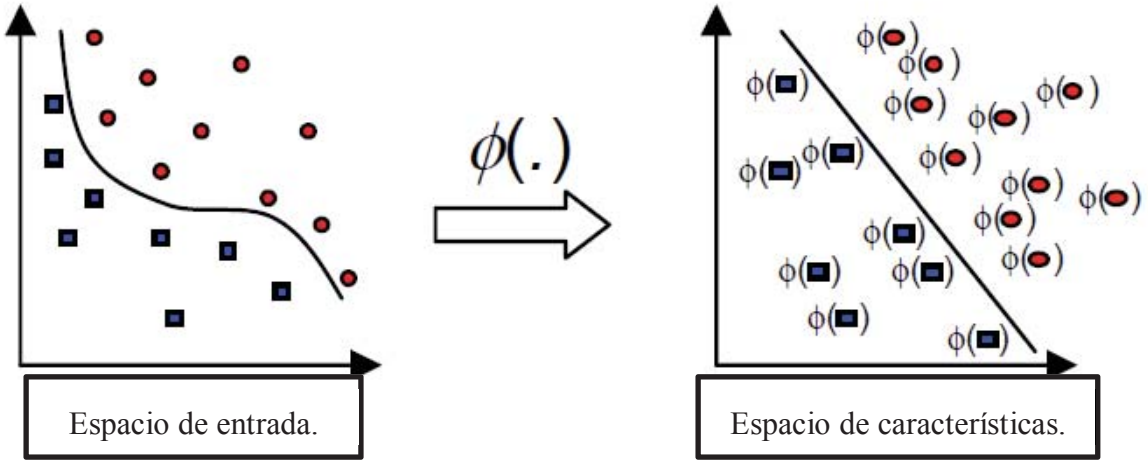


Figura 2.6 Caso no lineal.

Truco del Kernel

Para enfrentar el caso no lineal, se necesita usar una función K ( ), llamada kernel que calcula el producto punto de los puntos de entrada en el espacio de características Z, esto es:

$$z_i * z_j = \varphi(x_i) * \varphi(x_j) = K\left(x_{i,}\, x_j\right) \tag{2.4.11}$$

Las funciones que satisfacen el teorema de Mercer pueden ser usadas como productos punto y por ende ser usada como kernel. Podemos usar el kernel polinomial de grado d:

$$K\left(x_{i,}\, x_j\right) = (1 + x_i * x_j)^d \tag{2.4.12}$$

Para construir un clasificador SVM. Entonces el hiperplano no lineal de separación puede ser encontrado como la solución de:

$$\text{Max: } W(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j K(x_i, x_j) \quad (2.4.13)$$

$$\text{Sujeto a: } \sum_{i=1}^{n} y_i \alpha_i = 0,\ 0 \leq \alpha_i \leq C,\ i = 1, \ldots, l$$

Y la función de decisión es:

$$f(x) = signo\left(\sum_{i=1}^{l} y_i \alpha_i K(x_i, x_j) + b\right) \quad (2.4.14)$$

Además, usando diferentes funciones kernel, el algoritmo de SVM puede construir una variedad de máquinas de aprendizaje. Algunos tipos de kernel son: Lineal, Polinomial, Función de Base Radial y Sigmoidal entre otros.

2.5 Recuperación Probabilística

En recuperación probabilística, se examinan algunos modelos de IR que han tenido importancia ya sea en la investigación como en la práctica. Un modelo probabilístico, puede ser usado como un modelo estadístico, en referencia con modelos de relevancia o modelos basados en el contenido de los documentos. El modelo de recuperación probabilística en la actualidad, es uno de los modelos más exitosos en IR.

2.5.1 Modelo de Relevancia

El modelo probabilístico tiene su punto de partida en la "Probabilidad del principio de Ranking", la cual indica que si un sistema IR responde a cada consulta, generando un ranking de los documentos de dicha colección en orden decreciente de acuerdo a su probabilidad de relevancia, la eficacia general del sistema para el usuario será maximizada.

Dada una consulta, se determina la probabilidad de la relevancia para cada documento en la colección, de acuerdo al tipo de documento. Entonces se necesita conocer la probabilidad que el usuario juzgará el documento como relevante para dicha consulta. Por lo que ahora el problema se centra en estimar esta probabilidad. Esto se realiza introduciendo tres variables aleatorias: D para documentos, Q para consultas y una variable aleatoria binaria R para el juicio de relevancia. Ahora se debe estimar la probabilidad de un juicio relevante positivo, el cual se expresa como sigue:

$$p(R=1 \mid D=d, Q=q) \quad (2.5.1)$$

$$p(R=1 \mid D, Q) \quad (2.5.2)$$

Similarmente, el complemento de la fórmula anterior es:

$$p(R=1 \mid D, Q) = 1 - p(R=0 \mid D, Q) \quad (2.5.3)$$

Teorema de Bayes es un teorema fundamental en la teoría de la probabilidad, y establece lo siguiente:

$$P(A \mid B) = \frac{P(B|A)P(A)}{P(B)} \tag{2.5.4}$$

Aplicando el Teorema de Bayes a la Ecuación 2.5.3 queda:

$$P(R=1 \mid D, Q) = \frac{P(D,Q|R=1)\,P(R=1)}{P(D,Q)} \tag{2.5.5}$$

$$P(R=0 \mid D, Q) = \frac{P(D,Q|R=0)\,P(R=0)}{P(D,Q)} \tag{2.5.6}$$

Ahora, en lugar de seguir trabajando directamente con probabilidades, se debe cambiar a ***logit***, formulación que simplificará la presentación y manipulación de las ecuaciones. Dada una probabilidad p, el logit de p se define como:

$$\text{logit}(p) = \log\left(\frac{p}{1-p}\right) \tag{2.5.7}$$

Donde la base del logaritmo puede ser elegida arbitrariamente, en este caso se utiliza una base específica para los ejemplos o experimentos que corresponde a la base 2.

Logit tiene un número importante de propiedades. Como p varía entre 0 y 1, logit(p) varía entre -∞ y ∞. Dado 2 probabilidades p y q, logit(p) > logit(q) ssi p > q. Así logit y probabilidad son de rango equivalente, rankeando por uno produce el mismo orden que rankeando por el otro.

Tomando el logit de la Ecuación 2.5.2 y aplicando el teorema de Bayes queda lo siguiente:

$$\log\frac{p(R \mid D,Q)}{1-p\,(R \mid D,Q)} = \log\frac{p\,(R \mid D,Q)}{p\,(\neg R \mid D,Q)} \tag{2.5.8}$$

$$\log\frac{p(D,Q \mid R)\,p(R)}{p\,(D,Q \mid \neg R)\,p(\neg R)} \tag{2.5.9}$$

Un beneficio inmediato de cambiar a logit es que el término p(D, Q) se anula y no necesita ser estimado. Se puede expandir las probabilidades conjuntas en la Ecuación 2.5.8 en probabilidades condicionales usando la igualdad p(D, Q | R) = p(D | Q, R) * p(Q | R). Expandiendo las probabilidades conjuntas en esta vía y aplicando el teorema de Bayes en una segunda oportunidad tenemos:

$$\log\frac{p(D,Q \mid R)\,p(R)}{p\,(D,Q \mid \neg R)\,p(\neg R)} = \log\frac{p(D,Q \mid R)\,p(Q \mid R)\,p(R)}{p\,(D,Q \mid \neg R)\,p(Q \mid \neg R)p(\neg R)} \tag{2.5.10}$$

$$= \log \frac{p(D\,|Q,R)\,p(R\,|Q)}{p\,(D\,|Q,\neg R)\,p(\neg R\,|Q)} \tag{2.5.11}$$

$$= \log \frac{p(D\,|Q,R)}{p\,(D\,|Q,\neg R)} + \log \frac{p(R\,|Q)}{p\,(\neg R|Q)} \tag{2.5.12}$$

El término "$\log \frac{p(R\,|Q)}{p\,(\neg R|Q)}$" es independiente de D. Se puede considerar como un indicador de la dificultad de la consulta. En cualquier caso puede ser ignorado para propósitos de ranking, dejándonos con la fórmula de ranking que se muestra a continuación:

$$\log \frac{p(D\,|Q,R)}{p\,(D\,|Q,\neg R)} \tag{2.5.13}$$

Ésta fórmula se encuentra en el corazón del modelo de recuperación probabilística, y se regresará a ella en varias ocasiones al examinar varios métodos para estimar éste valor.

3 Ranking en IR

3.1 Introducción

Ranking permite la clasificación de documentos ordenados en base a una consulta, ubicando a los documentos con mayor relevancia en los primeros lugares. Existen diferentes escenarios de ranking para la recuperación de documentos de interés:

- Escenario 1: rankear los documentos únicamente en función de su relevancia respecto a una consulta.
- Escenario 2: considerar las relaciones de similitud, estructura del sitio Web y la diversidad en conjunto con otros documentos en el proceso de ranking.
- Escenario 3: agregar varios candidatos en las listas rankeadas, así obtener mejores resultados.
- Escenario 4: encontrar en qué grado una característica de una página Web influye en el resultado del ranking.

3.2 Modelos convencionales de Ranking para IR

En la literatura de IR, muchos modelos de ranking se han propuesto [5], ya que pueden ser categorizados como modelos de consulta-dependiente y modelos de consulta-independiente.

3.2.1 Modelos de consulta-dependiente

Los primeros modelos recuperaban documentos basados en las ocurrencias de los términos de consulta en los documentos, como por ejemplo el modelo booleano. Básicamente estos modelos pueden predecir si un documento es relevante a la consulta o no, pero no pueden predecir el grado de relevancia. Para predecir el grado de relevancia, fue propuesto el modelo de espacio vectorial (SVM). Ambos documentos y consultas se representan como vectores en un espacio euclidiano, en el cual el producto escalar de dos vectores puede ser usado para medir sus similitudes.

Además de los ejemplos anteriores, muchos otros modelos también se han propuesto para calcular su relevancia entre una consulta y un documento. Entre ellos están, tomar la proximidad de los términos en consideración a la consulta, considerar la relación entre los documentos en términos de similitud de contenido, estructura del hipervínculo, estructura del sitio Web y el tópico de diversidad.

3.2.2 Modelos de consulta-independiente

Son modelos que clasifican sus documentos en base a su propia importancia. Tomaremos PageRank como ejemplo. Este modelo es especialmente aplicable en la búsqueda Web, ya que hace uso de la estructura de enlace de la Web para su clasificación.

PageRank usa la probabilidad, de que un usuario que navega en la red haciendo clics al azar en los enlaces llegue a una determinada página Web, de esta forma rankear las páginas.

3.3 Nivel de consultas basado en la posición en evaluaciones IR

Dado el gran número de modelos de ordenación que existe, es necesario contar con un mecanismo de evaluación estándar para seleccionar el modelo más eficaz. A continuación se presentan una serie de pasos a seguir para la selección del modelo:

- Recolectar una gran cantidad de consultas para formar un equipo de prueba.
- Para cada consulta q:
    - Recolectar todos los documentos asociados con la consulta.
    - Obtener la relevancia del juicio hecha por humanos para cada documento.
    - Utilizar un determinado modelo de ranking para rankear los documentos.
    - Medir la diferencia entre los resultados del ranking y la relevancia del juicio usando una medida de evaluación.
- Usar la medida de la media sobre todas las consultas en el conjunto de prueba para evaluar el desempeño del modelo de ranking.

Respecto a la relevancia del juicio, tres estrategias fueron utilizadas en este trabajo:

1. Si un documento es relevante o no a la consulta, es decir, el juicio binario 1 ó 0, o especificar el grado de importancia del documento, es decir, múltiples categorías ordenas, por ejemplo: perfecto, excelente, bueno, regular o malo.
2. Si un documento es más relevante que otro con respecto a una consulta.
3. Especificar el orden parcial o total de los documentos con respecto a una consulta.

Como el juicio manual consume mucho tiempo, es casi imposible juzgar a todos los documentos respecto a una consulta. En consecuencia, siempre hay documentos sin juzgar devueltos por el modelo de ranking (documentos irrelevantes al proceso de evaluación).

Respecto a la relevancia del juicio, una serie de medidas de evaluación se han propuesto y utilizado en la literatura IR. Es evidente que la compresión de estas medidas será muy importante para el aprendizaje de rankear, ya que en cierta medida, definen la verdadera función objetivo de la clasificación.

3.3.1 Medidas de Evaluación

Para evaluar los modelos de ranking se utilizan varias medidas de evaluación extraídas de [2], a continuación se detallan algunas medidas de evaluación para medir rendimiento de los algoritmos de ranking:

- Hamming Loss (hloss): indica número de etiquetas que no han sido bien clasificadas, respecto a la etiqueta real.

$$hloss(h) = \frac{1}{d}\sum_{i=1}^{d}\frac{1}{|L|}|h(x_i)\Delta\, Y_i| \tag{3.3.0}$$

- Mean Reciprocal Rank (MRR): indica la media de los valores individuales alcanzados para cada consulta de la colección. Para una consulta q implica que r(q), es el primer documento relevante recuperado para dicha consulta. Entonces MRR es definido como:

$$MRR = \frac{1}{r(q)} \tag{3.3.1}$$

- Mean Average Precision (MAP): trata de calcular una media de la precisión hallada a distintos niveles de cobertura. Para definir MAP, primero necesitamos definir precisión en la posición k (P@K):

$$P@k(q) = \frac{\#\{documentos\ relevantes\ en\ lo\ alto\ de\ las\ k\ posiciones\}}{k} \tag{3.3.2}$$

Luego, la Precisión Promedio (AP) se define como sigue:

$$AP(q) = \frac{\sum_{k=1}^{m} P@k(q) * \mathrm{l_k}}{\#\{documentos\ relevantes\}} \tag{3.3.3}$$

Donde m es el número total de documentos asociados a la consulta q y $\mathrm{l_k}$ es el juicio binario sobre la relevancia del documento en la posición k. Por lo que MAP se define como la media aritmética de la Precisión Promedio (AP).

A continuación se presenta un ejemplo de ésta medida de evaluación.

Precisión en la posición K (p@k):

Cabe señalar que cada cuadrado corresponde a un documento clasificado, el cuadrado de color verde corresponde a posición relevante del ranking, y el cuadrado de color rojo corresponde a una posición irrelevante en la lista de ranking

- p@3 = 2/3, documentos relevante 2, hasta la posición 3.
- p@4 = 2/4, documentos relevante 2, hasta la posición 4.
- p@5 = 3/5, documentos relevante 3, hasta la posición 5.

Average Precision (AP):

- Considera los p@k relevantes a la consulta solamente.

Ejemplo:

$$\text{AP} = \frac{1}{3} * \left(\frac{1}{1} * \frac{2}{3} * \frac{3}{5}\right) = 0.76 \quad (3.3.4)$$

- Discounted Cumulative Gain (DCG): si bien las medidas mencionadas antes se dirigen principalmente a decisiones binarias, DCG [6] puede aprovechar la relevancia del juicio en términos de múltiples categorías ordenadas, y tiene un factor de descuento de posición explícito en su definición. Más formalmente, supongamos que la lista rankeada para la consulta q es π, entonces DCG en la posición k es definido como sigue:

$$DCG@k(q) = \sum_{r=1}^{k} G\left(\pi^{-1}(r)\right)\eta(r) \quad (3.3.5)$$

Donde $\pi^{-1}(r)$ es el documento rankeado en la posición r de la lista π, G( ) es el rating de un documento y $\eta(r)$ es un factor de descuento de posición.

Al normalizar DCG@k con el máximo valor de él, llamado Z_k, se obtendrá otra medida llamada DCG Normalizada (NDCG), y esta es:

$$NDCG@k(q) = \frac{1}{Z_k}\sum_{r=1}^{k} G\left(\pi^{-1}(r)\right)\eta(r) \quad (3.3.6)$$

- Ejemplo:

Primero generamos nuestro Rating relevante, correspondiente a las etiquetas que le asignaremos a un documento respecto a una consulta.

$$\text{Ganancia} = \left(2^{r(j)} - 1\right) \quad (3.3.7)$$

Tabla 3.1 Rating relevante y ganancias.

Rating relevante	**Valor (ganancia)**
Perfecto	$2^5 - 1 = 31$
Excelente	$2^4 - 1 = 15$

Bueno	$2^3 - 1 = 7$
Justo	$2^2 - 1 = 3$
Malo	$2^0 - 1 = 0$

Agregamos una consulta cualquiera:

Consulta: {tvn}

Tabla 3.2 Calculo de ganancia acumulativa.

	URL	**Ganancia**	**Ganancia acumulativa**
# 1	http://www.tvn.cl	31	31
# 2	http://www.tvnormandia.nm	3	$31 + 3 = 34$
# 3	http://www.tvndeportes.cl	15	$34 + 15 = 49$
# 4	http://www.tvnnoticias.cl	15	$49 + 15 = 64$

Ahora calculamos el factor de descuento:

Factor de descuento: Log (2) / (Log (1+ rank)) (3.3.8)

Tabla 3.3 Cálculo del factor de descuento.

	URL	**Ganancia**	**Descuento de la ganancia acumulativa**
# 1	http://www.tvn.cl	31	31*1 = 31
# 2	http://www.tvnormandia.nm	3	$31 + 3 * 0.63 = 32.9$
# 3	http://www.tvndeportes.cl	15	$32.9 + 15 * 0.50 = 40.4$
# 4	http://www.tvnnoticias.cl	15	$40.4 + 15 * 0.43 = 46.9$

Finalmente, con los datos anteriormente calculados se puede definir una lista de ranking, con los documentos en este caso URL más relevantes respecto a la consulta.

Tabla 3.4 Documentos ordenados en base a sus etiquetas.

	URL	Ganancia	Max DCG
# 1	http://www.tvn.cl	31	31*1 = 31
# 2	http://www.tvndeportes.cl	15	$31 + 15 * 0.63 = 40.5$
# 3	http://www.tvnnoticias.cl	15	$40.5 + 15 * 0.50 = 48$

En resumen, hay algunas propiedades comunes en estas medidas de evaluación, estas son:

1. Todas estas medidas de evaluación se calculan a nivel de consulta, es decir, primero la medida se calcula para cada consulta y a continuación entre todas las consultas de la prueba.
2. Todas estas medidas están basadas en la posición, es decir, la posición rankeada es usada explícitamente. Las medidas basadas en la posición usualmente son no-continuas y no-diferenciables con respecto a las puntuaciones, esto hace que la optimización de estas medidas sea bastante difícil.

3.4 Aprender a Rankear

En esta sección se introduce el término de Machine Learning o Máquina de Aprendizaje, debido a la efectividad que tiene para automáticamente sintonizar los parámetros, combinar múltiples evidencias y evitar las sobre pruebas. Por lo tanto, parece bastante prometedor adoptar tecnologías de ML para resolver los problemas de ranking antes mencionados.

3.4.1 ML Framework

En gran parte de la investigación de ML (especialmente en el aprendizaje discriminativo), se ha prestado atención a los siguientes componentes claves para desarrollar el modelo de ranking:

1. Espacio de Entrada (Input Space), es el que contiene los objetos bajo investigación. Por lo general, los objetos están representados por vectores característicos, extraídos de las diferentes aplicaciones.

2. Espacio de Salida (Output Space), su objetivo es aprender en base a los objetos de entrada.
3. Espacio de Hipótesis (Hypothesis Space), define las funciones que operan sobre los vectores de los objetos de entrada, y así hacer predicciones de acuerdo con el espacio de salida.
4. Función de Pérdida (Loss Function), mide el grado de predicción generado por la hipótesis.

3.4.2 Aprendiendo a Rankear en Framework

En los últimos años, cada vez más se ocupan las tecnologías de ML, para entrenar el modelo de ranking, por lo ha emergido una nueva área de investigación denominada "Aprender a Rankear". El aprendizaje para rankear se ha convertido en una de las áreas de investigación más activas en IR.

Sin embargo, la mayoría del estado del arte en los algoritmos de aprender a rankear, se ha preocupado de encontrar la mejor manera de combinar las características extraídas, a partir de una consulta realizada a pares de documentos, a través de entrenamiento discriminatorio.

Propiedades de los métodos para aprender a rankear:

- Basado en la característica: todos los documentos objetos de investigación están representados por vectores característicos, los que reflejan la relevancia de los documentos frente a una consulta. La capacidad de combinar un gran número de características, es una ventaja muy importante en el aprendizaje para rankear.

- Entrenamiento discriminatorio: es un método de aprendizaje para rankear tiene su espacio de entrada específico, la producción del espacio de salida, el espacio de hipótesis y la función de pérdida. En la literatura ML, los métodos de discriminación han sido ampliamente utilizados para combinar diferentes tipos de características, sin la necesidad de definir un marco probabilístico para representar los objetos y la exactitud de la predicción. Entrenamiento discriminativo es un proceso automático de aprendizaje basado en los datos de entrenamiento. Esto es muy exigente para los motores de búsqueda, porque todos los días reciben una gran cantidad de comentarios de los usuarios y los logs nos indican lo pobre de los rankings para alguna de las consultas. Es muy importante aprender de la retroalimentación en forma automática y así constantemente mejorar el mecanismo de ranking.

3.4.3 Enfoques para Aprender a Rankear

Muchos algoritmos para aprender a rankear pueden encasillarse en Framework. Con el fin de comprenderlos mejor, se lleva a cabo una categorización de estos algoritmos. En particular, se agrupan los algoritmos de acuerdo a los cuatro componentes presentados en ML, en 3 enfoques: el enfoque Pointwise, el enfoque Pairwise y el enfoque Listwise [7].

Cada enfoque tiene su propia definición de espacios de entrada y salida, usa diferentes hipótesis y emplea distintas funciones de pérdida. El espacio de salida es usado para facilitar el proceso de aprendizaje, el cual puede ser diferente respecto a la relevancia de los juicios sobre los documentos, es decir, incluso si utilizamos el mismo formato de juicios, se pueden obtener distintas etiquetas *ground truth* para los diferentes enfoques.

A continuación se describen a manera general los 3 enfoques respecto a los componentes de ML:

- Enfoque Pointwise:
    - Espacio de entrada: contiene el vector con las características de cada documento en forma individual.
    - Espacio de salida: contiene el grado de relevancia único de cada documento.
    - Espacio de hipótesis: contiene funciones que toman las características del vector de un documento como entrada y predicen el grado de relevancia del documento.
    - Función de pérdida: examina la precisión de la etiqueta *ground truth* para cada documento.
- Enfoque Pairwise:
    - Espacio de entrada: contiene un par de documentos, ambos representados como vectores característicos.
    - Espacio de salida: contiene el grado de relevancia entre cada par de documentos.
    - Espacio de hipótesis: contiene funciones bi-variables que toman un par de documentos como entrada y entrega como salida el orden relativo entre ellos.
    - Función de pérdida: mide la inconsistencia relativa del orden entre un par de documentos.
- Enfoque Listwise:
    - Espacio de entrada: contiene todo el grupo de documentos relacionados con la consulta q.
    - Espacio de salida: existen 2 tipos, uno que contiene los grados de relevancia de todos los documentos relacionados con una consulta, y el otro que contiene la lista ordenada de los documentos.
    - Espacio de hipótesis: contiene funciones multi-variables que operan en un grupo de documentos, y predicen sus grados de relevancia.
    - Función de pérdida: mide la diferencia entre la lista ordenada dada por la hipótesis y la lista *ground truth*.

3.4.3.1 Enfoque Pointwise

Cuando usamos las tecnologías de ML para resolver problemas de ranking, probablemente la manera más sencilla es comprobar si los métodos de aprendizaje existentes pueden ser directamente aplicados, esto es exactamente lo que el enfoque pointwise hace. Al aplicarlo, uno asume que el grado de relevancia exacta de cada documento es lo que se está prediciendo, aunque esto podría no ser necesario cuando el objetivo es producir una lista rankeada de los documentos.

A continuación se explican los algoritmos más representativos en 3 subcategorías:

1. Algoritmos basados en Regresión: en esta subcategoría, el problema de ranking se reduce a un problema de regresión, al considerar el grado de relevancia como números reales. A continuación se presentan algunos algoritmos:

    - Función de Regresión Polinomial.
    - Subconjunto de Ranking con Regresión.

2. Algoritmos basados en Clasificación: estos algoritmos no consideran la etiqueta *ground truth* como un valor cuantitativo, esto es más razonable que los algoritmos basados en regresión. A continuación se presentan algunos algoritmos:

    - Modelo Discriminativo para IR.
    - Clasificación Multi-Clase para Ranking (McRank).

3. Algoritmos basados en Regresiones Ordinales: estos algoritmos toman la relación ordinal entre las etiquetas *ground truth* durante el aprendizaje del modelo de ranking. A continuación se presentan algunos algoritmos:

    - Ranking basado en Perceptron (PRanking).
    - Ranking con Principios de Margen Grande.

3.4.3.2 Enfoque Pairwise

El enfoque Pairwise, no se centra en predecir con exactitud el grado de relevancia de cada documento, sino que se preocupa por el orden relativo entre dos documentos. En este sentido, está más cerca del concepto de ranking, que el enfoque Pointwise. En este enfoque de dos a dos, el problema de ranking se reduce a un problema de clasificación sobre pares de documentos. Es decir, el objetivo del aprendizaje, es minimizar el número de pares de documentos clasificados-perdidos, por lo tanto, el fin es hacer predicciones positivas en esos pares, en el cual el primer documento es más relevante que el segundo documento, y hacer predicciones negativas sobre otros pares. En el caso extremo, si todos los pares de documentos están clasificados correctamente, todos los documentos serán correctamente rankeados. Tener en cuenta que los pares de documentos no son independientes, lo cual viola el principio básico de clasificación. En este caso, aunque todavía se pueden utilizar algoritmos de clasificación para aprender el modelo del ranking, un framework diferente es necesario para analizar la generalización del proceso de aprendizaje.

A continuación se nombran los algoritmos más representativos:

1. Algoritmos Ordenados con Función de Preferencia.
2. RankNet y Frank.
3. RankBoost.
4. Ranking SVM (Máquina de Soporte Vectorial)

3.4.3.3 Enfoque Listwise

El enfoque Listwise puede ser dividida en dos subcategorías. Para la primera subcategoría, el espacio de salida contiene los grados de relevancia de todos los documentos relacionados con una consulta, y la función de pérdida es definida en la aproximación o límites de las medidas de evaluación en IR. Para la segunda subcategoría, el espacio de salida contiene la permutación de los documentos asociados con la consulta y la función de pérdida mide la diferencia entre la permutación dada por la hipótesis y la permutación del *ground truth*.

A continuación se presentan las dos subcategorías y sus algoritmos representativos:

- Optimización Directa de las Medidas de Evaluación en IR [8]: permite que el modelo de ranking, aprenda directamente de la optimización de las medidas que evalúan el desempeño del ranking. Las medidas de evaluación en IR, como NDCG y MAP, son basadas en la posición, y son no-continuas y no-diferenciables. Para enfrentar este desafío, primero podemos optimizar una continua y diferenciable aproximación de una medida de evaluación en IR, y luego usar tecnologías de optimización que permiten optimizar objetivos complejos. A continuación se nombran algunos algoritmos:
    - SoftRank.
    - SVM^{map} [9].
    - AdaRank.
    - Algoritmos basados en Programación Genética.
- Minimización de las Pérdidas en Listwise de Ranking: en esta subcategoría, la función de pérdida mide la inconsistencia entre la salida del modelo de ranking y la permutación *ground truth* de todos los documentos. A continuación se nombran algunos algoritmos:
    - ListNet.
    - ListMLE.

La mayor diferencia entre los enfoques, está dada por la función de pérdida. Las funciones de pérdidas, son principalmente usadas para guiar el proceso de aprendizaje, mientras que la evaluación del modelo de ranking aprendido se basa en las medidas de evaluación de IR.

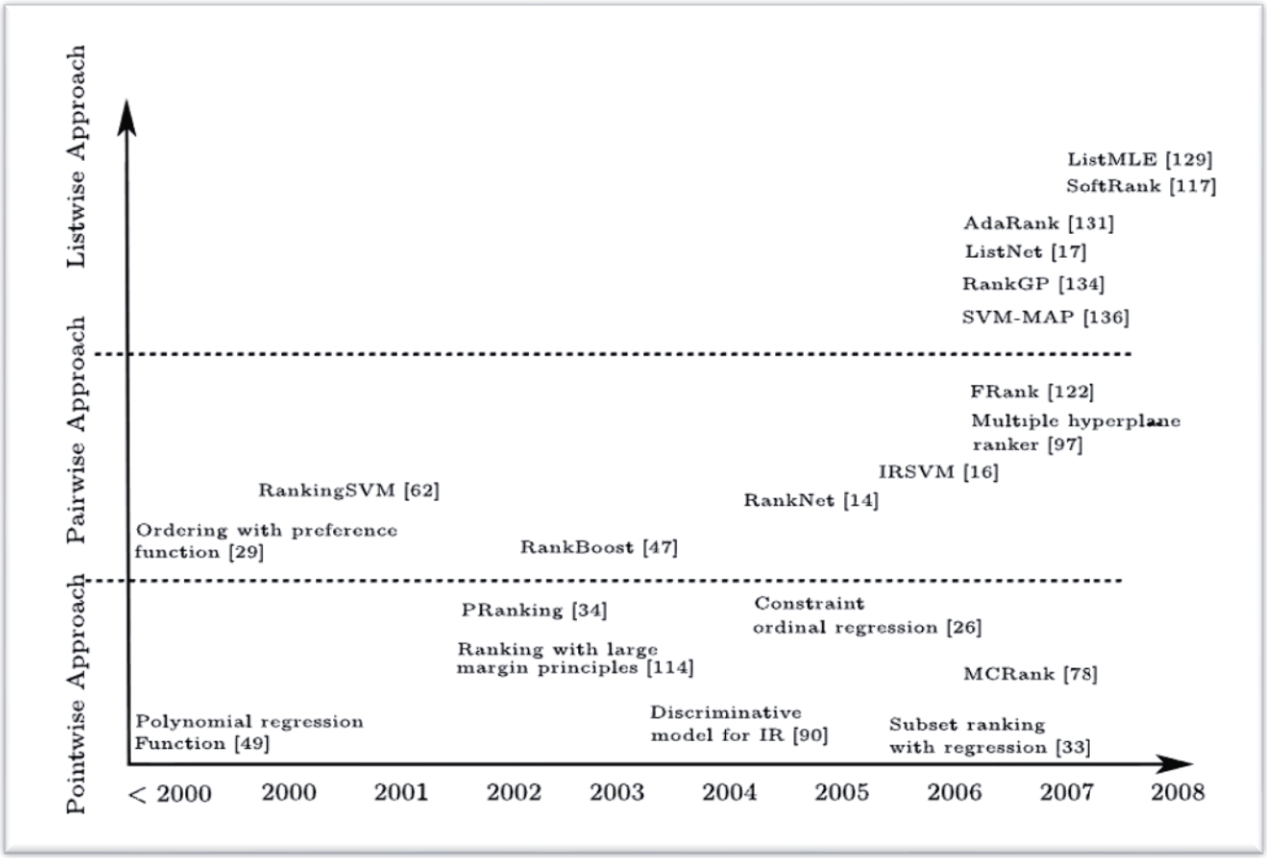


Figura 3.1 Algoritmos de Ranking.

4 Datos Multi-Etiquetados

4.1 Introducción

En clasificación multi-etiqueta, los documentos están asociados con un conjunto de etiquetas, es decir, cada ítem de un dataset multi-etiquetado puede ser miembro de múltiples categoría. Los métodos de clasificación multi-etiqueta son cada vez más requeridos por aplicaciones modernas, como por ejemplo en Medicina, las proteínas tienen muchas características y etiquetas. Actualmente esta es la naturaleza de muchos problemas reales en el mundo como la semántica en imágenes y videos, categorización de páginas web, marketing y categorización de música dentro de géneros o emociones entre otros.

4.2 Medición de Datos Multi-Etiquetados

Tal como en los datos de simple-etiquetado, en multi-etiquetado los datos pueden ser medidos por el número de ejemplos (N), el número de atributos en el espacio de entrada (M) y el número de etiquetas (L). A continuación se presentan medidas específicas para medir los datos multi-etiqueta.

- Label Cardinality (LCard): es una medida estándar en multi-etiquetado, introducida en (Tsoumakas and Katakis, 2007). Entrega el promedio simple del número de etiquetas asociadas con cada ejemplo.

$$\text{LCard} = \frac{\sum_{i=1}^{N} |y_i|}{N} \tag{4.2.1}$$

- Label Density (LDens): también introducida en (Tsoumakas and Katakis, 2007), relaciona a LCard, pero toma en cuenta el tamaño del espacio de etiquetas.

$$\text{LDens (D)} = \frac{1}{L} \text{LCard(D)} \tag{4.2.2}$$

Estas medidas entregan una buena idea de la frecuencia de etiquetas, pero no indica la regularidad o uniformidad del etiquetado.

4.3 Datasets y Aplicaciones

A continuación se muestran los datasets utilizados para ésta investigación y una reseña de cada uno. Los datasets fueron ordenados en base a su complejidad (N x L x M), siendo N el número de documentos, L el número de etiquetas y M el número de atributos.

Tabla 4.1 Datasets Multi-Etiqueta (n atributos numéricos) de [10].

Nombre	Dominio	Documentos (L)	Etiquetas (N)	Atributos (M)	Lcard	Ldens	Complejidad
Bibtex	texto	7395	159	1836	2,380	0,015	2.158.777.980
Corel5k	imagen	5000	374	499	3,522	0,009	933.130.000
Corel16k	imagen	6932	153	500	3,085	0,020	530.298.000
Enron	texto	1702	53	1001	3,386	0,064	90.296.206
Medical	texto	978	45	1449	1,255	0,028	63.770.490
Reuters	texto	9190	9	686n	1,056	0,117	56.739.060
Genbase	biología	662	27	1186	1,261	0,047	21.198.564
Scene	imagen	2407	6	294n	1,062	0,177	4.245.948

Los datasets utilizados provenían de distintas fuentes, como Scene (Boutell et al., 2004) es relativamente pequeño pero un dataset ampliamente usado en la clasificación de escenas, envolviendo los seis posibles escenarios como etiquetas: “beach”, “sunset”, “field”, “fall-foliage”, “mountain” y “urban”.

Genbase (Diplaris et al., 2005) es un dataset de microbiología cuyo interés radica en la función de genes, cada gen puede ser asociado a múltiples funciones. En éste dataset hay 27 etiquetas en el espacio de etiqueta.

Medical (Pestian et al., 2007) es un dataset de textos médicos recopilados por los Centros de Medicina Computacional 2007, para el desafío de procesamiento del Lenguaje Médico Natural.

Enron es un dataset compuesto por los cuerpos de e-mails, desarrollado por UC Berkeley Enron con su Proyecto de análisis de e-mail. Sus principales categorías son Coarse genre, Included/forwarded information, Primary topics y Emotional tone.

Reuters First 9, proviene del cuerpo moderno de Reuters RCV1 (Lewis et al., 2004), incluyendo la jerarquía de tópicos. Se redujo el espacio de atributos de cerca de 46.000 a 686, mediante el empleo de un filtro de selección de atributos de cada etiqueta, basado en ganancia de información, y luego tomando los 686 primeros atributos extraídos de las etiquetas, este tipo de selección se explica en (Tsoumakas and Vlahavas, 2007).

Corel5k y Corel6k son datasets de imágenes usado en el proyecto “Object Recognition as Machine Translation” por Pinar Duygulu, Kobus Barnard, Nando do Freitas y David Forsyth. Cada segmento de imagen es representado por 36 características. Entonces cada imagen tiene un diferente número de segmentos siendo éstos las etiquetas.

4.4 Distribución de Etiquetas

La distribución de etiquetas se refiere a la distribución de las frecuencias en la cual se definen las etiquetas dentro de los datos. Esto se caracteriza por las medidas descritas en la Sección 4.2.

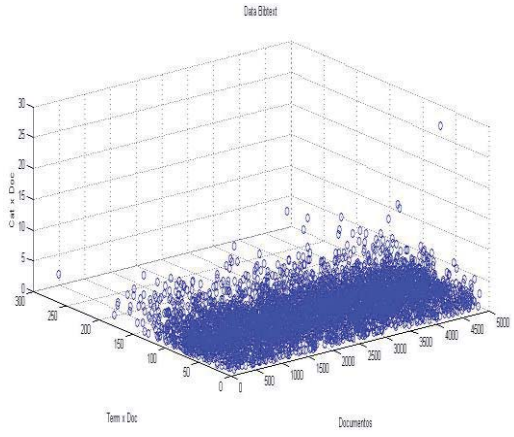
LCard varía considerablemente entre los conjuntos de datos, cercano a 1.0 (ej. Scene) donde la mayoría de ejemplos están asociados solamente con una etiqueta, y en otro casos hasta 4.0. LDens es usualmente muy bajo, el etiquetado suele ser poco denso. Baja cardinalidad y densidad de etiquetado es común en datasets de textos, donde la mayoría de ejemplos se ajusta naturalmente bajo un esquema de una etiqueta y el multi-etiquetado fue introducido para resolver las ambigüedades. Considerar el caso del dataset Scene, clasificación de imágenes, donde la mayoría de las imágenes son relevantes solo para una etiqueta, debido a que sus tipos de etiquetado son: "mountain", "field" o "sunset". Por ejemplo cuando una imagen es relevante para ambas etiquetas como "mountain" y "field" se utiliza múltiples etiquetas para resolver ambigüedades ocasionales.

Alta cardinalidad de etiquetas es a menudo observado en datasets con un amplio dominio. Ejemplos de esto, son los datasets biológicos, como Genbase donde se espera que los genes tengan múltiples funciones y en datasets como Enron sobre emails, donde las categorías o etiquetas toman la forma de un checklist.

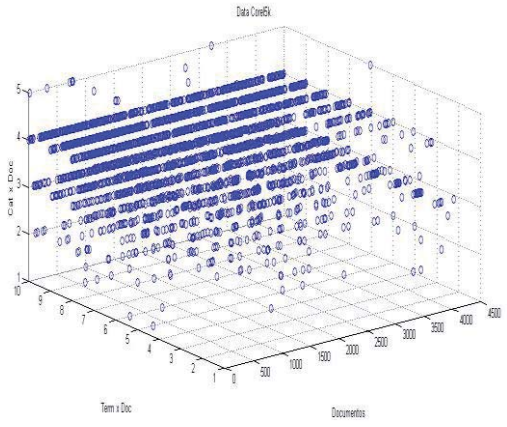
4.5 Características de los Datasets

A continuación se presenta una serie de gráficos que muestran el comportamiento, distribución, promedio y desviación estándar de los datasets utilizados para la investigación.

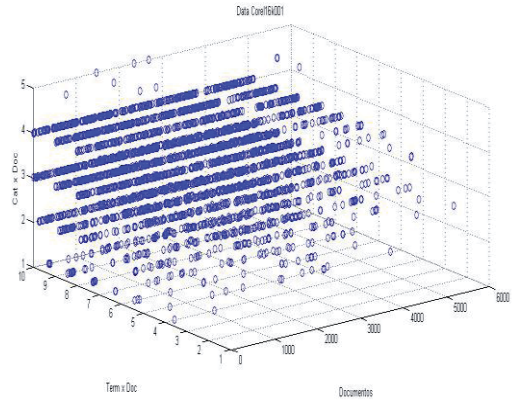
Primero, se muestran gráficos en 3D en los cuales se presenta en el Eje X los "Documentos", en el Eje Y "Términos por Documentos", es decir, el número de términos que posee cada documento y finalmente, en el Eje Z "Categorías o Etiquetas por Documentos", número de etiquetas que posee cada documento.



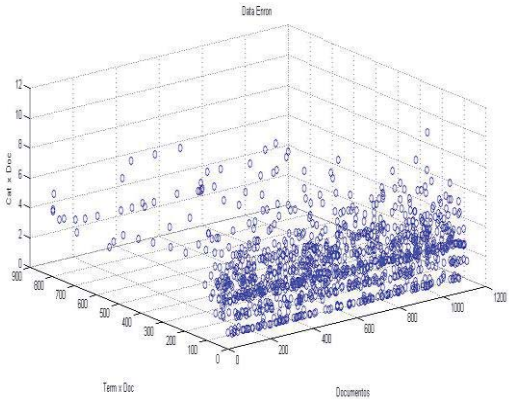
(1) Dataset Bibtext



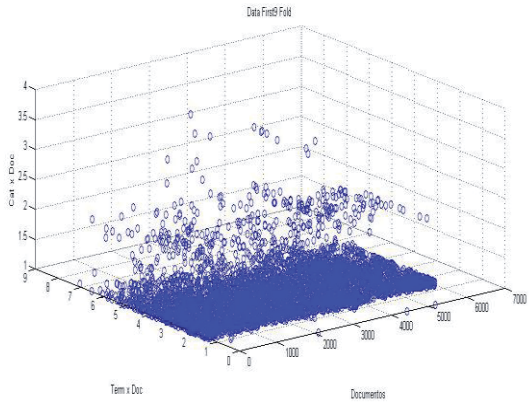
(2) Dataset Corel5k



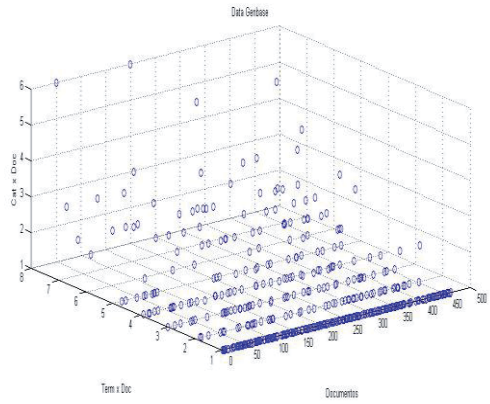
(3) Dataset Corel16k



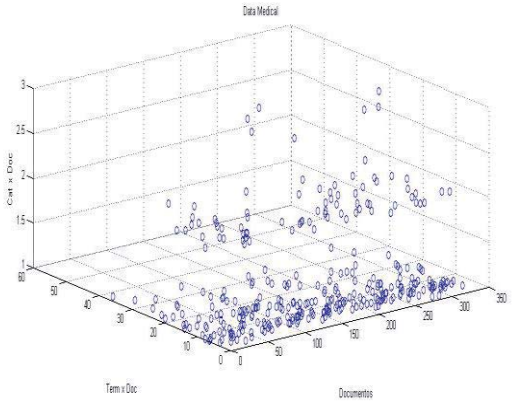
(4) Dataset Enron



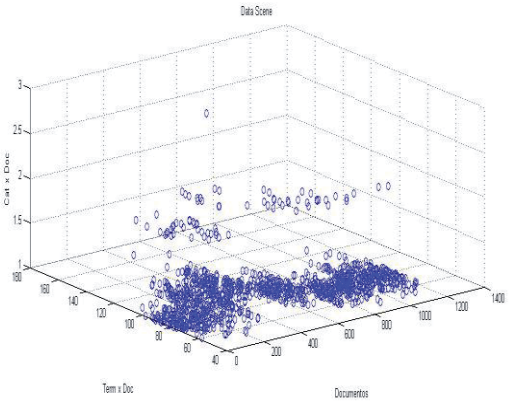
(5) Dataset First9



(6) Dataset Genbase



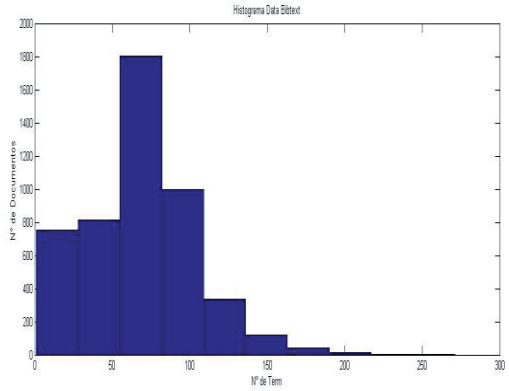
(7) Dataset Medical



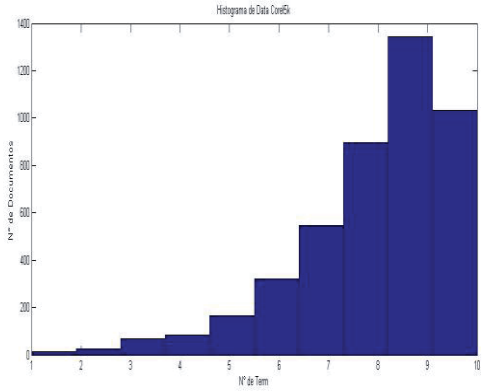
(8) Dataset Scene

Figura 4.1 Datasets en 3D.

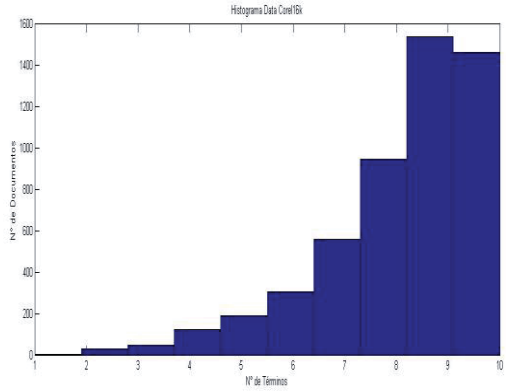
Segundo, se presentan los histogramas de los datasets utilizados, los histogramas muestran la frecuencia de los valores representados en barras. En el Eje X, se muestran el N° de Términos y en el Eje Y, se muestran la frecuencia de Documentos asociados a los Términos.



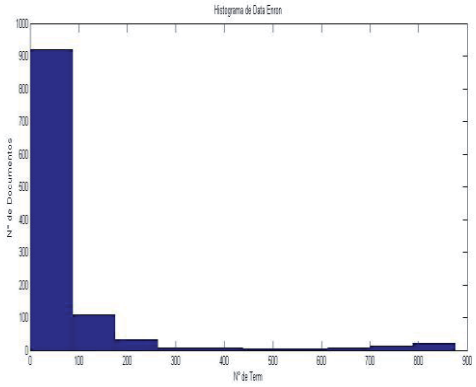
(1) Dataset Bibtext



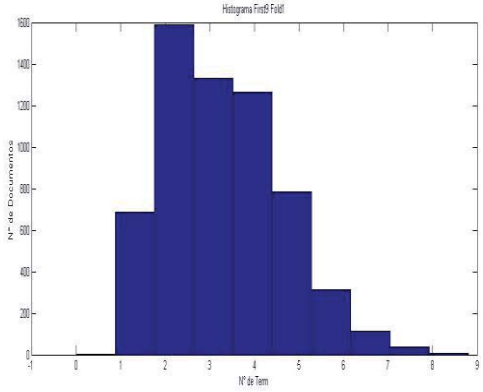
(2) Dataset Corel5k



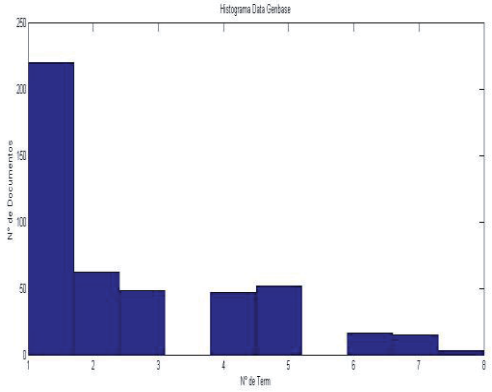
(3) Dataset Corel16k



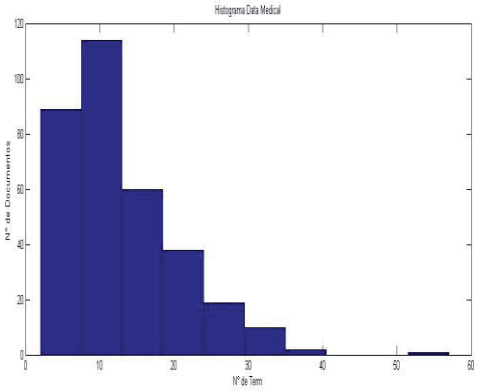
(4) Dataset Enron



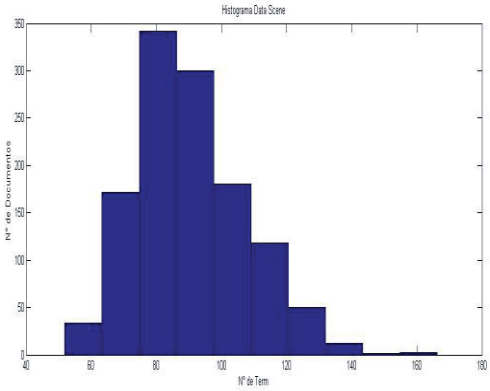
(5) Dataset First9



(6) Dataset Genbase



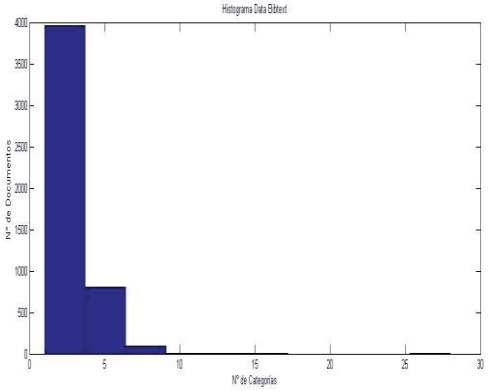
(7) Dataset Medical



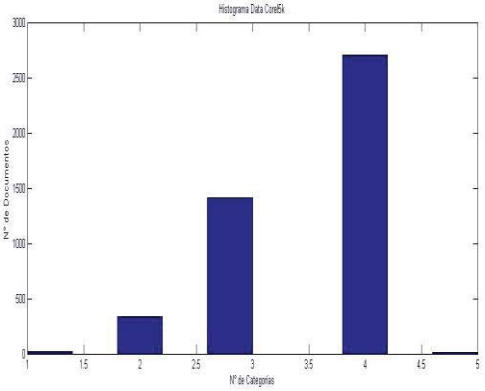
(8) Dataset Scene

Figura 4.2 Histograma Term x Doc

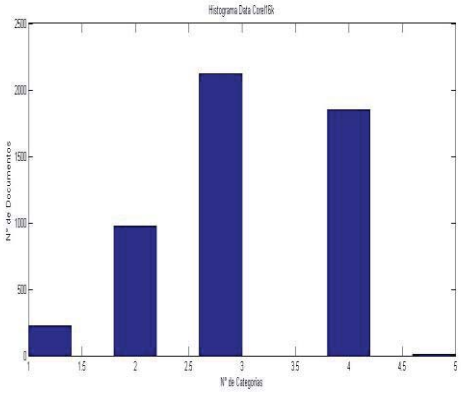
Tercero, se presentan los histogramas de los datasets utilizados, respecto a la relación Categoría y Documentos. En el Eje X, se muestran el N° de Categorías y en el Eje Y, se muestran la frecuencia de Documentos asociados a las Categorías.



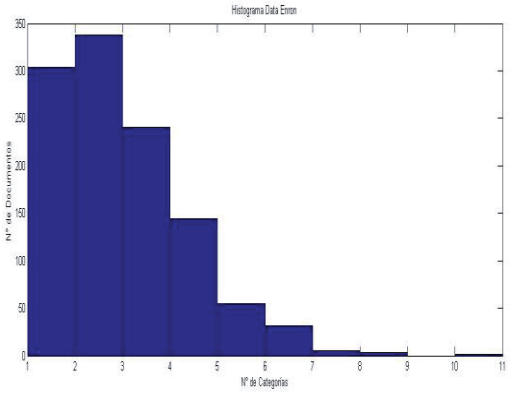
(1) Dataset Bibtext



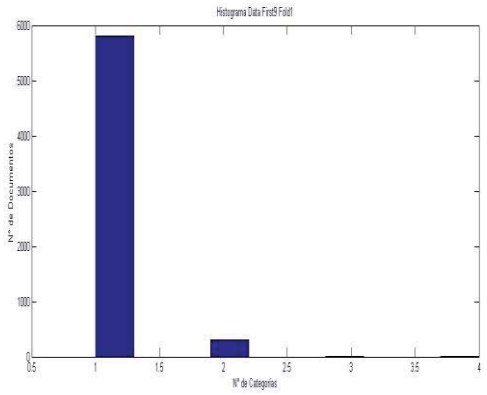
(2) Dataset Corel5k



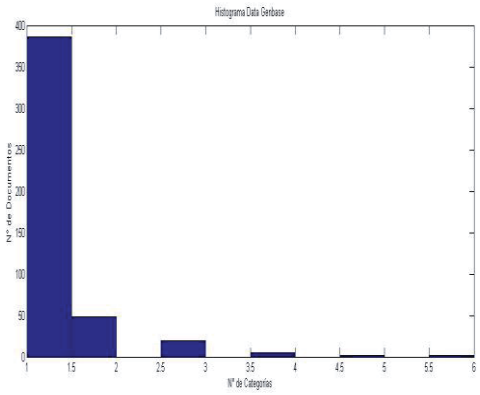
(3) Dataset Corel16k



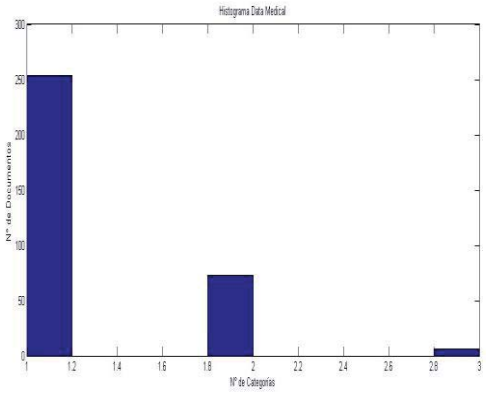
(4) Dataset Enron



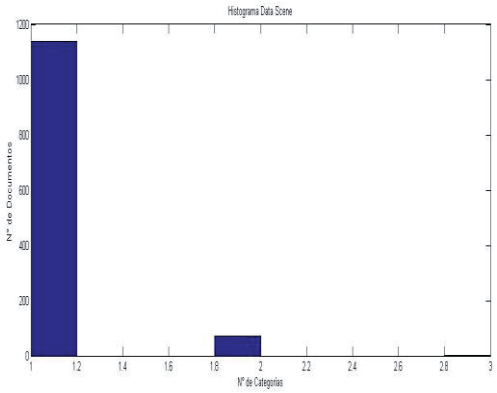
(5) Dataset First9



(6) Dataset Genbase



(7) Dataset Medical



(8) Dataset Scene

Figura 4.3 Histograma Cat x Doc.

Por último, se incluye una tabla con los valores de cada dataset, basado en el promedio y en su desviación estándar, primero referente con Términos y Documentos, luego Categorías y Documentos, datos que serán analizados en la sección de análisis de resultados.

Tabla 4.2 Información de los Datasets referente a Documentos y Términos.

DOC x TERM	**Bibtex**	**Corel5k**	**Corel16k**	**Enron**	**First9**	**Genbase**	**Medical**	**Scene**
Promedio	68,49	8,18	8,35	84,18	3,29	2,51	13,24	90,24
Desviación Estándar	35,27	1,72	1,67	152,02	1,31	1,83	7,75	16,77

Tabla 4.3 Información de los Datasets referente a Documentos y Categorías.

DOC x CAT	**Bibtex**	**Corel5k**	**Corel16k**	**Enron**	**First9**	**Genbase**	**Medical**	**Scene**
Promedio	2,38	3,52	3,09	3,39	1,06	1,26	1,26	1,03
Desviación Estándar	1,65	0,66	0,85	1,54	0,24	0,70	0,48	0,24

5 Medidas de Evaluación para Multi-Etiqueta

A continuación se presentan las medidas de evaluación que se utilizan en datasets multi-etiquetados, y que van a permitir medir el rendimiento del modelo desarrollado.

5.1 Introducción

En el contexto de etiquetado simple, el rendimiento predictivo es fácilmente manejado bajo una medida tradicional de precisión, donde cada ejemplo de prueba puede ser correcto o incorrecto. En el espacio de multi-etiqueta, el rendimiento predictivo puede ser medido de dos formas, basado en la evaluación de un conjunto de etiquetas, cada conjunto es evaluado separadamente o basado en la evaluación de relevancia binaria de cada etiqueta en forma individual.

5.2 Medidas de Evaluación para Multi-Etiqueta

Primero se debe considerar las medidas básicas de evaluación recién mencionadas, conjunto de etiquetas basado en precisión, la cual se refiere a "exact-match" (Ecuación 5.2.1) y etiqueta basada en precisión, medida como "Hamming Loss" (Ecuación 5.2.2). Estas medidas se representan de la siguiente forma:

$$Exact - Match\,(D) = \frac{1}{N}\sum_{i=1}^{N} 1_{\hat{y}_i = y_i} \tag{5.2.1}$$

$$Hamming - Loss(D) = \frac{1}{NL}\sum_{i=1}^{N} |\hat{y}_i \Delta y_i| \tag{5.2.2}$$

Una medida de multi-etiqueta de "Accuracy" fue introducida en (Godbole and Sarawagi, 2004) (Ecuación 5.2.3). Ésta medida toma la proporción del tamaño de la unión e intersección de la predicción y del conjunto de etiquetas actuales (representado por el operador lógico AND y OR operando en notación de bit), tomando de cada ejemplo un promedio sobre el número de ejemplos.

$$Accuracy(D) = \frac{1}{N}\sum_{i=1}^{N} \frac{|y_i \cap \hat{y}_i|}{|y_i \cup \hat{y}_i|} \tag{5.2.3}$$

La medida F-measure, comúnmente usada en recuperación de la información, es muy popular en clasificación multi-etiqueta (Tsoumakas and Katakis, 2007; Spy-romitros et al., 2008). Para cualquier vector asociado con $z \in \{0,1\}^T$, una etiqueta es relevante si $z_j = 1$ y la predicción $\hat{z}_j = 1$ (en el correspondiente vector) y desde esto se puede definir:

- Precision: es la fracción de las predicciones relevantes las cuales son actualmente relevantes.

$$Precision = \frac{(|z \cap \hat{z}|)}{|\hat{z}|} \tag{5.2.4}$$

- Recall: es la fracción de las relevancias actuales respecto a las predicciones.

$$Recall = \frac{(|z \cap \hat{z}|)}{|z|} \tag{5.2.5}$$

- F-measure (F1) es calculada como:

$$\mathrm{F1}(\hat{z}, \mathrm{z}) = \frac{2 * precision * recall}{precision + recall} \tag{5.2.6}$$

- F1-Micro:

$$\text{F1-Micro(D)} = \mathrm{F1}(\hat{z}, \mathrm{z}) \tag{5.2.7}$$

- F1-Macro x L:

$$\mathrm{F1} - \mathrm{Macro}^{xL}(D) = \frac{1}{L}\sum_{j=0}^{L} \mathrm{F1}(\hat{z}, \mathrm{z}) \tag{5.2.8}$$

- F1-Macro x N:

$$\mathrm{F1} - \mathrm{Macro}^{xN}(D) = \frac{1}{N}\sum_{j=0}^{N} \mathrm{F1}(\hat{z}, \mathrm{z}) \tag{5.2.9}$$

Finalmente, se presenta un ejemplo para entender cómo se calculan estas medidas de evaluación para datasets multi-etiquetas (Tabla 5.1 y 5.2).

Tabla 5.1 Instancias de Prueba.

$y_1 = (1,0,1,0)$ $\hat{y}_1 = (1,0,0,0)$	$y_2 = (1,0,1,1)$ $\hat{y}_2 = (1,0,1,1)$	$y_3 = (1,1,1,0)$ $\hat{y}_3 = (1,0,0,1)$

Tabla 5.2 Ejemplo de medidas de evaluación Multi-Etiquetas.

Exact-Match	$\frac{1}{3}1 = 0{,}333$
Hamming-Loss	$\frac{4}{12} = 0{,}333$
Accuracy	$\frac{1}{3}\left(\frac{1}{2}+\frac{3}{3}+\frac{1}{4}\right) = 0{,}583$

F1-Micro	$= 0{,}714$
F1-Macro x L	$\frac{1}{3}\,(0{,}67 + 1 + 0{,}4) = 0{,}689$
F1-Macro x N	$\frac{1}{4}\,(1 + 0{,}5 + 0{,}67 + 0) = 0{,}542$

6 Metodologías

A continuación, se presenta el modelo del prototipo a desarrollar.

6.1 Modelo del Prototipo

El modelo de ranking seleccionado es el Modelo de Independencia Booleano, el cual está basado en la ocurrencia de términos que aparecen en los documentos, éste modelo predice si un documento es relevante o no a una consulta. El proceso de clasificación incluye dos fases: entrenamiento y prueba. En la fase de entrenamiento o *training* un conjunto de datos inicial es usado para decidir que parámetros deberán ser ponderados y combinados con el objetivo de separar las clases, de esta manera construir un clasificador para la fase siguiente. El aprendizaje intenta descubrir una representación óptima a partir del conjunto de datos cuya etiqueta de clase es conocida por el investigador. En la fase de prueba o *testing*, el clasificador determinado en la fase de entrenamiento es aplicado a un conjunto de datos u objetos (conjunto de prueba) cuyas etiquetas de clase se desconoce. De esta forma clasificar los elementos y comparar con las etiquetas reales para determinar la efectividad del modelo. Por ejemplo, un conjunto típico de entrenamiento consiste en entrenar n consultas, y que éstas tengan asociados documentos representados por sus vectores característicos y a la vez con los correspondientes juicios de relevancia. Entonces un algoritmo específico de aprendizaje es empleado para aprender el modelo del ranking, y así la salida del modelo de ranking pueda predecir la etiqueta *ground truth* de los datos de entrenamiento, tan precisamente como sea posible, en términos de la función de pérdida. En la fase de pruebas, cuando las nuevas consultas entran, el modelo aprendido en la fase de entrenamiento debe ser capaz de ordenar los documentos de acuerdo a sus grados de relevancia respecto a la consulta, y retornar una lista rankeada de los documentos.

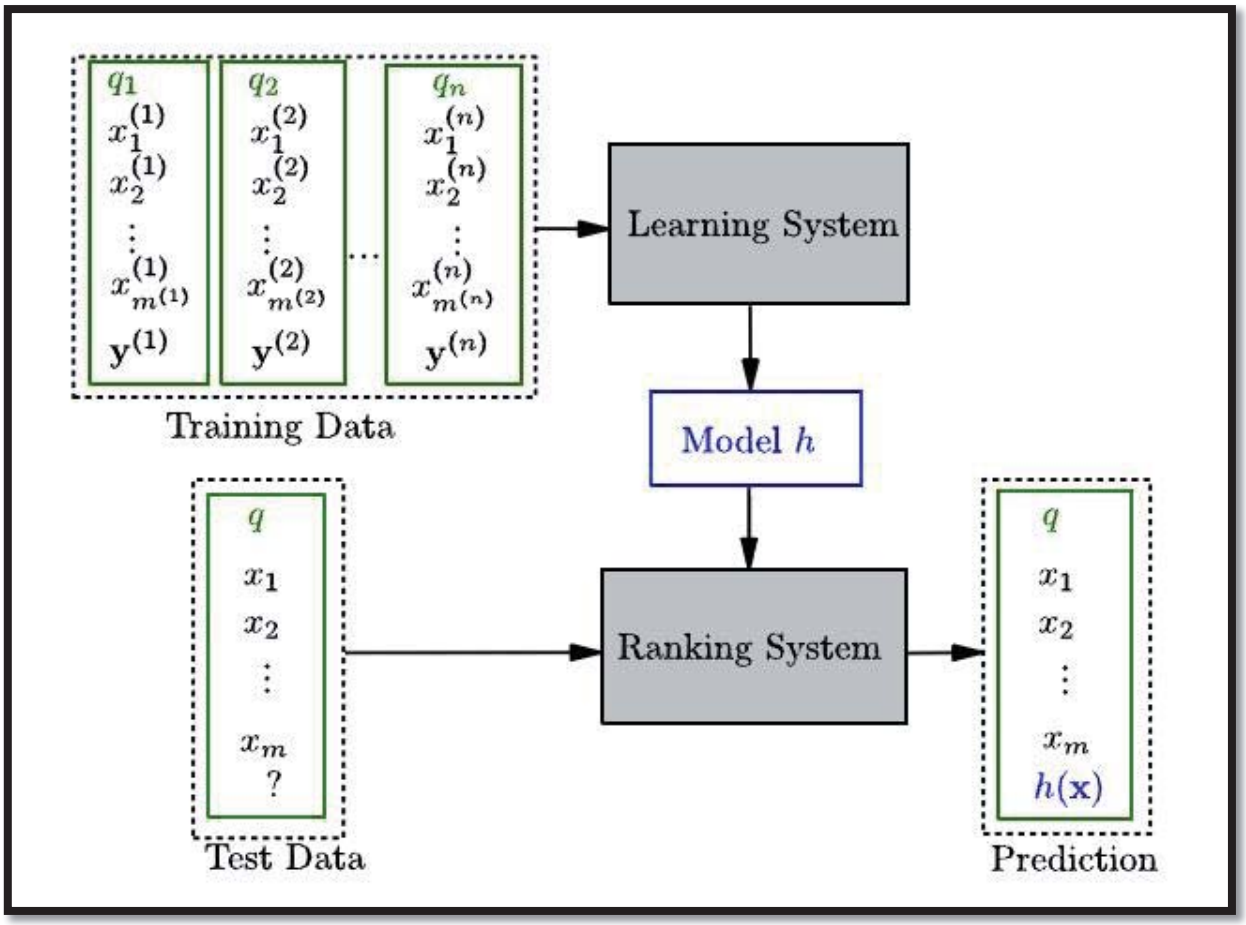


Figura 6.1 Modelo del Prototipo

6.1.1 Modelo de Independencia Booleano

A continuación se presenta el algoritmo seleccionado que se implementará en el prototipo. El primer paso es que el modelo estime la probabilidad de que un documento es relevante para una consulta, éste modelo está basado en lógica booleana y teoría clásica de conjuntos (las bases se explican en el Cap. 2.5.1), en la cual tanto los documentos y las consultas son consideradas como conjunto de términos, y la recuperación se basa en si los documentos contienen los términos de consulta.

Se deben cambiar algunas cosas del modelo base presentado en el marco teórico, como redefinir la variable aleatoria D, que representa los documentos, a un vector binario de variables aleatorias D = $\{D_1, D_2, \dots D_n\}$ con una dimensión para término en el vocabulario V y con $D_i = 1$ indicando que el término está presente y $D_i = 0$ indicando que el término está ausente. Similarmente, se redefine la variable aleatoria Q, que representa las consultas, a un vector binario de variables aleatorias Q = $\{Q_1, Q_2, \dots Q_n\}$ con $Q_i = 1$ término presente y $Q_i = 0$ término ausente. Se realizan dos grandes supuestos, el primero es un supuesto de independencia:

- Supuesto T: Dada la relevancia, los términos son estadísticamente independientes.

En otras palabras, dado un juicio de relevancia positivo, la presencia o ausencia de uno de los términos no depende de la presencia o ausencia de cualquier otro término. Similarmente, dado un juicio de relevancia negativo, la presencia o ausencia del término depende de los demás términos. Naturalmente, si no se condiciona la relevancia, los términos no son independientes porque hay usualmente múltiples términos para los cuales la probabilidad de presencia depende de su relevancia.

Este supuesto no refleja exactamente la realidad, por ejemplo, la presencia del término "shakespeare" en un documento incrementa la probabilidad que el término "william", "hamlet" y "stratford" pudieran aparecer en el documento. Sin embargo, el supuesto simplifica enormemente la estimación del valor de la Ecuación 2.5.13. La independencia de supuestos como éstos son comunes en la recuperación de información. Métodos desarrollados bajo estos supuestos a menudo proveen buenos desempeños a pesar de su naturaleza poco realista.

Como resultado de éste supuesto, se deben reescribir las probabilidades que aparecen en la Ecuación 2.5.13 como el producto de las probabilidades para las variables aleatorias individuales asociadas con las dimensiones de D:

$$P(D \mid Q, R) = \prod_{i=1}^{|V|} p(D_i \mid Q, R) \tag{6.1.1}$$

$$P(D \mid Q, \neg R) = \prod_{i=1}^{|V|} p(D_i \mid Q, \neg R) \tag{6.1.2}$$

La ecuación 2.5.13 se convierte en:

$$\log \frac{p(D \mid Q,R)}{p(D \mid Q,\neg R)} = \sum_{i=1}^{|V|} \log \frac{p(D_i \mid Q,R)}{p(D_i \mid Q,\neg R)} \tag{6.1.3}$$

El segundo supuesto fuerte conecta la aparición del término en la consulta a la probabilidad de aparición en un documento relevante, así clarificar el rol de la consulta como un puente entre la necesidad del usuario y la relevancia del documento.

Supuesto Q: La presencia de un término en un documento depende de la relevancia sólo cuando dicho término está presente en la consulta.

Para formalizar este supuesto se necesita arreglar la consulta, siendo Q = q = $\{q_1, q_2, \ldots q_n\}$, donde queda q_i, es 0 o 1. Como el resultado del supuesto Q, si $q_i = 0$ entonces:

$$P(D_i \mid Q, R) = P(D_i \mid Q, \neg R) \tag{6.1.4}$$

Y por lo tanto:

$$\log \frac{p(D_i \mid Q,R)}{p\,(D_i \mid Q,\neg R)} = 0 \tag{6.1.5}$$

El efecto de este supuesto es cambiar la suma en la Ecuación 6.1.3 desde una suma sobre todos los términos en el vocabulario a una suma de todos los términos en la consulta. Por lo que la fórmula de ranking se convierte en

$$\sum_{t \in q} \log \frac{p(D_t \mid R)}{p(D_t \mid \neg R)} \tag{6.1.6}$$

donde D_t es la variable aleatoria correspondiente al término t en el vector $\{d_1, d_{2,} \ldots\}$.

Como el supuesto T, el supuesto Q no refleja exactamente la realidad. Por ejemplo, si se busca sobre Pablo Neruda y se ingresa la siguiente consulta:

⟨"pablo","neruda","chileno" ⟩,

un documento relevante es más probable que contenga el término "poeta" que uno no relevante, incluso aunque éste término no aparezca en la consulta. Sin embargo, desde un punto de vista práctico una suma sobre términos de consulta parece más fácil de manejar que una suma sobre todos los términos en el vocabulario.

Luego, se debe arreglar D = d = $\{d_1, d_{2,} \ldots\}$ donde cada d_i es 0 o 1, extendiendo la notación se escribe $D_t = d_t$ para representar el valor de la variable aleatoria de $\{d_1, d_{2,} \ldots\}$ correspondiente al término t:

$$\sum_{t \in q} \log \frac{p(D_t = d_t \mid R)}{p(D_t = d_t \mid \neg R)} \tag{6.1.7}$$

Se debe ahora sustraer de la Ecuación 6.1.6 su propio valor cuando no aparezcan términos de la consulta en el documento y todos losD_t son 0;

$$\sum_{t \in q} \log \frac{p(D_t = d_t \mid R)}{p(D_t = d_t \mid \neg R)} - \sum_{t \in q} \log \frac{p(D_t = 0 \mid R)}{p(D_t = 0 \mid \neg R)} \tag{6.1.8}$$

Debido a que el valor de la Ecuación 6.1.6 cuando todos los términos de la consulta están ausentes pasan a ser una constante, por lo tanto la resta de un valor constante no tiene impacto en el ranking. Un reordenamiento de lo explicado queda como:

$$\sum_{t \in (q \cap d)} \log \frac{p(D_t = 1 \mid R)\, p(D_t = 0 \mid \neg R)}{p(D_t = 1 \mid \neg R)\, p(D_t = 0 \mid R)} - \sum_{t \in (q \setminus d)} \log \frac{p(D_t = 0 \mid R)\, p(D_t = 0 \mid \neg R)}{p(D_t = 0 \mid \neg R)\, p(D_t = 0 \mid R)} \tag{6.1.9}$$

Donde la suma de la izquierda es sobre todos los términos que aparecen en la consulta tanto como en los documentos y la suma de la derecha es sobre los términos que aparecen en la consulta pero no en el documento, por lo que el término de la derecha es 0 y la ecuación resultante es:

$$\sum_{t\in(q\cap d)} \log \frac{p(D_t=1\mid R)\, p(D_t=0\mid \neg R)}{p(D_t=1\mid \neg R)\, p(D_t=0\mid R)} \quad (6.1.10)$$

Este refinamiento de la Ecuación 2.5.13 bajo los supuestos de T y Q y a la vez considerando sólo la presencia y ausencia de términos es conocido como el "Modelo de Independencia Booleano".

A continuación se presenta un ejemplo para una clasificación Simple-Etiqueta tenemos:

Variables:

D: variable aleatoria que representa a un documento.

Q: variable aleatoria que representa a una consulta.

R: variable aleatoria que toma los valores 1 si el Documento *d* es relevante con respecto a la consulta *q* y 0 en caso contrario.

Entonces:

$$d \text{ es relevante para } q \text{ ssi } P(R=1\mid D=d, Q=q) > P(R=0\mid D=d, Q=q) \quad (6.1.11)$$

Esto quiere decir que un documento d es relevante para una consulta q, si y sólo si la probabilidad de que el documento sea relevante frente a una consulta debe ser mayor que la probabilidad de que el documento no sea relevante frente a una consulta.

Supuesto T: La aparición de términos en un documento es estadísticamente independiente [Naive-Bayes] [11].

$$D=\{d_{1,\ldots,} d_t\}, \qquad Q=\{q_{1,\ldots,} q_t\} \quad (6.1.12)$$

Supuesto Q: La relevancia de una consulta Q depende de la aparición de sus términos en un documento.

$$\text{Rank} = \sum_{t\in(q\cap d)} \log(P(d_t=1|R)P(d_t=0|R^c)/P(d_t=1|R^c)P(d_t=0|R)) \quad (6.1.13)$$

$$Rank = \sum_{t\in(q\cap d)} \log(P_t\,(1-P_t)\,/\mu_t\,(1-\mu_t)) \quad (6.1.14)$$

En donde:

Tabla 6.1 Cálculo de término relevante y no relevante.

Documento	**Relevante (R=1)**	**No-relevante (R=0)**
Término Presente (d_t=1)	$P_t = P(d_t=1\|R=1,q)$	$1-\mu_t = P(d_t=1\|R=0,q)$
Término Ausente (d_t=0)	$1- P_t = P(d_t=0\|R=1,q)$	$\mu_t = P(d_t=0\|R=0,q)$

Lo anterior referido al caso Simple-Etiqueta, llevándolo al caso Multi-Etiqueta tendríamos lo siguiente:

Variables:

D: variable aleatoria que representa a un documento $\{d_1,\ldots,d_t\}$.

Q: variable aleatoria que representa a una consulta.

R_L: variable aleatoria que toma los valores 1 si el Documento *d* es clasificado con la etiqueta *L* y 0 en caso contrario.

Entonces:

$$q \text{ es relevante para } L \text{ ssi } P(R_L=1|\, D, Q) > P(R_L=0|\, D, Q) \qquad (6.1.15)$$

En 6.1.15 indica si q es relevante para la etiqueta L si y sólo si la probabilidad del documento es relevante para la etiqueta L en base a una consulta, es mayor que la probabilidad de un documento no relevante para la etiqueta L en base a dicha consulta.

En donde la estimación sería:

$$Rank = \sum_{t\in(q\cap d)} \log(P_t\,(1-P_t)\,/\mu_t\,(1-\mu_t)) \qquad (6.1.16)$$

Donde la evaluación para cada etiqueta *L* sería:

Tabla 6.2 Cálculo de término relevante y no relevante en multi-etiqueta.

Documento	**Relevante (R_L =1)**	**No relevante (R_L =0)**
Term Presente (d_t=1)	$P_t = P(d_t=1|\, R_L=1,q)$	$1-\mu_t = P(d_t=1|\, R_L=0,q)$
Term Ausente (d_t=0)	$1- P_t =(d_t=0|R_L=1,q)$	$\mu_t= P(d_t=0|\, R_L=0,q)$

Cómo estimar μ_t y P_t a partir de los datos considerando múltiples etiquetas, consideremos por ejemplo 3 etiquetas:

Tabla 6.3 Estimación de μt y Pt en Multietiqueta.

Documento	**Relevante ($R_{L=1}$)**	**Relevante ($R_{L=2}$)**	**Relevante ($R_{L=3}$)**	**Total**
Term. Presente (d_t=1)	$s_{L=1}$	$s_{L=2}$	$s_{L=3}$	df_t
Term. Ausente (d_t=0)	$S_{L=1}-s_{L=1}$	$S_{L=2}-s_{L=2}$	$S_{L=3}-s_{L=3}$	$N-df_t$
Total	$S_{L=1}$	$S_{L=2}$	$S_{L=3}$	N

$$P_{t,L=1} = f_1(S_{t,L=1}, s_{t,L=1}) \quad (6.1.17)$$

$$P_{t,L=1} = \frac{s_{t,L=1}}{S_{t,L=1}}$$

$$\mu_{t,L=1} = f_2(S_{t,L=2} - s_{t,L=2}, S_{t,L=3} - s_{t,L=3}, |L|) \quad (6.1.18)$$

$$\mu_{t,L=1} = f(S_{t,L\neq 1,}|L|)$$

$$\mu_{t,L=1} = \frac{1}{|L-1|}\sum(S_t, L \neq 1 - s_t, L \neq 1 / S_t, L \neq 1)$$

Con las Ecuaciones 6.1.17 y 6.1.18, podemos realizar la estimación señalada en la Ecuación 6.1.16, y así poder clasificar un documento en una etiqueta.

A continuación, se presenta un ejemplo para entender el funcionamiento del modelo Booleano, se tienen 5 documentos clasificados en 3 categorías.

Tabla 6.4 Categoría x Documentos.

	Doc 1	**Doc 2**	**Doc 3**	**Doc 4**	**Doc 5**
Categoría 1	1	0	1	0	1
Categoría 2	0	1	0	1	0
Categoría 3	0	1	1	0	1

Cada documento esta compuesto por los siguientes términos:

Tabla 6.5 Documento x Términos.

	T1	**T2**	**T3**	**T4**	**T5**	**T6**	**T7**	**T8**	**T9**	**T10**
Doc 1	0	1	0	1	0	1	0	1	0	1
Doc 2	1	0	0	0	0	0	1	0	0	1
Doc 3	0	0	1	0	0	0	0	0	1	0
Doc 4	0	0	0	1	0	0	0	1	0	0
Doc 5	1	0	0	0	1	0	0	0	0	0

Entonces debemos construir, a partir del conjunto de documentos, la siguiente tabla para cada término.

Ejemplo: t=1

Tabla 6.6 Cálculo de término relevante y no relevante.

Documento	**Relevante ($R_{L=1}$)**	**Relevante ($R_{L=2}$)**	**Relevante ($R_{L=3}$)**
Término presente ($d_{t=1}$=1)	1	1	2
Término Ausente ($d_{t=1}$=0)	2	1	1

Para todos los demás términos, se muestran en la siguiente tabla:

Tabla 6.7 Cálculo de términos relevantes y no relevantes.

	$R_{L=1}$	$R_{L=2}$	$R_{L=3}$
$\mathbf{d_{t=1}=1}$	1	1	2
$\mathbf{d_{t=1}=0}$	2	1	1
$\mathbf{d_{t=2}=1}$	1	0	0
$\mathbf{d_{t=2}=0}$	2	2	3
$\mathbf{d_{t=3}=1}$	1	0	1
$\mathbf{d_{t=3}=0}$	2	2	2
$\mathbf{d_{t=4}=1}$	1	1	0
$\mathbf{d_{t=4}=0}$	2	1	3
$\mathbf{d_{t=5}=1}$	1	0	1
$\mathbf{d_{t=5}=0}$	2	2	2
$\mathbf{d_{t=6}=1}$	1	0	0
$\mathbf{d_{t=6}=0}$	2	2	3
$\mathbf{d_{t=7}=1}$	0	1	1
$\mathbf{d_{t=7}=0}$	3	1	2
$\mathbf{d_{t=8}=1}$	1	1	0
$\mathbf{d_{t=8}=0}$	2	1	3
$\mathbf{d_{t=9}=1}$	1	0	1
$\mathbf{d_{t=9}=0}$	2	2	2
$\mathbf{d_{t=10}=1}$	1	1	1
$\mathbf{d_{t=10}=0}$	2	1	2

De este modo, podemos construir la tabla de $P_{t,L}$ y $\mu_{t,L}$ para todas las combinaciones de t y L:

Tabla 6.8 Cálculo de Pt,L y μt,L.

	$R_{L=1}$	$R_{L=2}$	$R_{L=3}$
$\mathbf{P_{t=1,L=1,2,3}}$	0,333	0,500	0,667
$\boldsymbol{\mu}_{\mathbf{t=1,L=1,2,3}}$	0,417	0,500	0,583
$\mathbf{P_{t=2,L=1,2,3}}$	0,333	0,000	0,000
$\boldsymbol{\mu}_{\mathbf{t=2,L=1,2,3}}$	1,000	0,833	0,833
$\mathbf{P_{t=3,L=1,2,3}}$	0,333	0,000	0,333
$\boldsymbol{\mu}_{\mathbf{t=3,L=1,2,3}}$	0,833	0,667	0,833

$\mathbf{P_{t=4,L=1,2,3}}$	0,333	0,500	0,000
$\mu_{t=4,L=1,2,3}$	0,750	0,833	0,583
$\mathbf{P_{t=5,L=1,2,3}}$	0,333	0,000	0,333
$\mu_{t=5,L=1,2,3}$	0,833	0,667	0,833
$\mathbf{P_{t=6,L=1,2,3}}$	0,333	0,000	0,000
$\mu_{t=6,L=1,2,3}$	1,000	0,833	0,833
$\mathbf{P_{t=7,L=1,2,3}}$	0,000	0,500	0,333
$\mu_{t=7,L=1,2,3}$	0,583	0,833	0,750
$\mathbf{P_{t=8,L=1,2,3}}$	0,333	0,500	0,000
$\mu_{t=8,L=1,2,3}$	0,750	0,833	0,583
$\mathbf{P_{t=9,L=1,2,3}}$	0,333	0,000	0,333
$\mu_{t=9,L=1,2,3}$	0,833	0,667	0,833
$\mathbf{P_{t=10,L=1,2,3}}$	0,333	0,500	0,333
$\mu_{t=10,L=1,2,3}$	0,583	0,667	0,583

Entonces para realizar una predicción sobre un nuevo documento, realizamos lo siguiente:

Primero, deseamos conocer a que categoría pertenecen estos documentos:

Tabla 6.9 Documento x Términos a predecir.

	T1	**T2**	**T3**	**T4**	**T5**	**T6**	**T7**	**T8**	**T9**	**T10**
Doc Q1	0	1	0	1	0	1	0	1	0	1
Doc Q2	1	0	0	0	0	0	1	0	0	1

Ahora debemos calcular del documento Q1 para cada etiqueta de L:

$$Rank = \sum_{t\in(q\cap d)} \log(\mathrm{P}_t\,(1-\mathrm{P}_t)\,/\mu_t\,(1-\mu_t)) \qquad (6.1.19)$$

Tabla 6.10 Predicción para documento Q1.

	$R_{L=1}$	$R_{L=2}$	$R_{L=3}$	Q1	L =1	L = 2	L = 3
t = 1	-0,407	-0,602	-0,407	0	0	0	0
t = 2	0,000	-4,347	-4,289	1	0,000	-4,347	-4,289
t = 3	-0,164	-4,551	-0,164	0	0	0	0
t = 4	-0,294	-0,347	-4,532	1	-0,294	-0,347	-4,532
t = 5	-0,164	-4,551	-0,164	0	0	0	0
t = 6	0,000	-4,347	-4,289	1	0,000	-4,347	-4,289
t = 7	-4,532	-0,347	-0,294	0	0	0	0
t = 8	-0,294	-0,347	-4,532	1	-0,294	-0,347	-4,532
t = 9	-0,164	-4,551	-0,164	0	0	0	0,000
t = 10	-0,407	-0,551	-0,407	1	-0,407	-0,551	-0,407
					-0,995	**-9,938**	**-18,048**

Podemos concluir que el documento Q1, no pertenece a ninguna categoría, debido a que los valores son menores a 0.

Realizando lo mismo para el documento Q2, nos encontramos que:

Tabla 6.11 Predicción para documento Q2.

	$R_{L=1}$	$R_{L=2}$	$R_{L=3}$	Q2	L =1	L = 2	L = 3
t = 1	-0,407	-0,602	-0,407	1	-0,407	-0,602	-0,407
t = 2	0,000	-4,347	-4,289	0	0	0	0
t = 3	-0,164	-4,551	-0,164	0	0	0	0
t = 4	-0,294	-0,347	-4,532	0	0	0	0
t = 5	-0,164	-4,551	-0,164	0	0	0	0
t = 6	0,000	-4,347	-4,289	0	0	0	0
t = 7	-4,532	-0,347	-0,294	1	-4,532	-0,347	-0,294
t = 8	-0,294	-0,347	-4,532	0	0	0	0
t = 9	-0,164	-4,551	-0,164	0	0	0	0
t = 10	-0,407	-0,551	-0,407	1	-0,407	-0,551	-0,407
					-5,346	**-1,500**	**-1,108**

Por lo tanto, podemos concluir que el documento Q2 no pertenece a la categoría a ninguna categoría.

7 Implementación y Experimentación del Trabajo

A continuación, se presenta la implementación del modelo y las características del sistema tales como el detalle de los datos utilizados para las pruebas, el tipo de software y hardware usado. Por último, se evalúa el rendimiento del modelo a través de las medidas de evaluación presentadas en la Sección 5.

7.1 Datasets Utilizados

Se utilizaron todos los datasets presentados en la Sección 4, los cuales están formados por documentos (Doc), términos (Term) y sus categorías (Cat). Todos los datasets tienen la misma estructura, a continuación se detalla:

Tabla 7.1 Estructura de los datasets utilizados.

Particiones	Formato
train_data	(Doc x Term)
train_target	(Cat x Doc)
test_data	(Doc x Term)
test_target	(Cat x Doc)

7.2 Software y Hardware Utilizado

Respecto al Software utilizado, se trabajó con la herramienta "MATLAB" v7.10 en la plataforma Windows 7 de 32 bits, en la cual se programaron las funciones correspondientes para lograr el funcionamiento completo del modelo. A continuación se detallan dichas funciones:

Tabla 7.2 Funcionalidades Software desarrollado.

Nombre	Funcionalidad
Data_Medidas	Función que calcula el N° de documentos, N° de categorías, Lcard y Ldens.
Entrada	Cálculo de los términos relevantes presentes u ausentes en los documentos en función de las etiquetas.
Datos	Cálculo de los valores de $P_{t,L}$ y $\mu_{t,L}$ para todas las combinaciones de t y L.
Calclg	Cálculo del logaritmo basado en los valores de $P_{t,L}$ y $\mu_{t,L}$.

Ranking	Entrega la predicción de las etiquetas en base a la data de prueba.
Errores	Entrega los valores de las medidas de evaluación para multi-etiquetas respecto a la predicción.

En el ámbito al Hardware utilizado, éste corresponde a:

- Procesador : Intel Core 2 Duo.
- Memoria RAM : 3 Gb.
- Velocidad del Procesador: 1.66 GHz.

7.3 Pruebas de Rendimiento del Modelo

En esta sección se presentan las pruebas de rendimiento del modelo implementado, éstas se basaron en las medidas de evaluación presentadas en la Sección 5. Cabe señalar que se realizaron distintas pruebas al modelo y se seleccionó la que presentaba mejor rendimiento, las cuales consistían en 4 opciones:

1. Sin normalizar valores.
2. P_t y u_t normalizados.
3. Ranking normalizado.
4. Pt, ut y Ranking normalizado.

Referente a las opciones cabe señalar que algunos valores se normalizaban para transformar valores extremos, ya sea muy pequeños o grandes que distorsionan la predicción del modelo y llevarlos a una distribución normal. Se realizaron pruebas con los distintos datasets, y sólo se obtuvo buenos resultados para la opción 1, por lo que se descartaron las demás opciones. Luego se agregó un factor, cuya función era normalizar los resultados, se incluía justo antes de aplicar la función logaritmo, éste factor relaciona el número de categorías que tiene cada documento. Los resultados entregados por el modelo se muestran en la Tabla 7.3, primero sin factor y luego en la Tabla 7.4 con factor.

Tabla 7.3 Rendimiento Modelo BIM sin factor.

Dataset	Precision	Recall	Accuracy	F1-Micro	F1-Micro x L	F1- Micro x N	Exact-Match	Hamming Loss (%)
Bibtext	0,162	0,286	0,966	0,207	0,222	0,140	0,060	3,376
Corel16k	0,126	0,449	0,927	0,197	0,203	0,120	0,004	7,325
Corel5k	0,080	0,130	0,978	0,099	0,077	**0,019**	**0**	2,231
Enron	0,096	0,237	0,809	0,136	0,124	0,075	**0**	19,057
Genbase	0,225	0,616	0,883	0,330	0,286	0,509	**0**	11,719
Medical	**0,009**	**0,050**	0,818	**0,015**	**0,013**	0,058	**0**	18,233
Reuters_First9	0,669	0,828	0,932	0,740	0,756	0,604	0,600	6,774
Scene	0,182	0,844	**0,286**	0,300	0,297	0,258	**0**	**71,433**

Tabla 7.4 Rendimiento Modelo BIM con factor.

Dataset	Precision	Recall	Accuracy	F1-Micro	F1-Micro x L	F1- Micro x N	Exact-Match	Hamming Loss (%)
Bibtext	0,049	0,003	0,984	**0,005**	**0,003**	**0,001**	**0**	1,6157
Corel5k	0,120	0,318	0,972	0,174	0,172	0,010	**0**	2,839
Enron	0,330	0,287	0,918	0,307	0,294	0,034	**0**	8,225
Genbase	0,100	0,226	0,869	0,139	0,115	0,167	**0**	13,079
Medical	**0,003**	**0,015**	0,818	**0,005**	0,004	0,023	**0**	18,236
Reuters_First9	0,340	0,875	0,791	0,490	0,582	0,278	0,308	20,8982
Scene	0,181	1	**0,181**	0,307	0,304	0,306	**0**	**81,8562**

En el caso de la medida de evaluación Hamming Loss se encuentra en % y al contrario de las otras medidas mientras más grande sea el valor implica que el modelo tiene un mal rendimiento. Mientras más cercano a 0 implica que el modelo tiene un buen rendimiento, es decir, el modelo ha clasificado correctamente, esto se debe a que ésta medida suma las etiquetas mal clasificadas.

Se presentan los gráficos de los resultados sobre el rendimiento del modelo BIM, en base a las tablas presentadas anteriormente, en el cual se puede ver que con factor no mejoran la mayoría de los datasets, sólo el dataset Enron tiene una mejora sustancial respecto a sin factor.

La opción que tuvo mejores resultados fue el Modelo BIM sin factor, esto se ve reflejado en el gráfico de la Figura 7.1, en el cual las medidas de evaluación mejor evaluadas fueron Accuracy, para todos los datasets salvo en Scene, y la 2da medida mejor evaluada fue Recall. El dataset que ha manera general tuvo buenos resultados por parte del modelo fue el Reuters First9 y en cambio, el dataset Medical fue el que tiene los peores resultados. Respecto a las medidas de evaluación, el modelo realiza una mala predicción de

los conjuntos de vector de documentos, esto puede ser visto debido a que la medida Exact-Match en la mayoría de los datasets tiene un valor 0.

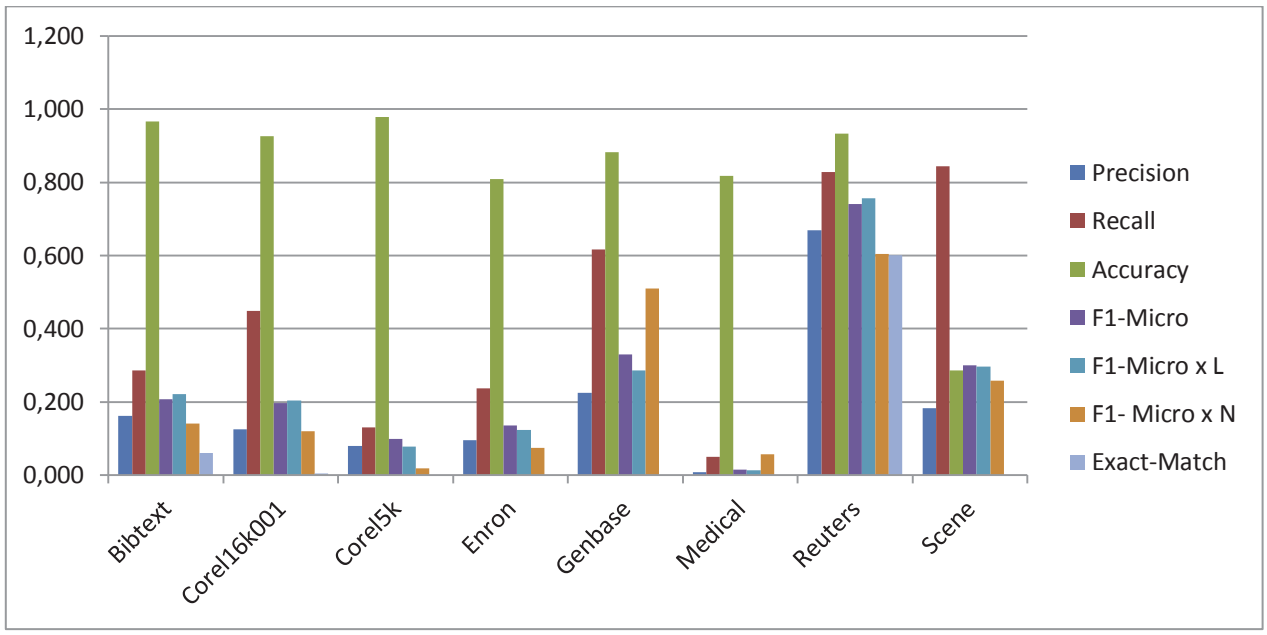


Figura 7.1 Gráfico del Rendimiento del Modelo BIM sin factor.

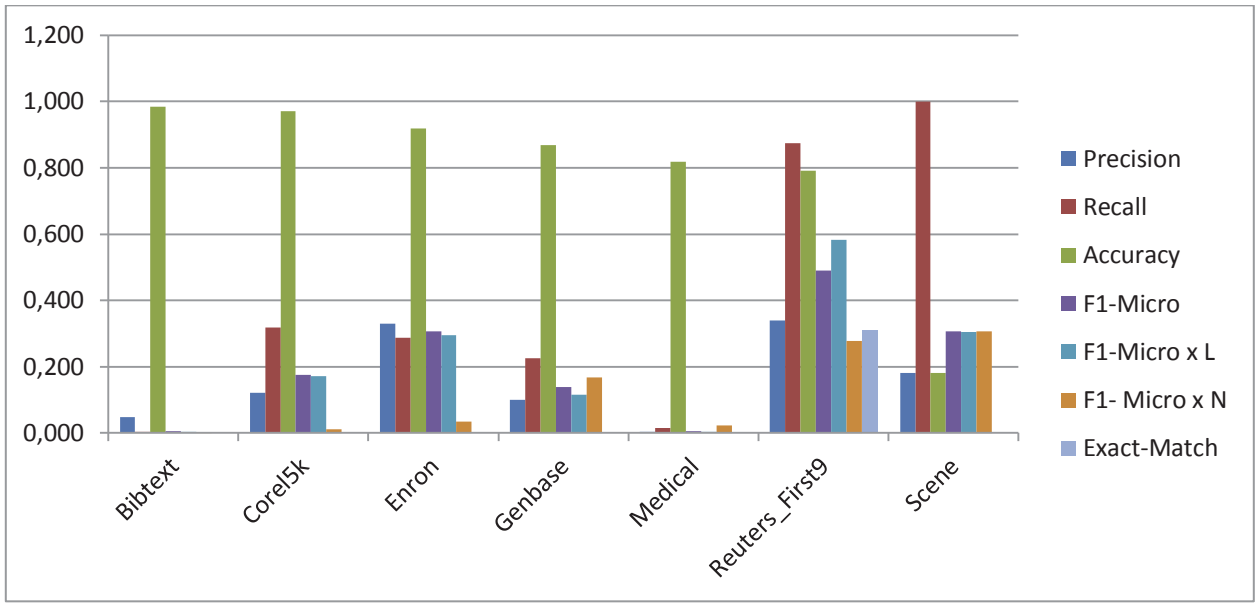


Figura 7.2 Gráfico del Rendimiento del Modelo BIM con factor.

En la Figura 7.3, muestra la medida de evaluación Hamming Loss, se puede apreciar que al aplicar el factor, no mejora el rendimiento del modelo. También, se puede determinar que la predicción sobre el dataset Scene, obtuvo el peor resultado, indicando esto que se realizó una pésima clasificación.

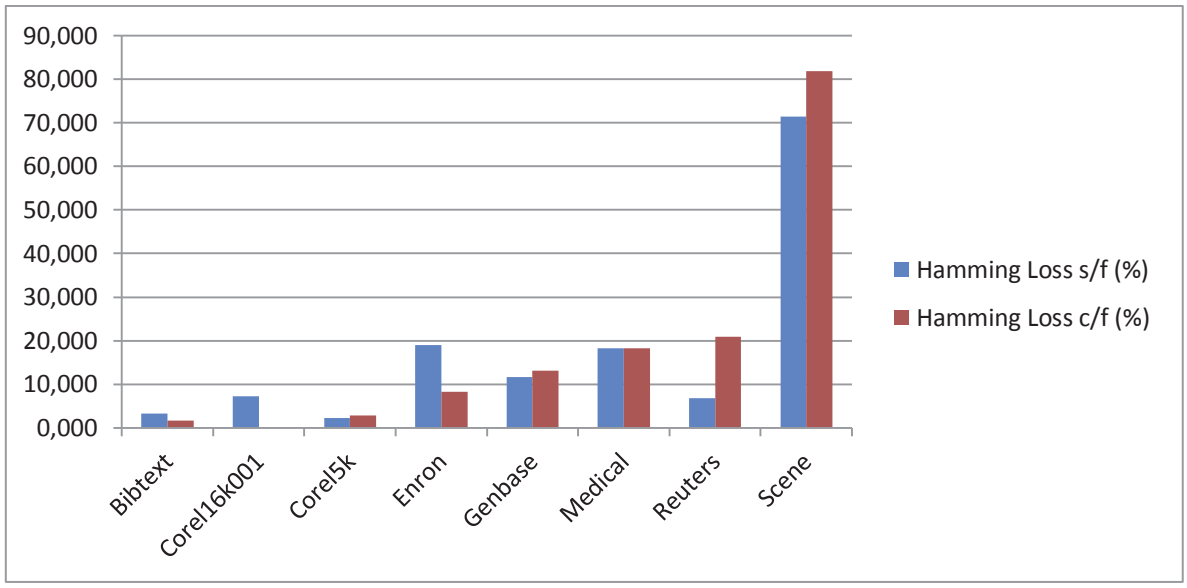


Figura 7.3 Rendimiento del Modelo BIM en base a h-loss.

8 Análisis de Resultados

En esta sección, primero se analiza el rendimiento del modelo en base a las medidas de evaluación presentadas en conjunto a los datasets utilizados, también se buscará si es que existe alguna relación en base a éstos factores. Luego se compara el modelo con otros modelos existentes, así establecer sus ventajas o desventajas.

8.1 Resultados del modelo BIM

Se partirá realizando un análisis de los datasets utilizados, dónde se presentan dos gráficos en los cuales se muestra el Promedio y Desviación Estándar de cada datasets, primero respecto a los términos que tienen los documentos (Figura 8.1), y en segundo lugar, respecto a las categorías que tienen los documentos (Figura 8.2).

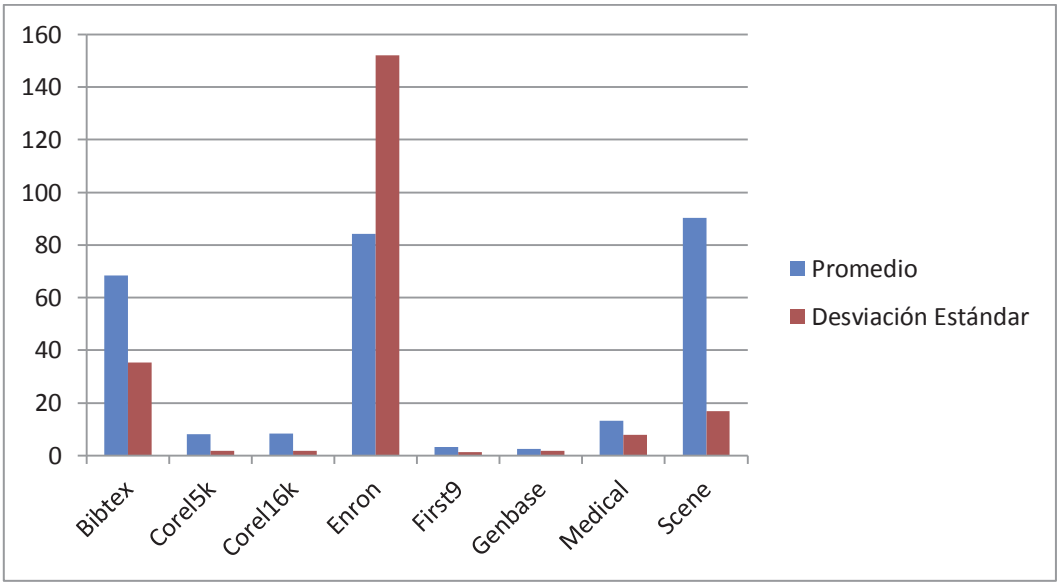


Figura 8.1 Análisis de los Datasets - Documentos vs Términos.

Se puede apreciar un comportamiento uniforme en los datasets (Figura 8.1), salvo en Enron el cual tiene una desviación estándar muy alta en comparación al promedio, esto indica que la cantidad de términos por documento varía enormemente.

Tabla 8.1Términos en cada Dataset.

	Bibtex	**Corel5k**	**Corel16k**	**Enron**	**First9**	**Genbase**	**Medical**	**Scene**
Términos	[1, 1836]	[1, 499]	[1, 500]	[1,1001]	[1, 686]	[1, 1186]	[1, 1499]	[1, 294]

En general, se puede deducir que el promedio de términos que tiene cada documento respecto total de términos posibles es similar en la mayoría de los datasets, salvo en Enron que la desviación estándar es mucho mayor que el promedio, otro caso diferente es el de Scene en el cual el promedio es mucho mayor que la desviación estándar.

En relación a las categorías que tiene cada documento (Figura 8.2), todos los datasets tienen un comportamiento similar, en los que se puede notar alguna diferencia son Corel5k, Corel16k y Enron, debido a la gran diferencia entre el promedio y la desviación estándar.

Baja cardinalidad y densidad de etiquetado es común en datasets de textos, donde la mayoría de ejemplos se ajusta naturalmente bajo un esquema de una etiqueta y el multi-etiquetado fue introducido para resolver las ambigüedades. Considerar el caso del dataset Scene, clasificación de imágenes, donde la mayoría de las imágenes son relevantes solo para una etiqueta, debido a que sus tipos de etiquetado son: "mountain", "field" o "sunset". Por ejemplo cuando una imagen es relevante para ambas etiquetas como "mountain" y "field" se utiliza múltiples etiquetas para resolver ambigüedades ocasionales.

Alta cardinalidad de etiquetas es a menudo observado en datasets con un amplio dominio. Ejemplos de esto, son los datasets biológicos, como Genbase donde se espera que los genes tengan múltiples funciones y en datasets como Enron sobre emails, donde las categorías o etiquetas toman la forma de un checklist.

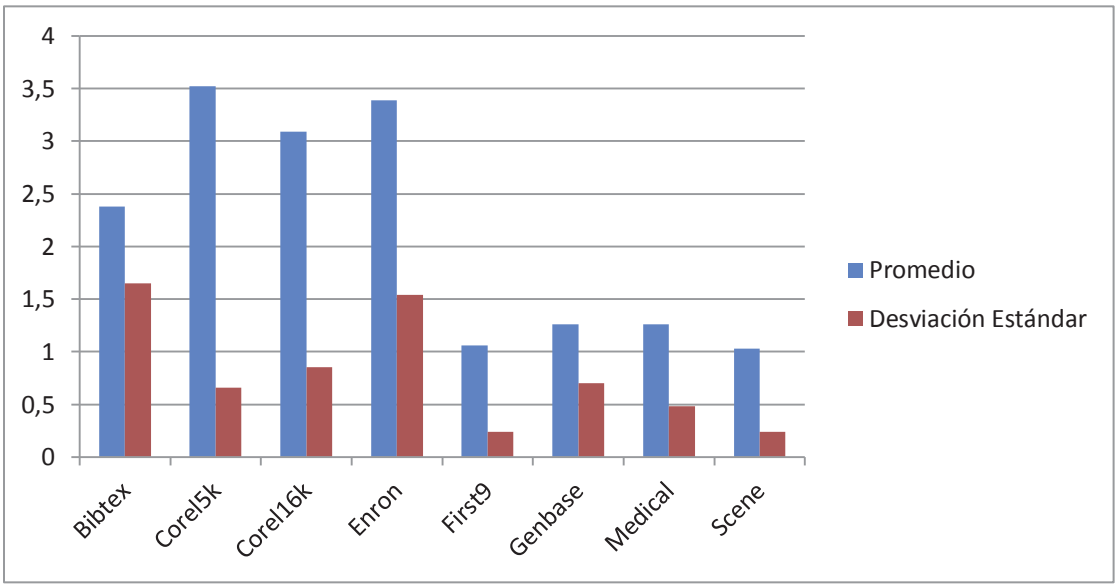


Figura 8.2 Análisis de los Datasets. Documentos vs Categorías.

Volviendo al modelo BIM y relacionados con los datasets, el que tuvo mejor valoración en base a las medidas de evaluación fue el dataset Reuters First9, si revisamos el gráfico en 3D de la Figura 4.1, podemos ver que existe una gran densidad de datos en un rango acotado, y que no tiene muchos valores que estén fuera del rango, una relación similar se puede apreciar con Bibtext. A diferencia de los datasets Enron, Medical y Genbase que los datos se encuentran dispersos, es decir, no existe una densidad alta de datos en un rango acotado.

Respecto a BIM y las medidas de evaluación, el dataset Medical presenta el peor desempeño en comparación a los demás datasets. Por otro lado, la predicción sobre Scene en base a la medida de evaluación Hamming Loss, presenta el peor valor un 71.43%, a diferencia de los otros datasets que la mayoría está por debajo del 10%.

La medida de evaluación peor evaluada es Exact-Match, esto quiere decir que el modelo no realiza una buena clasificación para los vectores de cada documento, es decir, la predicción que realiza en base a vectores es bajísima y en la mayoría de los casos nula.

En general, el modelo tuvo un rendimiento óptimo en la mayoría de las medidas de evaluación, y hay que considerar que fue probado con datasets de dominios distintos, en los cuales los que mejor resultados tuvieron fueron los de texto e imágenes, salvo algunas excepciones como Medical (Texto) y Scene (Imágenes).

8.2 Comparación con Modelos Relacionados

En esta sección, el modelo BIM se compara con otros modelos relacionados en base a las medidas de evaluación. Los otros modelos tienen distintos métodos o enfoques de ranking, estos modelos son: BR – Binary Relevance, EPS – Ensembles of Pruned Sets, RakEL – Random k-labEL subsets, CLR – Calibrated Label Ranking y MlkNN – Lazy multi-label method [12], el cual adapta el kNN – k-Nearest Neighbor.

Respecto a la Tabla 8.2, se puede apreciar que el modelo BIM en 5 de 6 datasets tiene el mejor valor de Accuracy respecto a los demás modelos. El modelo que tiene peores valores es el MlkNN [13]. Cabe señalar que a la medida de evaluación Accuracy, entrega una proporción entre la intersección de la predicción y la etiqueta real en relación a la unión de ésta, por lo que el modelo BIM, tiene su valor más bajo con Scene, siendo éste un dataset de imágenes al igual que Corel5k y Corel16k, pero a diferencia éstos datasets tienen un accuracy muy alto.

Tabla 8.2 BIM vs Otros Modelos: Accuracy.

Dataset	BIM	EPS_2	EPS_1	$RakEL_1$	BR	$RakEL_2$	CLR	MlkNN
Bibtext	0,97	0,32	0,30	0,29	0,32	0,30		**0,18**
Corel16k001	0,93							
Corel5k	0,98							
Enron	0,81	0,44	0,45	0,46	0,39	0,42		**0,35**
Genbase	**0,88**	0,95	0,97	0,98	0,98	0,98		0,95
Medical	0,82	0,75	0,75	0,76	0,73	0,75		**0,62**
Reuters_First9	0,93	0,50	0,50	0,45	0,32	**0,30**		0,44
Scene	**0,29**	0,74	0,73	0,72	0,58	0,72	0,71	0,71

En base a los resultados de la Tabla 8.3, el modelo BIM tiene el peor rendimiento en base a Exact-Match. Dicha medida es la peor evaluada, por lo que BIM no realiza una buena predicción en base a los vectores de cada documento, siendo éste un punto importante para mejorar.

Tabla 8.3 BIM vs Otros Modelos: Exact-Match.

Dataset	BIM	EPS_2	EPS_1	$RakEL_1$	BR	$RakEL_2$	CLR	MlkNN
Bibtext	0,06	0,13	0,14	0,09	0,12	0,10		**0,04**
Corel16k001	0,004							
Corel5k	**0**							
Enron	**0**	0,14	0,14	0,14	0,11	0,11		0,01
Genbase	**0**	0,90	0,94	0,97	0,97	0,97		0,90
Medical	**0**	0,64	0,65	0,66	0,65	0,65		0,50
Reuters_First9	0,60	0,38	0,38	0,36	0,27	**0,25**		0,29
Scene	**0**	0,68	0,69	0,66	0,51	0,67	0,66	0,64

Finalmente, el modelo BIM en base a F1-Macro x L, tiene mejor rendimiento con el Dataset Reuters_First9, en comparación al resto de los modelos, respecto a los otros datasets BIM, no tiene buenos resultados al realizar comparaciones de los vectores de los documentos en base a sus etiquetas.

Tabla 8.4 BIM vs Otros Modelos: F1-Macro x L

Dataset	BIM	EPS_2	EPS_1	$RakEL_1$	BR	$RakEL_2$	CLR	MlkNN
Bibtext	0,21	0,216	0,188	0,244	0,287	0,297		**0,152**
Corel16k001	0,20							
Corel5k	0,10							
Enron	0,14	0,14	0,16	0,21	0,2	0,21		**0,1**
Genbase	**0,33**	0,57	0,64	0,76	0,77	0,76		0,59
Medical	**0,01**	0,29	0,31	0,36	0,35	0,35		0,23
Reuters_First9	0,74	0,24	0,26	0,28	0,22	**0,21**		0,25
Scene	**0,30**	0,76	0,76	0,75	0,68	0,75	0,74	0,75

Concluyendo, el modelo BIM presenta muy buenos resultados en base a Accuracy, pero en relación a Exact-Match y F1-Macro x L, presenta valores muy bajos, por lo que se deduce que el modelo funciona bien al clasificar en forma individual, pero al momento de clasificar un conjunto de documentos, como son los vectores de cada documento no tiene un buen rendimiento, un aspecto importante que habría que mejorarse. Los datasets en los cuales BIM tiene peor rendimiento, son Genbase y Scene, el primero corresponde al dominio de biología en base a genes y el segundo al dominio de imágenes, esto queda demostrado en las 3 tablas presentadas anteriormente en comparación a los otros modelos.

9 Conclusiones

En el presente trabajo de investigación, se ha logrado establecer el marco conceptual en el ámbito de clasificación de textos. Se presentaron las definiciones y fundamentos de las distintas técnicas a utilizar en el marco teórico y en el estado del arte. Con esto se ha logrado cumplir con el primer objetivo de este proyecto.

A partir del estudio y análisis realizado sobre los enfoques de Ranking, se determinó que la principal diferencia entre cada uno radica en el espacio de entrada y salida. El enfoque Pointwise, es el que más se aleja del concepto de ranking ya que tiene como espacio de entrada un documento y como salida el grado de relevancia de éste. El enfoque Pairwise tiene como espacio de entrada un par de documentos y como salida la dupla de documentos en forma ordenada. Por último, el enfoque Listwise posee las mejores características debido a que tiene como espacio de entrada un conjunto de documentos y como espacio de salida el conjunto de documentos ordenados en base al grado de relevancia. La razón de esto se debe a que Listwise permite comparar todos los documentos en forma simultánea, permitiendo generar una lista de ranking de los documentos más exacta y precisa. Como resultado de esta primera parte de la investigación se concluyó que el enfoque Listwise es el que mejor representa el concepto de ranking, en relación a lo señalado anteriormente.

En una segunda parte del informe, se presenta el modelo BIM basado en la recuperación probabilística y en el trabajo con datasets simple-etiqueta. En esta investigación se trabajó con datasets multi-etiqueta, por lo que se le realizaron modificaciones al modelo original. Como resultado de esta modificación, se pudo apreciar que el modelo BIM tuvo un óptimo desempeño al clasificar en forma individual los documentos, esto se puede ver en medidas de evaluación como Hamming-Loss y Accuracy. En cambio, al realizar la clasificación de cada documento en un conjunto de categorías, el modelo no tuvo un buen rendimiento, esto se puede apreciar en medidas como Exact-Match y F1 tanto para etiquetas como documentos. Respecto al dominio de los datasets utilizados, los que mejor clasifica BIM son los textos e imágenes, salvo en el caso del dataset de imagen Scene. Las características de los datasets clasificados correctamente tenían una gran densidad de datos en un rango acotado y además, contenían muy pocas distorsiones que no afectaban mayormente la predicción. En cambio, algunos datasets que fueron mal clasificados sus datos se presentaban en forma dispersa, muy separados entre sí. Esto se ve reflejado por la relación entre promedio y la desviación estándar de cada dataset.

Finalmente se concluye que el modelo BIM modificado para el uso de datasets multi-etiqueta, tiene un óptimo rendimiento al clasificar en forma individual los documentos en cada categoría.

10 Referencias

[1] Sebastiani F. "Text Categorization". Encyclopedia of Database Technologies and Applications. Idea Group Publishing, Hersbey, US, pp. 683-687, 2005.

[2] Trotman A., "Learning to rank", Information Retrievel, Vol.8, pp. 359-381, 2005.

[3] Joachims T. "Text Categorization with Support Vector Machines: Learning with many relevant features". Proceedings of ECML-98, 10th European Conference on Machine Learning. Chemnitz, Germany, Issue 1398, pp. 137-142, 1998.

[4] Betancourt G. "Las Máquinas de Soporte Vectorial". Scientia et Technica, Cap XI, No 27, Abril 2005. ISSN 0122-1701.

[5] Liu T-Y. "Learning to Rank for Information Retrieval". Foundations and Trends in Information Retrieval, Vol 3, No.3. pp. 225-331, 2009.

[6] Jarvelin K. y Kekalainen J., "Cumulated gain-based evaluation of IR techniques". ACM Transactoions on Information Systems. Vol. 20, No. 4, Octubre 2002. Pages 422-446.

[7] Liu T-Y., Qin T., Xiong W-Y y Li H, "Ranking with multiple hyperplanes", en SIGIR 2007, pp. 279-286, 2007.

[8] Liu T-Y., Qin T. y Li H, "A general approximation framework for direct optimization of information retrieval measures", Technical Report, Microsoft Research, MSR-TR-2008-164, 2008.

[9] Yue Y., Finley T., Radlinski F. y Joachims, "A Support vector method for optimizing average precision", en SIGIR 2007, pp.271-278, 2007.

[10] Tsoumakas, G., Vilcek, J., Xioufis, E. S., and et al. (2009b). MULAN: A java library for multi-label learning. "http://mulan.sourceforge.net/datasets.html".

[11] Lewis, D. D. (1998). Naive (Bayes) at forty: The independence assumption in information retrieval. In ECML '98: 10th European Conference on Machine Learning, pages 4–15. Springer.

[12] Zhang, M.-L. and Zhou, Z.-H. (2007a). ML-KNN: A lazy learning approach to multi-label learning. Pattern Recognition, 40 (7): 2038–2048.

[13] Tsoumakas, G., Dimou, A., Spyromitros, E., Mezaris, V., Kompatsiaris, I., and Vlahavas, I. (2009a). Correlation-based pruning of stacked binary relevance models for multi-label learning. In MLD '09: 1st ECML/PKDD Workshop on Learning from Multi-Label Data, pages 101–116.