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

“Análisis del PageRank como factor de peso en la clasificación automática de textos”

ANDRÉS JEREZ LEIVA

PROFESOR GUÍA: RODRIGO ALFARO ARANCIBIA

PROFESOR CORREFERENTE: IGNACIO ARAYA ZAMORANO

CARRERA: INGENIERÍA CIVIL INFORMÁTICA

PROYECTO II

SEPTIEMBRE, 2018

Índice

Resumen ..... **iii**
Lista de Figuras ..... **iv**
Lista de Tablas ..... **v**
1 Introducción **1**
2 Descripción del problema **2**
2.1 La clasificación de texto de forma manual 2
2.2 Clasificación manual versus clasificación automática 3
3 Definición de objetivos **5**
3.1 Motivación 5
3.2 Objetivos generales 5
3.3 Objetivos específicos 5
4 Marco Teórico **6**
4.1 La Inteligencia Artificial 6
4.2 Las Máquinas de Aprendizaje 6
4.3 La Clasificación Automática de Texto 7
4.3.1 Extracción de Características 7
4.3.2 Pre-procesamiento 7
4.3.3 Indexado 8
4.3.4 Reducción de dimensionalidad 11
4.4 El PageRank 11
4.4.1 Funcionamiento general 12
4.4.2 Importancia 13
4.4.3 Indexado 13
4.5 Técnicas de Aprendizaje 18
4.5.1 Máquina de Vectores de Soporte 19
4.5.2 Árboles de Decisión 20
4.5.3 K Vecinos más Cercanos 21
4.5.4 Modelo Probabilístico de Naive Bayes 22
4.5.5 Algoritmo de Rocchio 23
4.5.6 Redes Neuronales 23
4.5.7 Random Forest 24
5 Metodología **26**

6 Desarrollo de la Solución 27
6.1 Software 27
6.1.1 Procesamiento de Lenguaje Natural 27
6.1.2 Librería NLTK 27
6.1.3 Librería Numpy 28
6.1.4 Librería Pandas 28
6.1.5 Librería Scikit-Learn 28
6.1.6 Librería SciPy 29
6.2 Selección del Data Set 29
6.3 Reducción de Palabras Vacías 29
6.4 Extracción de Frases o Palabras Claves 35
6.5 Construcción de Grafos Conceptuales 35
6.5.1 Aplicación del Factor PageRank 35
6.5.2 Grafos Conceptuales Ponderados 36
6.6 Cálculo del TF-PageRank 36
6.7 Fase de Entrenamiento – train 36
6.8 Fase de Pruebas – test 36
6.9 Clasificación de los Documentos - Algoritmo “Naive Bayes” 37
7 Resultados 38
7.1 Modelo TF-PR & Naive Bayes 38
7.1.1 Resultados 38
7.1.2 50 Top Words TF-PR 40
7.1.2 50 Top Words PageRank 41
7.2 Modelo TF-IDF & Naive Bayes 42
7.2.1 Resultados 42
7.2.2 50 Top Words 43
7.3 Comparaciones y Análisis 44
8 Trabajo Futuro 47
9 Conclusión 48
10 Referencias 49

Resumen

Hoy en día es posible encontrar una innumerable cantidad de datos e información gracias al internet, es por esto que surge la necesidad de administrar todo ese contenido, de manera tal que se pueda volver a acceder a este contenido con mayor facilidad, reduciendo los tiempos de búsqueda considerablemente. Una de las maneras que se pueden utilizar para la administración y categorización de la información es la clasificación automática de texto, que a diferencia de su competencia (la clasificación manual), es mucho más eficaz. Además de esto, es necesario utilizar ciertas medidas que logren diferenciar y valorar estos contenidos de la web con el objetivo de detectar lo más relevante, es decir, considerar aquello que tenga más calidad por sobre cantidad. El PageRank es una de las medidas que se utilizan para este trabajo y consiste en determinar la relevancia de un sitio web. Esta técnica utiliza un sistema de evaluación directamente proporcional entre relevancia y evaluación, es decir, mientras mayor sea su valor, más importante será el sitio. El experimento al que se apunta, consiste en alterar la finalidad del PageRank de trabajar con sitios web, a realizar los mismos procesos, pero aplicándolos a documentos y artículos, generando un indicador llamado TF-PageRank, que permita dar peso al grafo de palabras claves que se generará luego de reducir las palabras vacías (stopwords) para finalmente analizar el resultado y clasificar el artículo según corresponda.

Palabras Claves: Clasificación automática de texto, aprendizaje de máquinas, inteligencia artificial, pagerank, term frequency, procesamiento de lenguaje natural, grafos conceptuales, grafos ponderados.

Abstract

Today it is possible to find an innumerable amount of data and information thanks to the internet, that is why it arises the need to manage all that content, in such a way that you can access this content more easily, reducing the times of Search considerably. One of the ways that can be used for the management and categorization of information is automatic text classification, which, unlike its competence (manual classification), is much more effective. In addition to this, it is necessary to use certain measures that manage to differentiate and value these contents of the web with the objective of detecting the most relevant, that is, to consider what has more quality over quantity. PageRank is one of the measures that are used for this work and is to determine the relevance of a website. This technique uses a system of evaluation directly proportional between relevance and evaluation, that is, the greater the value, the more important the site. The experiment aimed at, is to alter the purpose of PageRank to work with websites, to perform the same processes but applying them to documents and articles, generating an indicator called TF-PageRank, which allows to give weight to the graph of keywords that will be generated after reducing the empty words (stop words) to finally analyze the result and classify the article accordingly.

Keywords: *Automatic text classification, machine learning, artificial intelligence, pagerank, term frequency, natural language processing, conceptual graphs, weighted graphs.*

Lista de Figuras

Figura 2.1 Proceso para la clasificación manual 3
Figura 4.1 Representación de un grafo al aplicar PageRank 12
Figura 4.2 Componentes del algoritmo de PageRank 13
Figura 4.3 Ejemplo de hiperplano de separación 20
Figura 4.4 Ejemplo de hiperplano de separación, de entre los infinitos posibles 20
Figura 4.5 Ejemplo de árbol de decisión 20
Figura 7.1 Rendimiento TF-PR & Naive Bayes 38
Figura 7.2 Matriz de Confusión TF-PR & Naive Bayes 39
Figura 7.3 Rendimiento TF-IDF & Naive Bayes 41
Figura 7.4 Matriz de Confusión TF-IDF & Naive Bayes 42
Figura 7.5 Gráfico de Líneas - Rendimiento TF-PR & Naive Bayes 44
Figura 7.6 Gráfico de Líneas – Rendimiento TF-IDF & Naive Bayes 44

Lista de Tablas

Tabla 4.1 Representación de matriz 14
Tabla 4.2 Representación de matriz completada 14
Tabla 7.1 Top Words TF-PR & Naive Bayes 40
Tabla 7.2 Top Words PageRank 41
Tabla 7.3 Top Words TF-IDF & Naive Bayes 43

1 Introducción

El increíble aumento en la cantidad de documentos electrónicos y páginas web está siendo un fenómeno casi incontrolable, esto, debido a los grandes aportes de investigación, lo que conlleva a una constante generación de información; y a la conectividad a Internet, cosa que hace un par de años atrás no era un medio de comunicación al cual todos podían acceder. Es por este motivo, que fue necesario generar técnicas que permitan organizar y clasificar contenidos de acuerdo a su temática de fondo o categoría.

Gracias a los grandes aportes de la inteligencia computacional, podemos reducir, de una manera considerable, los costos asociados y el tiempo que demanda la ejecución de esta acción de clasificar textos, mediante el uso estos algoritmos, los cuales tienen como objetivo, "aprender" o reconocer patrones redundantes.

Hoy en día ya es posible encontrar la clasificación de texto de forma manual y también, la automática. La primera, consiste básicamente en organizar la información utilizando como recursos, a humanos, quienes tendrán la tarea de leer los contenidos y asignarlos a las categorías que correspondan, lo que se torna en un proceso muy largo, además de costoso. La segunda, consiste en un conjunto de algoritmos, sistemas y procesos que sean capaces de clasificar un documento dentro de diversas categorías o clases, dependiendo del contenido y temática que presenten, utilizando una serie de técnicas de aprendizaje automático y de procesamiento de lenguaje natural.

Últimamente, el estudio de la clasificación automática de texto ha sido un foco de investigación para esta área de la informática, la cual ha dado buenos frutos. Gracias a lo anterior, es que nace el factor PageRank (de entre muchos otros), como propuesta de solución a la problemática que se plantea en este documento.

El PageRank es un valor numérico que representa la importancia que una página web tiene dentro de internet, de acuerdo al ranking que poseen sus palabras claves. Consiste en construir un grafo en base a las páginas que circulan en la web, las cuales estarán representadas por nodos, mientras que las aristas vendrán siendo los enlaces de entrada y salida de estas páginas.

La presente investigación busca analizar qué tan eficiente puede ser el PageRank al considerarlo en la clasificación automática de texto, proponiendo una nueva forma para calcular el peso de las palabras, combinándolo con el factor de peso TF (term frequency).

El trabajo está constituido de la siguiente manera: en una primera instancia se da paso a la descripción y definición de la problemática a tratar, considerando una comparación entre clasificación automática y manual, luego se presenta el marco teórico, el cual describe una serie de conceptos asociados a la investigación. En el siguiente apartado se describe el desarrollo de la solución, donde se puntualizan los recursos necesarios para implementarla. Además, se detalla la metodología de manera cronológica, y finalmente, se presentan las conclusiones que se desprenden de los resultados de la investigación.

2 Descripción del problema

El problema de clasificar textos de manera automática es un tema que ha sido objeto de estudio durante mucho tiempo dentro del área de la informática, debido a que actualmente existe una cantidad innumerable de páginas web y documentos en formato electrónico, lo que imposibilita la clasificación de manera manual. Este estudio, ha estado relacionado de manera directa con la aplicación de la Inteligencia Artificial y las Máquinas de Aprendizaje, quienes buscan generar algoritmos con capacidad de aprendizaje las cuales buscan desarrollar algoritmos que sean capaces de "aprender" o reconocer ciertos patrones que sean recurrentes de una serie infinita de clases que es posible encontrar, considerando que por cada una se posee un gran volumen de texto, el cual ha sido clasificado de manera previa.

Podríamos definir, entonces, a la clasificación automática de textos, como aquella tarea que tiene el objetivo de asignar y organizar, de manera automática, textos en categorías predefinidas. Para lograr esto, es necesaria la utilización de algoritmos que sean capaces de clasificar los textos mediante ciertas técnicas de aprendizaje, como se mencionó anteriormente.

En muchas ocasiones, esta clasificación se obtiene producto de un análisis estadístico, de manera tal que se recolectan ciertos datos que pueden ser de gran utilidad para el estudio, dentro de los cuales encontramos, a modo de ejemplo: la frecuencia de palabras en el documento, la frecuencia de las palabras en las categorías, etc. Además, cabe mencionar que existen diferentes modelos, índices y técnicas que circulan en la actualidad a través de la internet y que son bastante utilizadas para la clasificación de textos, dentro de las cuales encontramos conceptos como los siguientes: Máquina de Vectores de Soporte, Árboles de Decisión, K Vecinos más Cercanos, Modelo Probabilístico de Bayes, Algoritmo de Rocchio, Redes Neuronales, factor PageRank, factor tf-idf, Random Forests, entre otros.

2.1 La clasificación de texto de forma manual

La clasificación ejecutada por humanos es un proceso que busca organizar todo el conocimiento existente posible en el universo, contemplando un determinado orden sistemático. Consiste en segmentar información distinguiendo ciertas características que se presenten dentro del contenido, para posteriormente agrupar en una clase o categoría aquellos contenidos que presenten características en común.

Si lo anterior lo traducimos a términos de análisis documental, deberíamos realizar los siguientes pasos para la clasificación de documentos.

- Analizar un documento, con el objetivo de obtener una idea clara del contenido que se presenta.
- Realizar una síntesis de los contenidos expuestos en el texto como idea principal.
- Comparar la idea principal del contenido con alguna categoría de clasificación, con el objetivo de identificar a qué categoría se asemeja más.

Con estas tareas, es posible almacenar la documentación de manera estructurada y ordenada, para posteriormente ser utilizada y ubicada con facilidad. Lo mencionado con anterioridad lo podemos representar a través del siguiente esquema:

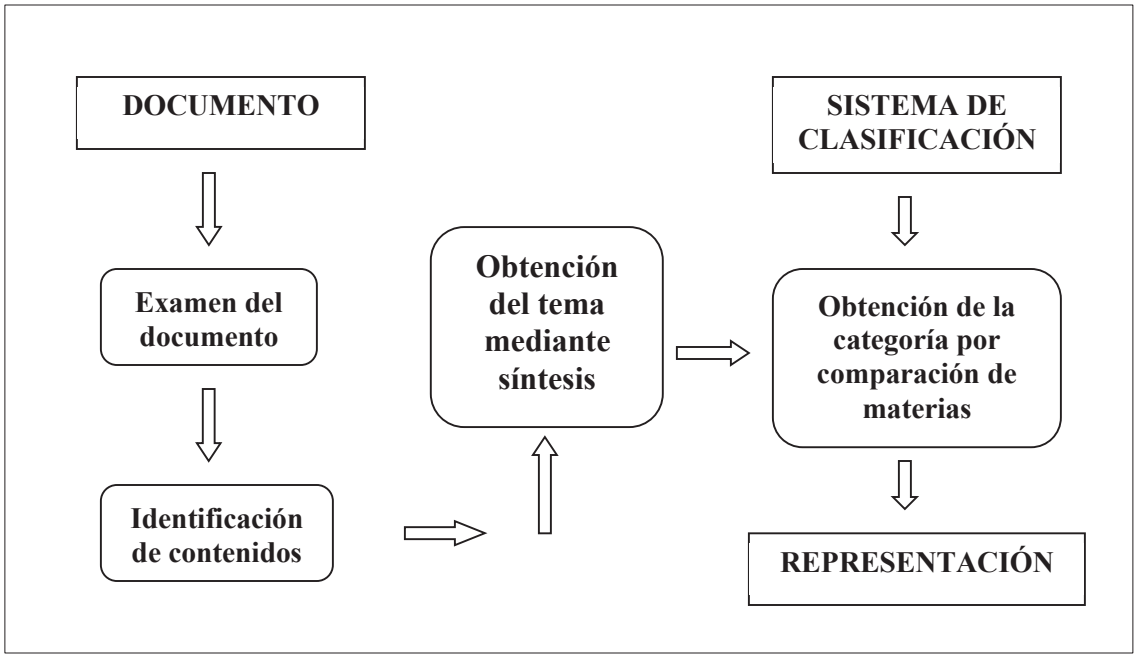

Figura 2.1. Proceso para la clasificación manual.

En el esquema, podemos notar que se ha descompuesto la tarea de clasificar texto en sub tareas, con el objetivo de visualizar de mejor manera la comparación entre la clasificación manual versus la automática.

2.2 Clasificación manual versus clasificación automática

Uno de los problemas principales de la clasificación de texto manual, es la labor de analizar el documento y lograr identificar hacia qué tema apunta sus contenidos. Un sistema de clasificación automático posee una tarea similar a lo que se llama "Indización Automática".

En este proceso, se representa el documento escogiendo una parte de los términos de los que se compone, considerando además su ponderación para tratar de expresar aquellos contenidos que contiene. Además, parece claro que este proceso no se puede comparar con lo que un humano realiza, ya que la lectura vendría siendo secuencial, es decir, la forma común y corriente de leer un texto, mientras que el cálculo de la ponderación o el peso de la importancia de las palabras vendría siendo estadística.

Vamos a suponer que el sistema lleva a cabo este proceso de ponderación. Ahora, el sistema posee cierta representación de los documentos de forma apropiada, la que se vería reflejada en sus contenidos. Hay que considerar que el sistema no percibe lo que estos contenidos representan, pero los expresa de tal manera que simula este aprendizaje. Luego, el próximo paso consiste en crear un enlace de esta representación, en donde los contenidos

del documento se encuentran de manera implícita, con una categoría definida que los logre sintetizar.

Como el sistema aun no logra comprender los contenidos del documento, es fácil darse cuenta que no se podrá llevar a cabo la compleja tarea de pasar del análisis de éste hasta realizar la síntesis, tal cual lo haría un humano. Es más, el clasificador no podrá dar ese paso jamás, porque no logrará identificar qué materia podría sintetizar de mejor manera el documento. Esto es debido a que carece de herramientas necesarias que logren ese objetivo, a menos que logre encontrar en la suma de los términos de indización, una representación apropiada a esa materia, cosa que nosotros como humanos no hacemos al momento de clasificar contenidos. Sin embargo, un clasificador tampoco lograría tomar la decisión de saber cuál es el tema de manera evidente, ya que no identificaría las materias ni su significado.

Otra importante dificultad, es la tarea de conectar la temática del documento con alguna categoría de clasificación. Quizás nosotros ya estamos acostumbrados a realizar este tipo de tareas, por ende, no es un proceso muy difícil, ya que las clasificaciones estarán orientadas a localizar las materias registradas de manera más fácil. Un humano, por ejemplo, tendría que comprobar a qué corresponde la materia del documento, comparando con cada una de las diferentes clases principales de clasificación, de manera tal que encuentre la correcta, luego tendría que ir descendiendo de acuerdo a sus subclases, hasta finalmente acertar con aquella a la que más se adecúe a la temática del texto. Es decir, se debe comparar la materia del documento con aquella subyacente a ella, en una categoría hasta que se encuentre una que le corresponda. En cambio, un clasificador artificial no podría realizar dicha labor de manera directa, ya que como se mencionó anteriormente, el sistema desconocería las materias y sus significados.

Entonces, como es posible darse cuenta, las tareas de un clasificador humano no pueden ser llevadas a cabo por un clasificador automático de manera directa, pues, el lado inteligente de un clasificador automático no pretende simular el comportamiento de un humano, sino que trata de aprovecharse de éste para lograr las mismas conclusiones a través de diversos caminos. Lo anterior lo podemos lograr considerando la tomando en cuenta una serie de documentos de entrenamiento. Dichos documentos fueron preclasificados de manera previa por expertos, de esta forma, el sistema los representará en sus propios términos mediante algunas técnicas que van de la mano de la estadística en la indización, para posteriormente ser utilizados.

3 Definición de objetivos

A continuación, se presentan los objetivos generales y específicos, correspondientes al objeto de estudio.

3.1 Motivación

Como es posible darse cuenta, la organización y/o administración de la información es un trabajo muy difícil de realizar, ya que para esto se requiere la correcta aplicación de la taxonomía, lo que implica, además, el uso de recursos como lo son los humanos, quienes deberán tener los conocimientos necesarios para efectuar dicha tarea.

Con el nacimiento de la clasificación automática de texto se genera un abanico de posibilidades que contemplan desde la automatización para clasificar contenidos, como también herramientas de soporte que impliquen reducir tiempos y mejorar calidad en los procesos para la clasificación manual.

Por consiguiente, la utilización de técnicas y mecanismos automáticos, permitirán refinar la clasificación de contenidos y sus categorías a lo largo del tiempo.

3.2 Objetivos generales

Proponer el factor de peso "TF-PageRank" con el fin de implementarlo en el proceso para la clasificación automática de textos y comparar sus resultados con respecto a otros métodos de ponderación.

3.3 Objetivos específicos

- Investigar la clasificación de textos y las máquinas de aprendizaje.
- Comprender la metodología del cálculo del PageRank.
- Interpretar de manera correcta el funcionamiento de los clasificadores de textos.
- Desarrollar un sistema capaz de clasificar textos de manera automática, utilizando el factor propuesto.
- Analizar los resultados obtenidos por el sistema generado.
- Comparar los resultados obtenidos con los que se obtendrían utilizando algún sistema de clasificación alternativo.
- Concluir con un análisis acerca de la eficiencia y rendimiento del sistema propuesto.

4 Marco Teórico

En este apartado se darán a conocer todos los conceptos utilizados para llevar a cabo la presente investigación de proyecto. Se abordarán en profundidad los conceptos y palabras claves como lo son la inteligencia artificial, máquinas de aprendizaje, la clasificación automática de texto, el factor PageRank y otras técnicas aplicables para la clasificación automática.

4.1 La Inteligencia Artificial

La inteligencia artificial, consiste en el diseño de procesos que, al momento de ejecutarse a través de una máquina o arquitectura física, puede llegar a producir resultados que maximizan una cierta medida de rendimiento**.** Estos procesos están basados en secuencias de entradas de datos percibidos y almacenados por la mencionada arquitectura.

Además, agrupa un conjunto de técnicas que, mediante circuitos electrónicos y programas avanzados de computadora, buscan imitar procedimientos similares a los procesos inductivos y deductivos del cerebro humano. Se basa en la investigación de las redes neuronales humanas y a partir de ahí, busca copiar electrónicamente el funcionamiento del cerebro.

4.2 Las Máquinas de Aprendizaje

Cuando el ser humano adquiere conocimientos, habilidades, actitudes o valores a través del estudio, de la experiencia o la enseñanza, se dice que aprende. Este proceso puede resultar una tarea fácil para el humano, sin embargo, lograr que una máquina aprenda y experimente como lo hacen las personas es una interrogante que existe desde los inicios de las computadoras. Actualmente, no existe una máquina capaz de aprender de la misma manera que lo hace el hombre, sin embargo, se han creado algoritmos eficaces para algunas tareas de aprendizaje.

En términos muy generales, se puede decir, que un programa aprende, si el desempeño obtenido para realizar alguna tarea, mejora con la experiencia.

De manera formal, se dice que un programa aprende de la experiencia E con respecto a una clase de tareas T y una medida de desempeño P, si su desempeño en las tareas T, medido con P, mejora con la experiencia E.

Se puede decir entonces que el Aprendizaje Computacional estudia los procesos computacionales que hay detrás del aprendizaje en humanos y en las máquinas. Esta disciplina juega un papel importante en muchas áreas de la ciencia.

4.3 La Clasificación Automática de Texto

La clasificación de textos surge de la necesidad de separar documentos de un tema o clasificación específica de un conjunto de documentos de diferentes temas. Al lograr clasificar los documentos por temas, la búsqueda de información se puede realizar de manera más sencilla.

Debido al elevado número de documentos que pueden pertenecer a una colección de documentos, sobretodo en formato electrónico, realizar la clasificación en forma manual, provoca que la tarea sea complicada, costosa y que requiera mucho tiempo, por lo que surge la idea de hacerlo automáticamente.

Así es como surge el área de Clasificación Automática de Textos, en la cual se han utilizado diferentes métodos estadísticos y más recientemente técnicas de Aprendizaje Computacional.

El primer paso para realizar la tarea de Clasificación Automática de Textos utilizando técnicas de Aprendizaje Computacional, consiste en obtener los atributos que describan el texto a clasificar, así como transformarlos a una representación adecuada para ser utilizados por los algoritmos de Aprendizaje Computacional. A este paso previo se le llama extracción de características. En la siguiente sección se explica con mayor detalle cómo se realiza la extracción de características en la Clasificación Automática de Textos. Posteriormente se presentan los algoritmos más utilizados en el área de Clasificación Automática de Textos.

4.3.1 Extracción de Características

El proceso de extracción de características consta, generalmente, de tres etapas, las cuales son las siguientes:

- Pre-procesamiento
- Indexado
- Reducción de dimensionalidad

4.3.2 Pre-procesamiento

El pre-procesamiento consiste fundamentalmente en eliminar aquellos elementos que generalmente no contienen información para la tarea de la clasificación. Consta de tres posibles fases básicas:

- Eliminación de etiquetas. Si los documentos utilizados contienen algún tipo de etiquetas o cabeceras (ej. etiquetas de HTML o XML), estás podrán ser removidas, debido a que en algunos casos no proporcionan información útil para la clasificación.

- Eliminación de palabras vacías. Las palabras vacías son palabras que son muy frecuentes y que por lo general no contienen información, por ejemplo: pronombres, preposiciones, conjunciones, artículos, etc.

- Lematización de palabras. Por lematización nos referimos al proceso de remover los sufijos para reducir una palabra a su lema o raíz. Por ejemplo, comprender, comprenderlo y comprendió tienen la raíz comprend.

4.3.3 Indexado

Quizás la representación de documentos más comúnmente usada es la llamada modelo vectorial. En el modelo vectorial, los documentos son representados por vectores de palabras y una colección de documentos son representados por una matriz A (palabra por documento), donde cada entrada representa las ocurrencias de una palabra en un documento, i.e.,

$$A = \begin{pmatrix} a_{11} & \cdots & a_{1m} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nm} \end{pmatrix},$$

Donde a_{ik} es el peso de la palabra i en el documento k.

Existen diversas formas que sirven para calcular el peso a_{ik} de la palabra i en el documento k, pero muchas de las aproximaciones están apoyadas en base a dos observaciones experimentales:

- Entre más ocurre una palabra en un documento, más relevante es en el tema del documento.

- Entre más veces ocurre la palabra en los documentos de la colección, será menos relevante.

Algunas de las formas más comunes de calcular la ponderación de las palabras son a través de los siguientes esquemas:

4.3.3.1 Ponderado Booleano

Este es el esquema más simple y consiste en asignar 1 a a_{ik} si la palabra ocurre en el documento y 0 en otro caso:

$$a_{ik} = \begin{cases} 1 & si \quad f_{ik} > 0 \\ 0 & en \quad otro \quad caso \end{cases}$$

Donde f_{ik} es la frecuencia de la palabra i en el documento k.

4.3.3.2 Ponderado por Frecuencia de Palabra

Otro tipo de esquema sencillo es utilizar la frecuencia de la palabra presente dentro del documento:

$$a_{ik} = f_{ik}$$

Donde f_{ik} es la frecuencia de la palabra i en el documento k.

4.3.3.3 Ponderado TFxIDF

Las metodologías anteriores no consideran la frecuencia de la palabra para todos los documentos que se tengan en la colección. Una de las aproximaciones más utilizadas a la hora de calcular los pesos de los términos es el TFxIDF (Term Frequency x Inverse Document Frequency), el cual tiene como objetivo asignar un peso a una palabra en el documento en proporción al número de ocurrencias de la palabra dentro del documento, y en proporción inversa al número de documentos en la colección en la que la palabra ocurre al menos una vez.

Este factor lo podemos desglosar en los siguientes acrónimos: TF e IDF. El primero de estos es el Term Frequency (TF), o en español, Frecuencia de Aparición de un Término, el cual corresponde a la suma de la totalidad de las ocurrencias, o, dicho de otra forma, es el número de veces en que se repite cierto término en el documento, lo que permite determinar su capacidad de representación. A este tipo de factor de aparición también lo denominamos "Frecuencia de aparición relativa" porque está ligado a un documento en específico y no a la colección completa.

Una vez que el documento ha sido normalizado se puede efectuar el cálculo respectivo. Luego se lleva a cabo el conteo e la cantidad de veces en que el término aparece dentro del texto. Cabe mencionar que, es necesario calcular este factor TF para cada término dentro del documento, tal cual como se representa en la siguiente ecuación:

$$\mathrm{tf}_{(n)} = \sum_{(n)}^{D1}$$

En donde, la frecuencia de aparición de un término (n) en un documento (D1) será la suma de las ocurrencias de dicho término.

El factor IDF de un término es indirectamente proporcional al número de documentos en los que aparece. Esto quiere decir que mientras más pequeña sea la cantidad de documentos, así como también la frecuencia absoluta de las veces en que aparece el término, más alto será el valor del factor IDF, y viceversa, mientras más alta sea la frecuencia absoluta relativa a una alta presencia en la totalidad de los documentos que se tienen en la colección, el valor de su factor discriminatorio tendrá un menor valor.

$$\mathrm{IDF} = \log_{10} \frac{\mathrm{N}}{\mathrm{DF}_{(\mathrm{n})}} + 1$$

- ✓ En donde N es el número total de documentos de la colección.
- ✓ DF (Document Frequency) corresponde al número de documentos en los que aparece el término (n) a lo largo de toda la colección
- ✓ +1 corresponde al factor correctivo
- ✓ Con respecto al $\log_{10}$, también suele utilizarse el logaritmo en base 2, ya que la idea es conseguir un coeficiente bajo para manejarlo de manera más fácil.

Finalmente, con respecto al factor TF-IDF se puede mencionar que el peso de un término dentro de un documento será aquel producto de su frecuencia de aparición en ese documento (TF) y su frecuencia inversa de documento (IDF), lo que podemos apreciar con la siguiente ecuación:

$$\text{TF-IDF}_{(\mathrm{n,d})} = \mathrm{TF}_{(\mathrm{n,d})} \mathrm{x}\, \mathrm{IDF}_{(\mathrm{n})}$$

- ✓ En donde $\text{TF-IDF}_{(\mathrm{n,d})}$ corresponde al peso de un término (n) en un documento (d).
- ✓ $\mathrm{TF}_{(\mathrm{n,d})}$ corresponde a la frecuencia de aparición de un término (n) en un documento (d).
- ✓ $\mathrm{IDF}_{(\mathrm{n})}$ corresponde al factor IDF de un término (n).

Este cálculo corresponde a una medida numérica que expresa cuán relevante es una palabra para un documento en una colección, y opera de la siguiente manera: el valor TF-IDF aumentará proporcionalmente al número de veces en que una palabra aparezca dentro del documento, pero será compensada por el valor de la frecuencia de la palabra en la colección de documentos, lo que va a logra permitir manejar el hecho de que algunas palabras puedan ser generalmente más comunes que otras.

4.3.3.4 Ponderado TF-RFL

Corresponde a la relevancia de la frecuencia de una categoría o etiqueta, la cual viene siendo una representación propuesta por la que constituye una nueva representación para el problema de múltiples categorías. La métrica con la que actúa es la siguiente:

$$tf - rfl_{tdl} = f_{td} \log_2\left(2 + \frac{a_{t,l}}{\max(1, mean(a_{t,\lambda_{j/l}}))}\right)$$

En donde $mean(a_{t,\lambda_{j/l}})$ corresponde al número promedio de documentos que contienen el término *t* para cada documento clasificado en categorías diferentes a *l.*

4.3.4 Reducción de dimensionalidad

Uno de los problemas que se deben abordar durante la clasificación de textos viene siendo la alta dimensionalidad en el espacio de atributos, lo que como consecuencia hace que el procesamiento sea extremadamente costoso, computacionalmente hablando. A partir de esto es que existe la necesidad de reducir el conjunto original de atributos, a este proceso se le llama reducción de dimensionalidad. Existen una serie de mecanismos que sirven para reducir las dimensionalidades, algunos de estos son los que a continuación se describen:

4.3.4.1 Umbral de frecuencia de documento

La frecuencia de documento para un término es el número de documentos en los cuales las palabras aparecen. Dado un cierto umbral de frecuencia de documento, se calcula la frecuencia de documento para cada palabra dentro del conjunto de datos de entrenamiento y se eliminan aquellas palabras en donde la frecuencia de documento sea menor que el umbral determinado. Este método está basado en el supuesto de que las palabras raras generalmente no tienen información para la predicción de categorías.

4.3.4.2 Ganancia de información

La Ganancia de Información consiste en medir el número de bits de información para predecir la categoría por medio de la presencia o ausencia de una palabra en el documento.

Sea $c_1,\ldots, c_k$ el conjunto de posibles categorías. La ganancia de información de una palabra w es definida como:

$$IG(w) = -\sum_{j=1}^{K} P(c_j)\log P(c_j) + P(w)\sum_{j=1}^{K} P(c_j \mid w)\log P(c_j \mid w) + P(\overline{w})\sum_{j=1}^{K} P(c_j \mid \overline{w})\log P(c_j \mid \overline{w})$$

Donde $P(c_j)$ es la probabilidad de la clase c_j (fracción de documentos en la colección que pertenece a la clase c_j) y $P(w)$ es la probabilidad de la palabra (fracción de documentos en los cuales la palabra w ocurre). $P(c_j|\,\overline{w})$ es la probabilidad de la clase dada la palabra (fracción de documentos de clase c_j que tiene al menos una ocurrencia de la palabra w) y $P(c_j|\,\overline{w})$ es la probabilidad de la clase c_j dada la no ocurrencia de la palabra w (fracción de documentos de clase c_j que no contienen la palabra w).

La ganancia de información se calcula para cada palabra del conjunto de entrenamiento, y se eliminan aquellas palabras que tengan menor ganancia de información de acuerdo a un umbral.

4.4 El PageRank

El término PageRank hace mención a una patente que pertenece a la compañía de Google, la cual fue creada e introducida por Larry Page y Sergey Brin (desarrolladores de Google) en el año 1999. Esta patente contempla una serie de algoritmos matemáticos los

cuales son utilizados para ordenar de manera numérica la relevancia de las diversas páginas web que se pueden encontrar. Google evalúa de forma numérica cada uno de los enlaces que direccionan a una página, y la suma de toda esta, entrega el PageRank. La medida va del cero al diez y así se permite conocer la relevancia de una página web.

El PageRank, entonces, es un sistema de valoración que permite medir qué tan relevante puede llegar a ser una página web, considerando los elementos con respecto de la cantidad y la calidad de los enlaces que apunten a la página web.

4.4.1 Funcionamiento general

El PageRank posee un orden, en el cual utiliza un sistema de links como indicador de relevancia de una página web determinada. Explicando brevemente lo anterior, Google interpreta un link de una página web "X" a una página web "Y" como si fuera un voto para la "Y", pero también tiene en cuenta cuál es la página web que da el voto, (también se le denomina "backlink"), lo que quiere decir que los votos de las páginas web con PageRank alto ayudan a destacar en ella otras páginas web. Cada link que apunta a una página web suma una cantidad numérica la cual calcula Google y de esta forma se aplica el PageRank.

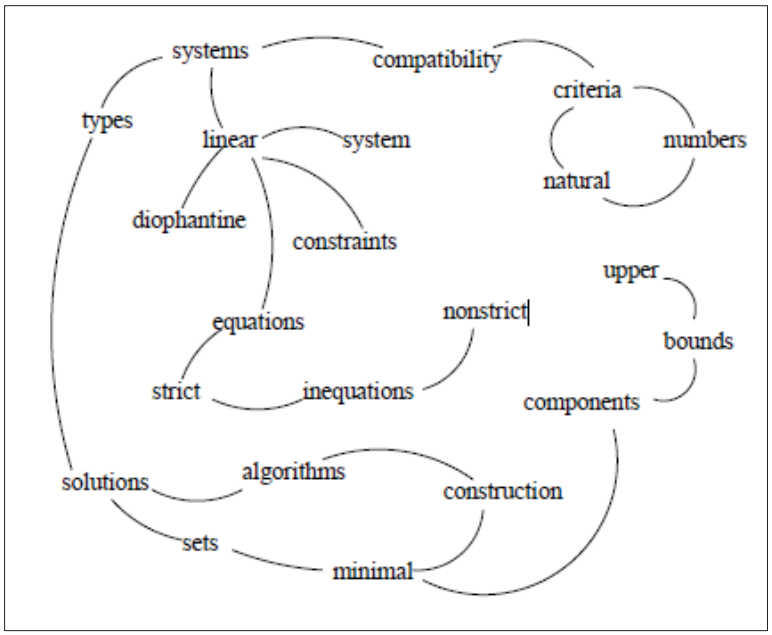

Figura 4.1 Representación de un grafo al aplicar PageRank

Cabe mencionar que no solamente el número de links a una página web influye en el PageRank de esta, sino que se consideran también otro tipo de factores como los que se pueden observar a través de la siguiente imagen:

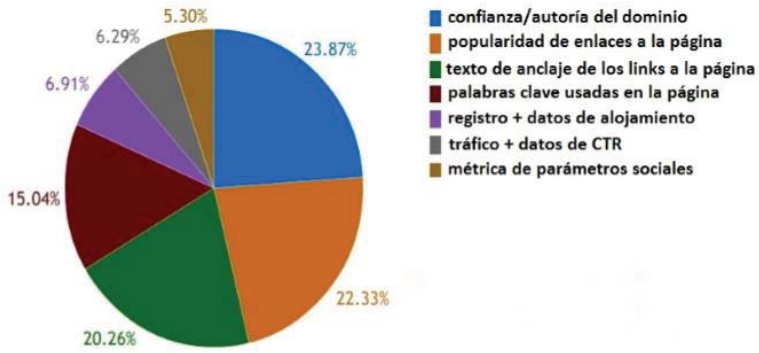

Figura 4.2 Componentes del algoritmo de PageRank

4.4.2 Importancia

La calidad del contenido de una página web es algo que se debe tener en cuenta todo el tiempo cuando se navega en la web. Precisamente, el PageRank ayuda a conocer la relevancia que posee una página web y poder confiar en la calidad de su contenido.

- El PageRank es uno de los factores primordiales del algoritmo de Google, que sirve para que una página salga más arriba en los resultados de búsqueda.
- Google indexa más contenido de una página web si esta tiene PageRank alto.
- Por medio del PageRank se puede conocer la relevancia que tiene para Google una página web.
- La relevancia de cada página web en el PageRank depende de los links que incluye o recibe.

4.4.3 Indexado

La métrica con la que actúa es la siguiente:

$$PR(A) = (1 - d) + d \sum_{i=1}^{n} \frac{PR(i)}{C(i)}$$

En donde:

- *PR(A)* corresponde al PageRank de la página *A*.
- *d* corresponde a un factor de amortiguación que tiene un valor entre 0 y 1.
- *PR(i)* corresponde a los valores que tienen cada una de las páginas *i* que direccionan a *A*.

- *C(i)* corresponde al número total de los enlaces que salen de la página *i* (independientemente de que no vaya hacia *A*).

Para entender de mejor manera su funcionamiento se presentará el siguiente ejemplo.

Se cuenta con 5 sitios web:

- Wikipedia
- Emol
- Youtube
- Facebook
- PUCV

Se asignará un 1 cuando una página esté relacionada con otra, como por ejemplo Wikipedia está siendo llamada desde Youtube por medio de un link. En el caso contrario se asigna 0, en el caso de este ejemplo no existe ninguna relación desde Wikipedia hacia Emol. Además, cuando la página está relacionada con ella misma también se asigna 0. Como por ejemplo Wikipedia está siendo relacionada con Wikipedia, lo que de momento resultaría algo como la siguiente tabla.

4.1 Representación de matriz.

	Wikipedia	Emol	Youtube	Facebook	PUCV
Wikipedia	0		1		
Emol	0				
Youtube					
Facebook					
PUCV					

Se completa la tabla, visualizando la diagonal de ceros que se genera descendentemente, como la que se aprecia a continuación:

4.2 Representación de matriz completada.

	Wikipedia	Emol	Youtube	Facebook	PUCV
Wikipedia	0	0	1	1	1
Emol	0	0	0	0	1
Youtube	1	0	0	1	1
Facebook	1	0	1	0	0
PUCV	1	1	0	0	0

Luego, esta matriz será identificada con la letra M y se suman las columnas, quedando de la siguiente forma:

$$\mathcal{M} = \begin{bmatrix} 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \end{bmatrix} \Rightarrow \mathcal{M} = \begin{bmatrix} 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \end{bmatrix}$$

$$3 \quad 1 \quad 2 \quad 2 \quad 3$$

Se divide cada elemento de la matriz por la suma de la columna, la cual corresponda y de esta manera se obtendrá una matriz que se identifica con la letra M'.

$$\mathcal{M}' = \begin{bmatrix} 0 & 0 & 1/2 & 1/2 & 1/3 \\ 0 & 0 & 0 & 0 & 1/3 \\ 1/3 & 0 & 0 & 1/2 & 1/3 \\ 1/3 & 0 & 1/2 & 0 & 0 \\ 1/3 & 1 & 0 & 0 & 0 \end{bmatrix}$$

El siguiente paso es encontrar los valores de un vector $\boldsymbol{v}$ que represente el PageRank de cada una de las páginas. Pero como no se sabe cuánto vale este vector, solo se muestran cinco valores para cada una de las páginas.

$$\boldsymbol{v} = (a,b,c,d,e)$$

Posteriormente, se asigna una letra a cada página siguiendo el mismo orden en que se presentaron estas páginas:

- Wikipedia = a
- Emol = b
- Youtube = c
- Facebook = d
- PUCV = e

Este problema se llevará a uno de valores propios y vectores propios, en donde $\boldsymbol{v}$ es el vector propio y λ el valor propio el cual es un número que pertenece a los reales:

Se multiplica la matriz M por el vector $\boldsymbol{v}$:

$$M'\boldsymbol{v} = \lambda \mathrm{I}_d \boldsymbol{v}$$

Donde $\lambda \in \mathbb{R}$.

Se suma el inverso aditivo a λ Identidad de la derecha, obteniendo el resultado:

Suma del inverso de $\lambda I_d v$:

$$M'v - \lambda I_d v = \lambda I_d v - \lambda I_d v$$

Extrayendo el factor común:

$$M'v - \lambda I_d v = \lambda I_d v - \lambda I_d v \quad (= 0)$$

Y resulta:

$$(M' - \lambda I_d)v = 0$$

Lo que pertenece a un sistema homogéneo. Por lo tanto, si se reemplazan las matrices y el vector correspondiente en la ecuación, se obtiene:

$$\left(\begin{bmatrix} 0 & 0 & 1/2 & 1/2 & 1/3 \\ 0 & 0 & 0 & 0 & 1/3 \\ 1/3 & 0 & 0 & 1/2 & 1/3 \\ 1/3 & 0 & 1/2 & 0 & 0 \\ 1/3 & 1 & 0 & 0 & 0 \end{bmatrix} - \begin{bmatrix} \lambda & 0 & 0 & 0 & 0 \\ 0 & \lambda & 0 & 0 & 0 \\ 0 & 0 & \lambda & 0 & 0 \\ 0 & 0 & 0 & \lambda & 0 \\ 0 & 0 & 0 & 0 & \lambda \end{bmatrix}\right) \begin{bmatrix} a \\ b \\ c \\ d \\ e \end{bmatrix} = 0$$

Se resta M' a la matriz identidad λ:

$$\begin{bmatrix} -\lambda & 0 & 1/2 & 1/2 & 1/3 \\ 0 & -\lambda & 0 & 0 & 1/3 \\ 1/3 & 0 & -\lambda & 1/2 & 1/3 \\ 1/3 & 0 & 1/2 & -\lambda & 0 \\ 1/3 & 1 & 0 & 0 & -\lambda \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \\ e \end{bmatrix} = 0$$

Luego se determina el valor M' y de λI_d para hallar el valor de λ:

$$|\mathcal{M}' - \lambda I_d| = \begin{vmatrix} -\lambda & 0 & 1/2 & 1/2 & 1/3 \\ 0 & -\lambda & 0 & 0 & 1/3 \\ 1/3 & 0 & -\lambda & 1/2 & 1/3 \\ 1/3 & 0 & 1/2 & -\lambda & 0 \\ 1/3 & 1 & 0 & 0 & -\lambda \end{vmatrix}$$

Se iguala a cero:

$$|\mathcal{M}' - \lambda I_d| = \begin{vmatrix} -\lambda & 0 & 1/2 & 1/2 & 1/3 \\ 0 & -\lambda & 0 & 0 & 1/3 \\ 1/3 & 0 & -\lambda & 1/2 & 1/3 \\ 1/3 & 0 & 1/2 & -\lambda & 0 \\ 1/3 & 1 & 0 & 0 & -\lambda \end{vmatrix} = 0$$

Se obtiene el siguiente polinomio a partir de la operación anterior:

$$\lambda^5 + \frac{37\lambda^3}{36} + \frac{2\lambda^2}{9} + \frac{7\lambda}{36} - \frac{1}{18} = 0$$

De esto podemos encontrar 5 posibles valores para λ:

- $\lambda_1 = 1.$
- $\lambda_2 = -\frac{2}{3}$
- $\lambda_3 = -\frac{1}{2}$
- $\lambda_4 = -\frac{1}{3}$
- $\lambda_5 = \frac{1}{3}$

De estos valores consideramos el que presente mayor valor absoluto, es decir, el $\lambda_1 = 1$:

$$\lambda = Max|\lambda_i| = \lambda_1 = 1$$

Ya que se encontró el valor de **λ**, se reemplaza en el resultado anterior, en donde se obtiene lo siguiente:

$$(M' - \lambda \mathrm{I}_d)$$

$$\begin{bmatrix} -1 & 0 & 1/2 & 1/2 & 1/3 \\ 0 & -1 & 0 & 0 & 1/3 \\ 1/3 & 0 & -1 & 1/2 & 1/3 \\ 1/3 & 0 & 1/2 & -1 & 0 \\ 1/3 & 1 & 0 & 0 & -1 \end{bmatrix}$$

Y se multiplica por el vector v:

$$(M' - \lambda \mathrm{I}_d)v = 0$$

$$\begin{bmatrix} -1 & 0 & 1/2 & 1/2 & 1/3 \\ 0 & -1 & 0 & 0 & 1/3 \\ 1/3 & 0 & -1 & 1/2 & 1/3 \\ 1/3 & 0 & 1/2 & -1 & 0 \\ 1/3 & 1 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \\ e \end{bmatrix} = 0$$

Luego se resuelve:

$$\begin{bmatrix} -a + c/2 + e/3 + d/2 \\ -b + e/3 \\ a/3 - c + e/3 + d/2 \\ a + /3 + c/2 - d \\ a/3 + b - e \end{bmatrix} = 0$$

De esta manera, se ha obtenido un vector que representa cinco incógnitas y cinco ecuaciones igualadas a 0:

$$-a + c/2 + e/3 + d/2 = 0$$
$$-b + e/3 = 0$$
$$a/3 - c + e/3 + d/2 = 0$$
$$a + /3 + c/2 - d = 0$$
$$a/3 + b - e = 0$$

Por lo tanto, las soluciones del sistema serían:

a = 16
b = 1
c = 16/3
d = 14/3
e = 3

Se reemplazan los valores obtenidos en el vector de incógnitas que planteamos al principio:

$$\mathcal{v} = (6, \frac{16}{3}, \frac{14}{3}, 3, 1)$$

Finalmente, los resultados de relevancia serían los siguientes: en primer lugar, se encuentra Wikipedia, segundo lugar Youtube, tercer lugar Facebook, cuarto PUCV y finalmente Emol.

4.5 Técnicas de Aprendizaje

En esta sección se describen algunas técnicas y algoritmos de aprendizaje que se utilizan con frecuencia para el manejo y clasificación automática de información.

4.5.1 Máquina de Vectores de Soporte

Las máquinas de vectores de soporte tienen su origen en los trabajos sobre la teoría del aprendizaje estadístico y fueron introducidas en los años 90 por Vapnik y sus colaboradores. Aunque originalmente las SVMs fueron pensadas para resolver problemas de clasificación binaria, actualmente se utilizan para resolver otros tipos de problemas (regresión, agrupamiento, multiclasificación). También son diversos los campos en los que han sido utilizadas con éxito, tales como visión artificial, reconocimiento de caracteres, categorización de texto e hipertexto, clasificación de proteínas, procesamiento de lenguaje natural, análisis de series temporales, etc. De hecho, desde su introducción, han ido ganando un merecido reconocimiento gracias a sus sólidos fundamentos teóricos.

Dentro de la tarea de clasificación, las SVMs pertenecen a la categoría de los clasificadores lineales, puesto que inducen separadores lineales o hiperplanos, ya sea en el espacio original de los ejemplos de entrada, si estos son separables o cuasi-separables (ruido), o en un espacio transformado (espacio de características), si los ejemplos no son separables linealmente en el espacio original.

Mientras la mayoría de los métodos de aprendizaje se centran en minimizar los errores cometidos por el modelo generado a partir de los ejemplos de entrenamiento (error empírico), el sesgo inductivo asociado a las SVMs radica en la minimización del denominado riesgo estructural. La idea es seleccionar un hiperplano de separación que equidista de los ejemplos más cercanos de cada clase para, de esta forma, conseguir lo que se denomina un margen máximo a cada lado del hiperplano. Además, a la hora de definir el hiperplano, solo se consideran ejemplos de entrenamiento de cada clase que caen justo en la frontera de dichos márgenes. Estos ejemplos reciben el nombre de vectores de soporte. Desde un punto de vista práctico, el hiperplano separador de margen máximo ha demostrado tener una buena capacidad de generalización, evitando en gran medida el problema de sobreajuste a los ejemplos de entrenamiento.

Desde un punto de vista algorítmico, el problema de optimización del margen geométrico representa un problema de optimización cuadrático con restricciones lineales que puede ser resuelto mediante técnicas estándar de programación cuadrática. La propiedad de convexidad exigida para su resolución garantiza una solución única, en contraste con la no unicidad de la solución producida por una red neuronal artificial entrenada con un mismo conjunto de ejemplos.

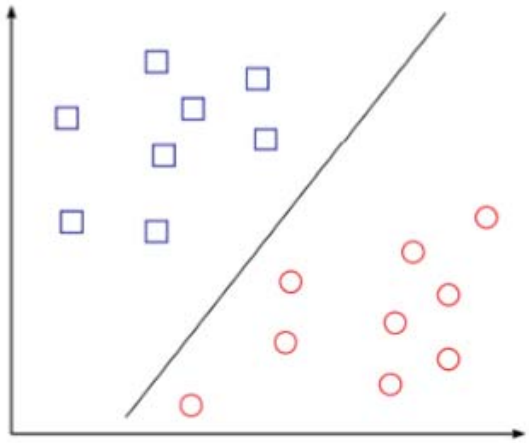

Figura 4.3 Ejemplo de hiperplano de separación.

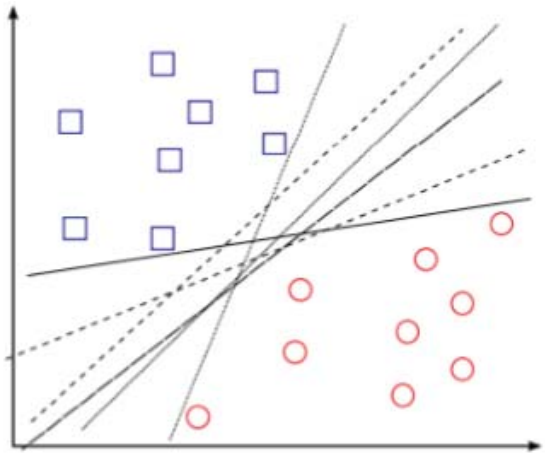

Figura 4.4 Ejemplo de hiperplano de separación, de entre los infinitos posibles.

4.5.2 Árboles de Decisión

Los árboles de decisión (también llamados de clasificación o de identificación) constituyen una aproximación radicalmente distinta a todas las estudiadas hasta el momento. Es uno de los métodos de aprendizaje inductivo supervisado no paramétrico más utilizado. Como forma de representación del conocimiento, los arboles de clasificación destacan por su sencillez. A pesar de que carecen de la expresividad de las redes semánticas o de la lógica de primer orden, su dominio de aplicación no está restringido a un ámbito concreto, sino que pueden utilizarse en diversas áreas: diagnóstico médico, juegos, predicción meteorológica, control de calidad, etc.

Un árbol de decisión es una forma de representar el conocimiento obtenido en el proceso de aprendizaje inductivo. Puede verse como la estructura resultante de la partición recursiva del espacio de representación a partir del conjunto (numeroso) de prototipos. Esta partición recursiva se traduce en una organización jerárquica del espacio de representación que puede modelarse mediante una estructura de tipo árbol. Cada nodo interior contiene una pregunta sobre un atributo concreto (con un hijo por cada posible respuesta) y cada nodo hoja se refiera a una decisión (clasificación).

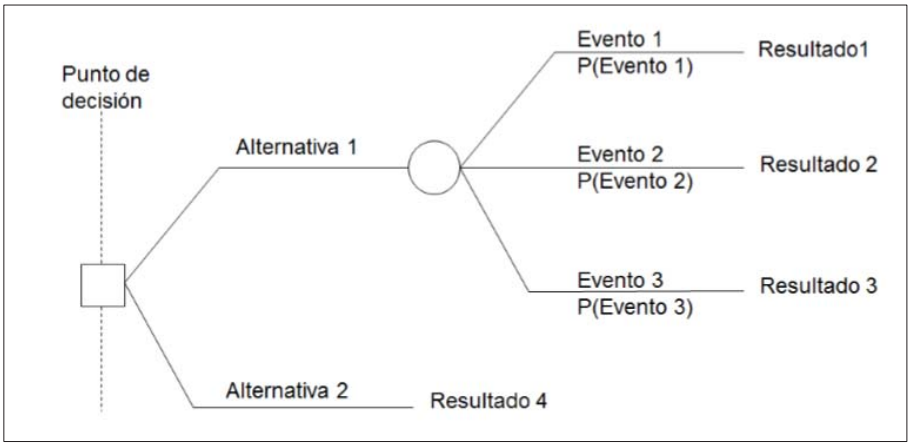

Figura 4.5 Ejemplo de árbol de decisión.

La clasificación de patrones se realiza en base a una serie de preguntas sobre los valores de sus atributos, empezando por el nodo raíz y siguiendo el camino determinado por las respuestas a las preguntas de los nodos internos, hasta llegar a un nodo hoja. La etiqueta asignada a esta hoja es la que se asignará al patrón a clasificar.

La metodología a seguir para la construcción del árbol de decisión, puede resumirse en dos pasos:

1. Aprendizaje: Consiste en la construcción del árbol a partir de un conjunto de prototipos. Constituye la fase más compleja y la que determina el resultado final.

2. Clasificación: Consiste en el etiquetado de un patrón, independiente del conjunto de aprendizaje. Se trata de responder a las preguntas asociadas a los nodos interiores utilizando los valores de los atributos del patrón. Este proceso se repite desde el nodo raíz hasta alcanzar una hoja, siguiendo el camino impuesto por el resultado de cada evaluación

4.5.3 K Vecinos más Cercanos

El método KNN es un método de aprendizaje inductivo supervisado que sirve para estimar la función de densidad F(x / C_j) de las predictoras x por cada clase C_j. Este es un método de clasificación no paramétrico, que estima el valor de la función de densidad de probabilidad o directamente la probabilidad a posteriori de que un elemento x pertenezca a la clase C_j a partir de la información proporcionada por el conjunto de prototipos. En el proceso de aprendizaje no se hace ninguna suposición acerca de la distribución de las variables predictoras.

En el reconocimiento de patrones, el algoritmo k-nn es usado como método de clasificación de objetos (elementos) basado en un entrenamiento mediante ejemplos cercanos en el espacio de los elementos. K-nn es un tipo de "Lazy Learning", en donde la función se aproxima solo localmente y todo cómputo es diferido a la clasificación.

Los ejemplos de entrenamiento son vectores en un espacio característico multidimensional. Cada ejemplo esta descrito en términos de 'p' atributos considerando 'q' clases para la clasificación. Los valores de los atributos del i-ésimo ejemplo (donde $1 \leq i \leq n$) se representan por el vector p dimensional.

$$x = \left(x_{1i}, x_{2i}, \ldots, x_{pi}\right) \in X$$

El espacio es particionado en regiones por localizaciones y etiquetas de los ejemplos de entrenamiento. Un punto en el espacio es asignado a la clase C si está es la clase más frecuente entre los 'k' ejemplos de entrenamiento más cercano. Generalmente se usa la distancia euclideana:

$$d(x_i, x_j) = \sqrt{\sum_{r=1}^{p} (x_{ir} - x_{jr})^2}$$

La fase de entrenamiento del algoritmo consiste en almacenar los vectores característicos y las etiquetas de las clases de los ejemplos de entrenamiento. En la fase de clasificación, la evaluación del ejemplo (del que no se conoce su clase) es representada por un vector en el espacio característico. Se calcula la distancia entre los vectores almacenados y el nuevo vector, y se seleccionan los 'k' ejemplos más cercanos. El nuevo ejemplo es clasificado con la clase que más se repite en los vectores seleccionados.

Este método supone que los vecinos más cercanos nos dan la mejor clasificación y esto se hace utilizando todos los atributos; el problema de dicha suposición es que es posible que se tengan muchos atributos irrelevantes que dominen sobre la clasificación: dos atributos relevantes perderían peso entre otros veinte irrelevantes.

Para corregir el sesgo que se puede generar se puede asignar un peso a las distancias de cada atributo, dándole así mayor importancia a los atributos más relevantes. Otra posibilidad consiste en tratar de determinar o ajustar los pesos con ejemplos conocidos de entrenamiento. Finalmente, antes de asignar pesos se recomienda identificar y eliminar los atributos que se consideran irrelevantes.

4.5.4 Modelo Probabilístico de Naive Bayes

El clasificador Naive Bayes se construye usando el conjunto de entrenamiento para estimar la probabilidad de cada clase dados los valores de atributos (palabras) del documento de una nueva instancia. Usamos el Teorema de Bayes para estimar las probabilidades:

$$P(c_j \mid d) = \frac{P(c_j)P(d \mid c_j)}{P(d)}$$

El denominador en la ecuación anterior no distingue entre categorías y puede ser eliminado. Este método asume que los atributos son condicionalmente independientes, dada la clase. Esto simplifica los cálculos.

$$P(c_j \mid d) = P(c_j)\prod_{i=1}^{M} P(d_i \mid c_j)$$

Una estimación $\hat{P}(c_j)$ para $P(c_j)$ puede ser calculada de la fracción de documentos de entrenamiento que es asignada a la clase c_j:

$$\hat{P}(c_j) = \frac{N_j}{N}$$

Donde N_j es el número de documentos de entrenamiento para los cuales la clase es c_j y N es el número total de documentos de entrenamiento.

Una estimación $\hat{P}(d_i \mid c_j)$ para $P(d \mid c_j)$ está dada por:

$$\hat{P}(d_i \mid c_j) = \frac{1 + N_{ij}}{M + \sum_{k=1}^{M} N_{kj}}$$

Donde N_{ij} es el número de veces de la palabra i ocurrida dentro de los documentos de la clase c_j en el conjunto de entrenamiento. Para evitar el problema de la probabilidad cero se utiliza Laplace (agregar un 1). M es el número de términos en el vocabulario.

A pesar de que la suposición de independencia condicional es generalmente falsa para la aparición de la palabra en documentos, el clasificador Naive Bayes es sorprendentemente efectivo.

4.5.5 Algoritmo de Rocchio

Es un algoritmo perteneciente a la clasificación supervisada, el cual se basa en construir vectores que traten de representar cada clase a partir de documentos de entrenamiento. Para el vector de cada clase, los documentos de entrenamiento para esa clase se utilizan como ejemplos positivos, en cambio, los documentos de entrenamiento de las demás clases se usan como ejemplos negativos.

El vector representativo de una clase se construye sumando los pesos de los términos de los ejemplos positivos. De éste, se restan los pesos de los términos de los ejemplos negativos. Al aplicar coeficientes multiplicadores, es posible dar más o menos importancia a los ejemplos positivos o a los negativos.

El resultado es un vector de términos con pesos como el utilizado en el modelo vectorial. Ahora, para clasificar un nuevo documento, no hay más que estimar la similitud entre el vector de ese documento y los vectores de cada una de las clases.

4.5.6 Redes Neuronales

Las redes neuronales artificiales o RNA surgen ante la necesidad de utilizar sistemas que operasen como el cerebro humano, ya que hasta ahora se había conseguido métodos capaces de superar al cerebro humano en cálculos que se consideraban muy difíciles para él, pero en cambio no se podían realizar tareas que el cerebro ejecuta de forma simple a todas horas, como por ejemplo reconocer a una persona o saber discriminar si una persona se ha roto una pierna en base al resultado de una exploración o de una prueba.

Las características más importantes del funcionamiento del cerebro como sistema de computación son las siguientes:

- ✓ es robusto
- ✓ es flexible, se adapta al entorno
- ✓ puede tratar información ambigua o incompleta
- ✓ es pequeño, compacto y consume poca potencia

Las redes neuronales pretenden conseguir todas estas características.

Como el comportamiento del cerebro se debe a la interacción de millones de células nerviosas, las RNA pretenden desarrollar un equivalente algorítmico de los procesos de reconocimiento y aprendizaje.

De tal manera, podríamos definir la red neuronal artificial como una implementación, en hardware o software, de un sistema de procesamiento de datos que intenta emular las funciones computacionales elementales de la red nerviosa del cerebro humano. Su propiedad esencial es que implementan un nuevo paradigma de computación muy útil en problemas que no se adecuan bien a las estructuras convencionales de cálculo. Mediante este procedimiento las redes neuronales pueden extraer información estructural de masas de datos complicados o imprecisos, que sinteticen una descripción del fenómeno que ha generado estos datos, es decir, simulan las funciones de un experto en el tema, capaz de enfrentarse con cierto grado de eficacia a situaciones nuevas.

De tal manera, las RNA pretenden imitar la estructura y funcionamiento del cerebro humano con el fin de resolver problemas prácticos mediante la construcción de sistemas de procesamiento de la información paralelos (sobre diferentes sectores a la vez), distribuidos (de modo que si una zona se pierde no cae toda la red) y adaptativos (aprenden de la experiencia pudiendo generalizar conceptos a partir de casos particulares) que puedan representar un cierto comportamiento inteligente.

Las relaciones que se establecen entre los componentes que describen y definen la información son muy importantes para poder seleccionar posteriormente, de forma inteligente la información que contiene la base de conocimiento.

La clasificación automática de la información, y el modo en que esta es recuperada son los campos donde más se han aplicado las RNA.

El fin último de las aplicaciones descritas a continuación es desarrollar un modelo de representación del conocimiento que se asemeje a la forma en la que opera la mente humana, potenciando el sistema de browsing y posibilitando la interactividad y las relaciones multidimensionales en la recuperación de la información.

4.5.7 Random Forest

Uno de los métodos más populares usados por los científicos de datos es el algoritmo Random Forest, uno de los mejores algoritmos de clasificación, capaz de organizar grandes cantidades de datos con exactitud.

Es una combinación de árboles predictores en la que cada árbol depende de los valores de un vector aleatorio probado independientemente y con la misma distribución para cada uno de estos. El algoritmo para inducir un random forest fue desarrollado por Leo Breiman y Adele Cutler, siendo Random forest su marca de fábrica.

Este algoritmo mejora la precisión en la clasificación mediante la incorporación de aleatoriedad en la construcción de cada clasificador individual. Esta aleatorización puede introducirse en la partición del espacio (construcción del árbol), así como en la muestra de entrenamiento. El Random Forest comienza con una técnica de aprendizaje automático estándar llamada "árbol de decisiones", que, en cuanto al conjunto, corresponde a un aprendizaje. En un árbol de decisión, una entrada se introduce en la parte superior y hacia abajo a medida que atraviesa el árbol de los datos, los cuales se acumulan en conjuntos más pequeños.

El Método Random Forest es fácil de aprender y de usar, tanto por profesionales como por personas con menos experiencia. Con poca investigación y tiempo de desarrollo, puede ser utilizado por personas con poca base estadística. En pocas palabras, este método nos permite hacer de manera segura predicciones más precisas y sin la mayoría de los errores básicos comunes a otros métodos.

5 Metodología

En este apartado, se describe de manera cronológica el procedimiento que se ha llevado a cabo para desarrollar e implementar la solución.

- Buscar, leer y analizar documentos, estudios o investigaciones que hagan referencia a la clasificación automática de texto.

- Investigación en profundidad del PageRank.

- Investigación de los recursos y herramientas que son más útiles para trabajar con el análisis y procesamiento de texto.

- Selección del set de datos para las etapas de entrenamiento y de prueba (train/test).

- Codificación del software utilizando las herramientas que se investigaron.

- Fase experimental, entrenar el software y luego aplicar las pruebas respectivas para comprobar su desempeño.

- Analizar los resultados obtenidos y compararlos con algún sistema de clasificación alternativo.

6 Desarrollo de la Solución

A continuación, se describen los puntos más característicos y relevantes que se consideraron para poner en marcha esta solución.

6.1 Software

Para dar solución a esta problemática se ha pensado en construir un programa computacional desarrollado en la última versión de Python, (la cual corresponde a la 3.6), que permita clasificar artículos de manera automática, utilizando todas las riquezas que este lenguaje ofrece en lo que a librerías se refiere. A continuación, se describen los recursos más importantes, como por ejemplo las librerías, que serán necesarios para llevar a cabo la implementación de la solución.

6.1.1 Procesamiento de Lenguaje Natural

Es un campo de las ciencias de la computación, inteligencia artificial y lingüística que estudia las interacciones entre las computadoras y el lenguaje humano. El PLN se ocupa de la formulación e investigación de mecanismos eficaces computacionalmente para la comunicación entre personas y máquinas por medio de lenguajes naturales. El PLN no trata de la comunicación por medio de lenguajes naturales de una forma abstracta, sino de diseñar mecanismos para comunicarse que sean eficaces computacionalmente que se puedan realizar por medio de programas que ejecuten o simulen la comunicación.

Los modelos aplicados se enfocan no solo a la comprensión del lenguaje de por sí, sino a aspectos generales cognitivos humanos y a la organización de la memoria. El lenguaje natural sirve solo de medio para estudiar estos fenómenos. Hasta la década de 1980, la mayoría de los sistemas de PLN se basaban en un complejo conjunto de reglas diseñadas a mano. A partir de finales de 1980, sin embargo, hubo una revolución en PLN con la introducción de algoritmos de aprendizaje automático para el procesamiento del lenguaje.

6.1.2 Librería NLTK

El kit de herramientas de lenguaje natural, o más comúnmente NLTK, es un conjunto de bibliotecas y programas para el procesamiento del lenguaje natural (PLN) simbólico y estadísticos para el lenguaje de programación Python. NLTK incluye demostraciones gráficas y datos de muestra. Se acompaña de un libro que explica los conceptos subyacentes a las tareas de procesamiento del lenguaje compatibles al toolkit, además de programas de ejemplo.

NLTK está destinado a apoyar la investigación y la enseñanza en PLN o áreas muy relacionadas, que incluyen la lingüística empírica, las ciencias cognitivas, la inteligencia artificial, la recuperación de información, y el aprendizaje de la máquina. NLTK se ha utilizado con éxito como herramienta de enseñanza, como una herramienta de estudio

individual, y como plataforma para los sistemas de investigación de prototipos y construcción.

6.1.3 Librería Numpy

Uno de los módulos más importantes de Python es Numpy. El origen de Numpy se debe principalmente al diseñador de software Jim Hugunin quien diseñó el módulo Numeric para dotar a Python de capacidades de cálculo similares a las de otros softwares como MATLAB. Posteriormente, mejoró Numeric incorporando nuevas funcionalidades naciendo lo que hoy conocemos como Numpy.

Numpy es el encargado de añadir toda la capacidad matemática y vectorial a Python haciendo posible operar con cualquier dato numérico o array (posteriormente veremos qué es un array). Incorpora operaciones tan básicas como la suma o la multiplicación u otras mucho más complejas como la transformada de Fourier o el álgebra lineal. Además, incorpora herramientas que nos permiten incorporar código fuente de otros lenguajes de programación como C/C++ o Fortran lo que incrementa notablemente su compatibilidad e implementación.

6.1.4 Librería Pandas

Pandas es una librería de python destinada al análisis de datos, que proporciona unas estructuras de datos flexibles y que permiten trabajar con ellos de forma muy eficiente. Dentro de lo que puede brindar, ofrece las siguientes estructuras de datos:

- Series: Son arrays unidimensionales con indexación (arrays con índice o etiquetados), similar a los diccionarios. Pueden generarse a partir de diccionarios o de listas.

- DataFrame: Son estructuras de datos similares a las tablas de bases de datos relacionales como SQL.

- Panel, Panel4D y PanelND: Estas estructuras de datos permiten trabajar con más de dos dimensiones.

6.1.5 Librería Scikit-Learn

Es una biblioteca de aprendizaje de máquina de software libre para el lenguaje de programación de Python. Cuenta con varios algoritmos de clasificación, regresión y clustering, incluyendo máquinas de vectores de soporte, random forest, k-means, etc., y está diseñado para operar con las bibliotecas numéricas y científicas de Python, NumPy y SciPy.

6.1.6 Librería SciPy

Es una biblioteca de Python gratuita y de código abierto utilizada para las ciencias de la computación y la informática.

SciPy contiene módulos para optimización, álgebra lineal, integración, interpolación, funciones especiales, procesamiento de señal e imagen, solucionadores de ODE y otras tareas comunes en ciencia e ingeniería.

6.2 Selección del Data Set

Para el desarrollo del proyecto, se seleccionó el conjunto de datos de "20 News Group", que comprende alrededor de 18.000 publicaciones de grupos de noticias sobre 20 temas divididos en dos subconjuntos: uno para el entrenamiento y el otro para pruebas o test (o para evaluación de desempeño). La división entre el train y el conjunto de test se basa en los mensajes publicados antes y después de una fecha específica.

A continuación, se presentan las categorías que están contenidas en el conjunto de datos:

1. comp.graphics
2. comp.os.ms-windows.misc
3. comp.sys.ibm.pc.hardware
4. comp.sys.mac.hardware
5. comp.windows.x
6. misc.forsale
7. rec.autos
8. rec.motorcycles
9. rec.sport.baseball
10. rec.sport.hockey
11. talk.politics.misc
12. talk.politics.guns
13. talk.politics.mideast
14. talk.religion.misc
15. alt.atheism
16. soc.religion.christian
17. sci.crypt
18. sci.electronics
19. sci.med
20. sci.space

6.3 Reducción de Palabras Vacías

Palabras vacías o Stop Words se les llama a aquellas palabras sin significado como artículos, pronombres, preposiciones, etc. que son filtradas antes o después del procesamiento de datos en lenguaje natural (texto). A Hans Peter Luhn, uno de los pioneros en recuperación de información, se le atribuye la acuñación de la locución inglesa stop words y el uso del concepto en su diseño. Está controlada por introducción humana y no automática.

No hay una lista definitiva de palabras vacías que todas las herramientas de procesamiento de lenguajes naturales incorporen. No todas las herramientas de PLN usan una lista de palabras vacías. Algunas herramientas evitan usarlo específicamente para soportar búsquedas por frase. El uso de un algoritmo de stemming puede reducir parte de la base lógica o dependencia de una lista de palabras vacías a filtrar.

6.4 Extracción de Frases o Palabras Claves

La extracción de frases claves (*keyphrases*) consiste en a partir de un pedazo de texto, como el caso de un artículo de una revista, producir una lista de *keywords* o keyphrases que capturan los temas principales discutidos en el texto. En el caso de artículos de investigación, muchos autores proveen keywords asignados manualmente, pero la mayoría de textos no contienen keyphrases preexistentes.

En el caso extractor de keyphrases con un enfoque extractivo se seleccionarían ciertas frases como keyphrases, las cuales son "jaladas" directamente desde el texto. En contraste, un sistema de keyphrases con enfoque abstractivo de alguna manera internaliza el contenido y genera keyphrases que pueden ser más descriptivas y más parecidas a lo que una persona produciría. Puede ser que estos términos no aparezcan en el texto y requieran un entendimiento profundo, lo que hace bastante difícil que una computadora produzca keyphrases de este tipo. Las keyphrases tienen muchas aplicaciones, como lo son las siguientes:

- Mejorar la búsqueda de documentos al proveer un resumen corto.
- Mejorar la recuperación de información, debido a que, si los documentos tienen keyphrases asignadas, un usuario puede buscar por medio de las keyphrases para producir más hits o resultados confiables que una búsqueda de texto completo.
- La extracción de keyphrases automáticos puede ser útil en generar entradas de índice para un largo corpus de texto.

6.5 Construcción de Grafos Conceptuales

Los grafos conceptuales son estructuras que se utilizan frecuentemente para la representación del conocimiento basado en lógica. Siguiendo con lo anterior, los textos se representan de forma natural, simple y con una representación semántica detallada. En estas estructuras, existen dos tipos de nodos: las relaciones conceptuales, las cuales se generalmente se representan por óvalos, y los conceptos, los que están representados por rectángulos. Un concepto se conecta a otro concepto a través de una relación conceptual, y una relación conceptual debe conectarse estrictamente con algún concepto.

6.5.1 Aplicación del Factor PageRank

La idea principal es aplicar generar un algoritmo que represente a un documento a través de la conexión de un grafo, lo mencionado previamente, en el cual los nodos corresponderán a oraciones, palabras u otro tipo de unidad, y las aristas serán los enlaces que se generen de acuerdo al contenido entre las unidades, o más bien dicho, las relaciones sintácticas. Entonces, considerando lo anterior, se utilizará el algoritmo de ponderación denominado PageRank para identificar la importancia de un nodo, en donde aquellos con mayor ponderación se considerarán más relevantes y pasarán a ser parte de los nodos o palabras claves del documento a clasificar, de esta forma se podrá decir que el grafo estará

“ponderado” o “pesado”. Cabe mencionar que el factor PageRank es un valor que oscila entre 0 y 1, en donde, mientras más cerca se esté del 1, más importancia tendrá la palabra.

6.5.2 Grafos Conceptuales Ponderados

Los grafos conceptuales ponderados son Grafos Conceptuales, en donde se asocian pesos a las aristas que conectan a los nodos creando un flujo denominado ‘flujo semántico’, en este caso, el tipo de pesaje corresponde al del factor PageRank. Un flujo semántico es básicamente el peso que acumulan los nodos y que se transmite hacia otros nodos aumentando o disminuyendo su valor al pasar por alguna relación conceptual.

En este contexto, las relaciones conceptuales representan principalmente la semántica del texto, relaciones tales como agente, objeto, lugar, atributo, tema, etc. De ahí que un peso con valor alto en la arista indica interés por el flujo de la relación en cuestión. Además, en este tipo de grafos, los nodos entregan un peso que define su preferencia, es decir, el grado de interés o importancia de ciertas temáticas definidas por los conceptos.

6.6 Cálculo del TF-PageRank

Una vez calculado el PageRank para cada término, se procede a calcular el factor TF (señalado en el punto 4.3.3.2) para cada palabra, para así después combinar este factor con el PageRank y generar el factor propuesto a través de la multiplicación de ambos, denominado TF-PageRank (TF-PR). Esto generará un vector de pesos para cada documento procesado, conteniendo el peso de las palabras más importantes.

6.7 Fase de Entrenamiento – train

El set de datos “20 News Group” se divide en dos carpetas, la primera es para la fase de entrenamiento (carpeta “train”) y la segunda para la fase de pruebas (carpeta “test”), ambas con las mismas 20 clases agrupadas en subcarpetas. Cabe mencionar que la data se acotó a 10.000 archivos, por lo tanto, de la carpeta “train” se procesan 8.000 archivos del conjunto de datos, lo que corresponderá al 80% del total de la data, distribuidos de igual cantidad para cada una de las clases.

6.8 Fase de Pruebas – test

Esta etapa consiste en procesar 100 archivos por cada clase, es decir, se procesan 2.000 archivos, lo que corresponde al 20% del conjunto de datos. La idea es generar un vector con las palabras más importantes para cada uno de ellos, además de un vector con su respectivo peso TF-PageRank. Al momento de generar un vector de palabras y de pesos por cada archivo, se crean dos documentos de extensión “txt”, el primero contendrá el total de palabras importantes para esa clase, y el segundo el total de pesos, esto vendría siendo una concatenación de todos los archivos por categoría, tanto para palabras como también para sus pesos.

6.9 Clasificación de los Documentos - Algoritmo “Naive Bayes”

Hasta esta etapa, ya se han procesado los archivos para el train y el test, por lo tanto, lo que ahora resta es utilizar algún clasificador automático de texto para que asigne una categoría a los documentos de la etapa de test. Para esto, se escogió el algoritmo que utiliza el clasificador automático “Naive Bayes” o “Bayesiano”, el cual, combinándolo con el “TF-PR” nos entregará una predicción de lo que sería la clasificación de los archivos de testing. Finalmente, para obtener los resultados y conocer el rendimiento del algoritmo, se le envían dos parámetros al clasificador, el primero vendría siendo el “*y_true*”, el que corresponde a un vector de largo dos mil, con las etiquetas clasificadas de manera correcta para cada archivo; y el segundo es el vector “*y_pred*” que vendría siendo la predicción mencionada anteriormente, del mismo largo.

7 Resultados

En este apartado se describen los resultados obtenidos en el proceso de clasificación. Cabe mencionar que para obtener las métricas que se mostrarán a continuación, se utilizaron módulos de la librería Scikit-Learn.

7.1 Modelo TF-PR & Naive Bayes

En primer lugar, se presenta el rendimiento obtenido del modelo propuesto.

7.1.1 Resultados

A continuación, se muestra el rendimiento que se obtuvo utilizando el factor de pesaje propuesto TF-PR con el clasificador Bayesiano:

Accuracy : 68.0 %

	precision	recall	f1-score	support
alt.atheism	0.78	0.79	0.79	100
comp.graphics	0.51	0.76	0.61	100
comp.os.ms-windows.misc	1.00	0.19	0.32	100
comp.sys.ibm.pc.hardware	0.49	0.57	0.53	100
comp.sys.mac.hardware	0.46	0.57	0.51	100
comp.windows.x	0.81	0.57	0.67	100
misc.forsale	0.30	0.93	0.46	100
rec.autos	0.79	0.81	0.80	100
rec.motorcycles	0.74	0.90	0.81	100
rec.sport.baseball	0.86	0.79	0.82	100
rec.sport.hockey	0.94	0.79	0.86	100
sci.crypt	0.94	0.82	0.88	100
sci.electronics	0.67	0.54	0.60	100
sci.med	0.91	0.67	0.77	100
sci.space	0.82	0.83	0.83	100
soc.religion.christian	0.98	0.64	0.78	100
talk.politics.guns	0.78	0.72	0.75	100
talk.politics.mideast	1.00	0.60	0.75	100
talk.politics.misc	0.75	0.50	0.60	100
talk.religion.misc	0.72	0.61	0.66	100
avg / total	0.76	0.68	0.69	2000

Figura 7.1 Rendimiento TF-PR & Naive Bayes.

- ✓ Precision: representa la habilidad del clasificador de no etiquetar como positiva una muestra que es negativa. En este caso, podemos ver que el promedio de este indicador es de un 76%.

- ✓ Recall: representa la habilidad del clasificador para encontrar todas las muestras positivas, en otras palabras, cuántos fueron positivos de la predicción obtenida. En este caso, podemos ver que el promedio de este indicador es de un 68%.

- ✓ F1-score: se puede interpretar como un promedio ponderado entre los valores de precision y recall, en donde el F1-score alcanzará su mejor valor en 1 y el peor puntaje en 0. En este caso, podemos ver que el promedio de este indicador es de un 69%.

- ✓ Support: representa el número de ocurrencias de cada clase en y_true. En este caso, y como se mencionó anteriormente, para efectos del test, se consideraron 100 archivos de cada categoría.

- ✓ Accuracy: esta función calcula la precisión del subconjunto, en donde, el conjunto de etiquetas predicho para una muestra debe coincidir exactamente con el conjunto correspondiente de etiquetas en y_true, en este caso, podemos ver que el indicador arroja un resultado de un 68%.

La matriz de confusión asociada es la siguiente:

```
Confusion Matrix :
 [[79  0  0  0  1  0  6  0  3  1  0  1  0  1  3  0  0  0  0  5]
 [ 0 76  0  1  3  5 13  1  0  1  0  0  0  0  0  0  0  0  0  0]
 [ 0 12 19 35 13  5 10  1  0  0  0  0  3  0  2  0  0  0  0  0]
 [ 0  5  0 57 17  0 15  0  0  0  0  0  6  0  0  0  0  0  0  0]
 [ 0  2  0  7 57  0 26  0  1  1  0  0  5  0  1  0  0  0  0  0]
 [ 0 28  0  2  7 57  5  1  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  3  2  0 93  2  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  1  0  1  1  0 11 81  3  0  0  0  1  0  1  0  0  0  0  0]
 [ 0  1  0  0  0  0  4  5 90  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  1  0 13  0  2 79  5  0  0  0  0  0  0  0  0  0]
 [ 0  1  0  1  1  1 10  0  1  4 79  0  2  0  0  0  0  0  0  0]
 [ 0  3  0  0  2  0  9  0  2  0  0 82  0  0  1  0  1  0  0  0]
 [ 0  7  0  5  9  0 23  1  0  0  0  0 54  0  1  0  0  0  0  0]
 [ 0  4  0  1  3  1 14  1  4  0  0  0  3 67  0  0  0  0  2  0]
 [ 1  3  0  0  1  0  5  2  2  0  0  0  3  0 83  0  0  0  0  0]
 [ 8  2  0  0  1  0  4  0  1  1  0  0  0  2  3 64  1  0  2 11]
 [ 1  0  0  0  0  1  9  1  6  1  0  3  2  0  0  0 72  0  1  3]
 [ 2  1  0  1  5  0 11  1  2  2  0  0  0  0  0  0  1 60 11  3]
 [ 1  2  0  0  1  0 13  6  4  2  0  1  1  0  2  0 15  0 50  2]
 [ 9  2  0  2  0  0 12  0  1  0  0  0  1  4  4  1  2  0  1 61]]
```

Figura 7.2 Matriz de Confusión TF-PR & Naive Bayes.

7.1.2 50 Top Words TF-PR

A continuación, se presenta un top 50 de las palabras más importantes que se obtuvieron en la fase de test, con su respectivo peso:

Tabla 7.1 Top Words TF-PR & Naive Bayes.

TF - PR			
Feature	**Weight**	**Feature**	**Weight**
homosexuality	15.782	universe	6.954
bill	15.524	gun	6.663
sleep	15.209	zinaida	6.546
period	12.432	vitamin	6.441
cancer	11.526	evidence	6.048
deal	10.606	care	5.543
channel	10.238	board	5.525
sin	10.196	address	5.512
evolution	9.843	seat	5.416
game	9.439	fire	5.396
gun	9.373	moment	5.248
law	9.344	drive	5.152
chip	9.026	bus	5.121
exhibit	8.976	cassette	5.115
modem	8.076	locking	4.674
mode	7.824	moment	4.544
noise	7.797	printer	4.488
huebner	7.688	display	4.233
memory	7.495	daruwala	4.064
color	7.482	fool	4.045
bus	7.221	peace	3.269
bus	7.221	network	3.195
character	7.128	blah	3.132
game	7.097	government	2.926
price	7.088	drive	2.358

Como se puede apreciar, el peso de las palabras fluctúa entre el [0,+∞[, alcanzando su máximo valor con la palabra “homosexuality”, con un factor de peso de un 15.782.

7.1.2 50 Top Words PageRank

A continuación, se presenta un top 50 de las palabras más importantes que se obtuvieron en la fase de test, con su respectivo peso PageRank:

Tabla 7.2 Top Words PageRank.

PageRank			
Feature	**Weight**	**Feature**	**Weight**
format	0.361	card	0.236
huebner	0.3422	drive	0.2358
prism.gatech.edu	0.3216	board	0.2305
sleep	0.3019	mode	0.2304
keyboard	0.296	bit	0.2288
cs.indiana.edu	0.2952	howlin.cs.unlv.edu	0.2251
checking	0.2952	jcs	0.2245
isu	0.2952	encryption	0.2232
address	0.2878	sdcc15.ucsd.edu	0.2229
staff.tc.umn.edu	0.2771	zapotec.math.byu.edu	0.2229
foundation	0.2771	cassette	0.2223
prism.gatech.edu	0.275	robert	0.2222
texture	0.272	power	0.2222
memory	0.2699	sctc.com	0.2217
printer	0.265	world.std.com	0.2217
test	0.2639	usc.edu	0.2217
board	0.2619	software	0.2196
font	0.2559	encryption	0.2194
geds01.jsc.nasa.gov	0.2559	aludra.usc.edu	0.2184
daruwala	0.2516	lisa	0.217
number	0.2482	subject	0.2153
tv	0.2464	barrier	0.2153
backup	0.2455	deal	0.2146
difference	0.2404	drive	0.2141
mail	0.2383	list	0.2139

Como se puede apreciar, el peso de las palabras fluctúa entre el [0,+∞[, alcanzando su máximo valor con la palabra “format”, con un factor de peso de un 0.361.

7.2 Modelo TF-IDF & Naive Bayes

A modo de comparación, se realiza un estudio acerca del rendimiento de un algoritmo alternativo para la clasificación de los documentos, el cual consiste en utilizar el factor de peso TF-IDF, combinándolo con el clasificador "Naive Bayes".

7.2.1 Resultados

Los resultados fueron los siguientes:

Accuracy : 92.10000000000001%

	precision	recall	f1-score	support
alt.atheism	0.90	0.92	0.91	100
comp.graphics	0.81	0.82	0.82	100
comp.os.ms-windows.misc	0.96	0.88	0.92	100
comp.sys.ibm.pc.hardware	0.88	0.85	0.86	100
comp.sys.mac.hardware	0.87	0.89	0.88	100
comp.windows.x	0.86	0.93	0.89	100
misc.forsale	0.84	0.88	0.86	100
rec.autos	0.92	0.95	0.94	100
rec.motorcycles	0.98	0.97	0.97	100
rec.sport.baseball	0.97	0.97	0.97	100
rec.sport.hockey	0.96	0.99	0.98	100
sci.crypt	0.95	0.95	0.95	100
sci.electronics	0.94	0.89	0.91	100
sci.med	0.97	0.93	0.95	100
sci.space	0.93	0.96	0.95	100
soc.religion.christian	0.94	0.98	0.96	100
talk.politics.guns	0.92	0.97	0.94	100
talk.politics.mideast	0.97	0.99	0.98	100
talk.politics.misc	0.93	0.90	0.91	100
talk.religion.misc	0.94	0.80	0.86	100
avg / total	0.92	0.92	0.92	2000

Figura 7.3 Rendimiento TF-IDF & Naive Bayes.

Como muestra la imagen, podemos apreciar que el porcentaje de exactitud obtenido es de un 92,1%, lo que representa un buen resultado para la clasificación. Al igual que los resultados obtenidos en las variables precision, recall y del f1-score, corresponden a un 92% para todas. Cabe mencionar que, para esta comparación, y como se mencionó anteriormente, la base de datos utilizada ha sido la misma, aplicando los mismos parámetros, es decir, para cada categoría se seleccionaran 100 archivos, de igual forma que se hizo con el modelo propuesto.

La matriz de confusión asociada a este modelo es la siguiente:

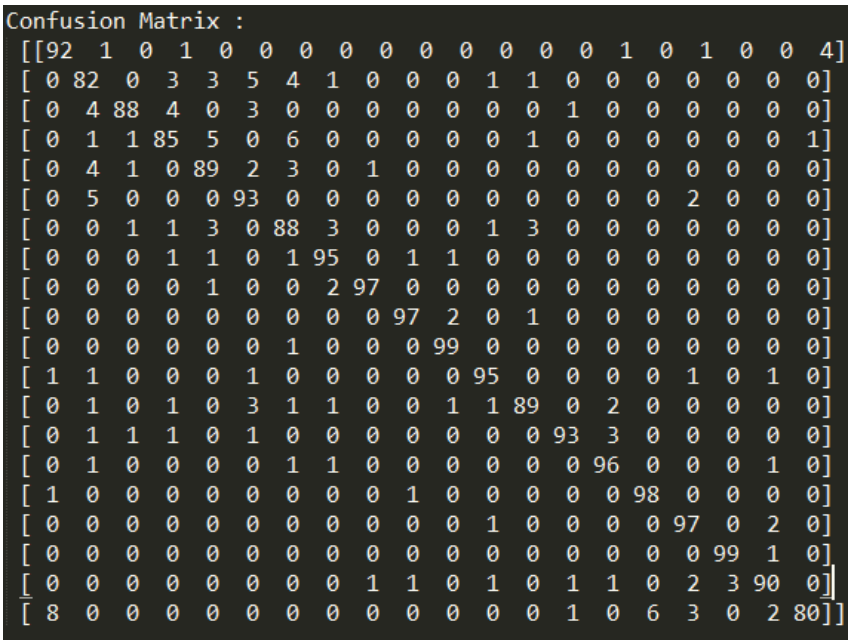

Figura 7.4 Matriz de Confusión TF-IDF & Naive Bayes.

7.2.2 50 Top Words

Tabla 7.3 Top Words TF-IDF & Naive Bayes.

TF - IDF					
Feature	**Weight**	**Feature**	**Weight**	**Feature**	**Weight**
noise	2,288	printing	1,054	way	0,9633
way	2,244	distribution	1,022	belief	0,9618
file	2,001	text	0,9998	prism.gatech.edu	0,9597
mouse	1,793	phone	0,9997	pc	0,959
eye	1,658	address	0,9995	mail	0,9532
drive	1,506	game	0,9863	dealer	0,947
mouse	1,471	quack	0,9846	card	0,944
god	1,407	game	0,9845	gld	0,941
phone	1,396	engine	0,9842	body	0,9303
god	1,239	bike	0,984	fire	0,93
keyboard	1,184	backup	0,982	church	0,9296
date	1,163	drug	0,9765	bitmap	0,9226
date	1,163	book	0,9757	thee	0,9168
bit	1,144	disk	0,9738	bit	0,9162
video	1,136	drive	0,9702	thou	0,9141
prism.gatech.edu	1,083	battery	0,968	rider.cactus.org	0,9135
oil	1,071	format	0,9648		

7.3 Comparaciones y Análisis

En primer lugar, una de las principales diferencias que pueden apreciarse de manera fácil, tiene que ver con los pesos que se obtienen de las palabras, en donde podemos comparar la palabra "homosexuality" con un factor de 15.782 del modelo TF-PR, versus la palabra "noise" con un peso de 2,288 para el factor TF-IDF, ambas consideradas como las más importantes de cada modelo.

Ahora, si se comparan los resultados del mayor valor IDF versus el mayor del factor PageRank, debo mencionar que el factor PageRank alcanza resultados más altos que los del IDF, con un valor máximo de 0.361 para la palabra "Format", versus el mayor valor de IDF que es 0.00427, para la palabra "Sleep", lo que representa una considerable diferencia.

La pregunta es, ¿a qué se debe esta rotunda diferencia entre los factores? Una posible respuesta que he logrado comprobar puede ser la siguiente: el factor TF-IDF asigna menos peso a una palabra si ocurre en una gran cantidad de los documentos en su corpus. En este caso, se procesaron dos mil archivos, y cien para cada categoría lo que representa una gran cantidad de documentos. Sin embargo, esto no necesariamente significa que la palabra no es importante para distinguir su clase. De hecho, una palabra que es común en su corpus, pero que también ocurre sustancialmente más a menudo en una clase que en la otra, podría muy bien ser muy valiosa para distinguir las clases.

En segundo lugar, para evaluar la calidad de la clasificación de cada modelo, en cada variación del corpus, algoritmo, y parámetros de pre-procesamiento, se han examinado tres medidas: precisión, recall y f1-score. La "precisión" (porcentaje de exactitud de documentos correctamente clasificados) es la más utilizada para evaluar algoritmos de aprendizaje, pero por si sola puede no ser la más representativa en la categorización de documentos ya que es muy común que en cada clase encontremos muchos más ejemplos negativos que positivos. Por ello, se utiliza "F1-Score", (número de documentos correctamente categorizados como pertenecientes a una determinada clase, dividido por el número total de documentos etiquetados como pertenecientes a tal clase) y "recall" (número de documentos correctamente categorizados como pertenecientes a una determinada clase, dividido por el número total de documentos que realmente pertenecen a esa clase, es decir, incluyendo los que el algoritmo no haya conseguido identificar como tales).

Los resultados obtenidos entre ambos modelos se contrastan en los siguientes gráficos:

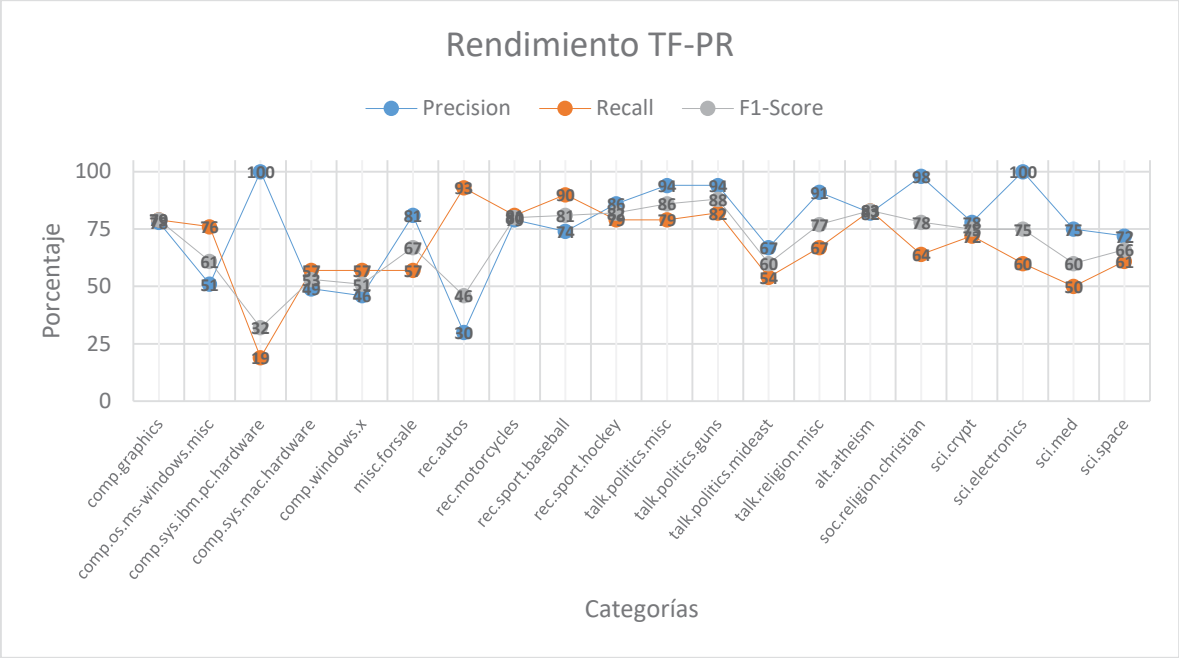

Figura 7.5 Gráfico de Líneas - Rendimiento TF-PR & Naive Bayes.

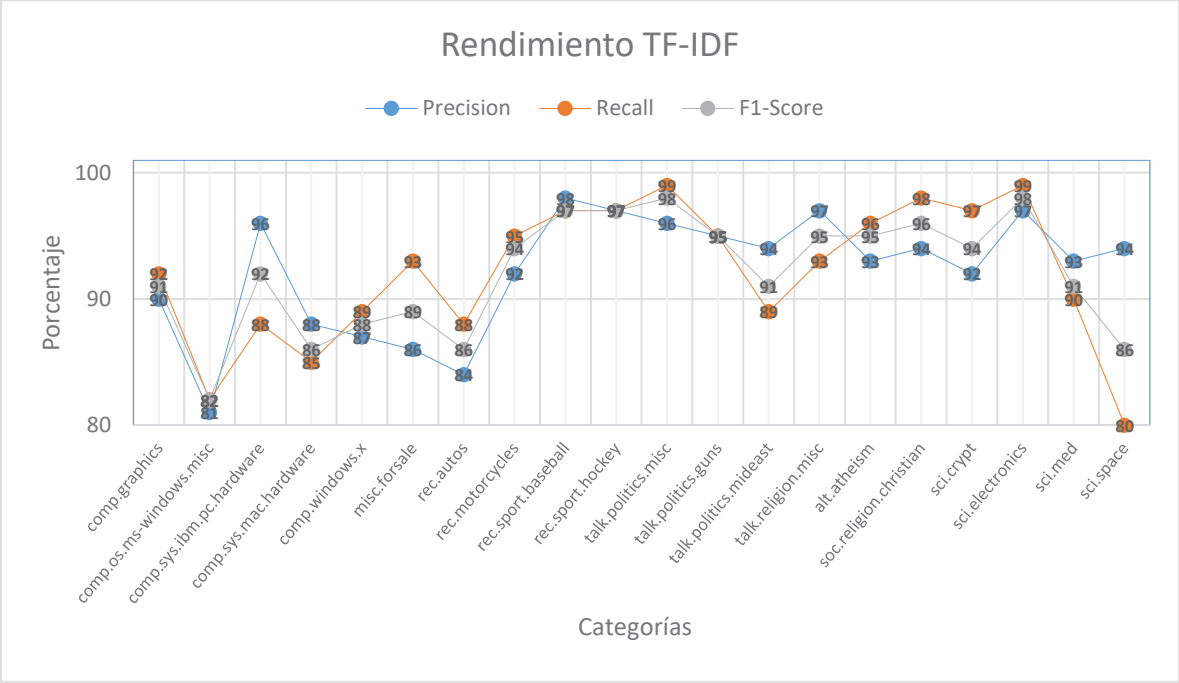

Figura 7.6 Gráfico de Líneas – Rendimiento TF-IDF & Naive Bayes.

Según las medidas obtenidas, y como muestran los gráficos, podemos apreciar que el rendimiento del algoritmo TF-IDF posee mejores resultados en comparación con el modelo propuesto. La explicación de este comportamiento puede darse principalmente por lo que se dijo en el primer punto, que el algoritmo de pesaje TF-IDF, a pesar de no tener valores tan altos en los pesos como los que se obtienen del TF-PR, entrega palabras que son más representativas para cada una de sus clases, ya que al algoritmo no le basta simplemente que una palabra se repita muchas veces dentro del corpus ni en los corpus de los demás documentos. Por el otro lado, en el algoritmo de TF-PR se muestran valores muy elevados de pesaje, aunque esto no indica que demuestre un buen rendimiento.

8 Trabajo Futuro

Es interesante ver cómo trabajan estos sistemas de clasificación automática de texto ya que alivian en gran medida el trabajo humano que se debería realizar para clasificar documentos de manera manual. Es por esto mismo que este tema de investigación va tomando importancia cada vez más, y de una manera acelerada, en donde en estos últimos años hemos podido darnos cuenta de los grandes aportes ya sea en papers y contenido en general que es posible encontrar en la web, en pos de enriquecer el conocimiento en la minería de textos.

Una de las tareas a futuro que se podría realizar para aportar con mayores análisis a la investigación, podría ser el variar el tamaño de la data, es decir, ver cómo se comportan los modelos descritos si consideramos entrenar y testear mayor o menor cantidad de documentos. Es muy probable que existan diferencias, quizás pequeñas, pero analizables en cuanto a los rendimientos que se puedan obtener.

Una segunda tarea a futuro sería generar una base de datos que contenga una mayor cantidad de palabras vacías, y luego extraerlas o eliminarlas de los documentos procesados. De esta manera, se acotaría el universo de posibles palabras importantes, además de reducir la dimensionalidad de los textos procesados. Esto, con el objetivo de aumentar la efectividad del modelo propuesto, aumentar su nivel de comparabilidad frente a otros modelos alternativos, y demostrar que puede ser factible para la clasificación de texto de manera automática.

9 Conclusión

La minería de textos, y la categorización de documentos en particular, son un campo de investigación y aplicación prometedor dado que más y más organizaciones están interesadas en aprovechar el gran cuerpo de conocimiento que disponen. Las técnicas de recuperación de la información y de aprendizaje automático facilitan en gran parte la tarea de extracción de conocimiento. Por lo tanto, se comenzó a analizar el cálculo del PageRank para trabajarlo directamente con documentos o artículos en vez de utilizarlo con sitios web. En efecto, se detalla la propuesta de utilizar este algoritmo como factor de peso en el proceso para la clasificación de textos de manera automática combinándolo con el factor TF (Term Frequency), de lo que resultaría el factor TF-PageRank.

El proceso de la clasificación de texto de manera automática es un largo y riguroso camino que se debe seguir, más aún si se implementa con el algoritmo del PageRank para ponderar las palabras más importantes, el cual conlleva a la generación de matrices y grafos. Con respecto a esto último, se ha logrado desarrollar la solución de manera más fácil con la utilización de las librerías que se mencionaron anteriormente, lo que ha favorecido de manera importante, en términos de tiempo, a la investigación.

En cuanto a los resultados que se obtuvieron en la fase de entrenamiento, podría decir que el modelo propuesto del factor TF-PageRank, si bien no presenta malos resultados, considerando que asignó categorías de manera correcta a más de la mitad del conjunto de datos utilizado, no es el más óptimo para clasificar documentos de manera automática, ya que, existen mejores alternativas, como lo es, por ejemplo, el factor de peso TF-IDF, que presenta, y de manera considerable, un mejor rendimiento y mejores resultados. Por lo tanto, como TF-PR no es el factor más adecuado para darle importancia a las palabras contenidas en un documento, se puede concluir que el factor PageRank tampoco es el más óptimo para proponerlo como factor de peso en la clasificación automática de textos.

Finalmente, hay que mencionar que el conocimiento y la experiencia obtenida de este proyecto son base para futuras implementaciones que pueden hacer extensivas estas técnicas aplicándolas a otros problemas de clasificación e interpretación de textos, así como también, para otros problemas de gestión del conocimiento.

10 Referencias

[1] Miriam Martín García. *Sistema de Clasificación Automática de Críticas de Cine*, 2009.

[2] Juan Pablo Cárdenas, Gastón Olivares y Rodrigo Alfaro. *Automatic text classification using words networks,* 2014.

[3] Clasificación Automática de Textos *pendientedemigracion.ucm.es/info/multi doc/tema-8-clasificacion-automatica*

[4] Blanca Gil Urdaciain. *Manual de Lenguajes Documentales,* 2004.

[5] Técnicas avanzadas de recuperación de información *http://ccdoc-tecnicasrecupera cioninformacion.blogspot.cl/2012/11/frecuencias-y-pesos-de-los-terminos-de.html*

[6] Natividad Novergues. *Algunas Aplicaciones de Redes Neuronales Artificiales*, 2000.

[7] Carlos G. Figuerola, José Luis Alonso Berrocal, Angel F. Zazo. *Clasificación Automática de Documentos.*

[8] Cristina González Rubio. *Clasificación automática de texto para el seguimiento de campañas electorales en redes sociales*, 2015.

[9] Nancy Barrera. *La Inteligencia Artificial*, 2016.

[10] Maribel Tirados. *Algoritmo Random Forest*, 2014.

[11] Araujo N. *Método Semisupervisado para la Clasificación Automática de Textos de Opinión,* 2009.

[12] Daniel Atuesta Rodríguez. *El algoritmo de Google*, 2015.

[13] Sebastián Ariel Pérez Vera. *Análisis y Clasificación de Textos con Técnicas Semi Supervisadas Aplicado a Área Atención al Cliente,* 2017.

[14] Sumarización *https://sites.google.com/site/ci2414i2012sumarizacion/ sumarizacion/extraccion-de-frases-claves.*

[15] Francisco Alonso Almeida, Ivalla Ortega Barrera, Elena Quintana Toledo. *Input a Word, Analyze the World: Selected Approaches to Corpus Linguistics,* 2016.

[16] Bird, Steven; Ewan Klein; Edward Loper. *Natural Language Processing with Python*, (2009)

[17] Perkins, Jacob. *Python Text Processing with NLTK 2.0 Cookbook*, 2010.

[18] Bird, Steven; Edward Loper; Jason Baldridge. *Multidisciplinary instruction with the Natural Language Toolkit*, 2008.

[19] Vallez, Mari; Rovira, Cristòfol, Codina, Lluís; Pedraza, Rafael. *"Procedimientos para la extracción de palabras clave de páginas web basados en criterios de posicionamiento en buscadores"*, 2010.

[20] M. Alicia Pérez Abelleira y Carolina A. Cardoso. "*Minería de texto para la categorización automática de documentos"*, 2010.