UNIVERSIDAD TÉCNICA PARTICULAR DE LOJA

La Universidad Católica de Loja

ÁREA TÉCNICA

TÍTULO DE INGENIERO EN SISTEMAS INFORMÁTICOS Y COMPUTACIÓN

Aplicación de técnicas de minería de texto para el agrupamiento de componentes académicos en base a los contenidos de planes docentes.

TRABAJO DE TITULACIÓN.

AUTOR: Caraguay Guamán, Robert Wladimir

DIRECTOR: Mora Arciniegas, María Belén, Mg.

LOJA – ECUADOR

2016

Septiembre, 2016

APROBACIÓN DE LA DIRECTORA DEL TRABAJO DE TITULACIÓN

Magister.

María Belén Mora Arciniegas.

DOCENTE DE LA TITULACIÓN

De mi consideración:

El presente trabajo de titulación: Aplicación de técnicas de minería de texto para el agrupamiento de componentes académicos en base a los contenidos de planes docentes, realizado por Caraguay Guamán Robert Wladimir, ha sido orientado y revisado durante su ejecución, por cuanto se aprueba la presentación del mismo.

Loja, agosto de 2016

f) . . . . . .

DECLARACIÓN DE AUTORÍA Y CESIÓN DE DERECHOS

"Yo Caraguay Guamán Robert Wladimir declaro ser autor (a) del presente trabajo de titulación: Aplicación de técnicas de minería de texto para el agrupamiento de componentes académicos en base a los contenidos de planes docentes, de la Titulación de Sistemas Informáticos y Computación, siendo María Belén Mora Arciniegas director (a) del presente trabajo; y eximo expresamente a la Universidad Técnica Particular de Loja y a sus representantes legales de posibles reclamos o acciones legales. Además certifico que las ideas, conceptos, procedimientos y resultados vertidos en el presente trabajo investigativo, son de mi exclusiva responsabilidad.

Adicionalmente declaro conocer y aceptar la disposición del Art. 88 del Estatuto Orgánico de la Universidad Técnica Particular de Loja que en su parte pertinente textualmente dice: "Forman parte del patrimonio de la Universidad la propiedad intelectual de investigaciones, trabajos científicos o técnicos y tesis de grado o trabajos de titulación que se realicen con el apoyo financiero, académico o institucional (operativo) de la Universidad".

f.....

Autor Caraguay Guamán Robert Wladimir
Cédula 1104960347

DEDICATORIA

El presente trabajo de titulación se lo dedico a Dios quien supo guiarme por el buen camino, y darme la fortaleza y coraje para superar los diferentes problemas que se me presentaron, y así haber llegado a este importante momento en mi vida de formación profesional.

A mis padres por ser el pilar más importante en mi vida ya que, gracias a sus consejos, ayuda, comprensión, amor, y apoyo incondicional permitieron que curse mis estudios universitarios y me convierta en un profesional.

A mis hermanas por estar siempre presente acompañándome con sus ocurrencias y ayuda incondicional. A todos mis familiares que de alguna manera con su granito de arena me apoyaron para llegar a estas instancias.

Robert Wladimir Caraguay Guamán

AGRADECIMIENTO

Le agradezco a Dios por bendecirme, acompañarme y haberme guiado a lo largo de mi carrera, por ser mi fortaleza para llegar a este importante momento en mi formación profesional.

A mi directora de trabajo de titulación Mg. María Belén Mora quien, gracias a sus contantes exigencias, consejos, conocimientos, ayuda y correcciones me ayudo a adquirir los conocimientos necesarios para poder culminar con éxito el presente trabajo.

Les agradezco a mis padres por ser mi camino base para mi vida, por brindarme su confianza, apoyo incondicional y la oportunidad de convertirme en un profesional y en la persona que soy. A mis hermanas por su ayuda y apoyo incondicional en todo momento.

Robert Wladimir Caraguay Guamán

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

ÍNDICE DE ECUACIONES

RESUMEN

La Minería de Texto es un área de investigación de las ciencias de la computación que aplica la lingüística computacional y el procesamiento de textos para la identificación y extracción de nuevo conocimiento a partir de colecciones de documentos, es decir, de información no estructurada. La Universidad Técnica Particular de Loja dispone de una gran cantidad de asignaturas que crean planes docentes en diferentes periodos académicos, generando un alto volumen de información por lo que se requiere una evaluación periódica de los contenidos empleados en cada plan docente ciclo a ciclo, con el propósito de comprobar que no exista un solapamiento de contenidos. En el presente trabajo se da solución a esta problemática, a través de la aplicación de técnicas de clustering de Minería de Texto a los contenidos de los planes docentes de la titulación de Ingeniería en Sistemas Informáticos y Computación, con el fin de validar que la información sea consistente y no existan contenidos semejantes entre áreas de estudio.

PALABRAS CLAVES:

Minería de texto, agrupamiento, Procesamiento de Lenguaje Natural (PLN), matriz términos por documentos (TDM), algoritmo k-means, descomposición en valores singulares (SVD), indexación semántica latente (LSI), framework Shiny, lenguaje de programación R.

ABSTRACT

Text mining is an area of research of the computer science that applies the computational linguistics and the processing of texts for the identification and extraction of new knowledge from collections of documents, i.e. of unstructured information. The Universidad Técnica Particular de Loja has a lot of subjects that creates teaching plans in different academic periods, generating a high volume of information requiring a periodic evaluation of the content used in each teaching plan cycle to cycle, with the purpose of verifying that there is no overlap of content. This work provides solution to this problem, through the application of techniques of clustering of text mining to the contents of the teaching plans of the engineering degree in computer systems and computing, in order to validate that the information is consistent and there are no contents similar between areas of study.

KEYWORDS:

Text mining, clustering, Natural language processing (NLP), Document term matrix (TDM), algorithm k-means, Singular value decomposition (SVD), Latent semantic indexing, framework shiny, R programming language.

CAPÍTULO 1: INTRODUCCIÓN

1.1. Introducción

Durante los últimos años las instituciones han venido generando una gran cantidad de información, provocada por el apogeo de las Tecnologías de la Información. Esta información al ser analizada y procesada de manera correcta puede crear conocimiento y además aportar una ventaja competitiva frente a otras instituciones. No obstante, al poseer gran cantidad de información surge el problema de distinguir la información que realmente se necesita para formar conocimiento. En la Universidad Técnica Particular de Loja (UTPL) se genera gran cantidad de información, uno es el caso de los componentes (asignaturas) que producen planes docentes en diferentes periodos académicos, lo que produce un elevado volumen de información que puede ser repetible si no hay una evaluación de los contenidos que se usan para el desarrollo de la aplicación docente.

El Procesamiento del Lenguaje Natural (PLN) y la Minería de Texto son técnicas que pueden ayudar a procesar esta información, debido a que se han convertido en metodologías indispensables para el tratamiento de información no estructurada. El Procesamiento del Lenguaje Natural permite la manipulación de lenguajes naturales utilizando herramientas computacionales, en donde entran en juego los lenguajes de programación (Cortez Vásquez, Vega Huerta, & Quispe Pariona, 2009). Mientras que la Minería de Texto a través de una serie de técnicas y algoritmos permite agrupar documentos similares en clústeres, categorizar automáticamente documentos de acuerdo a sus contenidos, etc. (Brun & Senso, 2004).

En el presente trabajo de titulación se da solución a la problemática expuesta mediante el uso de técnicas de Minería de Texto, analizando el contenido de los planes docentes de los componentes académicos de la titulación de Ingeniería en Sistemas Informáticos y Computación de la UTPL, los mismos que se encuentran almacenados en la base de datos de planes docentes de la universidad. Para cumplir con el objetivo del trabajo, se realiza el agrupamiento de los planes docentes en base a sus contenidos mediante la aplicación de la técnica de clustering de Minería de Texto, para lo cual como requisito es necesario aplicar antes la técnica del Procesamiento del Lenguaje Natural (PLN), que permite extraer la información importante de cada uno de los documentos.

Para el agrupamiento se realiza el estudio de varios algoritmos logrando seleccionar al algoritmo de agrupamiento por particiones k-means y al algoritmo LSI (Indexación Semántica Latente), que utiliza la técnica de la descomposición en valores singulares (SVD, con sus siglas en inglés) de matrices, para la indexación automática y la recuperación de documentos fundamentados en la noción de concepto. La técnica admite identificar modelos de relaciones

entre los términos y conceptos comprendidos en una colección de datos, es decir, a partir de una colección de documentos permite determinar los planes docentes que poseen contenidos similares.

Para ejecutar las técnicas del Procesamiento de Lenguaje Natural y de Minería de Texto, se ha seleccionado el lenguaje de programación R Project y la herramienta RStudio para la programación, porque aporta las características y paquetes necesarios para realizar la codificación y así lograr el agrupamiento de los planes docentes. Adicional para la visualización de los resultados obtenidos por cada una de las técnicas y algoritmos empleados, se utiliza el framework Shiny desarrollado por RStudio que facilita el desarrollo de aplicaciones web interactivas basadas en R Project.

1.2. Objetivos

1.2.1. Objetivo general.

Aplicar técnicas de minería de texto para el agrupamiento de componentes académicos en base a los contenidos de planes docentes.

1.2.2. Objetivos específicos.

- Investigar las técnicas de Minería de Texto, profundizando en la técnica de clustering.
- Analizar, seleccionar y aplicar técnicas de Procesamiento del Lenguaje Natural.
- Desarrollar un prototipo para comparar los contenidos de los componentes académicos.
- Analizar y validar los resultados obtenidos.

1.3. Metodología de desarrollo

La metodología de investigación a utilizar para el desarrollo del presente trabajo es la cuantitativa. Este tipo de metodología de investigación es empleada en las ciencias empíricas y se centra en los aspectos observables susceptibles a cuantificación. “La metodología de investigación cuantitativa se basa en el uso de técnicas estadística para conocer ciertos aspectos de interés sobre la población que se está estudiando” (Hueso & Cascant, 2012). Para el cumplimiento de los objetivos del presente trabajo, se procede a realizar los siguientes pasos metodológicos:

Primer paso: se define la motivación, alcance del proyecto (se indica claramente la razón del trabajo de titulación y que es lo que se va a lograr con el desarrollo del mismo), objetivos generales y específicos, problemática, resultados esperados y estructura del documento.

Segundo paso: se estudia la Minería de Texto y su aplicación en la agrupación de documentos a partir de un conjunto textual, lo que permite profundizar en el tema para identificar y abordar la problemática dentro del contexto de estudio. Una vez realizado el estudio de Minería de Texto e identificado el problema, se procede a elaborar el capítulo correspondiente al marco teórico.

Tercer paso: analiza y compara algoritmos de clustering existentes para seleccionar uno y llevar acabo el agrupamiento de los planes docentes. Además, se estudia y selecciona un algoritmo de reducción de dimensiones de matrices, para a través de vectores determinar los documentos de los componentes académicos con contenidos similares.

Cuarto paso: aplica la técnica del Procesamiento de Lenguaje Natural a los planes docentes, logrando obtener un conjunto de datos listos para ser agrupados con los algoritmos seleccionados.

Quinto paso: se realiza el agrupamiento de los planes docentes que poseen contenidos similares con los algoritmos seleccionados. Además, se realiza el diseño y desarrollo del prototipo para la visualización de los resultados obtenidos, mediante una interfaz web.

Sexto paso: se analiza los resultados obtenidos al aplicar el algoritmo de clustering y el algoritmo de reducción de dimensiones de matrices LSI (Indexación Semántica Latente).

1.4. Problemática

En la actualidad se encuentra una gran cantidad de información no estructurada generada por empresas, instituciones, organizaciones, etc., que representan un enorme conjunto de documentos electrónicos, que se encuentran almacenados en distintos lugares y que cada día crecen en cantidad y en formatos. Por ejemplo, actualmente en Internet existe y se dispone de una gran cantidad de información no estructura, generada por organizaciones, instituciones, etc., relacionada a una variedad de temas en concreto como: educación, entretenimiento, políticas, revistas, noticias, etc., que se encuentra almacenada en forma digital y de la cual se puede generar conocimiento.

En el ámbito de educación superior explícitamente en la Universidad Técnica Particular de Loja (UTPL) se dispone de una gran cantidad de componentes (asignaturas), que producen planes docentes en diferentes periodos académicos lo que genera un alto volumen de información, y por lo que es requerida una evaluación de los contenidos empleados en cada plan docente ciclo a ciclo, con el fin de comprobar que no exista un solapamiento de contenidos.

Debido a esta gran cantidad de datos no estructurados almacenados en forma digital, la minería de texto es ampliamente utilizada en la actualidad. Su uso explícitamente se da por las diversas técnicas que posee para el procesamiento de documentos no estructurados, generando valor o conocimiento a partir de una colección de documentos.

Es por esto que en el presente trabajo se aplica la técnica de Procesamiento del Lenguaje Natural (PLN) para el acondicionamiento de los planes docentes, y así emplear la técnica de clustering (Agrupamiento) y la de indexación semántica latente (LSI) en el contenido de los planes docentes.

1.5. Resultados esperados

Con el desarrollo del presente trabajo de aplicación de técnicas de minería de texto para el agrupamiento de componentes académicos en base a los contenidos de planes docentes, se espera conseguir los siguientes resultados:

- Aplicación de técnicas de Procesamiento del Lenguaje Natural (PLN) a los planes docentes, para obtener la matriz términos por documentos y emplear los algoritmos.

- Uso del algoritmo de clustering k-means, para dividir al conjunto de documentos en grupos y determinar cuáles son los planes docentes con contenidos similares.

- Aplicación del algoritmo de reducción de dimensiones de matrices LSI (Indexación Semántica Latente), para a través de vectores determinar cuáles son los documentos que contienen contenidos semejantes.

- Prototipo basado en Minería de Texto para presentar los resultados generados en cada una de las fases del proyecto en una interfaz web.

1.6. Antecedentes

En los últimos años se han desarrollado un sin número de trabajos investigativos sobre minería de texto, debido a la gran cantidad de información que las organizaciones generan y a la necesidad de obtener conocimiento a partir de la información. Se presenta a continuación los resultados de una revisión de investigaciones relacionadas con el objetivo de estudio ("Aplicar técnicas de minería de texto para el agrupamiento de componentes académicos en base a los contenidos de planes docentes"), con el propósito de empaparse sobre el conocimiento del mismo.

El trabajo investigativo de Benítez & Díez (2005) sobre técnicas de agrupamiento para el análisis de datos cuantitativos y cualitativos, que tiene como objetivo "Ofrecer una visión global sobre las técnicas de agrupamiento de datos y reconocimiento de patrones, explicando los distintos tipos y configuraciones que se encuentran actualmente para este tipo de algoritmos". Muestra las técnicas para el descubrimiento y la extracción de conocimiento a partir de base de datos (KDD), en donde la minería de datos es el principal paso de este proceso de descubrimiento o extracción de información útil. Se describe la composición de los algoritmos utilizados en la Minería de Texto, la cual es: a) ajustar el modelo para el conjunto específico de datos a analizar, b) criterio para la selección del modelo a usar; y c) criterio de satisfacción o función de coste a optimizar para la obtención del modelo.

Además se proporciona métodos para lograr cumplir cada uno de los objetivos (Clustering) de la Minería de Texto, y se menciona que muchas veces se usan combinaciones de varios de estos métodos para alcanzar los objetivos del trabajo (Benítez & Díez, 2005).

En el trabajo de Brun & Senso (2004) sobre Minería Textual se establece una definición de Minería de Texto, y se expone su objetivo que es el de extraer nuevo conocimiento a partir del análisis de corpus textuales, pero no de deducirlo. Se identifica la relación entre la Minería de Texto con otras áreas o disciplinas como la minería de datos, recuperación de información, inteligencia artificial y lingüística computacional.

Se analiza las principales funciones que deberían satisfacer las herramientas de Minería Textual y las salidas a esperar. Las funciones planteadas son las siguientes:

- Identificar "hechos" y datos puntuales a partir del texto de los documentos.
- Agrupar documentos similares (clustering).

- Determinar el tema o temas tratados en los documentos mediante la categorización automática de los textos.
- Identificar los conceptos tratados en los documentos y crear redes de conceptos.
- Facilitar el acceso a la información repartida entre los documentos de la colección, y la visualización de las relaciones entre los conceptos tratados en la colección.
- Visualización y navegación en colecciones de texto.

Finalmente se presentan las técnicas de la Minería de Texto, las cuales son de gran interés para abordar el caso de estudio, debido a que se hará uso de estas. Las técnicas propuestas son (Brun & Senso, 2004):

- Pre-procesamiento de los documentos.
- Identificación de nombres propios.
- Representación de los documentos mediante el modelo vectorial.
- Clustering o agrupación automática de documentos.

Otro estudio analizado es el de Lama (2013) sobre Clustering system based on Text Mining using the K means algorithm (Sistema de agrupación basado en Minería de Texto usando el algoritmo K means), cuyo objetivo es proporcionar una plataforma única para colocar los grupos de titulares de noticias similares y sus correspondientes enlaces a los sitios originales. Se centra principalmente en el uso de técnicas de Minería de Texto y en el uso del algoritmo K means para crear grupos o clases de titulares de artículos de noticias similares.

El estudio del proyecto se basa en aplicar Minería Textual, enfocada principalmente en la minería de datos y en la extracción de información. Los titulares de las noticias están en formato de archivo XML. Los cuales son procesados utilizando técnicas de pre-procesamiento de documentos, para finalmente agruparlos en clases según sus similitudes. Las técnicas de Minería de Texto empleadas en el proyecto son (Lama, 2013):

- Pre-procesamiento de documentos.
- Representación de documentos.
- Extracción de Información.
- Clustering, para realizar la agrupación de los titulares de noticias se utiliza el algoritmo K-means, destacando que es uno de los métodos más eficientes para la agrupación y de ahí su uso para cumplir con el propósito del proyecto.

Al aplicar el algoritmo K-means cada grupo se caracteriza por poseer un centroide único, en donde los elementos o titulares de noticias que están cerca al centro de gravedad de ese grupo pertenecen al mismo, en cambio los que se encuentran alejados del centroide pertenecen a otro clúster. La letra "K" en el algoritmo k-means indica la cantidad de grupos o clúster que se desea crear para agrupar a los diferentes titulares de noticias (Lama, 2013).

Los resultados obtenidos son grupos de titulares de noticias similares, que son presentados en consola y también en una página web, que muestra grupos de noticias con enlaces subyacentes a los portales de noticias que contienen la noticia correspondiente.

El trabajo realizado por Karanikas, Tjortjis, & Theodoulidis (n.d.), sobre Fast and effective Text Mining using linear-time document clustering (Minería de Texto rápida y eficaz mediante agrupación de documentos en tiempo lineal), describe la importancia de la agrupación de documentos que es permitir a los usuarios profundizar selectivamente para explotar temas específicos de interés sin necesidad de leer todos los documentos. Además, se sugiere dos fases para la técnica. En primer lugar, se menciona la extracción de las características de cada documento, las mismas que se utilizan para comparar los documentos y etiquetar su contenido para los usuarios. En segundo lugar, la aplicación de algoritmos de clustering para agrupar automáticamente los documentos en una jerarquía de grupos. Los algoritmos utilizados son el K-means y Scatter/Gather (buckshot, fractionation, and split/join).

Para seleccionar el centro de cada grupo se utilizan tres algoritmos: el primero es el de *azar* que recoge puntos k al azar de la entrada como centros iniciales de los grupos. El segundo es el de *buckshot* (perdigones), el cual recoge $\sqrt{kn}$ puntos al azar de la entrada de N elementos y los selecciona como centros iniciales. El tercer es el de *fraccionamiento* que construye una jerarquía de abajo hacia arriba a partir del conjunto de entrada inicial, en donde los grupos de alto nivel de esta jerarquía se convierten en los centroides iniciales de los clúster (Larsen & Aone, n.d.).

Se introduce una metodología para medir la calidad y eficacia de cada algoritmo para lo cual, a las jerarquías generadas por los mismos, se las compara con un conjunto de categorías previamente asignadas a los documentos por profesionales expertos. Concluyendo que el método de selección de centros por perdigones ofrece mejor equilibrio en tiempo y calidad. Finalmente los resultados se presentan en una interfaz gráfica de usuario intuitiva (Larsen & Aone, n.d.).

En el trabajo realizado por Cardoso & Pérez (2010) sobre “Minería de Texto para la categorización automática de documentos”, y cuyo objetivo es “implementar un buscador semántico que aproveche el resultado de los algoritmos de aprendizaje automático para la clasificación de documentos”. Se utiliza un corpus de más de 8000 documentos sobre resoluciones rectorales de la Universidad Católica de Salta. Los documentos están en distintos formatos: Microsoft Word, texto plano y PDF. Como resultado de la fase de análisis se obtiene un conjunto de archivos en formato XMl, los que contienen partes relevantes del texto original y sirven para construir el índice del motor de búsqueda. Para construir el clasificador se emplea los algoritmos Co-training, EM (Expectation maximization) y SMO (sequential minimal optimization).

- En el algoritmo SMO: “el aprendizaje de máquinas de vectores es un método supervisado que ha demostrado buenas propiedades para la categorización de documentos” (Cardoso & Pérez, 2010).
- En el algoritmo Co-training: “se reconoce que los atributos de algunos conjuntos de datos pueden descomponerse naturalmente en dos subconjuntos, y que cada uno de ellos sirve efectivamente para clasificar cada documento, es decir, son redundantemente predictivos” (Cardoso & Pérez, 2010).
- El algoritmo EM: se basa en Naive Bayes y según Cardoso (2010) se utiliza para “instruir a clases para un pequeño conjunto etiquetado y después ampliarlo a un conjunto grande de datos no etiquetados utilizando el algoritmo EM de clustering iterativo”.

Finalmente, en dicho trabajo se presenta un buscador semántico, en donde los usuarios pueden visualizar la información de los documentos de acuerdo a las búsquedas que realicen.

1.7. Organización del proyecto

La memoria del trabajo de titulación, se ha dividido en seis capítulos y diez anexos cuya estructura se describe a continuación:

Capítulo 1: análisis a detalle de varios trabajos relacionados con el tema de la presente investigación.

Capítulo 2: reseña de que es Minería de Texto, de la diferencia entre Minería de Texto (text mining) y minería de datos (data mining), las técnicas de Minería de Texto disponibles y de las tecnologías que utilizan el proceso de Minería de Texto.

Capítulo 3: analiza y compara las ventajas y desventajas que presentan los algoritmos de agrupamiento, para realizar la selección de uno y llevar a cabo el agrupamiento de los planes docentes. Además, se estudia el algoritmo de reducción de dimensiones de matrices LSI (Indexación Semántica Latente), para a través de vectores identificar los documentos con contenidos similares.

Capítulo 4: extrae el contenido de los planes docentes de la base de datos y se crea el corpus con los documentos para aplicar el Procesamiento del Lenguaje Natural (PLN).

Capítulo 5: realiza el agrupamiento de los planes docentes con los algoritmos k-means y LSI (indexación semántica latente). Además, se diseña y desarrolla el prototipo para presentar los resultados generados por los algoritmos.

Capítulo 6: analiza y compara los resultados obtenidos al aplicar los algoritmos k-menas y LSI.

Los anexos están estructurados de la siguiente manera:

- Anexo 1: presenta el contenido del archivo spanish.data del paquete tm (text mining) de R Project.
- Anexo 2: evidencia la lista de stop words a remover de los documentos.
- Anexo 3: muestra los bigramas y trigramas identificados en los planes docentes.
- Anexo 4: presenta las palabras y frases que forman el vocabulario de términos.
- Anexo 5: exhibe la matriz términos por documentos de la prueba número uno.
- Anexo 6: contiene la matriz términos por documentos de la prueba número dos.

- Anexo 7: exhibe la matriz términos por documentos correspondiente a la prueba número tres.

  Anexo 8: expone la matriz términos por documentos de la prueba número cuatro.
- Anexo 9: contiene la matriz términos por documentos de la prueba número cinco.
- Anexo 10: evidencia la matriz términos por documentos de la prueba número seis.
- Anexo 11: dirección url del código del prototipo.

CAPÍTULO 2: MARCO TEORICO

2.1. Minería de Texto

La Minería de Texto o Text Mining, es una nueva y excitante área de investigación de las ciencias de la computación que se define como el proceso de descubrimiento de patrones interesantes y nuevos conocimientos de un conjunto de documentos o corpus textuales. “La Minería de Texto es un campo interdisciplinario joven que se basa en: la recuperación de información, en la minería de datos, en el aprendizaje automático, en la estadística y en la lingüística computacional” (Gupta & Lehal, 2009).

La diferencia entre minería de datos y Minería de Texto, está en que la minería de datos descubre patrones interesantes en las bases de datos, en cambio en la Minería de Texto los patrones se descubren de una colección de textos en lenguaje natural tales como correos electrónicos, archivos HTML, documentos de texto, archivos PDF, etc. Las bases de datos están diseñadas para ser procesadas de forma automática por los programas, en cambio los documentos están escritos para ser leídos por las personas. De ahí que las herramientas de minería de datos están construidas para manejar datos estructurados de base de datos, y las de Minería de Texto pueden operar datos no estructurados o semiestructurados (Gálvez, 2008).

En base a Cardoso & Pérez (2010) “se estima que entre el 80% y el 90% de los datos de las organizaciones son no estructurados”. Es de ahí que la Minería de Texto es un área muy utilizada en las organizaciones, por la gran cantidad de información que estas poseen, ofreciendo la posibilidad de explotar estas grandes cantidades de textos, extrayendo conocimiento útil y aportando una ventaja competitiva. Sin embargo, gran parte de los esfuerzos de investigación y desarrollo se han concentrado en la minería de datos que manipula datos estructurados. El problema que se presenta en la Minería de Texto es que los datos están escritos en lenguaje natural (datos no estructurados); y está enfocado para que los seres humanos se comuniquen entre sí, es por esto que las computadoras están muy lejos de la comprensión del lenguaje natural.

En base a esto Cardoso & Pérez (2010) mencionan que “los seres humanos tienen la capacidad de distinguir y aplicar patrones lingüísticos al texto y pueden superar fácilmente los obstáculos que las computadoras no pueden manejar fácilmente como las jergas lingüísticas, las variaciones de ortografía y el significado contextual”. No obstante, a pesar de nuestras capacidades lingüísticas que nos proporcionan la disposición de comprenden los datos no estructurados, carecemos de la capacidad que poseen los equipos de procesar corpus

textuales a altas velocidades. La Figura 1 presenta un modelo para el proceso de Minería de Texto.

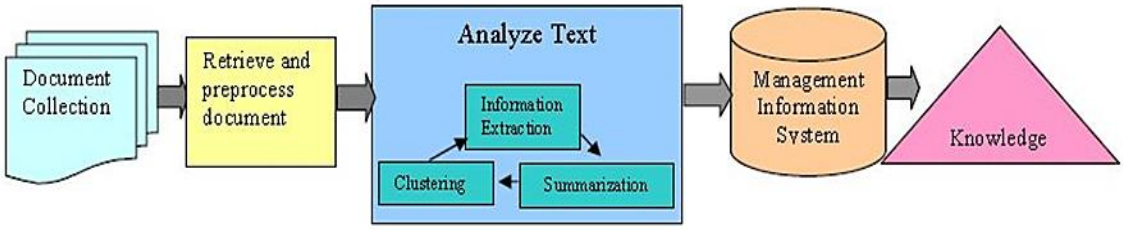


Figura 1. Ejemplo del proceso de Minería de Texto
Fuente: http://doi.org/10.4304/jetwi.1.1.60-76
Elaboración: Gupta & Lehal (2009)

2.1.1. Información estructurada y no estructurada.

La información estructurada es aquella que está perfectamente definida y sujeta a un formato específico. Cardoso & Pérez (2010) proporciona la siguiente definición: "la información estructurada se caracteriza por tener un significado que pretende no tener ambigüedad y que está representado explícitamente en la estructura o formato de los datos". Un ejemplo en el cual la información se encuentra estructurada son las bases de datos. En estas los campos cuentan con una definición específica como: fecha, valor numérico, cadenas de texto, etc. (Ruiz & González, n.d.).

En cambio, la información no estructurada es aquella que no puede ser almacenada en bases de datos relacionales. Las principales características de los datos no estructurados que expone Vidal (2014) son las siguientes:

- *Volumen y crecimiento:* el volumen de los datos no estructurados y la tasa de crecimiento es superior al de los datos estructurados.
- *Orígenes de datos:* su origen es muy diverso: documentos en lenguaje natural (Word, PDF, PPT), datos generados por redes sociales, datos generados en foros, correos electrónicos, etc.
- *Almacenamiento:* debido a su estructura no es posible almacenarlos en una arquitectura relacional.
- *Seguridad:* "el control de acceso a los mismos es complejo debido a cuestiones de confidencialidad y a la difícil clasificación del dato" (Vidal, 2014).
- *Terminología e idiomas:* "es una cuestión crítica tratando datos no estructurados de tipo texto. Es habitual llamar a lo mismo de diferentes formas, de tal modo que es necesario una racionalización de la terminología" (Vidal, 2014).

2.2. Técnicas de Minería de Texto

Para cumplir con el objetivo de la Minería de Texto es esencial la utilización de técnicas que proceden de la recuperación de información y de la lingüística computacional. Las técnicas para Minería de Texto son las siguientes (Brun & Senso, 2004):

1. Pre-procesamiento de los documentos.
2. Identificación de nombres propios.
3. Representación de los documentos mediante el modelo vectorial.
4. Clustering o agrupación automática de documentos.
5. Categorización automática.
6. Relaciones entre términos y conceptos.

2.2.1. Pre-procesamiento de los documentos.

La técnica consiste en la tokenización del texto, en la eliminación de signos de puntuación, en la extracción de stop words y en la lematización de las palabras de los documentos. Etxeberría (como se citó en Brun & Senso, 2004) menciona que "esta técnica consiste en extraer las palabras utilizadas en un documento, o segmentar el texto en distintas formas gráficas". Como resultado del pre-procesamiento el sistema informático debe convertir y devolver el documento en un formato de texto plano, no binario. En base a Lama (2013) el pre-procesamiento incluye las siguientes etapas:

2.2.1.1. Tokenización.

La tokenización en la Minería de Texto es el proceso de dividir un flujo determinado de texto o secuencia de caracteres en frases, palabras, símbolos u otros elementos importantes. Los cuales son almacenados en fichas que se agrupan juntos como una unidad semántica y son utilizados como entrada para su posterior procesamiento, tales como análisis o técnicas de Minería de Texto (Brun & Senso, 2004).

La tokenización generalmente ocurren a nivel de palabra, pero su definición varía de acuerdo al contexto. En idiomas como el inglés (y en la mayoría de los lenguajes de programación) donde las palabras están delimitadas por espacios en blanco, este enfoque es sencillo. "Sin embargo, la tokenización es más difícil para los idiomas como el chino, que no tienen límites de palabras" (Lama, 2013).

Ejemplo:

Entrada: "Francisco, papa y argentino"

Salida: Fichas con las palabras.

Francisco

Papa

Argentino

2.2.1.2. Eliminación de palabras vacías (Stop Word Removal).

En base a Brun & Senso (2004) “una tarea habitual en el pre-procesamiento de los documentos es la eliminación de palabras vacías, carentes de significado, como preposiciones, artículos, conjunciones, etc.”. A veces las palabras muy comunes representan muy poca importancia y contenido informativo al momento de seleccionar los documentos acordes a la necesidad del usuario, por lo que se las excluye completamente del vocabulario. A estas palabras se la denomina palabras vacías (Lama, 2013).

El objetivo principal de decidir crear un stop list radica en clasificar los términos de acuerdo a su frecuencia de recopilación, facilitando la identificación de los términos más empleados para posteriormente agruparlos en el stop list, los cuales serán excluidos durante la indexación. La separación de las palabras vacías del índice reduce considerablemente el tamaño del índice sin afectar la exactitud de las consultas del usuario. A continuación, se presentan algunas palabras vacías:

un	el	son
como	ser	desde
tiene	en	su
de	que	que
una	y	era
a	para	puede
que	es	eran
en	él	voluntad

2.2.1.3. Lematización.

Es el proceso de eliminar de manera automática las formas derivadas de una palabra para reducirlas a su forma original (lema). Es uno de los procesos principales del procesamiento

del Lenguaje Natural. “Consiste en dividir cada palabra en los lemas que la forman” (Brun & Senso, 2004). El algoritmo para realizar la lematización requiere de un diccionario o base de conocimiento sobre las diferentes alteraciones morfológicas, conjugaciones verbales, etc., que le permiten extraer de manera adecuada los lexemas que forman cada palabra. Sin embargo, Brun & Senso (2004) mencionan que:

> Es una tarea difícil implementar un lematizador para un nuevo idioma debido a que el proceso implica tareas complejas como el análisis morfológico de la palabra, es decir, comprender el contexto y determinar el papel de una palabra en una oración (que requiere, por ejemplo, el uso gramatical de la palabra).

Para aclarar el proceso de lematización vamos a considerar que disponemos de un conjunto de documentos que poseen el siguiente contenido:

Ejemplo:

- El perro corrió a la calle.
- El perrito come croquetas.
- La perra del vecino tiene crías.
- Los perros policías son entrenados arduamente.
- Las perras callejeras son un problema del municipio.

Como se observa en la Figura 2 se reduce las variantes morfológicas de una palabra a su raíz, lexema o lema y esta no tiene que tener significado. Esta técnica de pre-procesamiento ayuda a reducir el tamaño del índice en un 50 % (Lama, 2013).

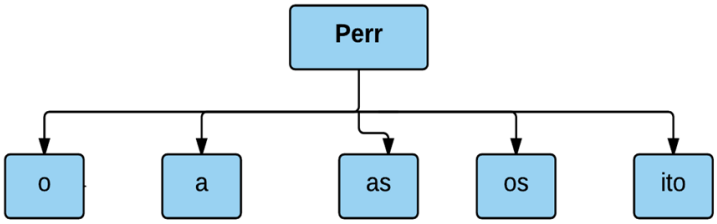


Figura 2. Lema de la palabra perro.
Elaboración: propia.

2.2.1.4. Segmentos repetidos o frases (N-Gramas).

Es un aspecto importante en el pre-procesamiento debido a que nos permite identificar secuencias de palabras, que aparezcan adyacentes en el contenido de los documentos y que utilizadas de esta forma generan un significado específico.

Por ejemplo:

- Ingeniería Química,
- Sistemas Operativos,
- Redes Neuronales,
- Instrumentos musicales,
- etc.

Al separar los segmentos repetidos en los diferentes términos que los conforman, originaria una descontextualización y pérdida del significado. Los programas de recuperación textual e indexación han prestado poco interés a este dilema separando este conjunto de palabras o frases. Sin embargo en el área de Minería de Texto, la extracción de estas expresiones o frases es importante, debido a que buscamos la agrupación de conceptos que representan el contenido de un documento (Vallez & Pedraza, 2007). Brun & Senso (2004) afirman que "identificar los segmentos repetidos que aparecen en un texto podría hacerse fácilmente teniendo acceso a un diccionario que permita identificar la categoría gramatical de cada palabra (sustantivo, adjetivo, preposición, verbo, etc.)".

El problema surge en determinar que *segmentos repetidos* proporcionan un significado exclusivo para ser tratados como conceptos o términos. "Para solucionar este problema, cabe la posibilidad de aplicar técnicas estadísticas que seleccionen únicamente aquellos segmentos de repetición que ocurren con mayor frecuencia en los documentos" (Brun & Senso, 2004).

2.2.2. Identificación de nombres propios.

La técnica consiste en extraer los nombres propios correspondientes a las personas, eventos, organizaciones, leyes, fechas, cantidades monetarias, instituciones, etc., del conjunto de documentos. "La Minería Textual también debería permitirnos identificar las relaciones que existen entre estos nombres propios y constatar así 'hechos' descritos en los documentos" (Brun & Senso, 2004).

Los nombres propios necesitan un procedimiento especial por la mayoría de sistemas de texto, debido a que no se puede interpretar su significado solo comprobando en los diccionarios de ortografía o intercambiándolos con sinónimos de la misma forma que las palabras comunes.

Los nombres propios permiten identificar rápidamente el significado o contenido de un determinado texto. Por esta razón la identificación de los nombres propios en todas sus variaciones es esencial y de gran valor para los sistemas de procesamiento de texto. Para Gálvez (2007) “una variante de nombre propio se podría definir como una cadena, que está conceptualmente relacionada con la forma correcta, o normalizada de ese nombre”. Para solucionar este problema se recomienda utilizar técnicas de equiparación aproximada, las cuales permiten establecer una correspondencia entre las variantes y los nombres propios correctamente almacenados en el diccionario.

Un tema complicado en la identificación de nombres propios, es la extracción de las relaciones que existen entre los términos. “En este sentido, es necesario recurrir a técnicas de parsing y análisis sintáctico de las sentencias, para identificar los verbos que sirven de nexo entre los nombres propios y tratar de deducir así posibles relaciones” (Brun & Senso, 2004).

2.2.3. Representación de los documentos mediante el modelo vectorial.

En base a Brun & Senso (2004) “una premisa en cualquier aplicación de recuperación y tratamiento documental es la necesidad de representar el contenido de los documentos mediante un modelo”. El modelo implementado y utilizado hoy en día en los sistemas de recuperación de información, como en el campo de la Minería de Texto es el modelo vectorial o también llamado modelo de espacio vectorial, el cual es un modelo algebraico por lo que, para extraer los documentos pertinentes del corpus textual, es muy importante transformar la versión del texto completo de los documentos a la forma vectorial.

En el modelo vectorial un documento se lo describe mediante un conjunto de términos o pesos de términos (estos deben estar almacenados en formato raíz o lema), los cuales son extraídos directamente del contenido del documento, o pueden ser asignados por un documentalista o programa informático. Brun & Senso (2004) indican que “cualquiera que sea el caso, el documento se representará mediante una secuencia de términos o “componentes” que corresponden con los distintos términos utilizados para describir el contenido del documento”. La representación más adoptada es una colección de documentos compuesta por *n* documentos indexados y *m* términos representados por una matriz documento-término (TDM, con sus siglas en inglés) de *n* x *m.*

Para Brun & Senso (2004) “en el modelo vectorial, cada documento se considera un vector y cada término que aparece en al menos un documento, será un componente del vector”. A

partir del modelo de espacio vectorial, los documentos se representan mediante la frecuencia de los términos (TF) y frecuencia inversa documento (IDF) (Blázquez Ochando, 2012).

En la representación de los documentos mediante el espacio vectorial, la recuperación de información se realiza a partir de la "comparación de la distancia que existe entre los vectores correspondientes a los documentos, y un vector utilizado para representar la ecuación de búsqueda" (Brun & Senso, 2004). Entendiéndose como recuperación de información al proceso mediante el cual, partiendo de un conjunto de textos fijos y de una necesidad de información, se devuelven los documentos que mejor satisfacen esa necesidad. En el caso de recuperar información a través de una consulta la representación de los documentos sería así (Aguirre Pérez, n.d.):

- documento = (peso_de_término_1, peso_de_término_2, ..., peso_de_término_n)
- consulta = (peso_de_término_1, peso_de_término_2, ..., peso_de_término_n)

Para establecer el grado de similitud entre una consulta dada por el usuario con respecto al conjunto de documentos, se calcula la distancia entre estos aplicando la fórmula de *medida de similitud*. La Ecuación 1 presenta la fórmula para medir la similitud.

$$SimCos(d_{(d)},q) = \frac{\sum_{n=1} \left(P_{(n,d)} \times P_{(n,q)}\right)}{\sqrt{\sum_{n=1} \left(P_{(n,d)}\right)^2 \times \sum_{n=1} \left(P_{(n,q)}\right)^2}}$$

Ecuación 1. Fórmula para el cálculo de la similitud del coseno.
Fuente: http://ccdoc-tecnicasrecuperacioninformacion.blogspot.com/2012/12/modelo-vectorial.html
Elaboración: Blázquez (2012)

Una de las ventajas del modelo espacio vectorial frente a otros modelos como el booleano, es que permite calcular la semejanza entre la ecuación de consulta y los documentos. Lo que "permite realizar un ranking u ordenación de los documentos recuperados, mostrando al principio de la lista aquellos documentos que presentan un alto grado similitud a la ecuación de búsqueda, y al final de la lista los que presentan baja similitud" (Brun & Senso, 2004). En el modelo booleano Blázquez (2012) menciona que un documento del corpus puede estar simbolizado por la presencia o ausencia de términos indexados en el diccionario de la siguiente manera:

- Documento {1,0,1,1,0,0,1,1,1,0,0,1,0,1,1}

En donde el número uno indica los términos que se encuentran en ese documento, y el número cero en cambio muestra la ausencia de términos. En cambio, el modelo vectorial utiliza el peso de cada uno de los términos para cada documento, exponiendo así la importancia de los mismos:

- Documento {4’343, 1’783, 0, 4’724, 0, 0, 6’934, 1’437, 5’934, 0, 2’421}

En este caso los valores distintos de cero indican la presencia de los términos y el número cero la ausencia.

Matriz Términos por Documentos

La matriz términos por documentos (mtd) es una forma de representar el modelo vectorial en la cual a los documentos se los expresa en función de vectores, que recogen la frecuencia de los términos que poseen. Los términos que poseen los documentos han pasado por el proceso de lematización y además son términos no vacíos, es decir, que aportan conocimiento al documento. La Figura 3 presenta la estructura de la matriz términos por documentos. Las filas corresponden a los documentos del corpus y las columnas a los términos de los documentos.

	Término 1	Término 2	Término 3
Documento 1	0	1	1
Documento 2	1	0	1
Documento 3	1	1	0

Figura 3. Estructura de la matriz términos por documentos.
Elaboración: propia.

2.2.4. Clustering o agrupación automática de documentos.

Montes (2013) define al agrupamiento como “el proceso de agrupar los datos en clases o en clústeres, de tal forma que los datos de un mismo clúster tienen una alta similitud y a su vez, son muy diferentes de los de otro clúster”. El análisis de Clustering es una técnica utilizada para agrupar o identificar documentos similares entre sí, que, a diferencia de la clasificación supervisada, este agrupa a los documentos durante su ejecución, es decir, no se tiene categorías predefinidas. La diferencia con la clasificación “es que en el caso del clustering no hay una división previa del espacio en categorías o clases; otra diferencia es también que los algoritmos de clasificación no agrupan los datos, sino que los clasifican uno a uno” (Benítez & Díez, 2005).

En la clasificación no supervisada o clúster no existen conjuntos prefinidos para asignar los objetos, estos son agrupados durante el proceso de ejecución. En cambio en la clasificación supervisada existen grupos predefinidos con anticipación en los cuales se van asignando los objetos (Berzal, n.d.).

En base a Montes (2013) “al hacer clúster, se puede identificar regiones densas y regiones dispersas en el espacio de características, y por lo tanto, descubrir distribución de patrones y correlaciones entre los atributos”. Un algoritmo básico de agrupación crea un vector de categorías para cada documento, y mide el peso del documento para determinar que se ajuste a su determinado grupo. Uno de los beneficios de la agrupación es que los documentos pueden aparecer en varias categorías, asegurando así que un documento útil no será omitido del resultado de búsqueda.

Benítez & Díez (2005) mencionan que “el funcionamiento de los algoritmos de clustering está basado en la optimización de una función objetivo”. Que habitualmente consiste en la suma ponderada de las distancias a los centros de los clústeres. Para lo cual el algoritmo asigna a cada objeto una medida de similitud al centroide de cada clúster, con el fin de establecer a cuál de los grupos identificados corresponde dicho objeto. Para realizar el cálculo de la medida de semejanza entre los objetos, se emplea la función de distancia de similitud del coseno descrita en el modelo vectorial.

Los algoritmos de agrupamiento según Puldón, Espín, & Jiménez (2012) se dividen en: agrupamiento particional, agrupamiento jerárquico, agrupamiento basado en densidad, agrupamiento basado en rejas y agrupamiento basado en modelos. A continuación, se describe a cada uno de ellos.

2.2.4.1. Agrupamiento jerárquico.

En base a Puldón et al., (2012) el agrupamiento jerárquico “crea una descomposición jerárquica de un conjunto de datos, formando un dendograma que divide recursivamente el conjunto de datos en conjuntos cada vez más pequeños”. En el cual para obtener los clústeres deseados se puede dividir en diferentes niveles al dendograma.

A continuación, se presentan algunos algoritmos de agrupamiento jerárquico (Benítez & Díez, 2005):

- **Birch (Balanced Iterative Reducing and Clustering using Hierarchies).**

Es un algoritmo sin supervisión que realiza agrupación jerárquica sobre grandes conjuntos de datos (Davitkov, 2011), el cual almacena para cada clúster o grupo un triplete de datos:

a) el número de objetos en el clúster,
b) la suma de todos los valores de los atributos de todos los objetos pertenecientes al grupo y,
c) la suma de los cuadrados de los atributos de los objetos que pertenecen al clúster.

Esta información le permite construir un árbol denominado CF-tree (Clúster Features tree) (Benítez & Díez, 2005). El procedimiento del algoritmo BIRCH es el siguiente (Davitkov, 2011):

1. El algoritmo comienza con un punto único de racimo (cada punto en la base datos es un clúster).

2. Los grupos de los puntos más cercanos son separados en otros grupos, y continúa, hasta que sólo queda un clúster.

3. El cómputo de los clústeres se realiza con ayuda de la matriz de distancia *(O (n^2) large)* y tiempo *O (n^2)*.

- **Rock (RObust Clustering using links).**

Es un algoritmo de agrupamiento de tipo jerárquico, diseñado para datos cualitativos. "Su funcionamiento se basa en el concepto de enlaces (links) entre objetos vecinos (neighbors). Se describe como vecinos a dos objetos, si el valor de una función de similitud entre los dos objetos excede cierto valor de límite θ" (Benítez & Díez, 2005).

La función de similitud y el valor del umbral están definidos por el usuario. ROCK interpreta primero todos los objetos como si fueran grupos independientes, y con cada una de las iteraciones combina los clústeres pretendiendo mejorar el resultado de la función objetivo. Para lo cual define una medida de bondad entre pares de grupos, en donde los pares que poseen la bondad más alta, son los que se combinan formando un nuevo clúster (Benítez & Díez, 2005).

- **CURE (Clustering Using Representatives).**

Benítez & Díez, (2005) mencionan que **"**es un algoritmo de clustering jerárquico que se basa en la selección de más de un elemento representativo de cada clúster. Como resultado, CURE es capaz de detectar grupos con múltiples formas y tamaños". Utiliza el muestreo aleatorio y la partición para acelerar la agrupación.

Para cada grupo *c,* se eligen puntos bien dispersos dentro de la agrupación y enseguida se reduce la distancia hacia el centro de la agrupación por una fracción *a*. La distancia entre dos grupos es entonces la distancia entre el par más cercano de puntos representativos de cada grupo. Los puntos distintivos *c* intentan capturar la forma física y la geometría de la agrupación. La disminución de los puntos dispersos permite deshace anomalías superficiales y además mitiga los efectos de los valores extremos (Possamai, 2006).

- **COBWEB.**

Garre, Cuadrado, & Sicilia (2007) indican que este algoritmo se identifica porque emplea un aprendizaje incremental, es decir, lleva a cabo la agrupación instancia a instancia. Construye una taxonomía de agrupaciones sin tener un número predefinido de clústeres. Los grupos están representados probabilísticamente por la probabilidad condicional P (A = v | C). Donde P (A = v | C) es la probabilidad de que una instancia tenga *v* de valor para su atributo A, dado que pertenece a la categoría C. Cuando mayor sea esta probabilidad más de dos instancias de una categoría comparten los mismos valores de atributo.

Durante su ejecución crea un árbol, en el cual las hojas simbolizan a los segmentos y el nodo raíz abarca por completo el grupo de datos de entrada. Al inicio el árbol solo contiene un solo nodo raíz, y las instancias se van agregando una a una por lo que árbol es actualizado en cada paso. En la actualización se mejora el lugar donde se incluye la nueva instancia, lo que puede conllevar a la reestructuración del árbol o simplemente a incluir la nueva instancia en un nodo ya existente (Garre et al., 2007).

2.2.4.2. Agrupamiento basado en densidad.

Los algoritmos pretenden encontrar grupos de datos teniendo en cuenta la distribución de los puntos, de tal modo que los grupos que se forman tienen una alta concentración de puntos en su interior, mientras que entre ellos aparecen zonas de baja densidad (Gallardo, 2009). Puldón et al. (2012) proporciona la siguiente definición, "se obtienen clústeres basados en regiones

densas de objetos en el espacio de datos, que están separados por regiones de baja densidad". A continuación, se presentan varios algoritmos basados en densidad:

- **DBSCAN (Density-Based Spatial Clustering of Applications with Noise).**

Es un algoritmo basado en densidad "que trata de encontrar todos los puntos centrales, donde los puntos centrales de un grupo son aquellos que tienen una región de vecindad que contiene un número mínimo de puntos para un radio determinado" (Gallardo, 2009).

Su proceso empieza seleccionando un punto *p* arbitrario, en donde si *p* es un punto central, se inicia eligiendo un grupo y se colocan todos los objetos denso-alcanzables desde *p* dentro de su grupo. En cambio, sí *p* no es un punto central se revista otro objeto del conjunto de datos. Este proceso se realiza hasta que todos los registros hayan sido procesados (Pascual, Pla, & Sánchez, 2007). De este modo en un conjunto de datos se distinguen estos tres tipos de puntos:

a) *punto central*: aquellos que se localizan en el interior de un grupo,
b) *punto de ruido*: aquellos que quedan fuera de los grupos formados y,
c) *punto de borde*: no son puntos centrales, pero pertenecen al área de vecindad de uno o más puntos centrales (Gallardo, 2009).

- **DENCLUE (DENsitybased CLUstEring).**

Utiliza una función de densidad local que considera sólo los puntos de datos que en realidad contribuyen a la función global de densidad. Es un algoritmo de dos fases, el cual dados N puntos de una base de datos D, provista de una distancia d, y $x \in D$, define la función de densidad, gradiente y punto atractor (Pascual et al., 2007).

En el primer paso se realiza un pre agrupamiento, en el cual se elabora un mapa perteneciente al espacio de los datos. El mapa se utiliza para acelerar el cálculo de la función de densidad, que requiere para acceder de manera eficiente a las porciones vecinas del espacio de datos. En el segundo paso se realiza la agrupación de los datos, el cual considera solamente los hipercubos más poblados y los conectados a hipercubos más poblados para determinar los grupos (Pascual et al., 2007).

- **OPTICS (Ordering Points To Identify the Clustering Structure).**

La motivación para emplear este algoritmo de densidad, se basa en la necesidad de introducir parámetros de entrada en casi todos los algoritmos de agrupamiento, los cuales son difíciles de calcular por el tamaño de los conjuntos de datos, lo que pretende resolver OPTICS es este problema basándose en el esquema del algoritmo DBSCAN (Pascual et al., 2007).

El algoritmo OPTICS produce un ordenamiento elevado de los elementos en el conjunto de datos que representan la estructura de la agrupación, y es bastante insensible a los parámetros de entrada. OPTICS crea una trama de accesibilidad, en la que los valles corresponden a los grupos. Esta trama es la representación gráfica de los elementos de datos, y la distancia de accesibilidad de un elemento se determina por la distancia a su punto más cercano del núcleo que ya ha sido considerado por el algoritmo OPTICS (Deepak & Roy, 2010).

2.2.4.3. Agrupamiento basado en rejas.

Puldón et al. (2012) mencionan que el agrupamiento basado en rejas, "cuantifica el espacio en un número finito de celdas y aplica operaciones sobre dicho espacio. Recientemente los algoritmos basados en este agrupamiento han sido presentados para datos espaciales".

A continuación se presenta algunos algoritmos basados en rejas (Benítez & Díez, 2005):

- **STING (STatistical INformation Grid).**

Particiona el espacio según niveles, en un número finito de celdas con una estructura jerárquica rectangular, donde cada celda se subdivide en cuatro celdas hijos, y la unión de las celdas hijos forman la celda padre. La celda raíz que es el nivel uno corresponde a toda el área espacial. Las células de nivel de hoja son de tamaño uniforme, y determinan a nivel global la densidad media de los objetos. El algoritmo STING conserva estadísticas de resumen para cada celda en su árbol jerárquico. Como resultado, los parámetros estadísticos de la celda padre pueden ser fácilmente calculados a partir de los parámetros de las celdas hijos (Cheng, Wang, & Batista, n.d.).

En STING en cada nivel se vuelven a particionar las celdas, construyendo un árbol jerárquico similar al del algoritmo BIRCH, una vez culminada la partición del espacio hasta el nivel de detalle deseado, se procede a formar los clústeres asociando celdas con información similar

mediante consultas especializadas, adoptando un enfoque de arriba hacia abajo para el agrupamiento (Benítez & Díez, 2005).

- **CLIQUE (CLustering In QUEst).**

Al igual que STING realiza particiones del espacio identificando de forma automática sub espacios de un espacio de datos de alta dimensión, permitiendo una mejor agrupación debido a que utiliza el principio Apriori. Cada nivel nuevo es una dimensión más hasta alcanzar las *n* dimensiones o características de los objetos (Liu, 2008).

La estructura de partición es en forma hiper-rectángulo. Su funcionamiento es el siguiente:

1. Empieza con una dimensión única y la divide en secciones buscando las más densas, o aquellas que contiene más objetos,
2. incluye la segunda dimensión en el análisis, particionando el espacio en rectángulos, e igual buscando los más densos,
3. luego sigue particionando con cubos en tres dimensiones,
4. cuando acaba con todas las características o dimensiones de los objetos, empieza a definir los grupos y las relaciones entre estos mediante semejanza de densidades y otra información extraída en todos los niveles o dimensiones (Benítez & Díez, 2005).

2.2.4.4. Agrupamiento de particiones.

Montes (2013) afirma que estos algoritmos permiten "organizar los objetos dentro de K grupos de tal forma que sea minimizada la desviación total de cada objeto desde el centro de su grupo o desde una distribución de grupos". A continuación, se presenta algunos algoritmos de agrupamiento de particiones:

- **K-Means (K-medias).**

Creado por Mac Queen en 1967, es uno de los algoritmos más conocidos y empleados entre los algoritmos de agrupamiento particionales, debido a que es de muy simple aplicación y eficaz (Pascual et al., 2007). García Cambronero & Gómez Moreno (n.d.) mencionan que K-means, sigue un procedimiento simple de clasificación de un conjunto de objetos en un determinado número K de clústeres, donde K es determinado a priori.

Su nombre hace referencia al número K de grupos o clústeres a buscar, que deben definirse con anticipación. La idea principal del algoritmo es definir k centroides y luego tomar cada objeto del conjunto de datos, y situarlo en el grupo de su centroide más cercano (Pascual et al., 2007). Para aplicar el algoritmo k-means se debe calcular la distancia euclidiana de todos los objetos.

Distancia Euclidiana.

La distancia euclidiana, es la distancia en línea recta o la trayectoria más corta posible entre dos puntos. Se basa en el teorema de Pitágoras, donde la distancia euclidiana es por lo general la longitud de la hipotenusa del triángulo. En el algoritmo k-means se utiliza habitualmente como una medida de dispersión de clúster para el agrupamiento, ya que disminuye la distancia media entre los puntos y centroides (Microsoft Azure, 2015). La Ecuación 2 muestra la fórmula para calcular la distancia euclidiana.

$$d_{ij} = \sqrt{\sum \left(X_{ik} - X_{jk} \right)^2}$$

Ecuación 2. Fórmula de la distancia euclidiana.
Elaboración: López, A.
Fuente: http://personal.us.es/analopez/ac.pdf

Donde d_{ij} corresponde a la distancia euclidiana entre las variables *i* y *j,* X_{ik} a una de las variables en estudio perteneciente al objeto *ik* y X_{jk} a la misma variable de estudio perteneciente al objeto *jk* (L. Morales, Canessa, Mattar, Orrego, & Matus, 2006).

El algoritmo k-means según Pérez, Cruz, Reyes, & Mexicano (2007) se compone de cuatro etapas:

1. **Iniciación:** este paso selecciona aleatoriamente k objetos del conjunto de datos, formando así los K clústeres iniciales y asignando un centroide por cada grupo (J. Pérez et al., 2007).
2. **Clasificación:** calcula todas las distancias euclídeas de todos los objetos a los k centroides, y se asigna pertinencia a cada objeto al clúster que tenga más cercano (Benítez & Díez, 2005).
3. **Cálculo de centroides:** para cada grupo generado en el paso anterior se vuelve a recalcula el centroide de cada clúster, como la media de todos los objetos que lo componen, buscando minimizar el valor de una función de coste, que es un sumatorio de todos los sumatorios de las distancias euclidianas de los objetos de cada clase al centroide de su respectiva clase (Benítez & Díez, 2005).

4. **Condición de convergencia:** repetir los pasos 2 y 3 hasta que los centros de todos los grupos permanezcan constantes o se cumplan alguna condición de parada (García Cambronero & Gómez Moreno, n.d.).

En base a Benítez & Díez (2005) la eficacia del algoritmo K-means depende de la capacidad del parámetro K, porque si este es mayor o menor que el número real de grupos, se crean grupos ficticios o se agrupan objetos que deberían pertenecer a clústeres distintos.

Inconvenientes: para García Cambronero & Gómez Moreno (n.d.) los principales inconvenientes del algoritmo k-means son los siguientes:

- Se necesita realiza sucesivas ejecuciones del algoritmo para así obtener el resultado más óptimo.
- Se necesita inicializar el número de prototipos o parámetros al principio de la ejecución del algoritmo.
- K-means es susceptible a valores extremos porque distorsionan la distribución de los datos.

- **CLARANS (Clustering Large Applications based up on RANdomized Search).**

Fue propuesto para mejorar la calidad y la escalabilidad del algoritmo CLARA. El algoritmo CLARANS toma como entrada el número *k* de los grupos deseados, pero existe el problema de que este parámetro es a menudo difícil determinar por el tamaño del conjunto de datos (El-Sharkawi & El-Zawawy, 2009).

Para solucionar este inconveniente CLARANS extrae una muestra del conjunto de datos y aplica el algoritmo PAM en la muestra, para encontrar los medoides (k objetos representativos) de la muestra. Si la muestra se selecciona de una manera suficientemente aleatoria, el medoide de la muestra aproximado puede ser el medoide de todo el conjunto de los datos. Finalmente para formar los grupos el algoritmo dibuja múltiples muestras y da la mejor agrupación como la salida (Ng & Han, 2002).

- **PAM (Partitioning Around Medoids).**

En base a Benítez & Díez (2005) el algoritmo PAM es una extensión del algoritmo K-means, en donde cada grupo o clúster está representado por un medoide en vez de un centroide. El algoritmo determina un objeto representativo para cada grupo o clúster. Este objeto se lo llama

medoide y está destinado a ser el objeto más céntrico dentro del grupo. Una vez que los medoides han sido elegidos, cada objeto no elegido se agrupa con el medoide que más se asemeja a sus características. La calidad de agrupación se mide por la disimilitud promedio entre un objeto y el medoide de su clúster (Ng & Han, 2002).

- **EM (Expectation-Maximization).**

El algoritmo forma parte de la familia de modelos que se conocen como *Finite Mixture Models*, los cuales se pueden utilizar para segmentar conjuntos de datos (Garre et al., 2007). Benítez & Díez (2005) afirman que "EM asigna cada objeto a un clúster predefinido, según la probabilidad de pertenencia del objeto a ese grupo concreto". Este método permite encontrar una estimación de máxima verosimilitud para un parámetro θ de una distribución (Camarena Ibarrola, n.d.).

El algoritmo EM comienza adivinando los parámetros de las distribuciones y los utiliza para calcular las probabilidades de que cada uno de los objetos pertenezca a un clúster, y utiliza el valor de esas probabilidades para volver a calcular los parámetros de las probabilidades, hasta que los objetos pertenezcan a un clase (E. Morales, 2012). El algoritmo EM consta de tres pasos (Navarro, 2001):

1. ***Paso de iniciación:*** toma los datos de entrada y los introduce en una matriz de datos característicos *X* de dimensiones *m* * *n*, donde *m* es la cantidad de datos y *n* es la dimensión de cada dato (Navarro, 2001).

2. ***Paso Expectativa:*** este paso "utiliza los valores de los parámetros iniciales o los proporcionados por el paso de Maximización de la iteración anterior, consiguiendo diferentes maneras de la función de densidad de probabilidad (FDP) buscada" (Garre et al., 2007).

3. ***Paso Maximización:*** aquí se obtiene nuevos valores de los parámetros a partir de los datos proporcionados en el paso de Expectativa.

El algoritmo EM puede identificar grupos o clases de distintas formas geométricas, lo que implica un alto coste computacional para conseguir un buen ajuste de los parámetros de los modelos (Benítez & Díez, 2005).

2.2.5. Categorización automática.

La presente técnica se utiliza en la Minería Textual para clasificar documentos en una serie de categoría preestablecidas. La técnica en la actualidad ha recibido gran atención debido al gran aumento de información digital disponible en las organizaciones, instituciones, etc., y por la necesidad de encontrar conocimiento en dicha información. La categorización de documentos puede definirse, como la tarea de separar documentos en grupos según la afinidad que guardan sus elementos. Es aplicable en distintos procesos (Brun & Senso, 2004).

- Clasificación e indización automática de documentos.
- Filtrado de contenidos (por ejemplo, en la distribución de noticias o newsfeeds).
- Asignación de páginas y sitios web a listas de categorías predefinidas en portales tipo Yahoo! o dmoz.org.
- Resolver la ambigüedad en palabras con polisemia.

Además Brun & Senso (2004) indican que "existen dos tipos de categorización: single-label y de multilabel". Dependiendo de la aplicación para la categorización de documentos, estos pueden clasificarse en una clase (single-label) o más clases (miltilabel).

Por ejemplo:

- Una noticia sobre la afición deportiva del presidente,

Puede clasificarse tanto en la columna de noticias políticas o en la columna de noticias deportivas. Este tipo de clasificación se denomina clasificación Multi-etiqueta (multi-label). Sin embargo, existen documentos que solo pueden ser clasificados en una sola categoría.

Por ejemplo:

- Una noticia sobre la clasificación de Ecuador al mundial,

Puede clasificarse solo en la columna de noticias deportivas. Este tipo de clasificación se denomina clasificación de etiqueta única (single-label).

2.2.6. Indexación Semántica Latente (LSI).

La Indexación Semántica Latente fue descrita en los años 1990 por Deerwester, Dumais, Furnas, Landauer y Harshman, como un método para recuperación de información (Botana, 2010). Para Paulsen & Ramampiaro (2009) LSI, es un enfoque para la indexación automática y para la recuperación de documentos fundamentados en la noción de concepto. Además, mencionan que fue constituida a partir del modelo de espacio vectorial (VSM), con el objetivo de superar los desafíos principales, como la recuperación de documentos basados en el contenido conceptual, en donde los términos individuales ofrecen pruebas poco confiables sobre el contenido de los documentos.

Para Sierra Araujo (2006) la Indexación Semántica Latente es una "variante del modelo vectorial". La cual simboliza documentos mediante una matriz (matriz de términos por documentos), y comprime los vectores que personalizan a los documentos en "vectores de dimensión reducida". LSI hace uso de cálculos estadísticos y de la técnica de la descomposición en valores singulares (Singular Value Descomposition, SVD) de una matriz, para identificar modelos de relaciones entre los términos (palabras) y conceptos comprendidos en una colección de datos, es decir, LSI busca emparejar el contenido de los documentos por conceptos en lugar de por términos, permitiendo que un documento sea recuperado si comparte conceptos con otro documento que es importante para una determinada consulta, y todo esto es posible gracias a la utilización de SVD. Sierra Araujo (2006) indica que LSI consta de las siguientes fases:

1. Construcción de la matriz
2. Descomposición en valores singulares de la matriz
3. Reducción de la dimensión del espacio
4. Búsqueda semántica

1. Construcción de la matriz

Sierra Araujo (2006) mencionan que para poder aplicar la indexación semántica latente (LSI), primero se debe crear una matriz *M* de *términos por documentos* para un conjunto de documentos, en la cual se puede identificar con qué frecuencia un término o n-grama aparece en los documentos. Se puede utilizar los siguientes pesos para la matriz términos por documentos:

- *Peso local:* "mide la importancia del término *i* en el documento *j"*, es decir, se tiene tres opciones: a) conservar la matriz de frecuencias original, b) transformarla a binaria, en donde solo se toma en cuenta la aparición o no del término, o c) "reducir las diferencias entre frecuencias utilizando la función de logaritmo" (Sierra Araujo, 2006).

- *Peso global:* "mide la importancia del término *i* a nivel del corpus" (Sierra Araujo, 2006), es decir, se tiene tres opciones: a) proporcionar el mismo nivel de importancia a todos los términos, b) normalizar los vectores que simbolizan a los términos y, c) calcular el idf (frecuencia inversa de documento) o calcular la entropía.

La figura 4 presenta un ejemplo de una matriz términos por documentos.

	Terms							
Docs	arquitectur	sgbd	arregl	bas	dat	relacional	clas	conexion	enrut
ArqSegRed.txt	0	0	0			0	0	1	4
BDAvan.txt	0	1	0			3	0	0	0
FundBD.txt	0	1	0			1	0	0	0
FunProg.txt	0	0	7			0	4	0	0
ProgAlg.txt	0	0	7			0	7	0	0
ProgAvan.txt	0	0	0			1	1	1	0
RedSistDist.txt	0	0	0			0	0	5	10

Figura 4. Ejemplo de una matriz términos por documentos (M), creada a partir un conjunto de documentos.
Elaboración: propia.

2. Descomposición en Valores Singulares

La descomposición en valores singulares (SVD), es una técnica de reducción de dimensiones de matrices. Botana (2010) menciona que SVD se emplea con el propósito de reducir el número de dimensiones de la matriz términos por documentos original, en un número mucho más manejable, pero sin que la matriz original pierda la información sustancial. Pero lo interesante no es solo reducir la dimensión de la matriz, sino "crear un espacio semántico vectorial en el que tanto términos como documentos están representados por medio de vectores que contengan sólo la información sustancial para la formación de conceptos" (Botana, 2010). Representar a un conjunto de documentos mediante el lenguaje vectorial permite que se realicen comparaciones por medio de distancias euclidianas, de cosenos y de diferentes medidas de similitud.

En álgebra se asegura que toda matriz rectangular $M \in R^{m \times n}$ puede ser escrita como producto de otras tres matrices, donde $\Sigma \in R^{m \times n}$ (S) es una matriz diagonal de valores singulares reales

y no negativos, es decir, mayores o iguales a cero (σ1 ≥ σ2 ≥. . . ≥ σn), y las matrices U y V son matrices constituidas por vectores singulares (Sierra Araujo, 2006).

- *U* simboliza a los términos en la nueva matriz.
- *V* simboliza a los documentos en la nueva matriz.
- *S* es una matriz diagonal que posee los valores singulares de *M* en orden descendente.

3. Reducción de la dimensión del espacio

El siguiente paso es realizar la reducción de la dimensión del espacio a las matrices U y V, lo cual se puede realizar una vez que se dispone de una descomposición en valores singulares de la matriz *M*, es posible aproximarla por otra matriz M_p de rango p ≤ r, calculada para los primeros *p* valores singulares (Sierra Araujo, 2006). La Ecuación 3 presenta la fórmula para reducir la dimensión del espacio a las matrices U y V.

$$M_p = U_p \sum_p V_p^T = \sum_{i=1}^{p} \quad \sigma_i \sigma_u \sigma_i^T$$

Ecuación 3. Fórmula para reducir la dimensión del espacio de una matriz.
Fuente: Aprendizaje Automático: conceptos básicos y avanzados
Elaboración: Sierra Araujo (2006)

En donde U_p y V_p son matrices formadas por las *p* primeras columnas de las matrices U y V. Como resultado de esta operación se pasa del espacio vectorial creado por la matriz *M,* al espacio creado por las columnas de M_p. De la elección eficiente del valor de *p* dependen los resultados (Sierra Araujo, 2006). Ahora se puede decir que dichas matrices ya no poseen ruido que interfiera para obtener la similitud de los documentos con una consulta.

La Figura 5 muestra gráficamente el proceso de reducción de dimensiones. En la imagen de la izquierda cada uno de los términos está representado por cuatro documentos o dimensiones (d1, d2, d3, d4). En cambio, en la imagen de la derecha los términos son simbolizados por dos dimensiones abstractas (c1, c2) pero de mayor beneficio funcional, donde a cada término se le deduce la probabilidad de estar representado en un concepto. Esto se puede observar en la figura de la derecha, en donde el término t2 se lo vincula con el documento d2, a pesar de que en la figura de la izquierda esto no se origine.

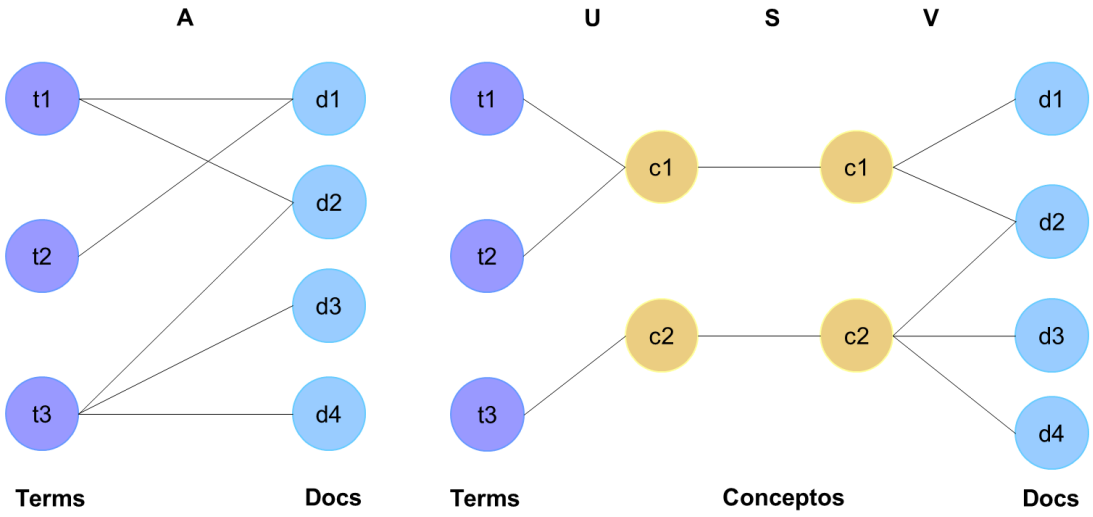


Figura 5. Representación gráfica de la reducción de dimensiones de una matriz realizar por SVD.
Fuente: La técnica del Análisis de la Semántica Latente (LSA/LSI) como modelo informático de la comprensión del texto y el discurso.
Elaboración: Botana (2010)

4. Búsqueda semántica

Una vez que se reduce la dimensión del espacio a las matrices se puede llevar a cabo "los cálculos de similitud semántica con los vectores de dicho espacio" (Sierra Araujo, 2006). En la extracción de información al ingresar una solicitud en el buscador, se supone que se ha entregado un *vector de búsqueda* $q^T = (q_1, q_2, q_3, \ldots, q_m)$, que LSI se encarga de construir mirando si cada término *i* de su base de datos aparece (es decir, si $q_i = 1$) o no en la solicitud de búsqueda (es decir, si $q_i = 0$). Se procede a utilizar el preprocesado a través de pesos y en seguida se traslada el vector **q** al espacio reducido y esto se realiza de igual forma con los documentos del corpus $q_p = q^T U_p \sum_p^{-1}$.

Sierra Araujo (2006) indica que el vector $\mathbf{q_p}$ contiene las coordenadas de la proyección del vector **q** en el espacio reducido. Una vez realizo esto se procede a calcular la semejanza que existe entre el vector de búsqueda y los vectores que simbolizan a los documentos. Para la similitud "se puede hacer uso del producto escalar entre los dos vectores como el coseno del ángulo θ que forman. Se recomienda utilizar el coseno porque se obtiene mejores resultados" (Sierra Araujo, 2006). La Ecuación 4 presenta la fórmula del coseno del ángulo θ.

$$\cos\theta = \frac{q_p^T d_p}{||q_p||\,||d_p||}$$

Ecuación 4. Fórmula del coseno del ángulo θ.
Fuente: Aprendizaje Automático: conceptos básicos y avanzados
Elaboración: Sierra Araujo (2006)

Al hacer uso del coseno Sierra Araujo (2006) menciona que si el ángulo θ que crean los dos vectores es pequeño, el coseno será próximo a 1, por lo que se interpretará que la petición de búsqueda y el documento son semánticamente similares. Los documentos que poseen un coseno superior a un umbral serán devueltos al usuario.

2.3. Tecnologías que utilizan el proceso de Minería de Texto

En base a Gupta & Lehal (2009) "en el campo del procesamiento del lenguaje natural se han producido tecnologías que enseñan a los ordenadores el lenguaje natural para que puedan analizar, comprender, e incluso generar texto". Si bien las diferencias entre el lenguaje humano e informático son extensas, se han desarrollado avances tecnológicos que han empezado a reducir esta diferencia.

A continuación se muestra algunas tecnologías que emplean y que se pueden utilizan en el proceso de Minería de Texto (Gupta & Lehal, 2009):

- Extracción de información (information extraction).
- Seguimiento del tema (topic tracking).
- Resumen (Summarization).
- Categorización (Categorization).
- Agrupación (Clustering).
- Vinculación del concepto (Concept linkage).
- Visualización de la información (Information visualization).
- Preguntas y respuestas (Question answering).

2.3.1. Extracción de información.

La técnica de extracción de información (EI) es empleada por las computadoras como punto de partida para el análisis de texto no estructurado, empleando software especializado para tratar estos grandes volúmenes de texto. "El software para la extracción de información identifica frases y relaciones dentro del texto clave. Lo hace mediante la búsqueda de secuencias predefinidas en el texto, un proceso denominado coincidencia de patrones" (Gupta & Lehal, 2009). Las aplicaciones para extraer la información relevante de los textos permiten identificar y definir las entidades (personas, empresas, etc.) y sus relaciones, revelando información semántica significativa.

Para Karanikas et al., (n.d.) “la extracción de información es el mapeo de textos del lenguaje natural (como informes, artículos de periódicos y revistas, etc.) en representación predefinida, estructurada, o plantillas, que, cuando se llena, representa un extracto de la información clave del texto original”. Las técnicas lingüísticas como la tokenización, la eliminación de palabras vacías, la lematización, los segmentos repetidos o frases, etc., alimentan a los sistemas de extracción de información. Siendo la técnica de pre-procesamiento lingüístico el primer paso para la extracción de información (R. Pérez, n.d.).

El proceso de extracción de información aborda el problema de transformar un conjunto de documentos de texto, a una base de datos estructurada. “La base de datos construida por el módulo de extracción de información, puede ser proporcionada al módulo de descubrimiento de conocimiento en la base de datos (KDD) para su posterior extracción de conocimiento” (Gupta & Lehal, 2009). La base de datos es creada a partir del conjunto de textos, mediante el uso de patrones de EI aplicados a cada uno de los documentos, para así crear una colección de registros estructurados, a los cuales se les puede aplicar técnicas de descubrimiento de conocimiento en la base datos (KDD) para descubrir relaciones interesantes (Gupta & Lehal, 2009). La Figura 6 presenta el proceso para la extracción de información.

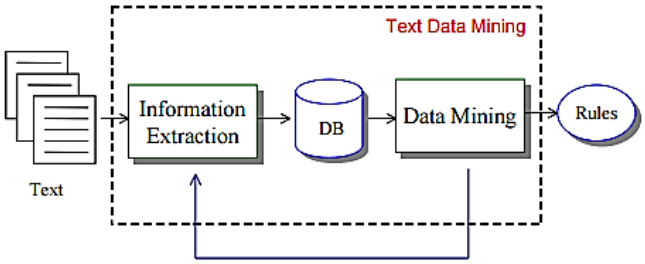


Figura 6. Visión general de la Extracción de Información basado en el marco de Minería de Texto.
Fuente: http://doi.org/10.4304/jetwi.1.1.60-76
Elaboración: Gupta & Lehal (2009)

2.3.2. Seguimiento del tema.

La presente técnica se basa en ofrecer a los usuarios recomendaciones sobre temas de interés para los mismos. Los sistemas de seguimiento de temas funcionan manteniendo o guardando la información de los perfiles de los usuarios, y basándose en la búsqueda de documentos, predice otros documentos relaciones y de interés para el usuario. Gupta & Lehal (2009) mencionan que “Yahoo! ofrece una herramienta de seguimiento de temas libre (www.alerts.yahoo.com) que permite a los usuarios elegir las palabras claves y les notifica cuando noticias relacionadas con esos temas hay disponibles”.

El proceso de seguimiento del tema es útil en un sinnúmero de áreas como, por ejemplo: se puede aplicar en la industria, creando un sistema para alertar a las compañías cuando un competidor lanza un nuevo producto o está en las noticias. Manteniendo así informada a la organización de los productos que hay en el mercado, de los cambios del mismo y además realizando un seguimiento de sus propios productos. En el área de la educación es gran utilidad para hacer seguimiento a las últimas investigaciones y publicaciones de temas de interés para docentes y estudiantes (Kaur & Gupta, 2012). A las herramientas de seguimiento de temas se las puede evidenciar mucho en páginas web, en donde se las utiliza para suscribir a los usuarios a temas específicos para que, por medio de correo electrónico, estos reciban notificaciones sobre los temas de interés. A continuación, se presentan algunas de estas herramientas (Macmanus, 2010):

- Google alerts[1].
- LazyFeed[2].
- Yahoo Pipes[3].
- Ensembli[4].
- Regator[5].
- GigaAlert[6].

Sin embargo, Patel & Sharma (2014) indican que la tecnología de seguimiento de temas "tiene aún limitaciones. Por ejemplo, si un usuario crea una alerta de *Minería de Texto*, el usuario recibirá varias noticias o documentos sobre minería de minerales, y muy pocos temas serán realmente sobre Minería de Texto". Este problema puede ser debido a la cantidad de ruido y a la dificultad de filtrar la información requerida. Las palabras claves son muy importantes en artículos, páginas web, revistas, etc., debido a que proporcionan una descripción de alto nivel del contenido a los lectores. Como en la actualidad los documentos en línea aumentan rápidamente de tamaño con el crecimiento del Internet, "la extracción de las palabras claves se ha convertido en una base de varias aplicaciones de Minería de Texto, tales como motores de búsqueda, categorización de texto, resúmenes, etc." (Gupta & Lehal, 2009).

La extracción de palabras claves manualmente es imposible por la gran cantidad de documentos que se generan en la web, por lo cual se necesita un proceso automatizado que

1 Google alerts: https://www.google.com/alerts
2 LazyFeed: https://www.crunchbase.com/organization/lazyfeed
3 Yahoo Pipes: http://pipes.yahoo.com/pipes/
4 Ensembli: https://www.crunchbase.com/organization/ensembli
5 Regator: http://regator.com/
6 GigaAlert: http://gigaalert.com/

extraiga las palabras claves de los artículos de noticias (Gupta & Lehal, 2009). A continuación, se presenta la arquitectura de un sistema de seguimiento de temas (Kaur & Gupta, 2012). La Figura 7 presenta la arquitectura de un sistema de seguimiento de tema.

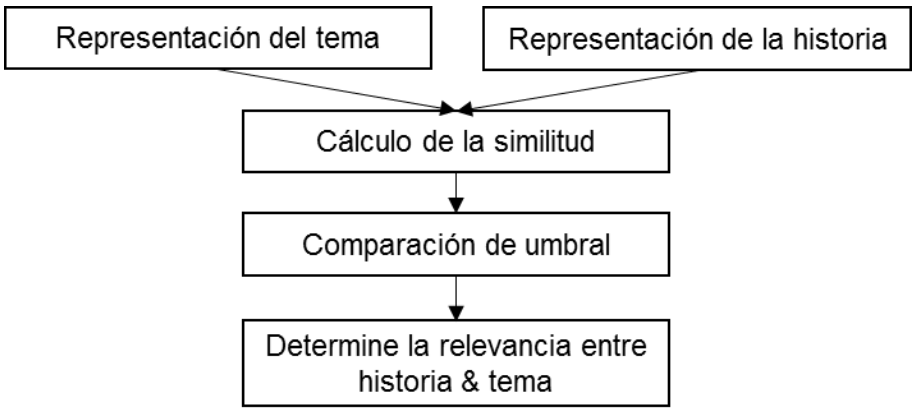


Figura 7. Arquitectura de un sistema de seguimiento de tema.
Fuente: http://www.ijarcsse.com/docs/papers/May2012/Volum2_issue5/V2I500532.pdf
Elaboración: Kaur & Gupta (2012)

2.3.3. Resumen (Summarization).

La aplicación de la Minería de Texto en la extracción de resúmenes es de gran utilidad para los usuarios, debido a que les permite de forma rápida averiguar si un determinado documento cumple o no con sus necesidades y así inviertan su tiempo en leerlo. Gupta & Lehal (2009) aluden que "con textos extensos, los sistemas de resumen de texto procesan y resumen el documento en el tiempo que le tomaría al usuario leer el primer párrafo". Generalmente las personas leen todo el contenido del documento para obtener una comprensión completa del texto, y luego proceder a sacar un resumen. La clave de estos sistemas es obtener un resumen detallado manteniendo sus puntos principales y el significado general del documento, evitando a las personas tener que leer documentos que no son de su interés.

Sin embargo, los ordenadores no tienen las capacidades lingüísticas que tienen los seres humanos para el procesamiento del lenguaje natural. Por esta razón Gupta & Lehal (2009) destacan que "una de las estrategias más utilizadas por las herramientas de resumen texto y extracción de frases, es la de extraer sentencias importantes de un artículo y ponderar estadísticamente estas frases". Por ejemplo, las herramientas de extracción de resumen pueden identificar en el contenido del documento la frase "en conclusión" y, determinar que el texto que le sigue a la frase son puntos principales del documento. La función de Microsoft Word de AutoSummarize es un ejemplo de resumen de texto.

El proceso de un sistema de resumen automático se divide en tres pasos (Gupta & Lehal, 2009):

1. En el paso de pre-procesamiento se consigue una representación estructurada del texto original.
2. En el paso de procesamiento el algoritmo transforma la estructura del texto en una estructura de resumen.
3. En el paso de generación se obtiene el extracto final de la estructura del resumen.

Además Gupta & Lehal,(2009) clasifican en dos grupos a los métodos de resumen:

a) "*Enfoques superficiales,* que se limitan al nivel sintáctico de representación y tratan de extraer las frases más destacadas del texto".
b) "*Enfoques más profundos,* que asumen un nivel semántico de representación del texto e involucra un procesamiento lingüístico de alto nivel".

2.3.4. Categorización (Categorization).

Cardoso (2010) proporciona la siguiente definición sobre la categorización. "La categorización de documentos de texto es una aplicación de la Minería de Texto que asigna a los documentos una o más categorías, etiquetas o clases, basadas en el contenido". El objetivo de la categorización automática de textos, es clasificar un conjunto de documentos en un número fijo de categorías predefinidas, donde cada documento puede pertenecer a una o más categorías. La categorización automática para Figuerola, Alonso, & Zazo, (2000) es "un proceso de aprendizaje, durante el cual el programa capta las características que distinguen cada categoría o clase de las demás, es decir, aquéllas que deben poseer los documentos para pertenecer a esa categoría".

Los sistemas de categorización tratan a los documentos como una bolsa de palabras, es decir, no procesan la información real como lo hace el proceso de extracción de información. Es por esto que Gupta & Lehal, (2009) sugieren que "la categorización sólo cuenta las palabras que aparecen y a partir de los recuentos, identifica los principales temas que el documento cubre".

La categorización se complementa con las tecnologías de resumen y seguimiento del tema, con la finalidad de especificar aún más la relevancia de un documento y así proporcionando a los usuarios información de su interés.

Por ejemplo vamos a considera un conjunto de documentos etiquetados de la siguiente manera (Gupta & Lehal, 2009):

- D = {d1, d2, d3, ……dn}

Donde estos documentos pertenecen al conjunto de clases de C:

- C = {c1, c2, c3, ……..cn}

El trabajo de los sistemas de categorización es entrenar al clasificador utilizando estos documentos, para así asignar categorías a cada uno de los documentos. En la fase de entrenamiento, "los documentos *n* se organizan en carpetas separadas *p*, donde cada carpeta corresponde a una clase y en el siguiente paso, el conjunto de datos de entrenamiento se prepara a través de un proceso de selección de características" (Gupta & Lehal, 2009).

2.3.5. Agrupación (Clustering).

En base a Montes (2013) Clustering "es el proceso de agrupar los datos en clases o en clústeres, de tal forma que, los datos de un mismo clúster tienen una alta similitud y a su vez, son muy diferentes de los de otro clúster". La tecnología de clustering es utilizada por los sistemas de gestión de información de las organizaciones, por las grandes cantidades de información que contienen (Gupta & Lehal, 2009).

A diferencia de la categorización automática, el proceso de agrupamiento crea grupos entre documentos de forma instantánea, es decir, los sistemas de agrupamiento resolverán que grupos se van a generar a partir de la similitud que determinen entre los documentos de la colección (Brun & Senso, 2004).

La agrupación tiene distintas aplicaciones, Brun & Senso (2004) muestran las siguientes:

- Encontrar la relevancia de los documentos implícitos en cada grupo tras la lectura de tan sólo uno de sus representantes.
- Identificar relaciones entre documentos en una colección que previamente se desconocían.
- Identificar duplicados potenciales y documentos que por tener información similar pueden no ser relevantes dentro del grupo.
- Mejorar la organización de los resultados devueltos por un motor de indexación.

Un ejemplo donde se emplea clustering es Carrot2[7], el cual es un sistema de recuperación basado en técnicas de clustering de documentos y contenidos web, sin requerir de bases de conocimiento externas tales como taxonomías o contenidos preclasificados. Carrot2 hace uso de algoritmos de agrupamiento jerárquico, con los que es capaz de agrupar los contenidos de los motores de búsqueda de Google o Bing (Ochando, 2011).

2.3.6. Vinculación del concepto (Concept linkage).

En base a Gupta & Lehal (2009) la vinculación del concepto "conectan documentos relacionados mediante la identificación de sus conceptos compartidos, y ayudan a los usuarios a encontrar información que quizá no habrían encontrado con el uso de métodos tradicionales de búsqueda". Concept linkage según Thilagavathi & Shanmuga (2014) "es un concepto valioso en la Minería de Textos especialmente en los campos de la biomedicina, en donde la investigación es elevada y por ende es imposible para los investigadores leer todo el material y hacer asociaciones con otras investigaciones".

La idea de conectar documentos que comparten conceptos, es que estas conexiones inesperadas puedan generar nuevas hipótesis al investigar con mayor profundidad y puedan conducir a nuevos conocimientos. Esta metodología se puede aplicar utilizando mapas complejos de visualización que muestran las redes de relaciones, entre un grupo de conceptos de una colección de documentos. Hacer uso de la visualización ofrece la posibilidad de describir relaciones que son desconocidas y no consideradas (Lavengood & Kiser, 2007).

Por ejemplo, una aplicación de Minería de Texto que aplique concept linkage, sería capaz de identificar fácilmente un vínculo entre documentos que contengan temas X e Y, e Y y Z, que son las relaciones que se conoce, pero la aplicación también podría identificar un posible vínculo entre X y Z, algo que un investigador no encontraría debido a la gran cantidad de información que tendría que ordenar para hacer la conexión (Thilagavathi & Shanmuga, 2014).

2.3.7. Visualización de la información (Information visualization).

Fan, Wallage, & Rich (2006) proporcionan la siguiente definición: "la Minería de Texto visual, o visualización de la información, coloca grandes fuentes textuales en una jerarquía visual o mapas y proporciona capacidades de navegación, además de la búsqueda simple".

[7] Carrot2: http://search.carrot2.org/stable/search

Una de las estrategias más comunes utilizadas en la Minería de Texto es identificar las entidades importantes en el texto y tratar de mostrar las conexiones entre esas entidades (Hearst, 2009). La herramienta VOSviewer[8] presenta una amplia funcionalidad en Minería de Texto, proporcionando soporte para la creación de mapas basados en corpus de documentos, facilitando la visualización de las relaciones que existen entre los términos de dicho documentos (Van & Waltman, 2011).

Van & Waltman (2011) dividen la construcción de un mapa para la visualización de información en cuatro pasos:

1. **Identificación de sintagmas nominales.** En este paso se realiza el etiquetado gramatical (identificación de verbos, sustantivos, adjetivos, etc.), luego se realiza un filtro lingüístico para seleccionar secuencias de palabras que se compongan de sustantivos y adjetivos, y finalmente se convierte los sintagmas nominales plurales en otros más singulares.
2. **Selección de las frases más relevantes del sustantivo.** Para cada frase nominal se determina la que tiene mayor número de ocurrencias y son agrupadas en clústeres.
3. **Mapeo y agrupación de los términos**. Se utiliza el marco unificado VOSviewer, en donde son cargados los datos de los documentos del corpus (Van & Waltman, 2011).
4. **Visualización de resultados del mapeo y clustering.** La herramienta VOSviewer ofrece varios tipos de visualizaciones. Además ofrece la posibilidad de acercar, desplazar y realizar búsquedas para tener un examen detallado del mapa (Van & Waltman, 2011).

2.3.8. Preguntas y respuestas (question answering).

Una de las áreas del procesamiento del lenguaje natural es realizar consultas o preguntas y respuestas (Q & A), que trata de encontrar la mejor respuesta a una pregunta dada. En la actualidad esta tecnología es muy utilizada en sitios web, permitiendo al usuario realizar una pregunta para enseguida obtener una respuesta. La tecnología puede utilizar múltiples técnicas de Minería de Texto (Gupta & Lehal, 2009).

Chali, Joty, & Hasan, (2009) mencionan que “los motores de búsqueda han demostrado ser adecuados, aunque no hay ninguna limitación en la expresividad del usuario en términos de formulación de consultas, existen ciertas limitaciones en lo que hace el motor de búsqueda

[8] VOSviewer: http://www.vosviewer.com/Home

con la consulta". Esto se debe a que, si se realiza consultas complejas se requieren de múltiples documentos para obtener la respuesta indicada. Por ejemplo, si se busca el número de componentes académicos de las universidades de Ecuador y Colombia, y en cual existe mayor número de desertores, la respuesta es probable que figure en varios documentos por lo que existe una limitación.

CAPÍTULO 3: ANÁLISIS Y SELECCIÓN DE LOS ALGORITMOS DE CLUSTERING

3.1. Análisis de algoritmos

En el presente capítulo se analiza los algoritmos de clustering disponibles para la agrupación de los planes docentes. De la clasificación de los algoritmos de agrupamiento disponibles, se ha considerado analizar los algoritmos de clustering jerárquico y por particiones, por las características y ventajas que ofrecen para llevar a cabo la agrupación de los planes docentes. Al culminar el análisis se realiza la selección de uno de ellos y se utiliza para cumplir con el objetivo del trabajo.

Adicional, se estudia al método de Indexación Semántica Latente (Latent Semantic Indexing - LSI), el cual es una variación del modelo de espacio vectorial que utiliza la técnica de descomposición en valores singulares (SVD), para identificar la similitud que existe entre los términos de los documentos del corpus.

3.1.1. Algoritmos jerárquicos.

Realizan una descomposición jerárquica en forma de árbol de los objetos. Se divide en dos categorías (Benítez & Díez, 2005):

1. **Divisivos (tom-down):** Empiezan considerando a todos los objetos como un único cluster o grupo, el cual se va ramificando en clústeres más pequeños.
2. **Aglomerativos (bottom-up):** Empiezan desde abajo hacia arriba, considerando a todos los objetos como clústeres separados, y difundiéndolos en grupos más grandes según una medida de distancia.

Ventaja

- La principal ventaja que se puede mencionar es la flexibilidad con respecto al nivel de granularidad, son fáciles de manejar y son aplicables a cualquier tipo de atributo (González, 2010).

Desventaja

- La desventaja que se encuentra es la no existencia de un criterio de parada y que luego de formar los clusters, no vuelven a ser visitados para mejorarlos (González, 2010).

3.1.2. Algoritmos por partición.

Su funcionamiento consiste en dividir un espacio en conjuntos o clústeres de datos. En donde, dados *n* objetos en un espacio dimensional se determina la partición de los objetos en K grupos, de modo que, si los objetos en un grupo son similares entre sí, se agrupan formando un determinado clúster. Estos algoritmos pueden presentan o no presentar un conocimiento a priori del número de clústeres, en el que debe ser dividido el conjunto de datos (Bedregal, 2008).

Ventaja

- Una vez formados los clústeres son revisitados posteriormente y los puntos reubicados para mejorar el agrupamiento (González, 2010).

Desventaja

- Tiene como desventaja que fallan cuando los puntos de un clúster están muy cerca del centroide de otro grupo.

Una vez estudiados los dos tipos de agrupamiento se opta por utilizar los algoritmos de agrupamiento basado en particiones, por las ventajas, por su funcionamiento, por la cantidad de algoritmos que existen y por el gran uso en el agrupamiento de documentos.

De los algoritmos de agrupamiento por particiones descritos en la sección 2.2.4.4, se selecciona varios de estos en base a sus características, para analizarlos y compararlos entre sí, con el propósito de seleccionar a uno.

Adicional a estos algoritmos se analiza el algoritmo LSI (Indexación Semántica Latente) con el propósito de determinar si también se puede hacer uso de esta técnica para agrupar los planes docentes: A continuación, se presentan los algoritmos seleccionados para el análisis.

- K-means.
- EM
- PAM
- LSI

3.2. Comparación de algoritmos

La Tabla 1 presenta las características de los algoritmos seleccionados.

Tabla 1. Comparación entre algoritmos

	K-means	EM (Expectativa - Maximización)	PAM (Particiones Alrededor de medoides)	LSI (Indexación Semántica Latente)
Características	❖ Agrupa un conjunto de datos en un número de k clústeres. ❖ Empieza con un conjunto aleatorio de centroides en cada uno de los grupos. ❖ Continúa reasignando los datos a los centroides basándose en la similitud entre los datos y el centroide.	❖ Asigna cada objeto a un clúster predefinido, según la probabilidad de pertenencia del objeto a ese grupo concreto. ❖ Permite encontrar una estimación de máxima verosimilitud para un parámetro θ.	❖ Es una extensión del algoritmo K-means. ❖ Establece un medoide para cada clúster. ❖ Cada objeto no elegido se agrupa con el medoide que más se asemeja a sus características.	❖ Permite la indexación automática y la recuperación de documentos basados en la noción del concepto. ❖ Su idea es pasar de una colección de términos a una colección de conceptos, para obtener la asociación entre términos y documentos. ❖ Para obtener la estructura semántica se utiliza SVD.
Funcionamiento	❖ Se compone de cuatro pasos: 1. En el paso de Inicio se selecciona aleatoriamente k objetos, formando los K clústeres originales y asignando un centroide por cada clúster. 2. En el paso de clasificación se calcula las distancias euclideanas de todos los k centroides. 3. En el paso de cálculo de centroides, para cada grupo formado en el paso anterior se vuelve a calcular el centroide. 4. En el paso de convergencia se repiten los pasos dos y tres hasta que se cumpla alguna condición de parada.	❖ Se compone de tres pasos: 1. En el paso de inicio, se introduce los datos en una matriz de datos de dimensiones m (cantidad de datos) * n (es la dimensión de cada dato). 2. En el paso E utiliza los parámetros iniciales o los proporcionados por el paso M, para conseguir la probabilidad deseada. 3. El paso M obtiene nuevos valores de los parámetros a partir de los datos proporcionados por el paso E.	❖ Se compone de dos pasos: 1. Se consigue un conglomerado inicial seleccionando los objetos que hacen decrecer la función objetivo tanto como sea posible hasta hallar *k* objetos. 2. Se realiza un reajuste y mejora de los clústeres construidos, devolviendo el objeto más central en cada grupo.	❖ Se compone de cuatro pasos: 1. Se construye la matriz de términos por documentos a partir del corpus. 2. Se aplica la descomposición en valores singulares a la matriz. 3. A las nuevas matrices resultado de aplicar SVD, se les reduce su dimensión del espacio. 4. Se realizar la búsqueda semántica para identificar la similitud existente entre los términos de los documentos.
Ventajas	❖ Logra resolver problemas de clustering de forma muy eficiente.	❖ Requiere poco espacio de almacenamiento y puede llevarse a cabo en un ordenador sencillo.	❖ La calidad de agrupación la mide por la disimilitud promedio entre un	❖ *Dimensiones latentes verdaderas*, LSI recobra la estructura semántica original del espacio y sus dimensiones

	K-means	EM (Expectativa - Maximización)	PAM (Particiones Alrededor de medoides)	LSI (Indexación Semántica Latente)
	❖ Su implementación es simple y rápida. ❖ La mayoría de herramientas de minería de datos incluyen este algoritmo. ❖ Puede adaptarse fácilmente a diferentes espacios/datos.	❖ Es numéricamente estable con cada iteración, es decir, en cada iteración aumenta la verosimilitud. ❖ Es fácil de implementar, y se basa en cálculos de datos completos. ❖ Puede usarse para proporcionar estimaciones de los valores de los datos perdidos.	objeto y el medoide de su clúster. ❖ Trabaja bien en base de datos pequeñas. ❖ Minimiza la suma de disimilitudes en vez de la disimilitud promedio.	originales (Barbara, 2000). ❖ LSI usa la *sinonimia*, que se refiere al efecto de que un determinado concepto puede estar descrito utilizando términos diferentes. ❖ Mediante la *polisemia* LSI busca remover el ruido de los datos y aumentar la calidad de la consulta. ❖ *Dependencia entre términos,* LSI ubica los términos en el espacio reducido de manera que expresen las correlaciones en su uso internamente de los documentos (Barbara, 2000).
Desventajas	❖ Necesita realiza sucesivas ejecuciones del algoritmo para así obtener el resultado más óptimo. ❖ Es aplicable cuando es posible calcular el centroide, como en el caso de los documentos, pero es de difícil aplicación en atributos categóricos. ❖ Se necesita inicializar o conocer el número de parámetros al principio de la ejecución del algoritmo. ❖ Es susceptible a valores extremos porque distorsionan la distribución de los datos.	❖ En algunos problemas, el paso de Esperanza (E) puede ser analíticamente intratable. ❖ Puede ser desesperadamente lento en problemas donde hay demasiada información incompleta. ❖ No tiene un procedimiento incluido para proporcionar una estimación de la matriz de covarianza de las estimaciones de los parámetros.	❖ Tiene un alto consumo computacional debido a la búsqueda de los medoides. ❖ Es lento cuando trabaja con información de bases de datos grandes.	❖ El *almacenamiento*, en LSI los vectores son número reales, en cambio las frecuencias originales de los términos son números enteros, por lo que esto añade costos de almacenamiento. ❖ La *eficiencia*, debido a que LSI compara la consulta con todos los documentos del corpus, en lugar de solo examinar los documentos que contienen algunos términos en común con la consulta. Esto imposibilita aplicar el modelo a grandes colecciones de datos (Barbara, 2000).

Elaboración: propia.

3.3. Algoritmo Seleccionado

Una vez que se realiza el estudio correspondiente a los algoritmos se selecciona los algoritmos **k-means** y **LSI** para realizar el clustering de los planes docentes, los cuales se seleccionan por características como: funcionamiento, fácil implementación, ventajas que posee frente a otros algoritmos, documentación existente y adaptabilidad al caso de estudio.

Cabe recalcar que los algoritmos elegidos tienen procedimientos diferentes para realizar la identificación de los planes docentes con contenidos similares. En el caso del algoritmo k-means divide el conjunto de datos en k grupos, agrupando a los términos que guardan relación. En cambio, el algoritmo LSI aplica la descomposición en valores singulares (SVD) y a través de vectores permite identificar cuáles son los documentos que presentan semejanza en sus contenidos.

CAPÍTULO 4: PROCESAMIENTO DE LENGUAJE NATUTAL SOBRE LOS DOCUMENTOS DE LOS PLANES DOCENTES

El presente capítulo especifica la fuente de datos, selecciona el framework a utilizar y aplica el procesamiento del lenguaje natural (limpieza y el pre-procesamiento de los datos). El PLN se ejecuta con el propósito de obtener un conjunto de datos limpios y procesados para aplicar los algoritmos K-means y LSI.

4.1. Especificación de la fuente de datos

El contenido de los planes docentes de los componentes académicos está almacenado en la base de datos MySQL[9] de *planes docentes* de la UTPL, de la cual se utilizan las tablas que poseen relación con los contenidos de los planes docentes de la titulación de Ingeniería en Sistemas Informáticos y Computación. La malla de la titulación posee 45 componentes académicos, de los cuales se ha decidido descartar a todos los componentes de Practicum, permaneciendo inicialmente con 36 componentes académicos para el análisis. Los componentes académicos a analizar son los siguientes:

1. Lógica de la programación.
2. Fundamentos informáticos.
3. Fundamentos matemáticos.
4. Fundamentos de programación.
5. Contabilidad.
6. Matemáticas discretas.
7. Estructura de datos y algoritmos.
8. Programación de algoritmos.
9. Física.
10. Cálculo.
11. Programación avanzada.
12. Electrónica digital.
13. Estadística.
14. Organización y administración empresarial.
15. Arquitectura de computadores.
16. Fundamentos de base de datos.
17. Economía, finanzas e inversiones.
18. Métodos cuantitativos.
19. Fundamentos de ingeniería de software.
20. Sistemas operativos.
21. Base de datos avanzadas.
22. Ingeniería de requisitos.
23. Teoría de autómatas y compiladores.
24. Fundamentos de redes y telecomunicaciones.
25. Gestión de proyectos.
26. Ingeniería web.
27. Gestión de tecnologías de información.
28. Redes y sistemas distribuidos.
29. Arquitectura de aplicaciones.
30. Inteligencia artificial.
31. Arquitectura y seguridad de redes.
32. Arquitectura y computación paralela.
33. Procesos de ingeniería de software.

[9] MySQL: https://www.mysql.com/

34. Inteligencia artificial avanzado.
35. Sistemas basados en conocimiento.
36. Auditoría informática.

Los planes docentes pertenecen a los siguientes periodos académicos:

- Abril 2014 – agosto 2014
- Octubre 2014 – febrero 2015
- Abril 2015 – agosto 2015

Al momento de identificar y seleccionar los componentes de la titulación para la extracción, surge el inconveniente de que siete planes docentes no constan en la base de datos. Los cuales son:

- Fundamentos matemáticos.
- Estructura de datos y algoritmos.
- Cálculo.
- Economía, finanzas e inversiones.
- Métodos cuantitativos.
- Fundamentos de redes y telecomunicaciones.
- Inteligencia artificial.

Las tablas que guardan relación con los planes docentes de los componentes académicos son las siguientes:

a) *Distri_tiempo_contenido:* la tabla posee los contenidos de los planes docentes a los cuales hay que relacionarlos con su respectivo componente académico. Los campos requeridos de la tabla son: *pac_id* y *contenido.* La Figura 8 presenta los campos que posee la tabla.

dtc_id	contenido	pac_id	tn2_id	contenidoHTML
122	Introducción a la Bioquímica.	31	1	Introducción a la Bioquímica.
123	Biomoléculas (Cap 2, 3, 4).	31	2	Biomoléculas (Cap 2, 3, 4).
125	Biomoléculas (Cap. 5 y 6).	31	3	Biomoléculas (Cap. 5 y 6).
128	Bioenergética.	31	4	Bioenergética.
129	Bioenergética (continuación).	31	5	Bioenergética (continuación).
130	Enzimología.	31	6	Enzimología.
132	Examen Bimestral.	31	8	Examen Bimestral.
133	Enzimología (continuación).	31	7	Enzimología (continuación).

Figura 8. Contenido de la tabla distri_tiempo_contenido.
Fuente: base de datos de la UTPL.
Elaboración: propia.

b) *Plan_acad_componente:* la tabla contiene el id del plan académico, el código al periodo que pertenecen, el código del componente académico en el sistema de gestión académica y otros campos relacionados con la asignatura. Los campos requeridos de la tabla son: *pac_id* y *sga_componente_id*, que permiten relacionar a cada componente académico con su respectivo contenido que se encuentra en la tabla *distri_tiempo_contenido*. La Figura 9 presenta los campos que contiene la tabla.

pac_id ▲ 1	sga_componente_id	sga_periodo_id	sga_codigo_com	conocimientos_previos
21	0ebda5f0-d9f5-0086-e053-ac10360d0086	08d494b3-9baf-0098-e053-ac10360d0098	PRE-TNBIF036	*NULL*
22	0ee570ba-c39c-007c-e053-ac10360c007c	08d494b3-9baf-0098-e053-ac10360d0098	PRE-TNBIF201	Dirigido a los (as) profesionales en formación d...
23	0ec2e9f3-8bb4-007e-e053-ac10360d007e	08d494b3-9baf-0098-e053-ac10360d0098	UTPL-TNGA011	Es conveniente que el alumno que se matricule de e...
24	0ebda5f0-d9fe-0086-e053-ac10360d0086	08d494b3-9baf-0098-e053-ac10360d0098	PRE-TNBIF063	Biología General, Biología Celular y Molecular I y...

Figura 9. Tabla plan_acad_componente.
Fuente: base de datos de la UTPL.
Elaboración: propia.

c) *Qr_titulacion:* la tabla contiene el id como el nombre de las titulaciones de la universidad. En nuestro caso se necesita explícitamente el *id* y *nombre* de la titulación de Sistemas Informáticos y Computación. La Figura 10 muestra el contenido de la tabla.

id	guid	nombre	id_periodo
1	a7c2dd58-eb43-004e-e043-ac10360d004e	INGENIERÍA EN ADMINISTRACIÓN DE EMPRESAS TURÍSTICA...	2
2	a7c2dd58-eb0b-004e-e043-ac10360d004e	PSICOLOGÍA	2
3	a7c2dd58-eb2e-004e-e043-ac10360d004e	RELACIONES PUBLICAS	2
4	a7c2dd58-eb47-004e-e043-ac10360d004e	INGENIERIA AGROPECUARIA	2
5	a7c2dd58-eb4c-004e-e043-ac10360d004e	GESTIÓN AMBIENTAL	2
6	a7c2dd58-eb32-004e-e043-ac10360d004e	CIENCIAS JURÍDICAS	2
7	a7c2dd58-eb34-004e-e043-ac10360d004e	ADMINISTRACIÓN DE EMPRESAS	2
8	a7c2dd58-eb26-004e-e043-ac10360d004e	COMUNICACIÓN SOCIAL	2

Figura 10. Contenido de la tablaqr_titulacion.
Fuente: base de datos de la UTPL.
Elaboración: propia.

d) *Qr_componente_edu:* la tabla contiene el identificador y el nombre de los componentes académicos, además posee el id de la titulación a la que pertenecen. Los campos requeridos de la tabla son: *id, nombrec* e *id_titulacion*, que permiten seleccionar los

componentes de la titulación de Ingeniería en Sistemas Informáticos y Computación. La Figura 11 presenta el contenido de la tabla.

id	guid	nombrec	id_variacion	id_titulacion	codigo	id_area
1	01c2ef56-3968-0084-e053-ac10360d0084	EMPRENDIMIENTO	50	27	PRE-TNEMP001	1
2	0188daca-242d-0048-e053-ac10360d0048	DESARROLLO ESPIRITUAL III	51	28	PRE-TNCH003	2
3	0138840a-463e-003c-e053-ac10360d003c	ALIMENTACIÓN SALUDABLE	1	1	PRE-TNIIA207	3
4	0138ae63-a964-00b6-e053-ac10360d00b6	MORFOLOGÍA Y EVOLUCIÓN VEGETAL	1	1	PRE-TNBIO219	3
5	019cb1b6-017b-00b4-e053-ac10360c00b4	REALIDAD NACIONAL Y AMBIENTAL	50	27	UTPL-TNAE002	1

Figura 11. Contenido de la tabla qr_componente_edu.
Fuente: base de datos de la UTPL.
Elaboración: propia.

Una vez identificados los campos requeridos de cada tabla, mediante consultas SQL se procede a extraer los campos y se crea la tabla *contenidos_planes_docentes,* la cual contiene los campos: *id, pac_id, nombrec, contenido, id_titulación y nombre titulación*. La Figura 12 presenta el contenido de la tabla.

id	pac_id ▲ 1	codigo	nombrec	contenido	id_titulacion	nombre
556	29	PRE-TNCCO0	PRÁCTICAS EN CITTES GP 3.2	Desarrollo de Gestión Productiva 3.2 como componen...	17	SISTEMAS INFORMÁTICOS Y COMPUTACIÓN
556	29	PRE-TNCCO0	PRÁCTICAS EN CITTES GP 3.2	Proyectos que necesitan colaboración de estudiante...	17	SISTEMAS INFORMÁTICOS Y COMPUTACIÓN
556	29	PRE-TNCCO0	PRÁCTICAS EN CITTES GP 3.2	Proyectos que necesitan colaboración de estudiante...	17	SISTEMAS INFORMÁTICOS Y COMPUTACIÓN
556	29	PRE-TNCCO0	PRÁCTICAS EN CITTES GP 3.2	Proyectos que necesitan colaboración de estudiante...	17	SISTEMAS INFORMÁTICOS Y COMPUTACIÓN
556	29	PRE-TNCCO0	PRÁCTICAS EN CITTES GP 3.2	Elaboración de presentaciones. Proyectos que neces...	17	SISTEMAS INFORMÁTICOS Y COMPUTACIÓN

Figura 12. Tabla contenidos_planes_docentes.
Fuente: Base de datos de la UTPL.
Elaboración: propia.

4.2. Creación del corpus

A partir de la tabla *contenidos_planes_docentes* que se presenta en la Figura 12 se genera un documento en formato de texto plano (.txt), por cada componente académico con sus respectivos contenidos. Los archivos son almacenados dentro una carpeta y forman un corpus de 29 documentos para ser pre-procesados. La Figura 13 muestra la colección de documentos.

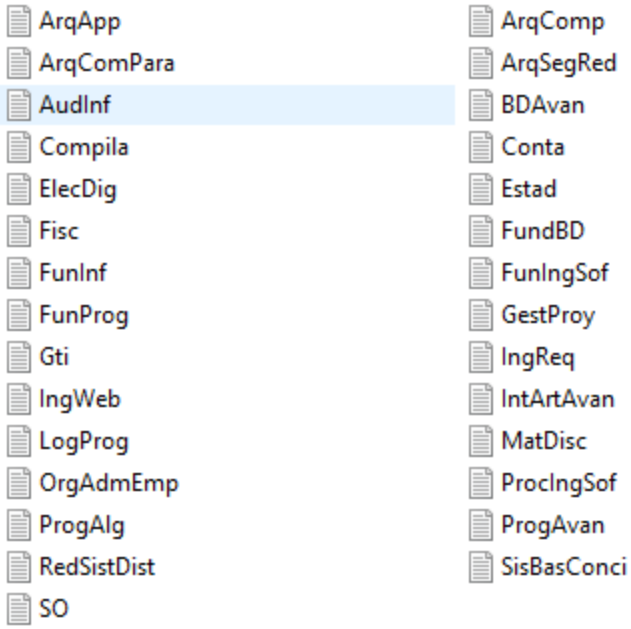


Figura 13. Colección de documentos correspondientes a los planes docentes. Elaboración: propia.

4.3. Framework

En la actualidad existen una gran cantidad de herramientas comerciales y de código abierto (open source) para aplicar Minería de Texto a conjuntos de datos no estructurados. Según Feinerer, Hornik, & Meyer (2008) la mayoría de los principales productos de computación estadística ofrecen capacidades de Minería de Texto que son las siguientes:

a) pre procesamiento,
b) agrupación,
c) resumen,
d) categorización y,
e) asociación.

La Tabla 2 presenta las características de las herramientas para Minería de Texto.

Tabla 2. Lista de herramientas disponibles para Minería de Texto y sus características.

Producto	Pre procesamiento	Asociación	Agrupación	Resumen	Categorización	API
Comercial						
Clearforest[10]	X	X	X	X		
Copernic Summarizer[11]	X			X		

[10] Clearforest: http://www.clearforest.com/
[11] Copernic Summarizer: http://www.copernic.com/en/products/summarizer/

Producto	Pre procesamiento	Asociación	Agrupación	Resumen	Categorización	API
dtSearch[12]	X	X		X		
Insightful Infact[13]	X	X	X	X	X	X
Inxight[14]	X	X	X	X	X	X
SPSS Clementine[15]	X	X	X	X	X	
SAS Text Miner[16]	X	X	X	X	X	
TEMIS[17]	X	X	X	X	X	
WordStat[18]	X	X	X	X	X	
Código abierto						
GATE[19]	X	X	X	X	X	X
RapidMiner[20]	X	X	X	X	X	X
Weka/KEA[21]	X	X	X	X	X	X
R/text mining[22]	X	X	X	X	X	X

Elaboración: propia.
Fuente: http://www.jstatsoft.org/index.php/jss/article/view/v025i05/v25i05.pdf

Para analizar la colección de documentos se ha seleccionado el framework informático estadístico de código abierto R Project[23], que proporciona la infraestructura básica necesaria para organizar, transformar, agrupar, categorizar y analizar los datos textuales. R Project a lo largo de los años ha demostrado ser uno de los entornos informáticos estadísticos más versátiles disponibles, y es por esta razón que se lo ha elegido para ejecutar el proceso de Minería de Texto en los planes docentes (Feinerer, Hornik, & Meyer, 2008). El paquete para Minería de Texto disponible en R Project es el *tm (text mining)*, el cual provee un marco que permite a los investigadores, estudiantes y profesionales aplica un sinnúmero de métodos existentes para estructuras de datos de texto. Se utiliza el entorno de programación RStudio[24] para la codificación.

Feinerer, Hornik, & Meyer (2008) mencionan que el análisis de Minería de Texto es un proceso que implica diversos pasos a seguir, que principalmente se genera por el hecho de que los documentos, desde la perspectiva del computador son colecciones de palabras no estructuradas y por lo cual se aplica técnicas de pre procesamiento de documentos para

[12] dtSearch: http://www.dtsearch.com/
[13] Insightful Infact: http://www.insightful.com/
[14] Inxight: http://go.sap.com/latinamerica/index.html
[15] SPSS Clementine: http://www-01.ibm.com/software/analytics/spss/products/modeler/
[16] SAS Text Miner: http://www.sas.com/en_us/software/analytics/text-miner.html
[17] TEMIS: http://www.temis.com/home
[18] WordStat: http://provalisresearch.com/products/content-analysis-software/
[19] GATE: https://gate.ac.uk/
[20] RapidMiner: https://rapidminer.com/
[21] Weka: http://www.cs.waikato.ac.nz/ml/weka/downloading.html
[22] R/text mining: https://cran.r-project.org/
[23] R Project: https://www.r-project.org/
[24] RStudio: https://www.rstudio.com/

obtener un conjunto de datos estructurados. Las fuentes de datos con las que se trabaja en la presente investigación, contienen datos no estructurados que requieren de un proceso de limpieza y de pre procesamiento de datos.

4.4. Limpieza de los datos

El proceso realiza la limpieza de cada uno de los documentos del corpus. La limpieza inicia cargando los documentos a R Project, luego se transforma el texto en minúsculas, se elimina los signos de puntuación, números y espacios en blanco adicionales del contenido de los documentos. La Figura 14 presenta un diagrama con el proceso a seguir para la limpieza de los datos.

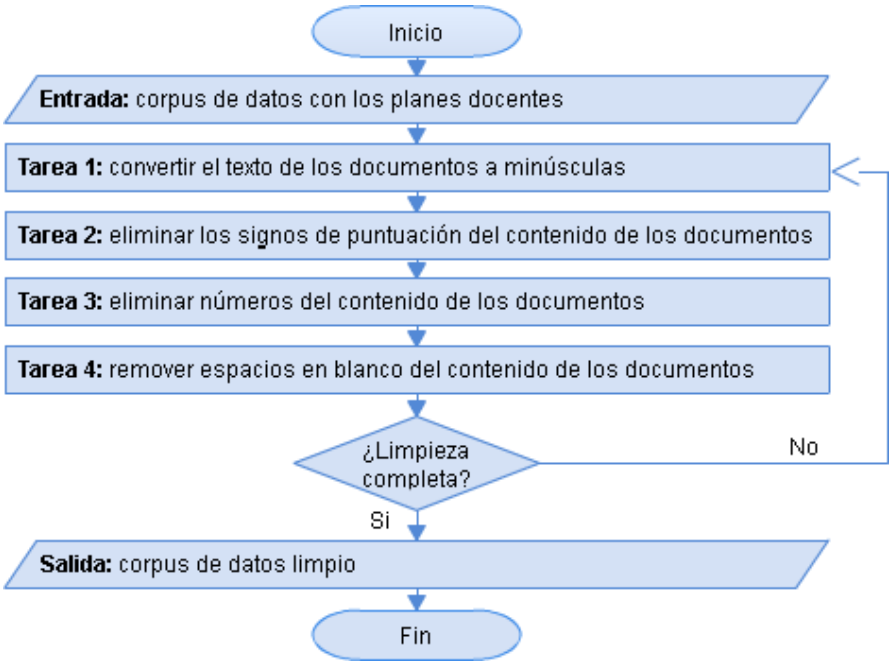


Figura 14. Diagrama para el proceso de limpieza del corpus.
Elaboración: propia.

Para realizar el proceso de limpieza primero se descarga e instala los paquetes para Minería de Texto *tm*[25] *(text mining)* y NLP[26] (Natural Language Processing). Una vez instaladas las librerías se procede a cargarlas para empezar con el proceso de limpieza del corpus. A continuación, se muestra la manera de cargar las librerías en R Project.

```
library(NLP) #lib. Procesamiento del Lenguaje Natural
Library(tm) #librería para Text Mining
```

25 Tm: https://cran.r-project.org/web/packages/tm/index.html
26 NLP: https://cran.r-project.org/web/packages/NLP/index.html

4.4.1. Entrada. Cargar archivos.

En la tarea se carga todo el conjunto de documentos (planes docentes) a R Project (corpus), con la ayuda de la función *DirSource*. La función adquiere una lista de archivos a través de una ruta (que corresponde al lugar donde se encuentran los archivos), e interpreta cada archivo como un documento (Feinerer & Hornik, 2015). El código para leer los archivos es el siguiente:

```
corpus = Corpus (DirSource ("/home/R/TextMining/Corpus",  "txt" , encoding = "UTF-8"))
```

El resultado de cargar los documentos es el siguiente:

```
> documentos
<<VCorpus>>
Metadata: corpus specific: 0, document level (indexed): 0
Content: documents: 29
```

4.4.2. Tarea 1. Transformar el texto a minúscula.

La tarea procede a transformar todo el texto de los documentos a minúscula con la ayuda de las funciones: *tm_map, content_transformer* y *tolower*.

- La interfaz *tm_map* aplica funciones de transformación al corpus.
- *Content_transformer* crea funciones que modifican el contenido del corpus.
- *Tolower* función que permite transformar cadenas de texto a minúscula (Feinerer & Hornik, 2015).

El código para convertir el texto de los documentos a minúscula es el siguiente:

```
corpus = tm_map (corpus, content_transformer (tolower))
```

La Figura 15 muestra el texto del plan docente de bases de datos avanzadas antes de aplicar la transformación a minúscula.

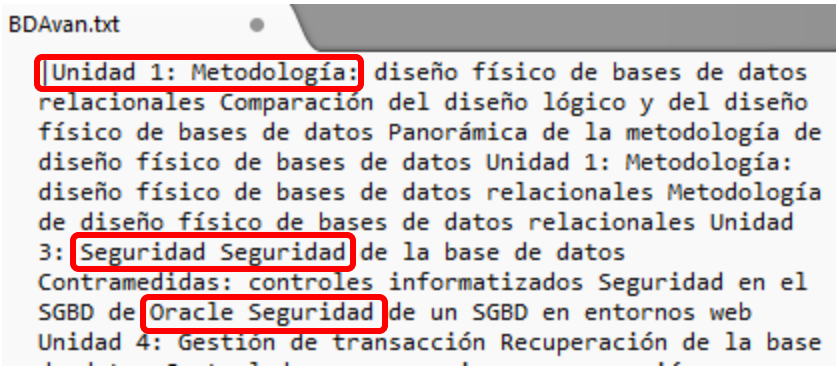


Figura 15.Contenido del componente académico bases de datos avanzadas sin transformar a minúsculas.
Elaboración: propia.

La Figura 16 presenta el resultado de convertir el texto del plan docente de bases de datos avanzadas a minúscula.

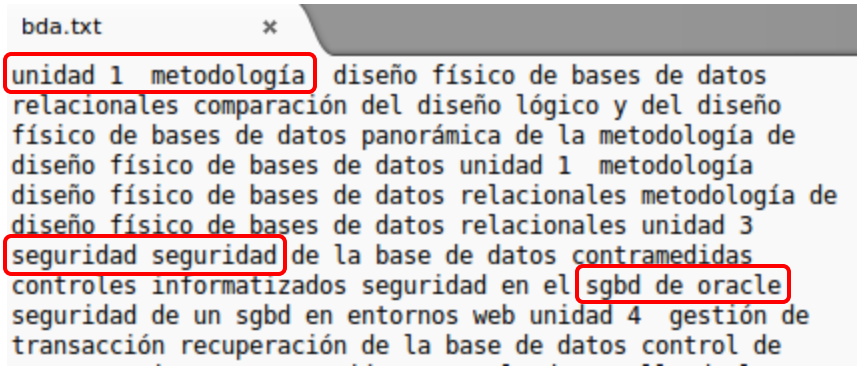


Figura 16. Contenido del componente académico de bases de datos avanzadas una vez transformado a minúsculas.
Elaboración: propia.

El proceso se aplica a cada uno de los 29 documentos que forman el corpus de planes docentes.

4.4.3. Tarea2. Eliminar signos de puntuación.

En la tarea se elimina todos los signos de puntuación de los documentos con la ayuda de la función *removePuntuation.*

- La función *removePuntuation* quita los signos de puntuación de un documento de texto. Por ejemplo: puntos (.), comas (,), punto y coma (;), paréntesis (()), guion (-), etc. (Feinerer & Hornik, 2015).

En los documentos se da el problema de que al eliminar los signos de puntuación varias palabras que están separadas por algún signo de puntuación, se unen formando una sola palabra y pierden su significado. En el siguiente ejemplo se presentan el problema.

Palabras en su formato original	**Palabras una vez eliminados los signos de puntuación**
entrada/salida productores-consumidores (RIP-OSPF)	entradasalida productoresconsumidores RIPOSPF

Por esta razón como primera acción se asigna un espacio en blanco a la derecha de todos los signos de puntuación con la finalidad de separar las palabras. El código para ubicar el espacio en blanco es el siguiente:

```
espacio <- content_transformer (function (x, pattern) gsub (pattern, " ", x))
corpusclean = tm_map (documentos, espacio, "[[: punct: ]]+")
```

El resultado de asignar el espacio en blanco es el siguiente:

Palabras en su formato original	**Palabras después de asignar espacio en blanco**
entrada/salida productores-consumidores (RIP-OSPF)	entrada/ salida productores- consumidores (RIP- OSPF)

Se procede a eliminar todos los signos de puntuación que existen en los documentos. El código para eliminar los signos de puntuación es el siguiente:

```
corpus = tm_map (corpus, removePunctuation)
```

La Figura 17 presenta el texto del plan docente de Programación de algoritmos antes de remover los signos de puntuación.

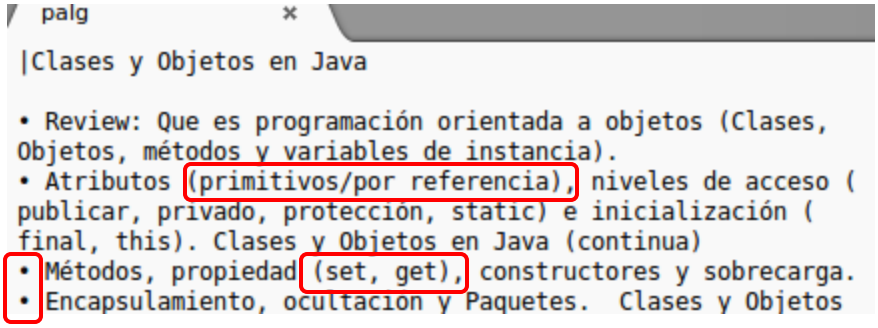


Figura 17. Contenido del plan docente de Programación de algoritmos sin remover los signos de puntuación.
Elaboración: propia.

La Figura 18 presenta el resultado de ejecutar el proceso de remover los signos de puntuación del plan docente de Programación de algoritmos.

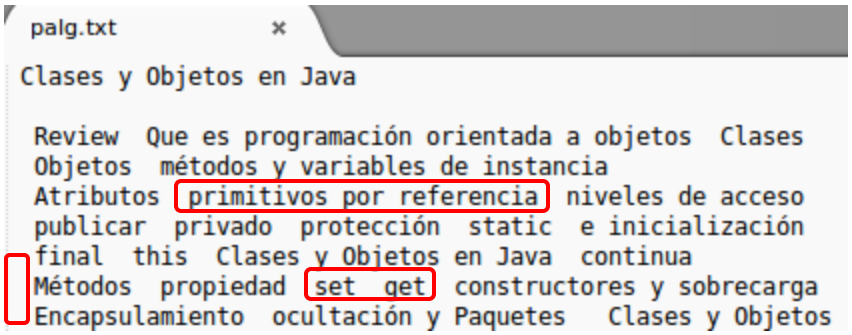


Figura 18. Contenido del plan docente de Programación de algoritmos una vez removidos los signos de puntuación.
Elaboración: propia.

El proceso se aplica a cada uno de los 29 documentos que forman el corpus de planes docentes.

4.4.4. Tarea 3. Eliminar números.

En la tarea se remueve todos los números de los documentos de texto del corpus mediante la función *removeNumbers*.

- La función *removeNumbers* eliminar todos los números de los documentos de texto (Feinerer & Hornik, 2015). El código para eliminar los números es el siguiente:

```
corpus = tm_map (corpus, removeNumbers)
```

La Figura 19 muestra el texto del documento correspondiente al plan docente de Lógica de la programación antes de remover los números.

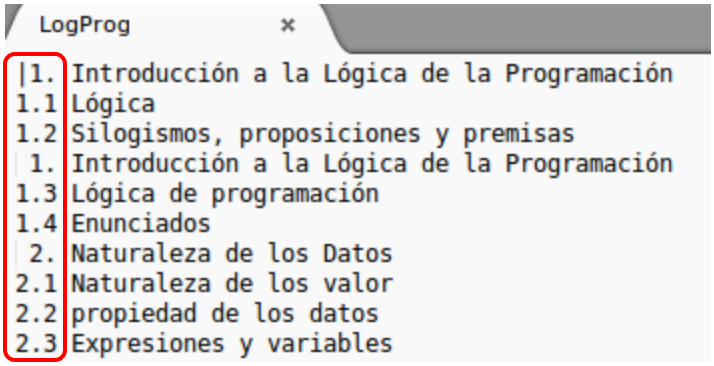


Figura 19. Contenido del plan docente Lógica de la programación sin remover los números.
Elaboración: propia.

La Figura 20 presenta el resultado de ejecutar el proceso de remover los números en el plan docente de Lógica de la programación:

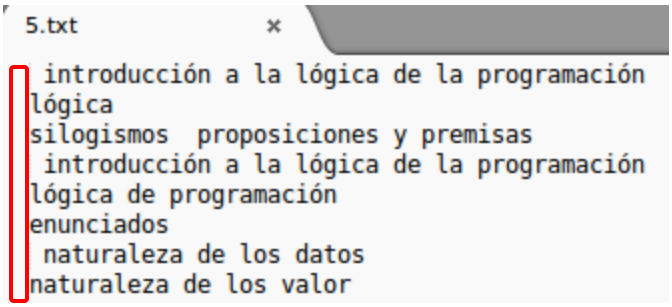


Figura 20. Contenido del plan docente de Lógica de la programación una vez removidos los números.
Elaboración: propia.

El proceso de igual manera se aplica a cada uno de los 29 documentos que forman el corpus de planes docentes.

4.4.5. Tarea 4. Eliminar espacios en blanco dobles.

Una vez eliminados los signos de puntuación y los números de los documentos de texto se elimina todos los espacios en blanco adicionales que existen en el texto de los documentos.

- La función *stripWhitespace* permite eliminar espacios en blanco adicionales dejando un solo espacio en blanco en el contenido de los documentos de texto (Feinerer & Hornik, 2015). El código para eliminar los espacios en blanco es el siguiente:

```
corpus = tm_map (corpus, stripWhitespace)
```

La Figura 21 presenta el texto del plan docente de Lógica de la programación antes de remover los espacios en blancos adicionales, en el cual se ha colocado cuadros de color rojo para indicar los espacios en blanco dobles.

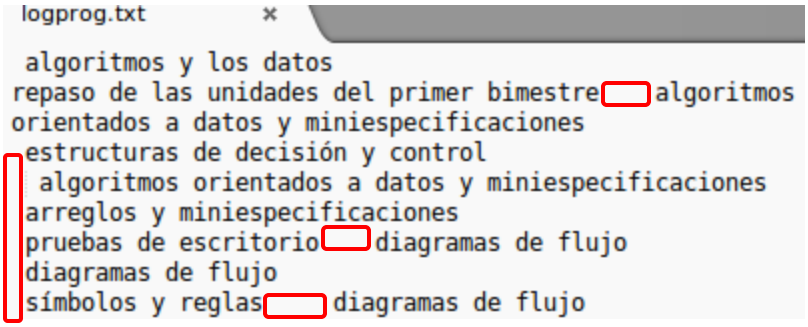


Figura 21. Contenido del plan docente de Lógica de la programación antes de remover los espacios en blanco adicionales.
Elaboración: propia.

La Figura 22 presenta el resultado de ejecutar el proceso de remover espacios en blanco adicionales en el plan docente de Lógica de la programación.

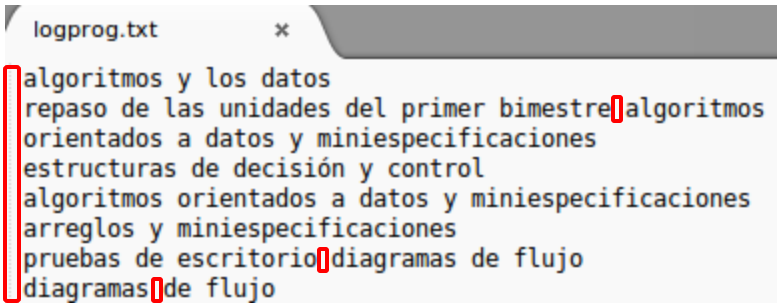


Figura 22. Contenido del plan docente de Lógica de la programación una vez removidos los espacios en blanco adicionales.
Elaboración: propia.

Después de aplicar los pasos de la limpieza de datos, se verifica si el contenido de los documentos de texto se encuentra libre de signos de puntuación y números, convertido a minúscula y sin espacios en blanco adicionales para considerar que están limpios y listos para aplicar la fase de pre procesamiento. Caso contrario se vuelve aplicar los pasos del proceso de limpieza de datos hasta que el corpus se encuentre limpio y listo para el pre procesamiento.

4.5. Pre-procesamiento de los datos

Las tareas que se aplican en la sección son: eliminar las palabras vacías o stop words, lematizar las palabras del texto, identificar bigramas y trigramas, definir el vocabulario para crear la matriz términos por documentos (MTD) y determinar la frecuencia de los términos de la matriz. La Figura 23 presenta un diagrama con las tareas a seguir para el pre procesamiento de los datos.

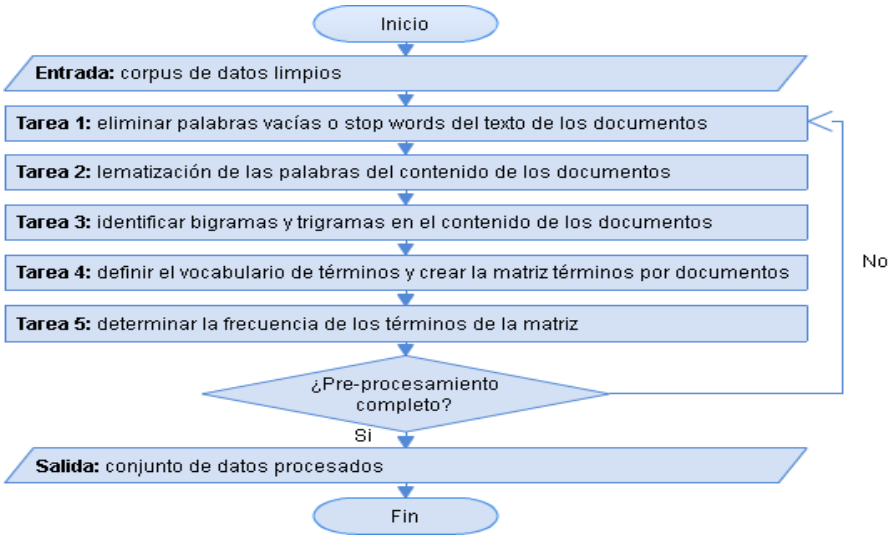


Figura 23. Diagrama para el pre procesamiento de los datos.
Elaboración: propia.

4.5.1. Entrada de datos.

Para empezar con el pre procesamiento se recibe como entrada el conjunto de datos generado por el proceso de limpieza. En nuestro caso la variable *corpuslimpio* contiene el conjunto de archivos limpios.

```
> corpuslimpio
<<VCorpus>>
Metadata: corpus specific: 0, document level (indexed): 0
Content: documents: 29
```

La Figura 24 muestra el contenido del plan docente de Organización y administración empresarial después de haber pasado por el proceso de limpieza.

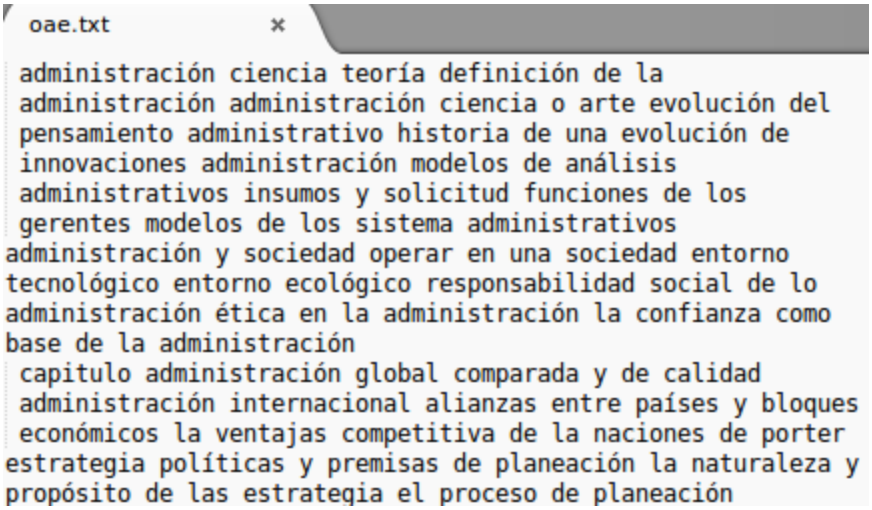


Figura 24. Plan docente de Organización y administración empresarial listo para el pre procesamiento.
Elaboración: propia.

4.5.2. Tarea 1. Remover stop words del conjunto de datos.

La tarea remueve todas las palabras vacías (stop words) del contenido de los documentos de texto del corpus con ayuda de las funciones *removeWords* y *stopwords* del paquete de Minería de Texto (tm).

- La función *removeWords* remueve stop words de un documento de texto (Feinerer & Hornik, 2015).

- La función *stopwords* posee varias clases de palabras vacías con soporte para diferentes idiomas. Los idiomas que soporta son: danish, dutch, english, finnish, french, german, hungarian, italian, norwegian, portuguese, russian, spanish, y swedish (Feinerer & Hornik, 2015).

El paquete tm (text mining) posee un archivo .dat por cada idioma que soporta, en este caso se hace uso del archivo *spanish.data* que contiene una lista de palabras vacías en el idioma español (Ver Anexo 1).

La Figura 25 presenta los archivos con stop words por cada idioma que soporta el paquete tm (text mining).

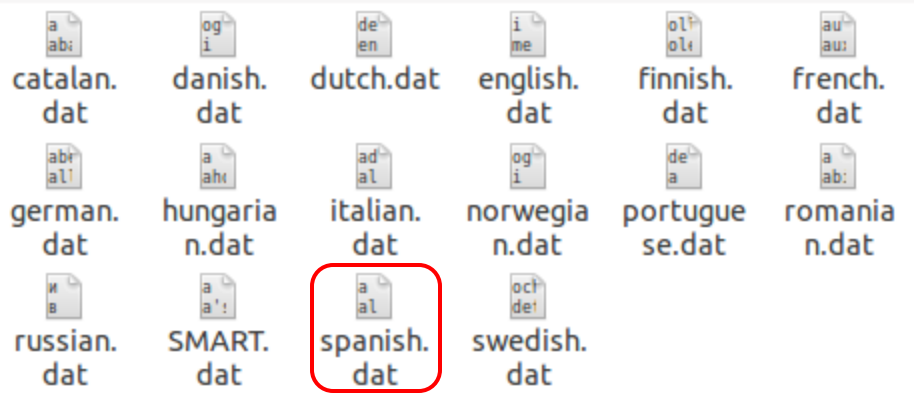


Figura 25. Lista de archivos que contienen stop words por cada idioma que soporta el paquete tm (text mining).
Elaboración: propia.

La Figura 26 muestra algunas palabras o stop words que forman el archivo spanish.dat.

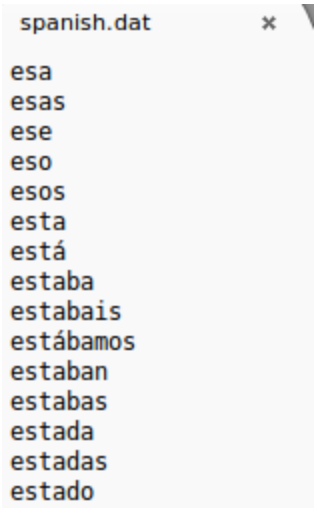


Figura 26. Extracto del contenido del archivo spanish.dat
Elaboración: propia.

Cargados los documentos a R Project, se procede a eliminar las palabras vacías de los documentos de texto y como el corpus de archivos contiene texto en el idioma español se especifica en el código el idioma *spanish* [Ejemplo: (`stopwords("spanish")`)], para que R Project cargue el archivo de stop words en español, realice la comparación y proceda a eliminarlas de los planes docentes. El código para eliminar las palabras vacías es el siguiente:

```
corpusprocesado = tm_map (corpuslimpio, removeWords, c(stopwords("spanish"),
"corpus"))
```

Una vez que se ejecuta el código se observa que aún los documentos poseen palabras vacías, que no han sido eliminadas por la razón que no constan dentro del archivo spanish.data. Por esta razón se procede a crear un archivo propio de stop words (Ver Anexo 2), que contiene una lista de palabras vacías que no aportan conocimiento en los planes docentes y se requiere eliminarlas. La Figura 27 presenta extracto de stop words que forman el archivo propio de palabras vacías.

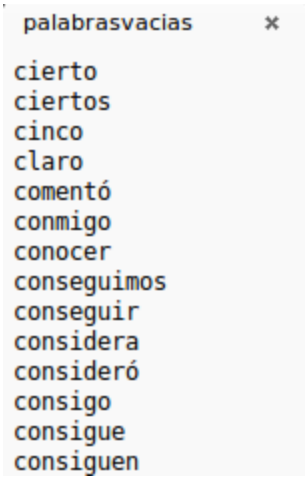


Figura 27. Extracto del contenido del archivo de stop words. Elaboración: propia.

Creado el archivo se carga a R Project en formato de codificación de caracteres UTF-8 para que reconozca las tildes. La lectura del archivo se la realiza con la función readLines.

- La función readLines permite leer algunas o todas las líneas de texto de una conexión o cadena de caracteres (Feinerer & Hornik, 2015).

Una vez cargado el archivo se prosigue a eliminar las palabras vacías de los planes docentes, para lo cual R compara las palabras (stop words) del archivo con el texto de los documentos

del corpus y las va eliminando. Una vez aplicado el proceso se considera que el corpus está libre de stop words.

El código para cargar el archivo y eliminar las palabras vacías es el siguiente:

```
archivopalabrasvacias = readLines ("palabrasvacias", encoding = "UTF-8")
archivopalabrasvacias = iconv (archivopalabrasvacias, to = "ASCII//TRANSLIT")
corpusprocesado = tm_map (corpusprocesado, removeWords,
archivopalabrasvacias)
```

La Figura 28 presenta el plan docente correspondiente al componente académico de Organización y administración empresarial antes de remover las palabras vacías.

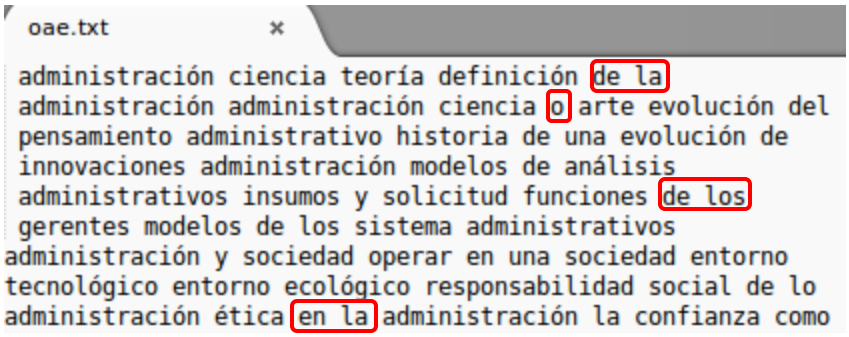


Figura 28. Plan docente del componente académico de Organización y administración empresarial sin remover stop words.
Elaboración: propia.

La Figura 29 presenta el resultado de ejecutar el proceso de remover stop words en el plan docente correspondiente al componente académico de Organización y administración empresarial.

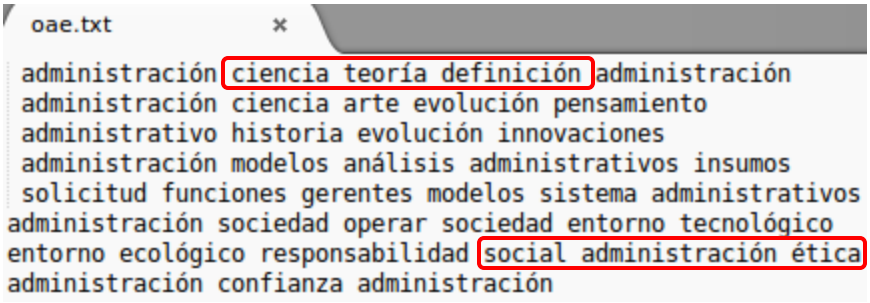


Figura 29. Plan docente del componente académico de Organización y administración empresarial una vez removidos stop words.
Elaboración: propia.

El proceso se aplica a cada uno de los 29 documentos que forman el corpus de planes docentes.

4.5.2.1. Corrección de faltas de ortografía.

Una vez removidos los stop words de los documentos se evidencia que gran cantidad de estos tienen faltas de ortografía en su contenido, y es por esto que se precede a revisar el contenido de cada documento con el propósito de ir corrigiendo las faltas de ortografía y evitando que se presenten problemas más adelante durante la ejecución del proceso. La corrección se realiza con ayuda de un editor de texto. A continuación, se presenta la lista de palabras que tienen faltas de ortografía:

Palabras mal escritas	**Palabras corregidas**
algortimos	algoritmos
programacion	programación
análisi	análisis
clasiicación	clasificación
comportamento	comportamiento
defiición	definición
gestions	gestión
maquinas	máquinas
motivacion	motivación
perpsectiva	perspectiva
Infomática	informática
Informatica	informática
electrico	eléctrico
polimosfismo	polimorfismo
revsión	revisión
twiter	twitter
varibales	variables
algebra	álgebra
etica	ética
foraneas	foráneas
percceptrón	perceptrón
jerarquias	jerarquías
busqueda	búsqueda

defenza	defensa
compentecias	competencias
menus	menús
comunicaciónes	comunicaciones
evaluaciónfinal	evaluación final

4.5.3. Tarea 2. Lematización.

La tarea elimina de manera automática las formas derivadas de las palabras que forman el contenido de los planes docentes para reducirlas a su lema original. Para aplicar la lematización de las palabras de cada documento primero se descarga e instala el paquete SnowballC[27] que permite encontrar el lema correspondiente de cada palabra.

Una vez instalado el paquete se procede a cargarlo en R Project para empezar con la lematización. El paquete soporta el idioma español así como los siguientes idiomas: Danish, Dutch, English, Finnish, French, German, Hungarian, Italian, Norwegian, Portuguese, Romanian, Russian, Swedish and Turkish (Bouchet, 2015).

El código para cargar la librería en R Project es el siguiente:

```
library(SnowballC) #librería para lematizar
```

El paquete SnowballC posee un archivo .Rdata por cada idioma que soporta, en el presente trabajo se hace uso del archivo spanish.Rdata que contiene una lista de palabras con sus respectivos lemas en español. La Figura 30 muestra los idiomas que soporta el paquete.

Figura 30. Lista de archivos que contienen los lemas por cada idioma que soporta el paquete SnowballC.
Elaboración: propia.

[27] SnowballC: https://cran.r-project.org/web/packages/SnowballC/index.html

La Figura 31 presenta algunas palabras con sus lemas que se encuentran en el archivo spanish.Rdata.

word	stem
abandona	abandon
abandonada	abandon
abandonadas	abandon
abandonado	abandon
abandonados	abandon
abandonamos	abandon
abandonan	abandon
abandonar	abandon
abandonarlo	abandon
abandonaron	abandon
abandono	abandon

Figura 31.Extracto del contenido del archivo spanish.Rdata
Elaboración: propia.

Una vez cargada la librería SnowballC se procede a la lematización de las palabras de los documentos de texto, y como el corpus de archivos contiene texto en el idioma español se especifica en el código el idioma *spanish* [Ejemplo: (language="spanish")], para que R cargue el archivo que contiene las palabras y lemas en español; realice la comparación y procesa a lematizar las palabras con ayuda de la función stemDocument.

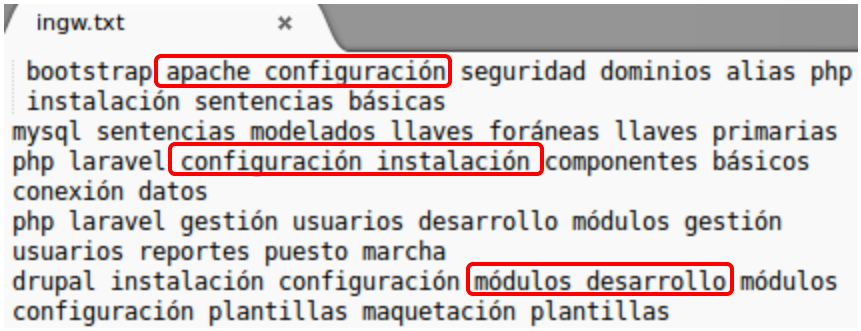
- La función *stemDocument* determina la raíz de las palabras en un documento de texto utilizando el algoritmo derivado de Porter (Feinerer & Hornik, 2015).

El código para lematizar las palabras es el siguiente:

```
corpusprocesado = tm_map (corpusprocesado, stemDocument, language="spanish")
```

La Figura 32 presenta el plan docente correspondiente al componente académico de Ingeniería Web. En el literal a) se presenta el texto del plan docente antes de aplicar el proceso de lematización, algunas palabras están encerradas en rectángulos de color rojo para mejorar su apreciación. En el literal b) se muestra el texto del plan docente después de aplicar el proceso de lematización.

a)



b)

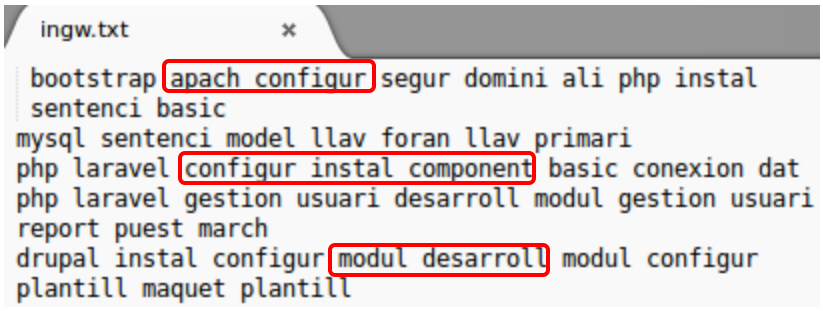


Figura 32. Plan docente del componente académico de Ingeniería Web. a) Contenido antes de aplicar el proceso de lematización. b) Contenido después de aplicar el proceso de lematización.
Elaboración: propia.

El proceso de lematización se aplica a cada uno de los 29 documentos que forman el corpus de planes docentes.

4.5.4. Tarea 3. Bigramas y trigramas.

La tarea identifica y selecciona una serie de n-gramas o secuencias de palabras que se encuentran adyacentes en el contenido de los planes docentes, y que utilizadas de esta forma crean un significado específico. Se ha considerado identificar bigramas y trigramas en el contenido de los planes docentes (Ver Anexo 3). A continuación, se presentan algunos bigramas y trigramas identificados:

- sistemas operativos
- casos uso
- base datos
- programación orientada objetos
- von Neumann
- inteligencia artificial
- sublime text
- visual basic net

Para formar los bigramas y trigramas en los documentos primero se descarga e instala el paquete RWeka[28], que posee una colección de algoritmos de aprendizaje automático para tareas de minería de datos escritos en Java, además contiene herramientas para el pre-procesamiento, clasificación, regresión, clustering y reglas de asociación (Hornik et al., 2015).

Una vez instalado el paquete se procede a cargarlo en R Project para proceder a formar los bigramas y trigramas. La manera de llamar a la librería es la siguiente:

```
library(RWeka) #Librería para formar n-gramas
```

La función a utilizar del paquete RWeka para formar los bigramas y trigramas es la siguiente:

- La función *NGramTokenizer* permite dividir cadenas en n-gramas con un mínimo y máximo número de gramas (Hornik et al., 2015).

La Figura 33 presenta el plan docente correspondiente al componente académico de electrónica digital, en el cual se identifican algunos bigramas que son guardados en un documento de texto plano para que posteriormente formen parte del vocabulario de términos y de la matriz términos por documentos.

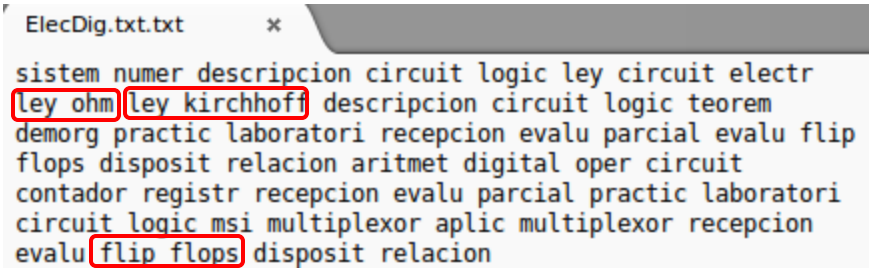


Figura 33. Plan docente del componente académico de Electrónica digital. Elaboración: propia.

El proceso de igual manera se aplica a cada uno de los 29 documentos que forman el corpus de planes docentes.

4.5.5. Tarea 5. Definir el vocabulario y crear la matriz términos por documentos.

4.5.5.1. Definir Vocabulario.

Para crear la matriz términos por documentos (MTD) primero se define un vocabulario con los términos (palabras) más relevantes de cada documento, permitiendo así descartar aquellas

[28]RWeka: https://cran.r-project.org/web/packages/RWeka/index.html

palabras que no aportan conocimiento a los planes docentes, pero que forman parte del contenido. Los términos seleccionados para formar el vocabulario son aquellos que aparecen como mínimo en dos documentos (aquí están incluidos los bigramas y trigramas), los cuales son guardados en un documento de texto plano (txt) y cargados a R Project.

El vocabulario está conformado por 110 términos lematizados (Ver Anexo 5) lo que nos indica el número de términos que tiene la matriz términos por documentos. El código para cargar el documento con el vocabulario es el siguiente:

```
vocabulario = readLines ("vocabulario", encoding = "UTF-8")
vocabulario = iconv (vocabulario, to = "ASCII//TRANSLIT")
```

Una vez cargado el vocabulario a R Project se procede a crear la matriz términos por documentos.

4.5.5.2. Matriz Términos por Documentos.

Se procede a construir la matriz términos por documentos a partir del vocabulario de términos, con ayuda de la función *DocumentTermMatrix* del paquete tm (text mining) de R Project.

- *DocumentTermMatrix* permite construir una matriz término-documento a partir de una colección de documentos (Feinerer & Hornik, 2015).

Para crear la matriz se envía como parámetro de control la variable *vocabulario* que contiene la lista de términos. Además, se envía la función *NGramTokenizer* para formar los bigramas y trigramas. El código para crear la matriz se presenta a continuación:

```
matrizterminodocumento = DocumentTermMatrix (corpusmatriz, control = list
(dictionary = c(vocabulario), tokenize = NGramTokenizer))
```

A continuación, se visualizar la información de la matriz una vez construida:

```
> matrizterminodocumento
<<DocumentTermMatrix (documents: 29, terms: 110)>>
Non-/sparse entries: 581/3537
```

```
Sparsity: 86%
Maximal term length: 23
Weighting: term frequency (tf)
```

Como se observa en el resultado la matriz términos por documentos está formada por 29 documentos y 110 términos.

Como la matriz términos por documentos es una forma de representar al modelo vectorial está compuesta de vectores que son los documentos, y para divisar a cada uno de los documentos con sus términos se la convierte en una matriz de caracteres facilitando su interpretación. La función *as.matrix* del paquete de text mining (tm) admite la transformación.

- La función *as.matrix* es una función genérica que devuelve una matriz de caracteres (Feinerer & Hornik, 2015).

El código para convertir la matriz es el siguiente:

```
matriz = as.matrix (matrizterminodocumento)
```

La Figura 34 muestra extracto de la matriz términos por documentos obtenida de los planes docentes.

```
> matriz
               Terms
Docs            administr algoritm almacen aplic arbol arquitectur arregl atribut bases_dat
  ArqApp.txt            0        0       0     2     0           2      0       1         0
  ArqComPara.txt        0        0       0     0     0           3      0       0         0
  ArqComp.txt           0        0       1     0     0           3      0       0         0
  ArqSegRed.txt         0        0       0     1     0           2      0       0         0
  AudInf.txt            0        0       0     3     0           0      0       0         0
  BDAvan.txt            0        0       2     1     0           2      0       0        16
  Compila.txt           0        1       0     0     3           0      0       0         0
  Conta.txt             0        0       0     0     0           0      0       0         0
  ElecDig.txt           0        0       0     1     0           0      0       0         0
  Estad.txt             0        0       0     0     1           0      0       0         0
  Fisc.txt              0        0       1     1     0           0      0       0         0
  FundBD.txt            0        0       0     0     0           2      0       2        17
  FunInf.txt            0        0       0     8     0           2      0       0         2
  FunIngSof.txt         0        0       0     1     0           1      0       0         0
  FunProg.txt           0        0       0     0     0           0      7       0         0
  GestProy.txt          1        0       0     0     0           0      0       0         0
  Gti.txt               0        0       0     1     0           2      0       0         0
  IngReq.txt            2        0       0     0     0           0      0       1         0
  IngWeb.txt            0        0       0     1     0           0      0       0         0
  IntArtAvan.txt        0        3       0     0     5           2      0       0         1
  LogProg.txt           0       22       0     0     0           0      2       0         0
  MatDisc.txt           0        0       0     0     6           0      0       0         0
  OrgAdmEmp.txt        20        0       0     0     0           0      0       0         0
  ProcIngSof.txt        2        0       0     0     0           0      0       0         0
  ProgAlg.txt           0        9       0     1     0           0      7       1         0
  ProgAvan.txt          1        0       0     2     0           1      0       0         1
  RedSistDist.txt       0        4       0     0     0           0      0       0         0
  SisBasConci.txt       0        0       0     0     0           0      0       0         0
  SO.txt                0        0       0     1     0           0      0       0         0
```

Figura 34.Matriz término documento de los documentos de los planes docentes.
Elaboración: propia.

4.5.6. Tarea 6. Frecuencia términos (tf).

La tarea calcula el número de veces que aparece una palabra o n-grama en cada uno de los planes docentes. Comúnmente las palabras frecuentes indican sobre qué temas en específico se habla en los documentos. Por ejemplo, en el plan docente de programación de algoritmos será muy común encontrar palabras como: java, algoritmos, clases, programación orientada a objetos, etc.

En el conteo de términos, cuantas más veces se encuentre un término en un documento es más factible que ese término sea relevante para el documento, pero no siempre es así debido a que una palabra que aparece ocho veces en un plan docente no quiere decir que sea ocho veces más importante que una que aparece una sola vez. Por esta razón se ha estimado considerar solo los términos (palabras o n-gramas) que tengan una frecuencia igual o mayor a tres, quedando descartados todos los términos con una frecuencia inferior. El código para obtener la frecuencia de las palabras y ordenarlas descendentemente es el siguiente:

```
frecuencia = colSums (frecuencia)
frecuencia = sort (frecuencia, decreasing = TRUE)
```

La Figura 35 presenta el resultado de ejecutar el código.

```
> frecuencia
        dat    estructur       sistem       proces      softwar     algoritm
        120           58           53           51           41           39
     requer      program       segur    administr    metodolog       aplic
         29           27           27           26           26           24
        web      funcion        oper     distribu       inform      arregl
         20           19           19           17           17           16
     ingeni   relacional     informat  transaccion      circuit       objet
         14           14           13           13           12           12
      comun      diagram        etic interrupcion        prueb       fisic
         10           10           10           10           10           9
      ejecu       empres      grafic  proposicion       recurs       tecnic
         8            8            8            8            8           8
    lenguaj       negoci     protocol      jerarqu       orient      atribut
         7            7            7            6            6           5
      estim        fluj     funcional        polit      practic      problem
         4            4            4            4            4           4
    vector
         3
```

Figura 35. Frecuencia de las palabras de los planes docentes ordenadas descendentemente.
Elaboración: propia.

Como se observa en la Figura 35 solo existen palabras cuya frecuencia es igual o mayor a tres.

4.5.6.1. Visualización de la frecuencia de los términos en un histograma.

Para una mejor visualización, comprensión y análisis de la frecuencia de los términos de los planes docentes, se opta por crear un histograma o plot en R a través del paquete ggplot2[29].

- El paquete *ggplot2*, permite la implementación de la gramática de gráficos en R. Combina las ventajas tanto de la gráfica de enrejado como de la base, favoreciendo construir un plot paso a paso desde varios orígenes de datos (Wickham & Chang, 2015).
- *Geom_bar* se utiliza para crear plots de área: gráficos de barras para la categoría x, e histogramas para continua e histogramas para constante y (Wickham & Chang, 2015).

El código para crear el histograma con la frecuencia de las palabras es el siguiente:

```
plot = ggplot (subset (frecupalab, frecuencia>2), aes (palabras, frecuencia))
plot = plot + geom_bar (stat = "identity", fill = "steelblue") + theme_minimal
()
```

La Figura 36 presenta el histograma con la frecuencia de los términos.

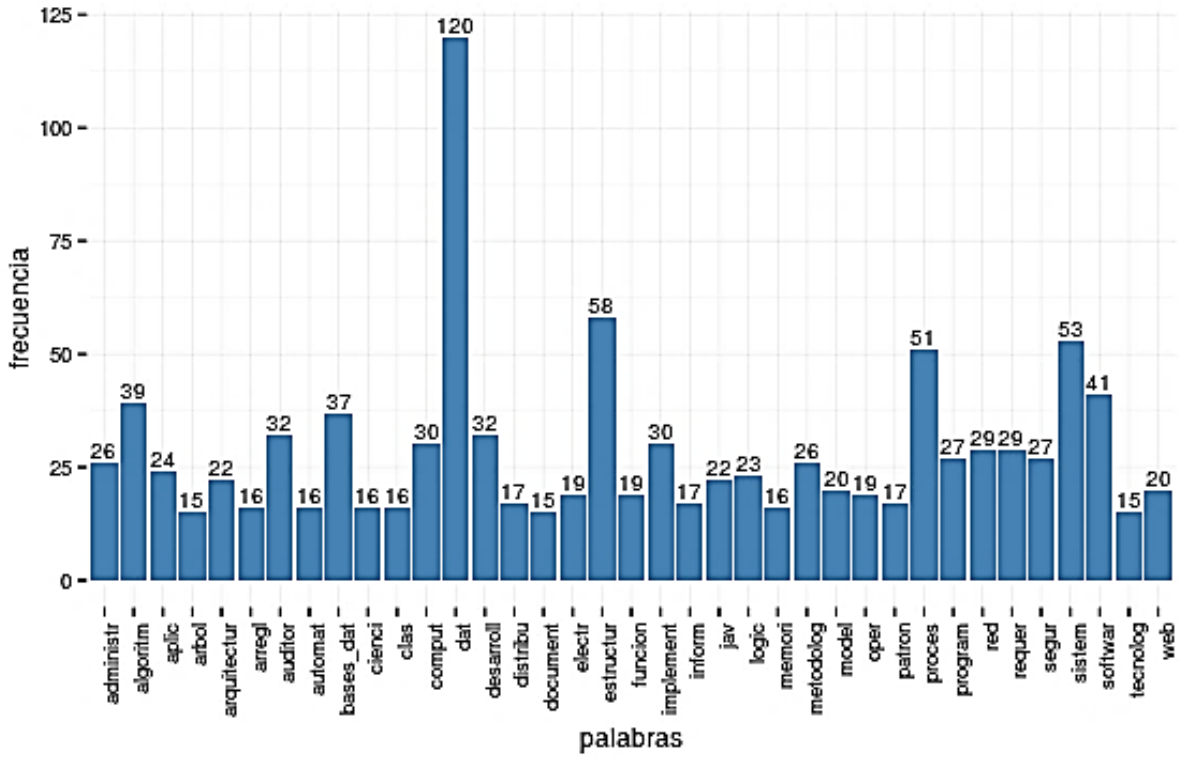


Figura 36. Histograma con frecuencia de términos mayores a 14.
Elaboración: propia.

[29] Ggplot2: https://cran.r-project.org/web/packages/ggplot2/ggplot2.pdf

CAPÍTULO 5: CLUSTERING DE LOS PLANES DOCENTES

El presente capítulo muestra el proceso de la técnica de clustering para el agrupamiento de los planes docentes en base a sus contenidos, mediante el algoritmo de clustering por partición K-means y la técnica de indexación semántica latente (LSI) en R Project. El primero algoritmo agrupa un conjunto de datos en un número de k clusters (grupos), en cambio el segundo algoritmo realiza la agrupación a nivel de documentos, en donde cada documento es representado mediante un vector.

Las técnicas son aplicadas a los contenidos de los planes docentes representados en la matriz términos por documentos, la cual es el resultado de aplicar el procesamiento del lenguaje natural (proceso de limpieza y pre-procesamiento de datos).

5.1. Algoritmo K-means en R Project

El agrupamiento de los contenidos de los planes docentes mediante el algoritmo k-means es realizado utilizando R Project, para lo que se requiere descargar e instalar los siguientes paquetes:

- **Paquete Stats.** El paquete contiene las funciones para cálculos estadísticos y la generación de números aleatorios. Específicamente se utiliza del paquete la función k-means clustering (R Core Team and contributors worldwide, 2016). Para cargar la librería se realiza de la siguiente manera:

```
library(stats)
```

- **Paquete Cluster.** El paquete posee métodos para el análisis de clusters (Maechler et al., 2015). En nuestro caso se utiliza para graficar los grupos encontrados por el algoritmo k-means. La siguiente línea de código carga la librería en RStudio.

```
library(cluster)
```

Una vez cargados los paquetes se procede a realizar el agrupamiento, para lo cual se hace uso de la función *dist* del paquete *stats* para calcular la distancia. La función recibe como primer parámetro de entrada una matriz numérica o trama de datos, en nuestro caso se envía la matriz términos por documentos obtenida en el pre-procesamiento de datos.

Como segundo parámetro se debe especificar el método o distancia que se va a utilizar: euclidean, maximum, manhattan, canberra, binary o minkowski. Para nuestro proyecto se utiliza la distancia euclidiana y se lo especifica en el código en la clase *method*.

El siguiente código presenta el proceso para calcular la distancia euclidiana en los términos de la matriz términos por documentos.

```
matriz = crearmtd(corpusmatriz) #Matriz numérica con los términos de los
planes docentes.
distancia = dist(t(matriz), method = "euclidean") #Distancia euclidiana
```

Una vez calculada la distancia euclidiana en los términos de la matriz se utiliza la función *kmeans* del paquete *stats*, para realizar el agrupamiento de los contenidos de los planes docentes. La función recibe como primer parámetro de entrada una matriz numérica de datos, en nuestro caso se envía la matriz de distancias, es decir, la matriz que posee la distancia euclidiana de cada término.

Como segundo parámetro se debe indicar el número de k centros (centroides) o grupos que queremos formar con los contenidos de los planes docentes. El valor que se asigna a k depende del número de planes docentes y de los resultados que se vayan obteniendo, es decir, el valor que se asigna a k debe generar los mejores resultados.

El siguiente código presenta el proceso para agrupar el contenido de los planes docentes en clústeres, en donde k = 8.

```
k = 8 #Número de clusters
agrupamiento = kmeans (distancia, k) #Algoritmo kmeans
```

5.2. Visualización gráfica de los grupos.

Para una mejor comprensión y análisis de los grupos obtenidos al aplicar el algoritmo k-means, se opta por crear un plot en R Project que permita observar gráficamente los grupos.

- La función **clusplot**, dibuja un clusplot (plot de agrupamiento) de dos dimensiones (Maechler et al., 2015). El siguiente código presenta el proceso para crear el plot con los grupos formados por el algoritmo k-means.

```
clusplot (as.matrix (distancia), agrupamiento$cluster, main = "K Means
Clustering", xlab = "Términos", ylab = "Clusters", color = TRUE, shade = TRUE,
labels = 4) #Plot kmeans
```

La Figura 37 presenta el diseño del plot que crea la función clusplot del paquete cluster, en la que se observa con facilidad los grupos formador por el algoritmo k-means, en este caso existen ocho grupos.

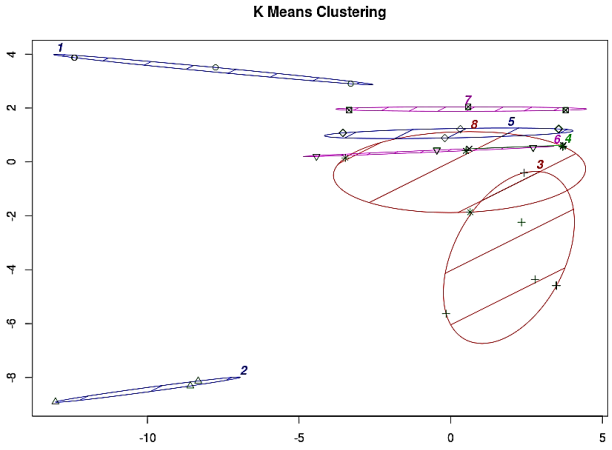


Figura 37. Gráfica de los grupos obtenida con la función clusplot del paquete cluster.
Elaboración: propia.

5.3. Clustering de pruebas

Para evidenciar el proceso de agrupamiento del algoritmo k-means se opta por ejecutar varias pruebas con un número determinado de planes docentes, con el objetivo de analizar los resultados obtenidos y la efectividad del algoritmo.

5.3.1. Prueba número uno.

La prueba número uno utiliza siete planes docentes que corresponden a las áreas de conocimiento de bases de datos, programación y redes. Se desarrolla la prueba con estos planes docentes con el objetivo de analizar los resultados que genera el algoritmo k-means al agrupar contenidos de diversas áreas de conocimiento. Los documentos son los siguientes:

- Arquitectura y Seguridad de Redes.
- Bases de Datos Avanzadas.
- Fundamento de Bases de Datos.
- Fundamentos de Programación.

- ❖ Programación de Algoritmos.
- ❖ Programación Avanzada.
- ❖ Redes y Sistemas Distribuidos.

Se aplica el proceso de limpieza de datos a cada uno de los documentos de la prueba:

1. Se inicia eliminando los signos de puntuación o caracteres especiales del contenido. La Figura 38 presenta el contenido del plan docente de Programación de algoritmos, el literal a) presenta el texto del documento antes de remover los signos de puntuación, y el literal b) muestra el contenido una vez removidos.

a)

palg

```
• Review: Que es programación
orientada a objetos (Clases,
Objetos, métodos y variables de
instancia).
• Atributos (primitivos/por
referencia), niveles de acceso (
publicar, privado, protección,
static) e inicialización (final,
this). Clases y Objetos en Java (
continua)
```

b)

palg.txt

```
Review  Que es programación
orientada a objetos  Clases
Objetos  métodos y variables de
instancia
Atributos  primitivos por
referencia  niveles de acceso
publicar  privado  protección
static  e inicialización  final
this  Clases y Objetos en Java
continua
```

Figura 38. Plan docente de Programación de algoritmos. a) Contenido antes de remover los signos de puntuación. b) Contenido una vez removidos los signos de puntuación y caracteres espaciales.
Elaboración: propia.

2. Se procede a transformar el texto de los planes docentes a minúsculas. La Figura 39 presenta el contenido del plan docente de Redes y sistemas distribuidos. El literal a) muestra el texto del documento antes de convertir a minúsculas, y el literal b) exhibe el texto del contenido transformado a minúsculas.

a)

rsd.txt

```
TRANSPORTE ORIENTADO A LA CONEXIÓN
9  1  La conexión TCP 9  2
Estructura del segmento TCP  9 3
Temporización 9 4  transacción de
datos fiable 9  5  Control de
flujo 9  6  Gestión de conexión
TCP
CONTROL DE CONGESTIÓN 10  1
```

b)

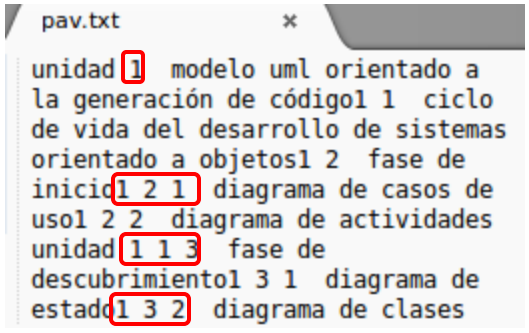
rsd.txt

```
transporte orientado a la conexión
9  1  la conexión tcp 9  2
estructura del segmento tcp  9 3
temporización 9 4  transacción de
datos fiable 9  5  control de
flujo 9  6  gestión de conexión
tcp
control de congestión 10  1
```

Figura 39. Contenido del plan docente de Redes y sistemas distribuidos. a) Contenido con caracteres en mayúsculas. b) Contenido transformado a minúsculas.
Elaboración: propia.

3. Se continúa removiendo toda numeración del contenido de los documentos. La Figura 40 muestra el plan docente de Programación avanzada. El literal a) presenta el texto del documento antes de remover los números. El literal b) exhibe el texto del plan docente una vez eliminados los números.

a)



b)

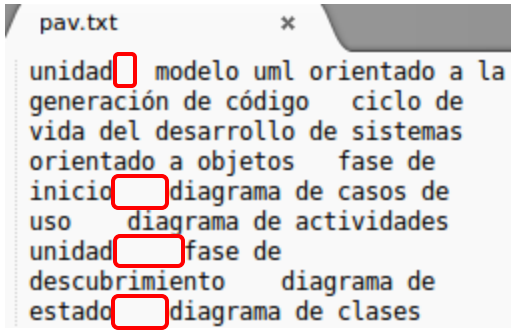
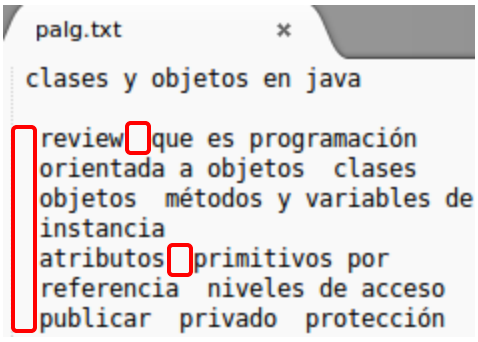


Figura 40. Plan docente de Programación avanzada. a) Contenido del documento antes de remover los números. b) Contenido del documento una vez removidos los números. Elaboración: propia.

4. Como resultado de aplicar las tareas anteriores se forman espacios en blanco adicionales que deben ser suprimidos. La Figura 41 presenta el contenido del plan docente de Programación de algoritmos. El literal a) muestra el contenido con los espacios en blanco a remover. El literal b) exhibe el resultado de suprimir los espacios en blanco.

a)



b)

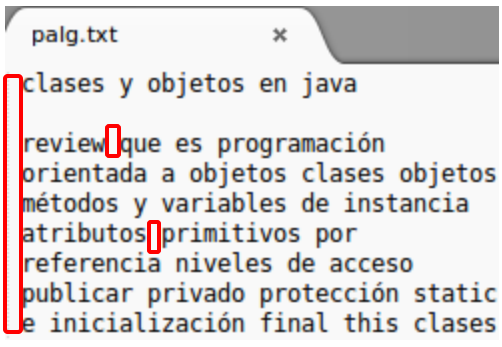
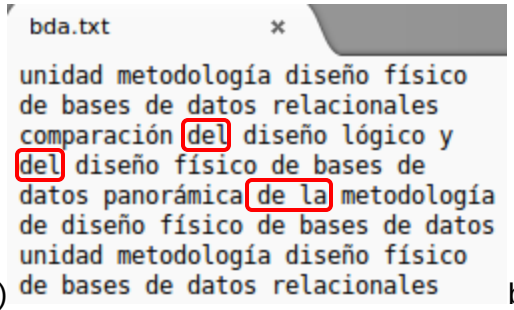


Figura 41. Plan docente de Programación de algoritmos. a) Contenido del documento con espacios en blanco. b) Contenido una vez suprimidos los espacios en blanco adicionales. Elaboración: propia.

Una vez que se obtiene un conjunto de documentos libres de caracteres innecesarios se aplica el pre-procesamiento de datos:

1. Se remueve stop words del contenido de los documentos. La Figura 42 muestra el plan de bases de datos avanzadas. El literal a) exhibe el contenido del documento con stop words, y el literal b) presenta el contenido una vez removidas las palabras vacías.

a)



b)

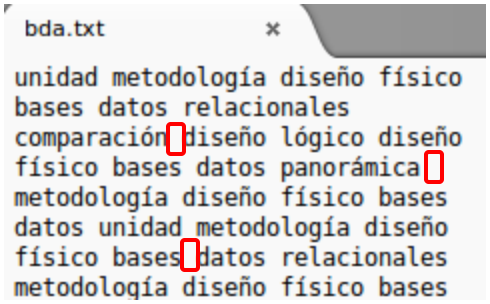
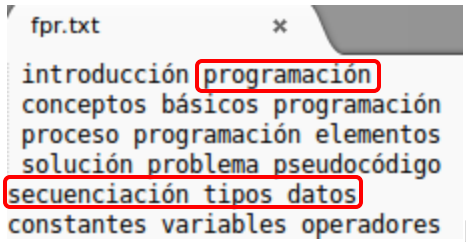


Figura 42. Plan docente de Bases de datos avanzadas. a) Texto del contenido antes de eliminar las palabras vacías. b) Texto del contenido una vez suprimidas las palabras vacías. Elaboración: propia.

2. Una vez eliminadas las palabras vacías se procede a lematizar las palabras del contenido. La Figura 43 presenta el contenido del documento de Fundamentos de la programación. El literal a) muestra las palabras del contenido sin lematizar, y el literal b) presenta el texto aplicado el proceso de lematización.

a)



b)

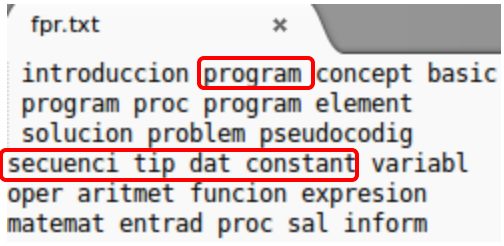


Figura 43. Contenido del plan docente de Fundamentos de la programación. a) Texto sin lematizar. b) Texto lematizado.
Elaboración: propia.

3. Se identifica bigramas y trigramas en el texto de los documentos. A continuación, se presenta los bigramas y trigramas de los documentos.

direccion ipv
enrut dinam
bas dat relacional
replic bas dat
bas dat movil
bas dat semant
arquitectur sgbd
model entid
relacion
model relacional
algebr relacional
calcul relacional
do whil
program orient
objet
cas uso
diagram clas
expresion regular
algoritm enrut
protocol
transaccion dat
cap red
cap transport
enrut intern

4. Se finaliza definiendo el vocabulario de términos y creando la matriz términos por documentos. La tabla 3 presenta los términos del vocabulario obtenidos de los planes docentes.

Tabla 3. Presenta los términos de cada plan docente que forman el vocabulario.

Plan docente	Términos de cada documento		
Arquitectura y Seguridad de redes	- direccion ipv - vlans	- conexión - internet	- enrut dinam - red
Redes y Sistemas Distribuidos	- algoritm enrut - udp - protocol transaccion dat	- cap red - cap transport - datagram - enrut intern	- multiplex - demultiplex - red - tcp
Fundamentos de Bases de Datos	- arquitectur sgbd - model entid relacion	- sql - model relacional - normaliz	- algebr relacional - calcul relacional
Bases de Datos Avanzadas	- sgbd - bas dat relacional	- bas dat movil - bas dat semant	- segur - oracl - replic bas dat
Fundamentos de Programación	- program - do whil - for	- arregl - clas - algoritm	- program orient objet - metod
Programación de algoritmos	- program - clas	- program orient objet	- herenci - metod
Programación Avanzada	- uml - cas uso	- diagram clas	- expresion regular

Elaboración: propia.

La Figura 44 presenta extracto de la matriz términos por documentos creada a partir del vocabulario de términos (Ver Anexo 6).

	Terms								
Docs	cap red	cap transport	cas uso	clas	conexion	datagram	demultiplex	diagram clas	direccion ipv
asr	0	0	0	0	1	0	0	0	1
bda	0	0	0	0	0	0	0	0	0
fbd	0	0	0	0	0	0	0	0	0
fpr	0	0	0	4	0	0	0	1	0
palg	0	0	0	7	0	0	0	0	0
pav	0	0	1	1	1	0	0	1	0
rsd	2	3	0	0	5	3	2	0	1

Figura 44. Extracto de la matriz términos por documentos creada a partir de los términos de los siete planes docentes.
Elaboración: propia.

Una vez ejecutado el pre-procesamiento de datos se ejecuta el algoritmo k-means para realizar el agrupamiento, en donde k = 3 debido a que después de realizar varias estimaciones para *k* (Ver Tabla 9) con este se obtiene buenos resultados. La Figura 45 presenta la estructura de los tres grupos.

Cluster 1	Cluster 2	Cluster 3
- algoritm - arregl - clas - for - metod - program - program orient objet - red - sql	- algebr relacional - arquitectur sgbd - bas dat móvil - bas dat relacional - bas dat semant - calcul relacional - replic bas dat - diagram clas - do whil - enrut dinam - herenci - normaliz - oracl - segur - cas uso - sgbd - uml - vlans - expresion regular - model relacional - model entid relación	- algoritm enrut - tcp - cap red - udp - cap transport - conexion - datagram - demultiplex - direccion ipv - enrut intern - internet - multiplex - protocol transaccion dat

Figura 45. Estructura de los tres clústeres formados por el algoritmo k-means.
Elaboración: propia.

La Figura 46 presenta el plot con los tres grupos formados por el algoritmo k-means, en donde cada grupo posee un color y figura diferente que los distingue.

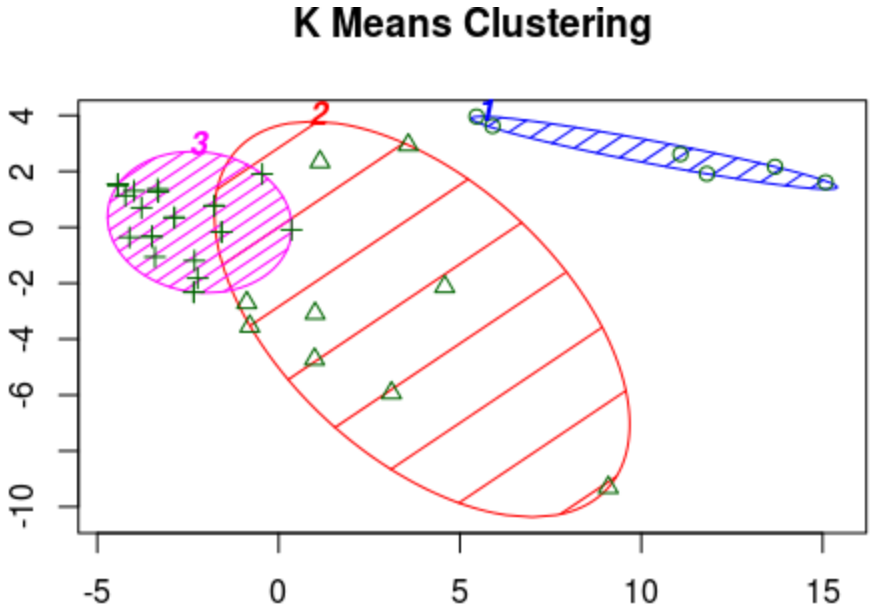


Figura 46. Grupos obtenidos al aplicar el algoritmo k-means, donde k = 3.
Elaboración: propia.

5.3.2. Prueba número dos.

Para la prueba número dos se utiliza seis planes docentes que corresponden a las áreas de conocimiento de física, programación y bases de datos. En la presente prueba se trabaja con planes docentes de otras áreas de conocimiento para analizar los resultados del algoritmo k-means. Los documentos empleados son:

- Bases de Datos Avanzadas.
- Física.

- ❖ Fundamentos de Bases de Datos.
- ❖ Fundamentos de Programación.
- ❖ Programación de Algoritmos.
- ❖ Programación Avanzada.

Se aplica el proceso de limpieza de datos a los documentos seleccionados con el propósito de obtener un conjunto de documentos libres de caracteres innecesarios:

1. Se elimina todo signo de puntuación y carácter especial del contenido de los documentos. La Figura 47 presenta el contenido del plan docente de Fundamentos de bases de datos. El literal a) muestra el texto del documento antes de remover los caracteres especiales. El literal b) exhibe el texto una vez suprimidos los caracteres especiales.

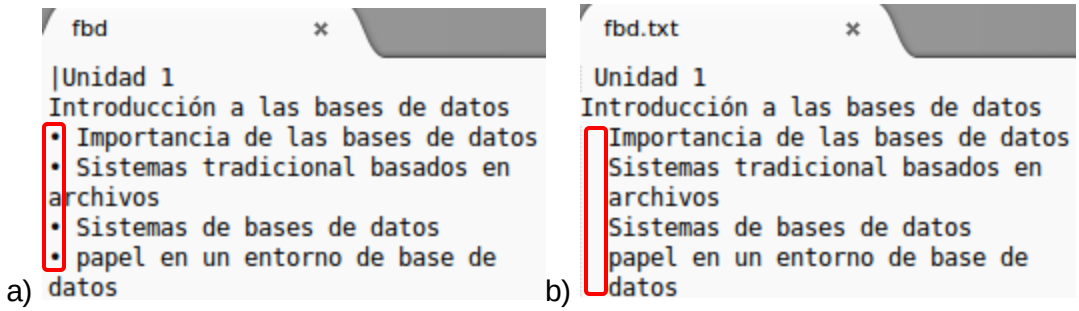


Figura 47. Contenido del plan docente de Fundamentos de bases de datos. a) Plan docente antes de remover los caracteres especiales. b) Documento una vez suprimidos los caracteres especiales.
Elaboración: propia.

2. Se transforma el contenido de los documentos a minúsculas. La Figura 48 muestra el proceso de transformar el texto de mayúsculas a minúsculas. El literal a) exhibe el contenido del plan docente antes convertir a minúsculas, y el literal b) una vez transformado a minúsculas.

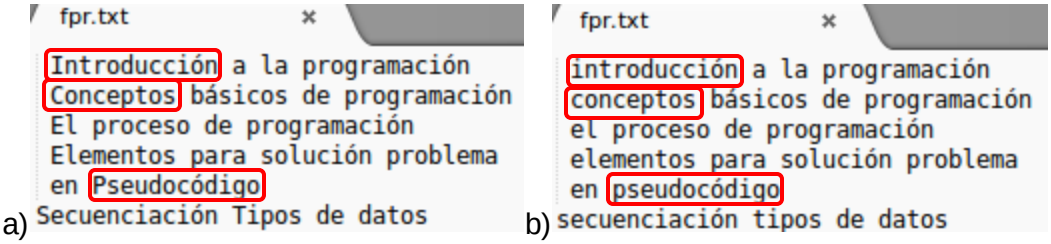


Figura 48. Plan docente de Fundamentos de la programación. a) Texto antes de convertir a minúsculas. b) Texto una vez transformando a minúsculas.
Elaboración: propia.

3. A continuación, se remueve toda la numeración del contenido de los planes docentes. La Figura 49 muestra los resultados de remover los números del plan docente de fundamentos de bases de datos.

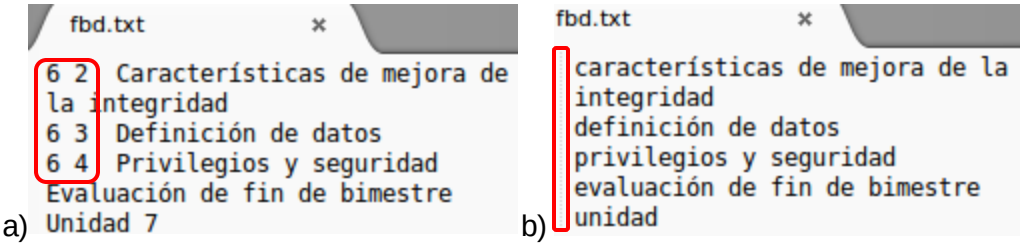


Figura 49. Contenido del plan docente de Fundamentos de bases de datos. a) Texto antes de suprimir los números. b) Texto después de remover los números.
Elaboración: propia.

4. Se finaliza removiendo todo espacio en blanco adicional del contenido de los documentos. La Figura 50 muestra los resultados de remover los espacios en blanco adiciones del plan docente de Física.

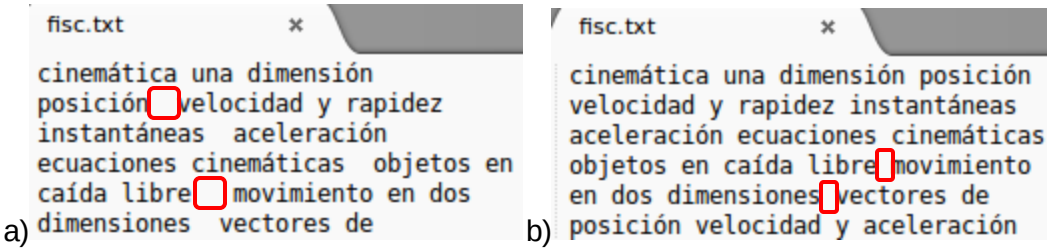


Figura 50. Contenido del plan docente de Física. a) Texto antes de remover los espacios en blanco. b) Texto después de suprimir los espacios en blanco adicionales.
Elaboración: propia.

Una vez ejecutado el proceso de limpieza se procede emplear las tareas del pre-procesamiento de datos:

1. Remover palabras vacías del contenido de los documentos. La Figura 51 presenta el resultado de remover las palabras vacías.

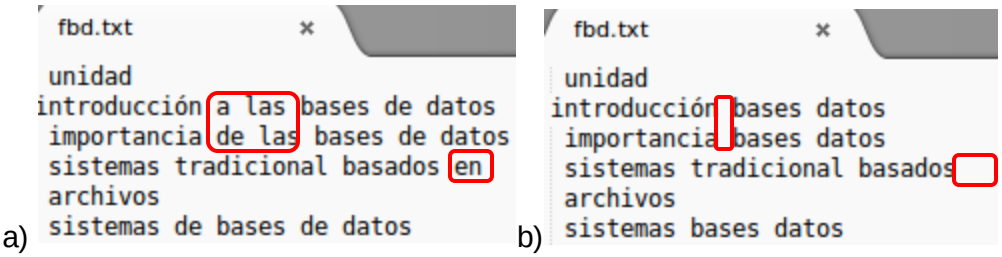
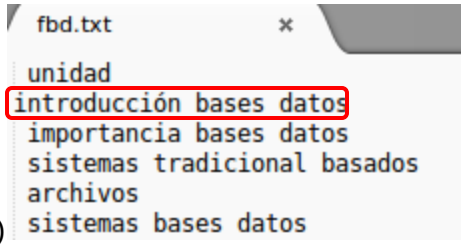


Figura 51. Plan docente de Fundamentos de bases de datos. a) Contenido con stop words. b) Contenido sin stop words.
Elaboración: propia.

2. Lematizar las palabras del contenido de los planes docentes. La Figura 52 presenta el resultado de la lematización.

a)



b)

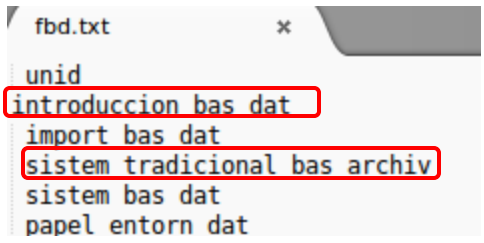


Figura 52. Contenido del plan docente de Fundamentos de bases de datos. a) Palabras del texto sin lematizar. b) Palabras del texto lematizadas.
Elaboración: propia.

3. Identificar bigramas y trigramas en el contenido de los documentos. Los bigramas y trigramas determinados en los planes docentes de la prueba son los siguientes:

bas dat relacional	model entid	program orient
replic bas dat	relacion	objet
bas dat movil	energ cinet	do whil
bas dat semant	carg electr	expresion regular
entorn bas dat	camp electr	diagram clas
model relacional	electr diferent	cas uso
algebr relacional	energ potencial	
calcul relacional	corrient electr	

4. Crear el vocabulario de términos y la matriz términos por documentos. La tabla 4 presenta los términos que forman el vocabulario.

Tabla 4. Términos del vocabulario obtenido de los seis planes docentes.

Plan docente	Términos de cada documento		
Bases de Datos Avanzadas	- sgbd - bas dat relacional	- replic bas dat - bas dat movil	- bas dat semant - oracl
Fundamentos de Bases de Datos	- entorn bas dat - model relacional	- calcul relacional - model entid relacion	- normaliz - segur - algebr relacional
Física	- cinemat - veloc - rapidez - aceler	- energ cinet - carg electr - camp electr	- electr diferent - energ potencial - corrient electr
Fundamentos de Programación	- program orient objet	- do whil - clas	- herenci
Programación de Algoritmos	- program orient objet	- clas	- herenci
Programación Avanzada	- uml - cas uso	- expresion regular	- subproces - diagram clas

Elaboración: propia.

La Figura 53 presenta extracto de la matriz términos por documentos creada a partir del vocabulario (Ver Anexo 7).

Terms									
Docs	corrient	electr	diagram	clas	do whil	electr	diferent	energ cinet	energ potencial
bda	0	0	0	0	0	0			
fbd	0	0	0	0	0	0			
fisc	2	0	0	2	2	2			
fpr	0	1	4	0	0	0			
palg	0	0	0	0	0	0			
pav	0	1	0	0	0	0			

Figura 53. Extracto de la matriz términos por documentos de la prueba número dos. Elaboración: propia.

Una vez ejecutado el pre-procesamiento de datos se ejecuta el algoritmo k-means para realizar el agrupamiento, en donde k = 3 debido a que después de ejecutar varias estimaciones para k (Ver Tabla 9) con el presente valor se obtienen resultados coherentes. La Figura 54 presenta la organización de los tres clústeres de la prueba.

Cluster 1	Cluster 2	Cluster 3
- aceler - camp electr - carg electr - cinemat - corrient electr - electr diferent - energ cinet - energ potencial - rapidez - veloc	- clas - do whil - normaliz - program orient objet - replic bas dat - sgbd	- algebr relacional - bas dat movil - bas dat relacional - bas dat semant - calcul relacional - cas uso - diagram clas - entorn bas dat - expresion regular - herenci - model entid relacion - model relacional - oracl - segur - subproces - uml

Figura 54. Estructura de los clústeres formados por el algoritmo k-means. Elaboración: propia.

La Figura 55 muestra el plot con los tres clústeres formados por el algoritmo k-means, en donde cada uno de los grupos presenta un color y figura diferente para facilitar su interpretación.

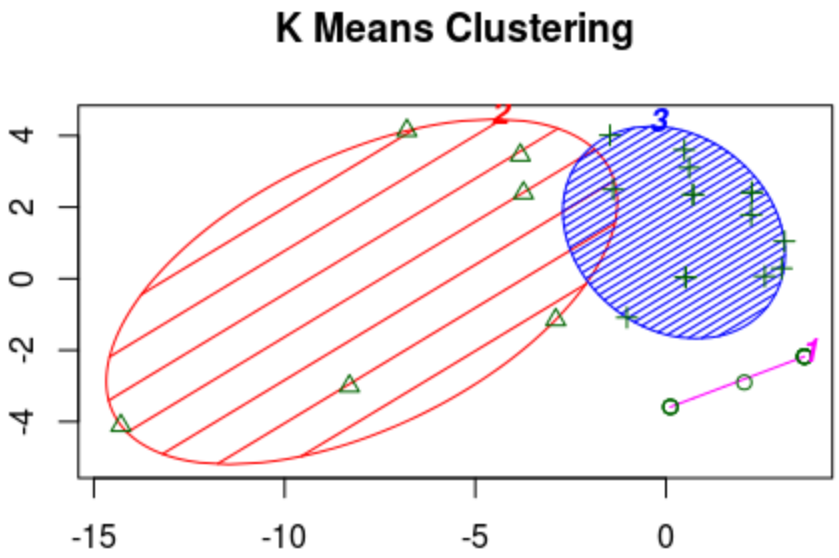


Figura 55. Gráfica de los resultados obtenidos al aplicar el algoritmo k-means, donde k = 3.
Elaboración: propia.

5.3.3. Prueba número tres.

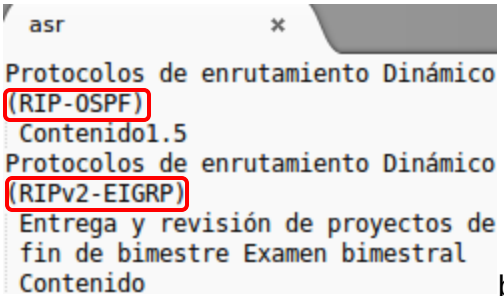
La prueba número tres utiliza siete planes docentes que pertenecen a las áreas de conocimiento de redes, inteligencia artificial, administración empresarial y desarrollo de software. Se utiliza en cada una de las pruebas planes docentes de diferentes áreas de conocimiento con el fin de identificar contenidos similares en todos los planes docentes que contiene la titulación seleccionada.

- Arquitectura y Seguridad de Redes.
- Redes y Sistemas Distribuidos.
- Organización y Administración Empresarial.
- Fundamentos de Ingeniería de Software.
- Ingeniería de Requisitos.
- Ingeniería Web.
- Inteligencia Artificial Avanzada.

Se aplica las tareas del proceso de limpieza de datos a cada uno de los planes docentes:

1. Eliminar caracteres especiales y signos de puntuación del contenido de los documentos. La Figura 56 presenta el resultado de remover los signos de puntuación y caracteres especiales de los planes docentes.

a)



b)

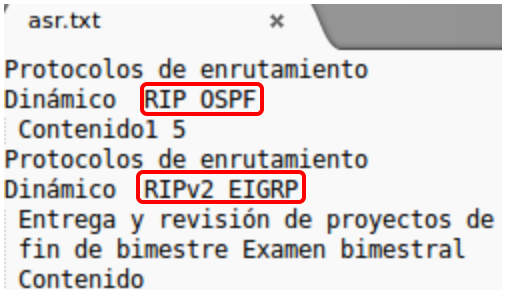
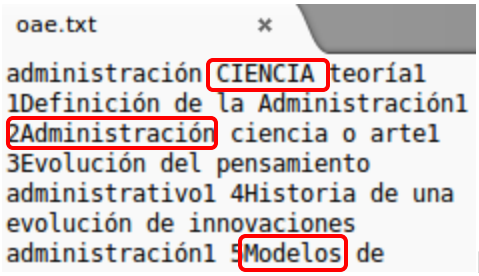


Figura 56. Plan docente de Arquitectura y seguridad de redes. a) Contenido sin remover los signos de puntuación. b) Contenido del plan docente una vez suprimidos los signos de puntuación y caracteres especiales.
Elaboración: propia.

2. Convertir el texto del contenido de los planes docentes a minúsculas. La Figura 57 muestra el resultado de convertir el texto a minúsculas.

a)



b)

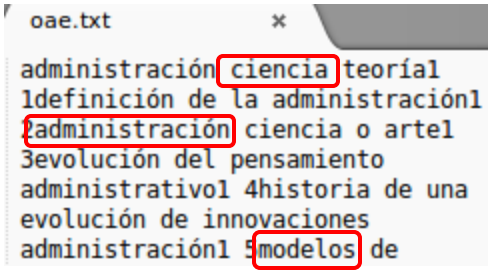
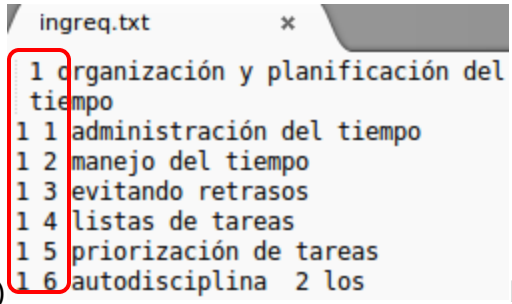


Figura 57. Plan docente de Organización y administración empresarial. a) Texto sin transformar a minúsculas. b) Texto después de transformar a minúsculas.
Elaboración: propia.

3. Remover del contenido de los documentos toda la numeración. La Figura 58 muestra el resultado de remover los números del contenido de los planes docentes.

a)



b)

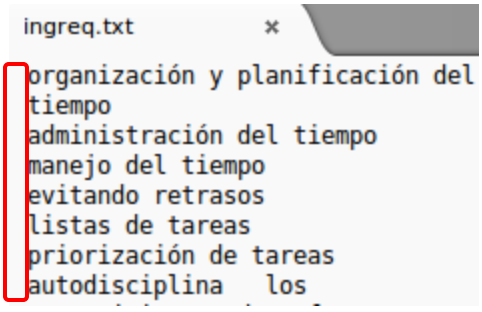
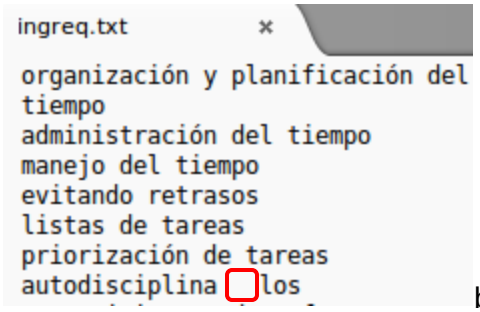


Figura 58. Plan docente de Ingeniería de requisitos. a) Texto del documento antes de suprimir los números. b) Texto del documento una vez removidos los números.
Elaboración: propia.

4. Eliminar espacios en blanco adicionales del texto de los documentos. La Figura 59 presenta el resultado de eliminar los espacios en blanco.

a)



b)

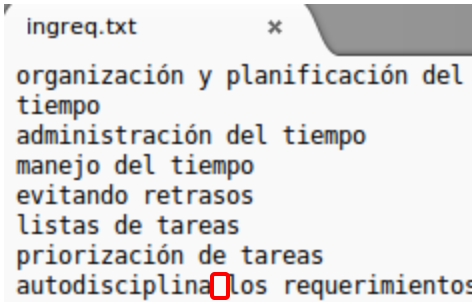
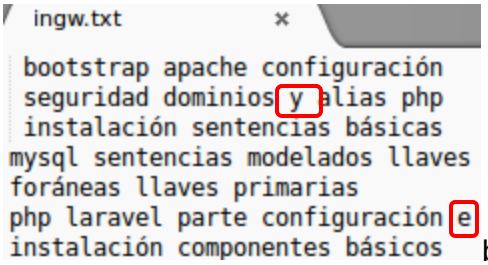


Figura 59. Contenido del plan docente de Ingeniería de requisitos. a) Texto sin remover los espacios en blanco. b) Texto una vez eliminados los espacios en blanco adicionales.
Elaboración: propia.

A continuación, se emplea las tareas del pre-procesamiento de datos sobre el conjunto de documentos limpios:

1. Primero se elimina los stop words del contenido de los documentos. La Figura 60 muestra el resultado de remover las palabras vacías del plan docente.

a)



b)

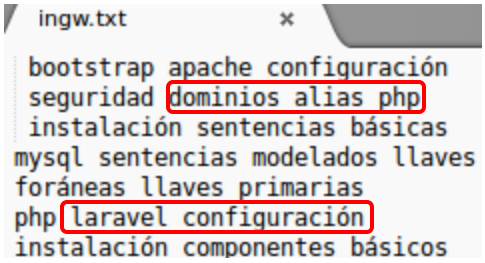
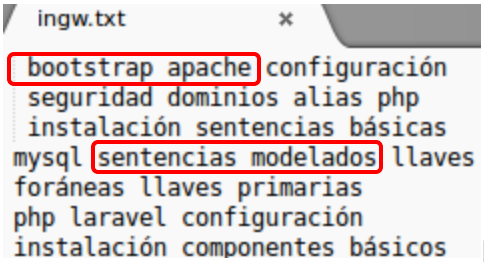


Figura 60. Contenido del plan docente de Ingeniería Web. a) Documento antes de remover las palabras vacías. b) Documento una vez suprimidas las palabras vacías.
Elaboración: propia.

2. Segundo se lematiza las palabras del contenido de los planes docentes. La Figura 61 muestra el resultado de lematizar las palabras del contenido de los documentos.

a)



b)

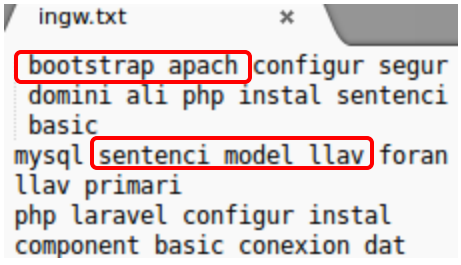


Figura 61. Contenido del plan docente de Ingeniería Web. a) Texto antes de aplicar la lematización. b) Texto una vez aplicado el proceso de lematización.
Elaboración: propia.

3. Tercero se identifica los bigramas y trigramas en el contenido de los documentos:

direccion ipv
ingeni softwar
prueb desarroll
ingeni requer
arbol clasif
regl neuronal
ajust pes
model ocult
markov
analisis clust
proces requer
document especif requer
captur requer
especif requer softwar
document requer
desarroll modul
gestion usuari
metodolog desarroll web
web app
factor human
tecnolog inform
departamentaliz
algoritm enrut
protocol
transaccion dat
cap red
cap transport

4. Cuarto se define el vocabulario de términos y se crea la matriz términos por documentos. La tabla 5 presenta los términos que forman el vocabulario.

Tabla 5. Planes docentes con sus respectivos términos que forman el vocabulario.

Plan docente	Términos de cada documento		
Arquitectura y Seguridad de Redes	- direccion ipv - vlans	- conexión - enrut dinam	- internet
Redes y Sistemas Distribuidos	- algoritm enrut - udp	- tcp - cap red - protocol transaccion dat	- cap transport - datagram
Organización y Administración Empresarial	- polit - premis - reingeni	- factor human - tecnolog inform	- departamentaliz - comerci
Fundamentos de Ingeniería de Software	- ingeni softwar	- ingeni requer	- prueb desarroll
Ingeniería de Requisitos	- tar - proces requer - prototip	- document especif requer - captur requer	- document requer - especif - especif requer softwar
Ingeniería Web	- php - instal - laravel - maquet	- desarroll modul - gestion usuari	- metodolog desarroll web - accesibil - web app
Inteligencia Artificial Avanzada	- arbol clasif - regl neuronal	- ajust pes - model ocult markov	- analisis clust

Elaboración: propia.

La Figura 62 presenta extracto de la matriz términos por documentos creada a partir del vocabulario (Ver Anexo 8).

Docs	Terms especif	factor	human	gestion	usuari	ingeni	softwar	instal	internet
ArqSegRed.txt	0	0	0	0	0	1
FunIngSof.txt	1	0	0	2	0	0
IngReq.txt	3	0	0	1	0	0
IngWeb.txt	0	0	3	0	3	0
IntArtAvan.txt	0	0	0	0	0	0
OrgAdmEmp.txt	0	2	0	0	0	0
RedSistDist.txt	0	0	0	0	0	2

Figura 62. Extracto de la matriz términos por documentos creada a partir del vocabulario de términos.
Elaboración: propia.

Con la matriz términos por documentos creada se ejecuta el algoritmo k-means para realizar el agrupamiento, en donde el valor para k = 6, porque después de llevar a cabo varias experimentaciones con el presente valor se obtienen los mejores resultados (Ver Tabla 9). La Figura 63 presenta la organización de los seis grupos formados por el algoritmo k-means.

Cluster 1
- captur requer
- document especif requer
- document requer
- especif
- especif requer softwar
- ingeni softwar
- proces requer
- prototip
- tar

Cluster 2
- ajust pes
- analisis clust
- arbol clasif
- model ocult markov
- regl neuronal

Cluster 3
- comerci
- departamentaliz
- factor human
- polit
- premis
- reingeni
- tecnolog inform

Cluster 4
- cap transport
- algoritm enrut
- conexion
- tcp
- datagram

Cluster 5
- accesibil
- gestion usuari
- laravel
- php
- metodolog desarroll web
- desarroll modul
- instal
- maquet
- web app

Cluster 6
- cap red
- direccion ipv
- ingeni requer
- prueb desarroll
- protocol transaccion dat
- udp
- enrut dinam
- internet
- vlans

Figura 63. Clústeres formados por el algoritmo k-means en la prueba número tres.
Elaboración: propia.

La Figura 64 presenta el plot con los seis grupos formados por el algoritmo k-means en la prueba, cada uno de los grupos se distinguen con facilidad debido a que poseen un color y figura diferente.

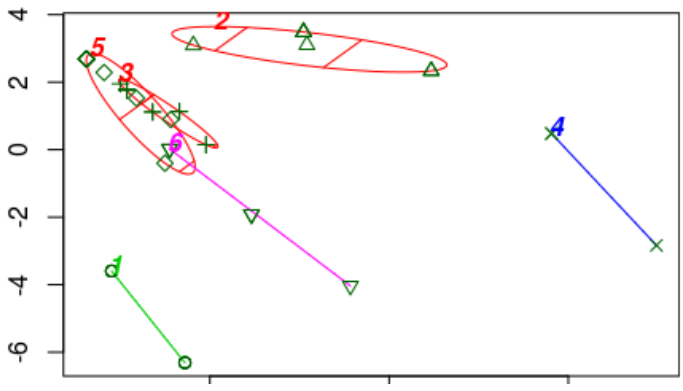


Figura 64. Resultados obtenidos al aplicar el algoritmo k-means, donde k = 6.
Elaboración: propia.

5.3.4. Prueba número cuatro.

Para la prueba número cuatro se utiliza planes docentes de las áreas de conocimiento de: teoría de autómatas, desarrollo de software, gestión de proyectos y programación. Algunas áreas de conocimiento contienen un solo plan docente con el fin de observar como el algoritmo k-means realiza el agrupamiento.

- Teoría de Autómatas y Compiladores.
- Fundamentos de Ingeniería de Software.
- Fundamentos de Programación.
- Gestión de Proyectos.
- Lógica de la Programación.
- Procesos de Ingeniería de Software.
- Programación de Algoritmos.
- Programación Avanzada.

Una vez cargados los documentos a R Project se aplica el proceso de limpieza datos. Las tareas que abarca el proceso son las siguientes:

1. Eliminar caracteres especiales y signos de puntuación. La Figura 65 presenta el resultado de eliminar los signos de puntuación del contenido de los planes docentes.

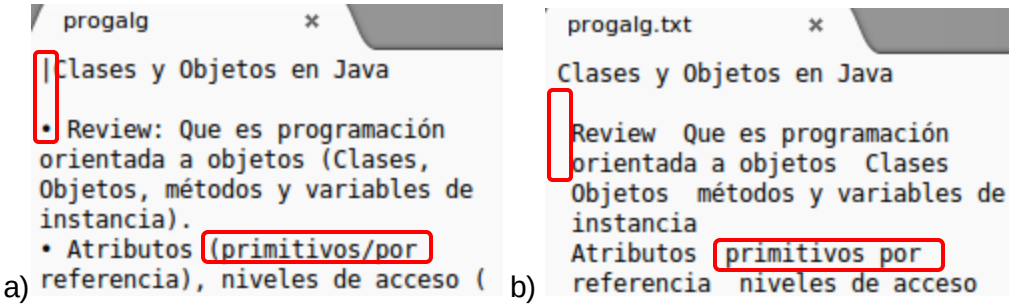
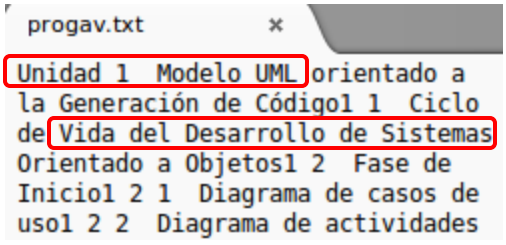


Figura 65. Contenido del plan docente de Programación de algoritmos. a) Texto antes de eliminar signos de puntuación y caracteres especiales. b) Texto una vez removidos los signos de puntuación y caracteres.
Elaboración: propia.

2. Transformar las palabras que estén en mayúsculas a minúsculas. La Figura 66 presenta el resultado de transformar el texto de mayúsculas a minúsculas en el contenido de los planes docentes.

a)



b)

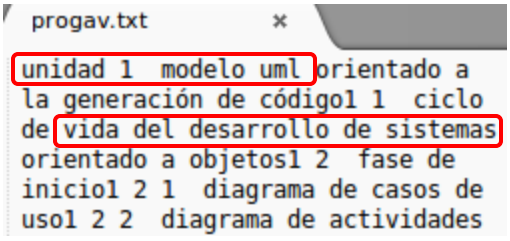
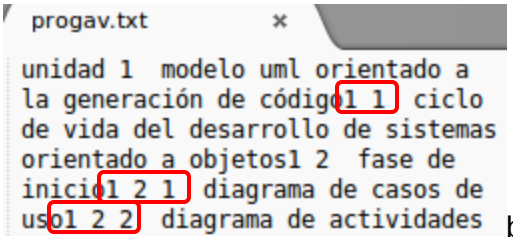


Figura 66. a) Contenido del plan docente de Programación avanzada. a) Texto con caracteres en mayúsculas. b) Texto convertido a minúsculas.
Elaboración: propia.

3. Suprimir los números del texto. La Figura 67 muestra el contenido del plan docente de Programación avanzada. El literal a) presenta el texto antes de remover los números y el literal b) una vez removidos.

a)



b)

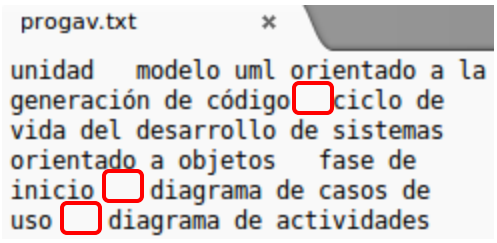
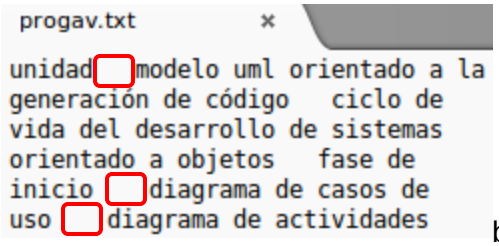


Figura 67. Contenido del plan docente de Programación avanzada. a) Texto antes de remover los números. b) Texto una vez suprimidos los números.
Elaboración: propia.

4. Al eliminar los números y caracteres especiales del contenido se forman espacios en blanco que hay que eliminarlos. La Figura 68 muestra el contenido del plan docente de Programación avanzada. El literal a) presenta el texto antes de suprimir los espacios en blanco. El literal b) exhibe el texto una vez removidos los espacios en blanco.

a)



b)

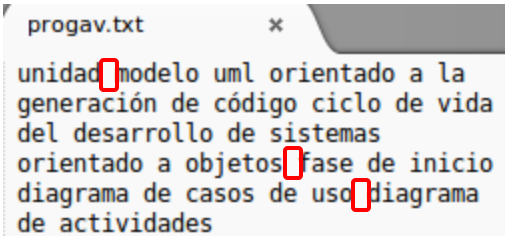
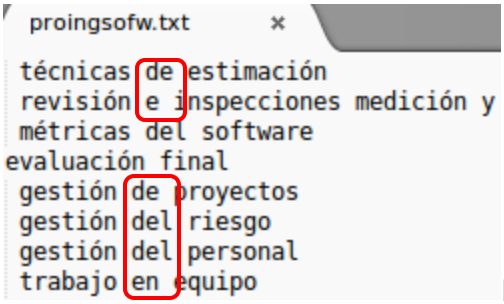


Figura 68. Plan docente de Programación avanzada. a) Contenido con espacios en blanco adicionales. b) Texto sin espacios en blanco.
Elaboración: propia.

Con el conjunto de datos libres de caracteres incensarios se ejecuta el pre-procesamiento de datos. Las tareas que abarca el proceso son las siguientes:

1. Remover del texto las palabras vacías. La Figura 69 muestra el contenido del plan docente de Procesos de ingeniería de software. El literal a) presenta el texto antes de remover los stop words y el literal b) exhibe el texto después de suprimir las palabras vacías.

a)



b)

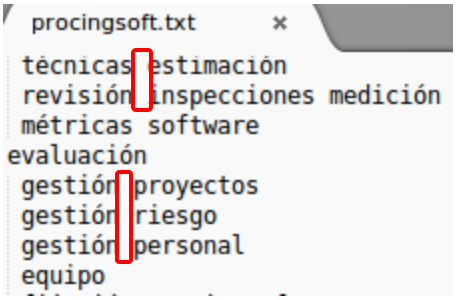
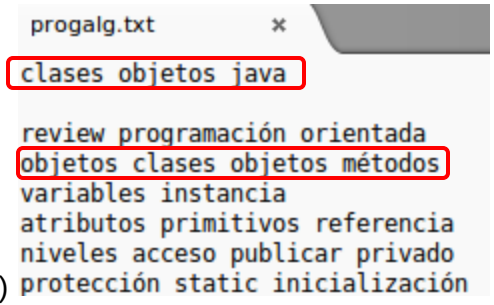


Figura 69. Plan docente de Procesos de ingeniería de software. a) Texto antes de remover las palabras vacías. b) Texto una vez suprimidas las palabras vacías.
Elaboración: propia.

2. Lematizar las palabras del texto. La Figura 70 presenta el resultado de aplicar el proceso de lematización a los términos de los planes docentes.

a)



b)

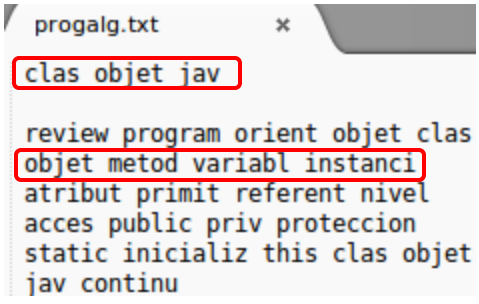


Figura 70. Plan docente de Programación de algoritmos, a) Texto sin lematizar las palabras del contenido, b) Texto una vez aplicado el proceso de lematización.
Elaboración: propia.

3. Identificar en el texto bigramas y trigramas. A continuación, se presenta los bigramas y trigramas encontrados:

automat finit	cas uso	gestion calid
teor automat	do whil	cierr proyect
expresion regular	program orient	logic program
analisis lexic	objet	regl procedent
analisis sintact	codig etic	algoritm orient dat
ingeni softwar	ingeni softwar	prueb escritori
prueb desarroll	control proyect	diagram fluj
ingeni requer	gestion riesg	api jav

program ficher jav	recurs human	mejor proces
algoritm orden	gestion proyect	calendariz proyect
diagram clas	gestion riesg	
expresion regular	gestion calid	

4. Determinar el vocabulario de términos y crear la matriz términos por documentos. La tabla 6 muestra el vocabulario de términos obtenido de los planes docentes de la prueba.

Tabla 6. Presenta el vocabulario de términos de los planes docentes.

Plan docente	Términos de cada documento		
Teoría de Autómatas y Compiladores	- automat finit - teor automat	- analisis lexic	- analisis sintact
Fundamentos de Ingeniería de Software	- ingeni softwar - ingeni requer	- prueb desarroll - clas	- program orient objet
Fundamentos de Programación	- do whil - for	- clas	- program orient objet
Gestión de Proyectos	- codig etic - cierr proyect - pmi	- cost - calendariz proyect	- control proyect
Lógica de la Programación	- logic program - silog - proposicion	- regl procedent - algoritm orient dat	- miniespecif - diagram fluj
Procesos de Ingeniería de Software	- calendariz proyect - recurs human	- gestion proyect - gestion riesg	- gestion calid - mejor proces
Programación de Algoritmos	- clas - program orient objet	- herenci	- api jav - program ficher jav
Programación Avanzada	- uml - cas uso	- diagram clas	- expresion regular

Elaboración: propia.

La Figura 71 presenta extracto de la matriz términos por documentos creada a partir del vocabulario (Ver Anexo 9).

	Terms									
Docs	prueb	escritori	regl	procedent	requer	silog	sistem	subproces	teor	automat	uml	variabl
Compila.txt	0	0	0	0	0	0	2	0	0			
FunIngSof.txt	0	0	4	0	2	0	0	0	0			
FunProg.txt	0	0	0	0	0	0	0	0	2			
GestProy.txt	0	0	0	0	0	0	0	0	0			
LogProg.txt	1	1	0	1	0	0	0	0	1			
ProcIngSof.txt	0	0	0	0	1	0	0	0	0			
ProgAlg.txt	0	0	0	0	0	0	0	0	1			
ProgAvan.txt	0	0	0	0	1	3	0	3	0			

Figura 71. Extracto de la matriz términos por documentos obtenida de los ocho planes docentes.
Elaboración: propia.

Una vez ejecutadas las tareas del pre-procesamiento de datos se aplica el algoritmo k-means para realizar el agrupamiento. En la prueba k = 4 debido a que una vez ejecutadas varias

pruebas con este valor se obtienen grupos formados de manera coherente (Ver Tabla 9). La Figura 72 presenta la estructura de los cuatro grupos construidos por el algoritmo k-means.

Cluster 1	Cluster 2	Cluster 3	Cluster 4
- algoritm orient dat - api jav - codig etic - expresion regular - gestion proyect - herenci - mejor proces - regl procedent - uml	- calendariz proyect - cierr proyect - control proyect - diagram clas - do whil - gestion calid - gestion riesg - ingeni requer - ingeni softwar - pmi - cas uso - program ficher jav - proposicion - prueb desarroll - recurs human - silog - cost - teor automat	- analisis lexic - analisis sintact - automat finit	- clas - diagram fluj - for - logic program - program orient objet

Figura 72. Estructura de los cuatro grupos formados por el algoritmo k-means.
Elaboración: propia.

La Figura 73 presenta el plot con los cuatro clústeres, donde cada uno de los grupos posee su respectivo número, color y figura que favorece su interpretación.

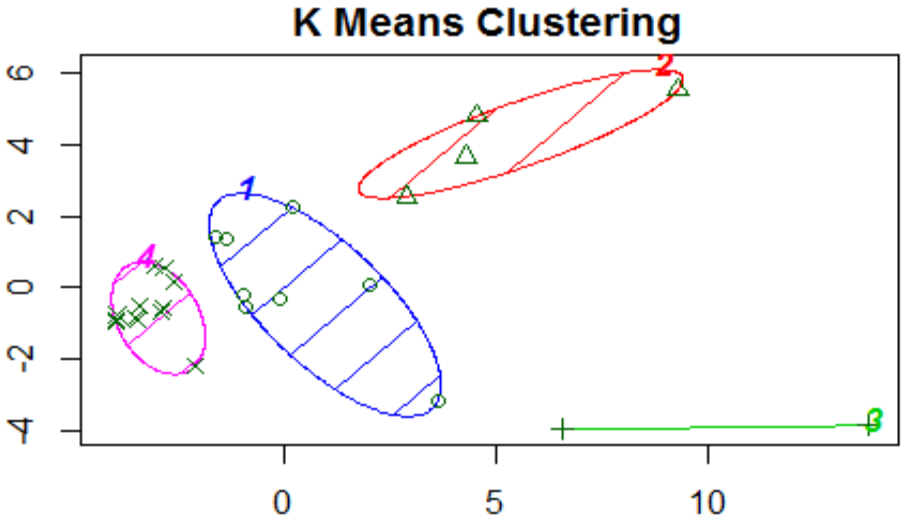


Figura 73. Gráfica de los clusters obtenidos al aplicar el algoritmo k-means, donde k = 4.
Elaboración: propia.

5.3.5. Prueba número cinco.

La prueba número cinco emplea dos planes docentes del área de conocimiento de bases de datos y dos del área de redes. Se trabaja solo con los cuatro planes docentes con el propósito de examinar la efectividad del algoritmo agrupando pequeñas cantidades de documentos.

- ❖ Arquitectura y Seguridad de Redes.
- ❖ Bases de Datos Avanzadas.
- ❖ Fundamentos de Base de Datos.
- ❖ Redes y Sistemas Distribuidos.

Una vez cargado el corpus de planes docentes a R Project se ejecuta las tareas del proceso de limpieza de datos:

1. Eliminar signos de puntuación y caracteres especiales. La Figura 74 presenta el contenido del plan docente de Arquitectura y seguridad de redes antes y después de remover los signos de puntuación.

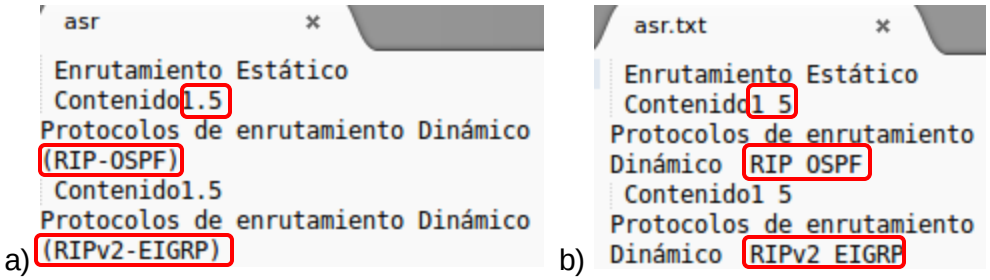


Figura 74. Contenido del plan docente de Arquitectura y seguridad de redes, a) Antes de remover los signos de puntuación y b) Después de suprimir los signos de puntuación y caracteres especiales.
Elaboración: propia.

2. Transformar el texto a minúsculas. La Figura 75 muestra el contenido del plan docente de Arquitectura y seguridad de redes antes y después de convertir el texto a minúsculas.

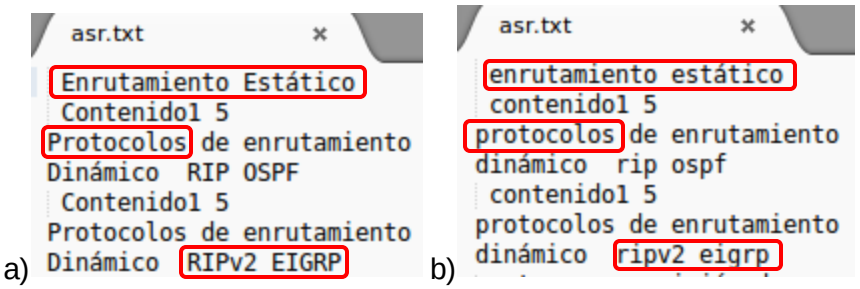


Figura 75. a) Contenido antes de convertir el texto a minúsculas. b) Contenido después de transformar el texto a minúsculas.
Elaboración: propia.

3. Eliminar números. La Figura 76 muestra el contenido del plan docente de Redes y sistemas distribuidos antes y después de remover los números.

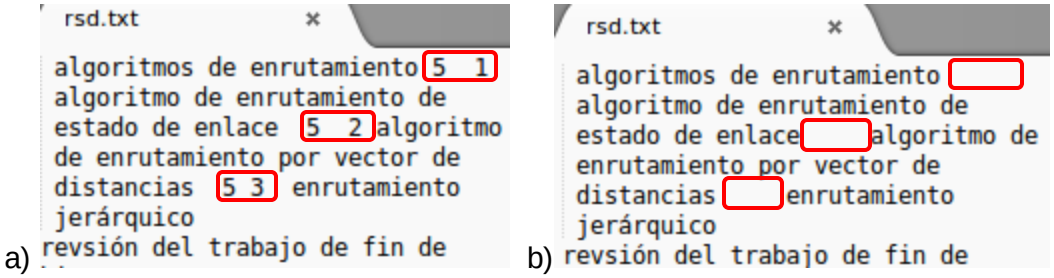


Figura 76. a) Texto antes de remover los números. b) Texto después de suprimir los números.
Elaboración: propia.

4. Eliminar espacios en blanco adicionales. La Figura 77 presenta el contenido del plan docente antes de remover los espacios en blanco y después de suprimirlos.

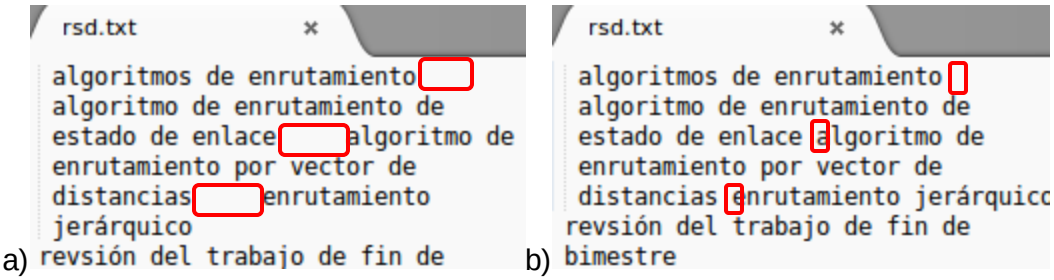


Figura 77. Plan docente de Redes y sistemas distribuidos. a) Texto antes de suprimir los espacios en blanco. b) Texto una vez removidos los espacios en blanco.
Elaboración: propia.

Al conjunto de datos limpio se aplica las tareas del pre-procesamiento de datos:

1. Remover palabras vacías. La Figura 78 muestra el contenido del plan docentes antes de remover las palabras vacías y después de eliminarlas.

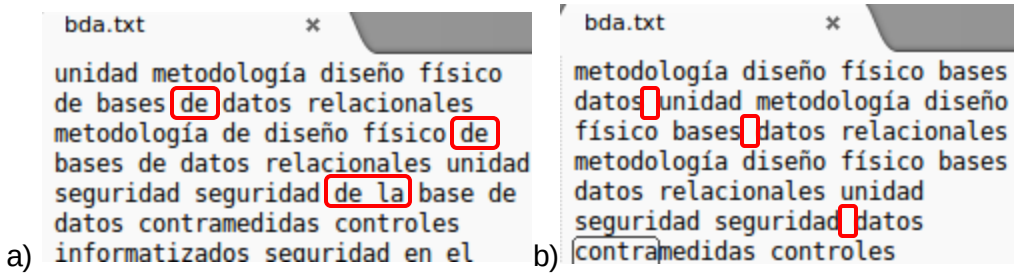


Figura 78 a) Texto sin remover las palabras vacías. b) Texto una vez suprimidas las palabras vacías.
Elaboración: propia.

2. Lematizar las palabras. La Figura 79 presenta el texto del plan docente de Bases de datos avanzadas antes y después de la lematización.

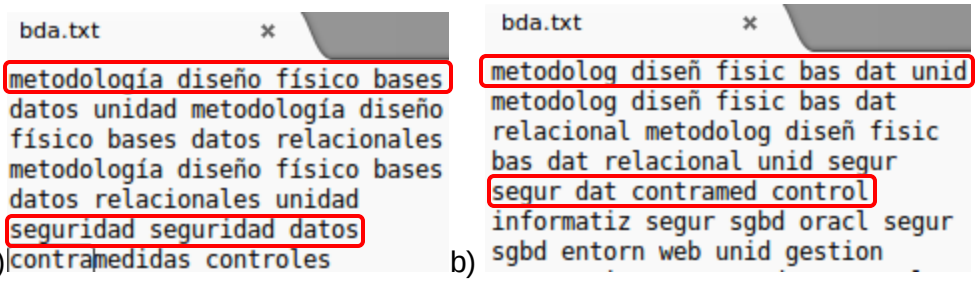


Figura 79. Contenido del plan docente de Bases de datos avanzadas. a) Texto antes de lematizar las palabras. b) Texto después de la lematización.
Elaboración: propia.

3. Determinar bigramas y trigramas. Los n-gramas identificados en los planes docentes son los siguientes:

direccion ipv	algoritm enrut	entorn bas dat
enrut dinam	protocol	model relacional
bas dat relacional	transaccion dat	algebr relacional
replic bas dat	cap red	calcul relacional
bas dat movil	cap transport	model entid
bas dat semant	enrut intern	relacion

4. Formar el vocabulario de términos y construir la matriz términos por documentos. La tabla 7 presenta los términos de cada uno de los documentos.

Tabla 7. Términos de cada uno de los planes docentes que forman el vocabulario

Plan docente	Términos de cada documento		
Arquitectura y Seguridad de Redes	- direccion ipv - vlans	- conexion - internet	- enrut dinam - rip
Bases de Datos Avanzadas	- bas dat relacional - segur - oracl	- sgbd - web - replic bas dat	- bas dat movil - bas dat semant
Fundamentos de Base de Datos	- entorn bas dat - sgbd - model relacional	- algebr relacional - calcul relacional - sql	- segur - model entid relacion - normaliz
Redes y Sistemas Distribuidos	- algoritm enrut - udp - protocol transaccion dat	- cap red - cap transport - datagram - tcp	- enrut intern - multiplex - demultiplex

Elaboración: propia.

La Figura 80 presenta síntesis de la matriz términos documentos creada a partir del vocabulario (Ver Anexo 10).

Docs	Terms protocol	transaccion	dat	replic	bas	dat	rip	segur	sgbd	sql	tcp	udp	vlans	web
ArqSegRed.txt			0		0	1	0	0	0	0	0	2	0	
BDAvan.txt			0		5	0	4	5	0	0	0	0	4	
FundBD.txt			0		0	0	1	4	9	0	0	0	0	
RedSistDist.txt			2		0	0	0	0	0	4	2	0	0	

Figura 80. Síntesis de la matriz términos por documentos de los cuatro planes docentes.
Elaboración: propia.

Con la matriz términos por documentos construida se ejecuta el algoritmo k-means para agrupar los contenidos de los planes docentes. En la prueba k = 2 porque después de realizar varias estimaciones con el presente valor se obtiene resultados coherentes y eficientes (Ver Tabla 9). La Figura 81 muestra la composición de los dos grupos.

Cluster 1		Cluster 2	
- algoritm enrut	-cap red	- algebr relacional	-bas dat movil
- cap transport	-conexion	- bas dat relacional	-bas dat semant
- datagram	-demultiplex	- calcul relacional	-model entid relacion
- direccion ipv	-enrut dinam	- model relacional	-normaliz
- enrut intern	-entorn bas dat	- oracl	-replic bas dat
- internet	-multiplex	- segur	-sgbd
- rip	-tcp	- sql	-web
- udp	-vlans		
- protocol transaccion dat			

Figura 81. Estructura de los dos grupos formados por el algoritmo k-means.
Elaboración: propia.

La Figura 82 presenta el plot creado a partir de los dos clústeres formados por al algoritmo k-means. Los términos de cada uno de los grupos estan representados por figuras y colores distintos facilitando la identificación en la gráfica.

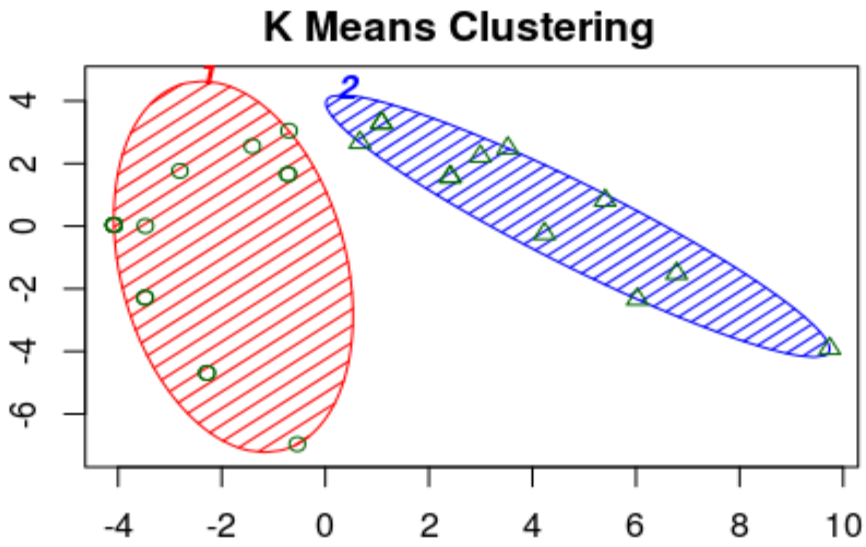


Figura 82. Gráfica de los clusters obtenidos al aplicar el algoritmo k-means, donde k = 2.
Elaboración: propia.

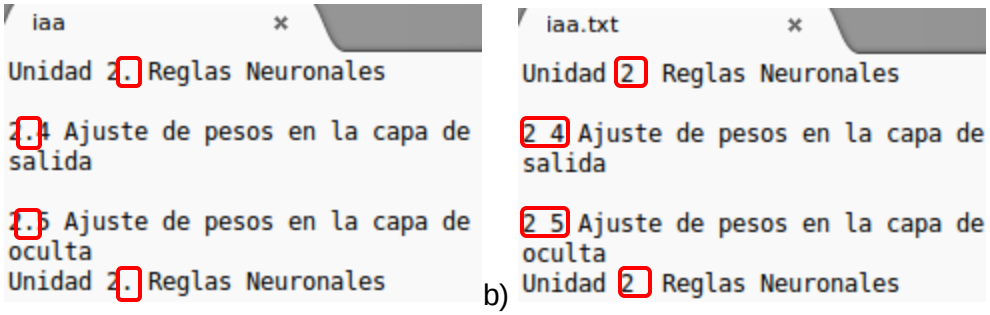
5.3.6. Prueba número seis.

La prueba número seis es la última de las pruebas desarrolladas, emplea cinco planes docentes pertenecientes a las áreas de conocimiento de: redes, bases de datos e inteligencia artificial.

- Arquitectura y Seguridad de Redes.
- Bases de Datos Avanzadas.
- Fundamentos de Bases de Datos.
- Inteligencia Artificial Avanzada.
- Redes y Sistemas Distribuidos.

Se aplica las tareas del proceso de limpieza de datos a los documentos de los planes docentes. Las tareas son las siguientes:

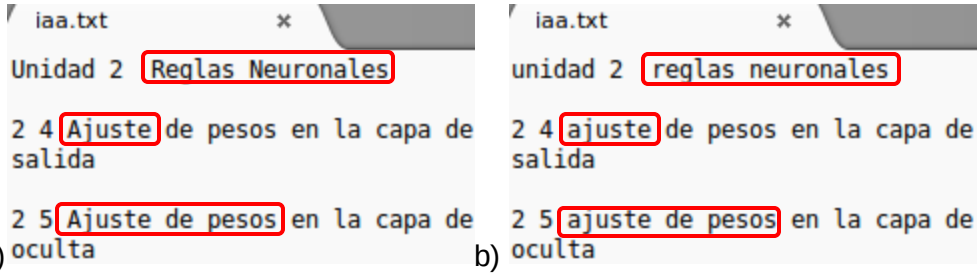
1. Se elimina todo signo de puntuación y carácter especial del contenido. La Figura 83 presenta el resultado de remover los signos de puntuación en los planes docentes.



a) b)

Figura 83. Contenido del plan docente de Inteligencia artificial avanzada. a) Texto antes de remover los caracteres especiales. b) Texto una vez suprimidos los caracteres especiales.
Elaboración: propia.

2. Se convierte las palabras que están en mayúsculas a minúsculas. La Figura 84 muestra el resultado de aplicar el proceso.

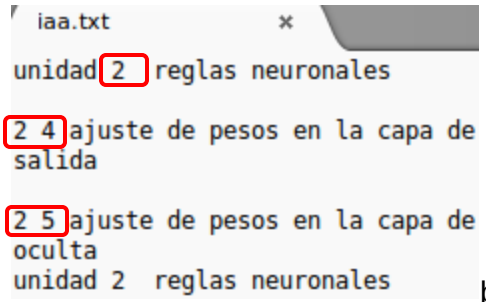


a) b)

Figura 84. a) Texto del plan docente de Inteligencia artificial avanzada antes de convertir a minúsculas. b) Texto una vez ejecutado el proceso.
Elaboración: propia.

3. Se remueve los números del contenido de los documentos. La Figura 85 presenta el resultado de ejecutar el proceso.

a)



b)

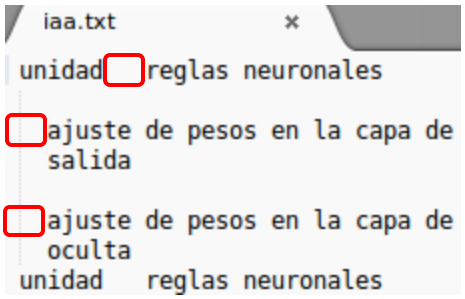
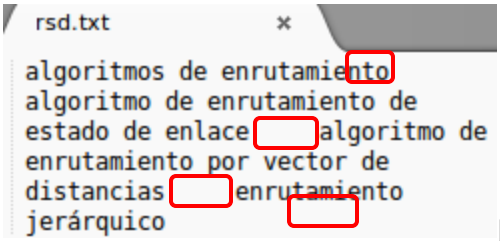


Figura 85. a) Texto antes de remover la numeración. b) Texto una vez suprimida la numeración.
Elaboración: propia.

4. Se suprime los espacios en blanco adicionales del contenido. La Figura 86 presenta el texto del plan docentes de Redes antes y después de remover los espacios en blanco.

a)



b)

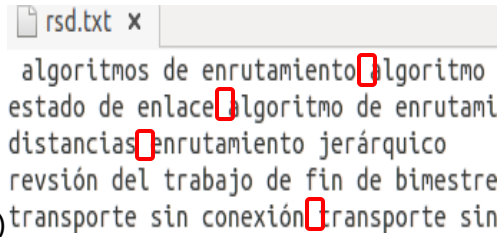
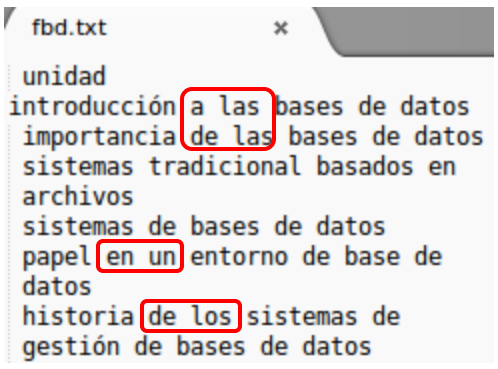


Figura 86. a) Texto de plan docente de Redes y sistemas distribuidos con espacios en blanco adicionales. b) Texto sin espacios en blanco adicionales.
Elaboración: propia.

Una vez que el contenido de los documentos está limpio se ejecuta el pre-procesamiento de datos. Las tareas del proceso son las siguientes:

1. Suprimir del contenido los stop words. La Figura 87 muestra los resultados de aplicar el proceso.

a)



b)

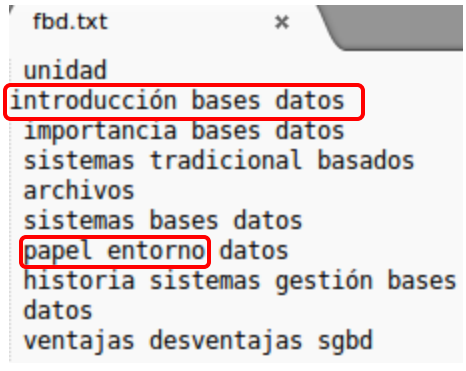
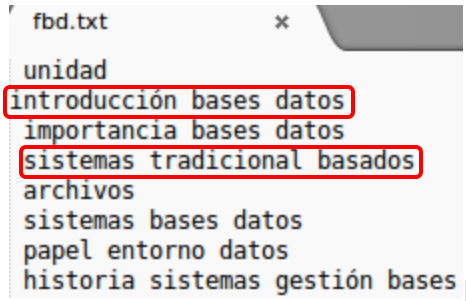


Figura 87. a) Contenido del plan docente antes de remover stops words. a) Texto una vez llevado a cabo el proceso.
Elaboración:

2. Lematizar las palabras del contenido. La Figura 88 muestra los resultados de aplicar la lematización.

a)



b)

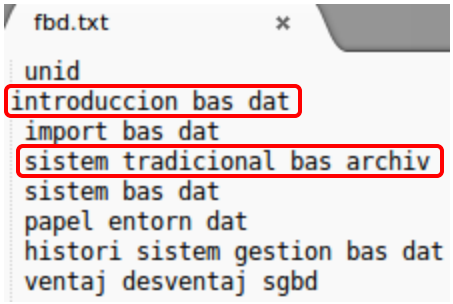


Figura 88. Contenido del plan docente de Fundamentos de bases de datos. a) Texto sin lematizar. b) Texto aplicado el proceso de lematización.
Elaboración: propia.

3. Determinar los bigramas y trigramas existentes en el contenido:

bas dat relacional	model entid	cap transport
replic bas dat	relacion	enrut intern
bas dat movil	direccion ipv	arbol clasif
bas dat semant	enrut dinam	regl neuronal
entorn bas dat	algoritm enrut	ajust pes
model relacional	protocol	analisis clust
algebr relacional	transaccion dat	model ocult
calcul relacional	cap red	markov

4. Crear la matriz términos por documentos a partir del vocabulario de términos. La tabla 8 presenta los términos del vocabulario.

Tabla 8. Términos del vocabulario obtenido del contenido de los planes docentes.

Plan docentes	Términos		
Arquitectura y Seguridad de Redes	- conexión - internet	- direccion ipv - enrut dinam	- vlans
Bases de Datos Avanzadas	- sgbd - bas dat relacional	- replic bas dat - bas dat movil	- bas dat semant - oracl
Fundamentos de Bases de Datos	- sgbd - entorn bas dat - model relacional	- algebr relacional - calcul relacional	- normaliz - segur - model entid relacion
Inteligencia Artificial Avanzada	- arbol clasif	- ajust pes - analisis clust	- model ocult markov

Plan docentes	Términos		
	- regl neuronal		
Redes y Sistemas Distribuidos	- algoritm enrut - udp	- tcp - cap red - cap transport	- datagram - enrut intern - protocol transaccion dat

Elaboración: propia.

La Figura 89 presenta extracto de la matriz términos por documentos creada a partir del vocabulario (Ver Anexo 11).

Docs	Terms ajust	pes	algebr	relacional	algoritm	enrut	analisis	clust	arbol	clasif
asr	0		0		0		0		0	
bda	0		0		0		0		0	
fbd	0		3		0		0		0	
iaa	2		0		0		4		3	
rsd	0		0		4		0		0	

Figura 89. Extracto de la matriz términos por documentos creada a partir del contenido de los cinco planes docentes.
Elaboración: propia.

Una vez obtenida la matriz térmnos por documentos en el pre-procesamiento de datos se ejecuta el algoritmo k-means para agrupar el contenido de los documentos. El valor óptimo para k = 3 porque después de efectuar varias estimaciones con el presente valor se forman clústeres coherentes (Ver Tabla 9). La Figura 90 muestra la estructura de los cuatro clústeres.

Cluster 1	Cluster 2	Cluster 3	
- algoritm enrut - cap red - cap transport - conexion - datagram - direccion ipv - enrut intern - internet - protocol transaccion dat - tcp - udp	- ajust pes - analisis clust - arbol clasif - model ocult markov - regl neuronal	- algebr relacional - bas dat relacional - calcul relacional - entorn bas dat - model relacional - oracl - segur - vlans	- bas dat movil - bas dat semant - enrut dinam - model entid relacion - normaliz - replic bas dat - sgbd

Figura 90. Estructura de los grupos formados por el algoritmo k-means.
Elaboración: propia.

La Figura 91 presenta los cuatro grupos dibujados en el plot, cada uno de los grupos posee su respectivo número, color y figura para facilitar su la visualización y análisis.

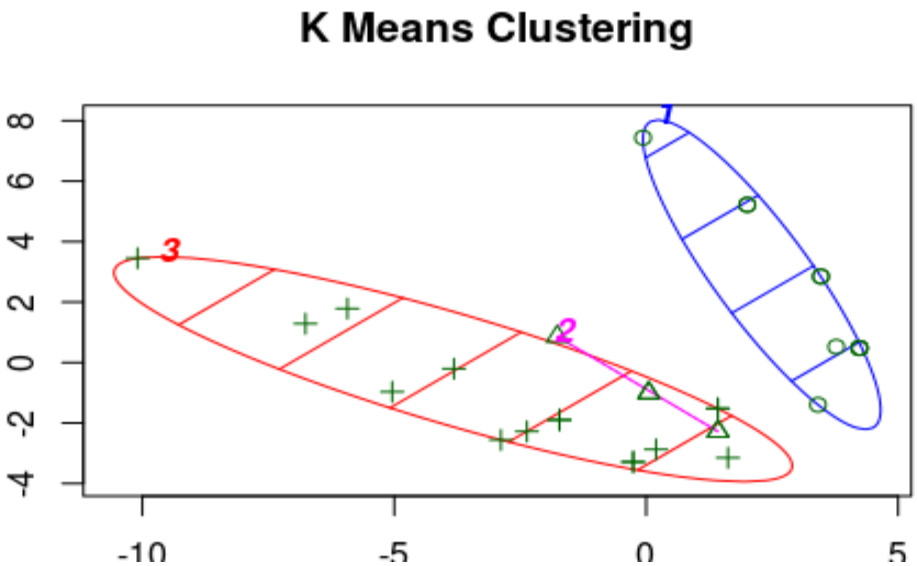


Figura 91. Gráfica de los clústeres obtenidos al aplicar el algoritmo k-means, donde k = 3. Elaboración: propia.

5.3.7. Evaluación de resultados para K

La Tabla 9 presenta la estimación de los valores para **K** en cada una de las pruebas realizadas con el algoritmo k-means. Los valores seleccionados en cada una de las pruebas se encuentran resaltados de color amarillo.

Tabla 9. Estimación de los valores para k en cada una de las pruebas del algoritmo k-means.

Prueba uno											
K = 3			**K = 4**				**K = 5**				
c1	**c2**	**c3**	**c1**	**c2**	**c3**	**c4**	**c1**	**c2**	**c3**	**c4**	**c5**
9	21	13	4	13	20	6	4	9	4	24	2
Prueba dos											
K = 3			**K = 4**				**K = 5**				
c1	**c2**	**c3**	**c1**	**c2**	**c3**	**c4**	**c1**	**c2**	**c3**	**c4**	**c5**
10	6	16	4	10	16	2	12	2	3	5	10
Prueba tres																	
K = 5					**K = 6**						**K = 7**						
c1	**c2**	**c3**	**c4**	**c5**	**c1**	**c2**	**c3**	**c4**	**c5**	**c6**	**c1**	**c2**	**c3**	**c4**	**c5**	**c6**	**c7**
9	16	5	5	9	9	5	7	5	9	9	7	9	8	5	3	6	6
Prueba cuatro																	
K = 3			**K = 4**				**K = 5**					**K = 6**					
c1	**c2**	**c3**	**c1**	**c2**	**c3**	**c4**	**c1**	**c2**	**c3**	**c4**	**c5**	**c1**	**c2**	**c3**	**c4**	**c5**	**c6**
3	24	8	9	17	3	5	13	3	6	9	3	15	4	2	2	4	7

Prueba cinco								
K = 2		K = 3			K = 4			
c1	c2	c1	c2	c3	c1	c2	c3	c4
16	13	13	10	6	6	12	6	5

Prueba seis											
K = 3			K = 4				K = 5				
c1	c2	c3	c1	c2	c3	c4	c1	c2	c3	c4	c5
11	5	15	11	10	5	5	5	5	4	8	9

Elaboración: propia.

5.4. Indexación Semántica Latente en R Project

Para identificar los planes docentes con contenidos similares con el algoritmo LSI en R Project se desarrollan las siguientes tareas:

5.4.1. Construcción de la matriz.

Para aplicar el algoritmo LSI se comienza por construir una matriz M de términos por documentos. En nuestro caso se utiliza la matriz términos por documentos obtenida del procesamiento del lenguaje natural.

5.4.2. Descomponer en valores singulares a la matriz.

A la matriz de términos por documentos se la descompone en valores singulares. La función SVD en R Project viene cargada en el paquete base.

- **Paquete R base:** contiene las funciones esenciales que permiten a R dar soporte para la programación básica aritmética, etc. (R Core Team and contributors, 2016).

- **Función svd:** permite calcular la descomposición singular del valor de una matriz rectangular (R Core Team and contributors, 2016). La función *SVD* recibe como argumento de entrada la matriz términos por documentos.

A continuación, se presenta el código para descomponer en valores singulares a una matriz términos por documentos:

```
matrizsvd = crearmtd (corpusmatriz) #Matriz término documento
```

```
algsi = svd(matrizsvd) #SVD
```

Una vez aplicada la técnica de descomposición en valores singulares a la matriz se obtiene como resultado las siguientes tres matrices:

- *Matriz U*: es una matriz ortogonal real de tamaño *m* x *m*.
- *Matriz V*: es una matriz ortogonal real de tamaño *n* x *n.*
- *Matriz D*: es una matriz real de tamaño *m* x *n,* pseudo-diagonal con elementos no negativos en la diagonal.

5.4.3. Reducción de la dimensión del espacio.

Una vez aplicada la descomposición a la matriz términos por documentos es necesario realizar la reducción de la dimensión del espacio a las matrices U y V, para lo cual se toma como referencia a los valores más altos de la matriz D. A continuación, se presenta el código para reducir la dimensión del espacio a la matriz U y V, en este caso se reduce a tres columnas.

```
matrizu<-algsvd$u[,1:3]#Reducción de la dimensión a la matriz U
matrizv<-algsvd$v[,1:3]#Reducción de la dimensión del espacio V
```

5.4.4. Visualización gráfica de los documentos en el plano.

Para mejor comprensión y análisis de los grupos obtenidos al aplicar la técnica de indexación semántica latente (LSI), se opta por crear un plot en R Project que facilite la visualización. El código que se utiliza para graficar los documentos en el plot es el siguiente:

```
plot (matrizu, asp=1, main = "Representación de los documentos en el plano",
pch=16, col=34); abline (h = 0, v = 0, lty =5, col=28)
text (matrizu, labels=rownames (matrizsvd), pos=1, col=153, font=1, cex=0.8)
```

La Figura 92 presenta el diseño del plot a utilizar para representar a los documentos en el plano mediante vectores y determinar la semejanza en sus contenidos.

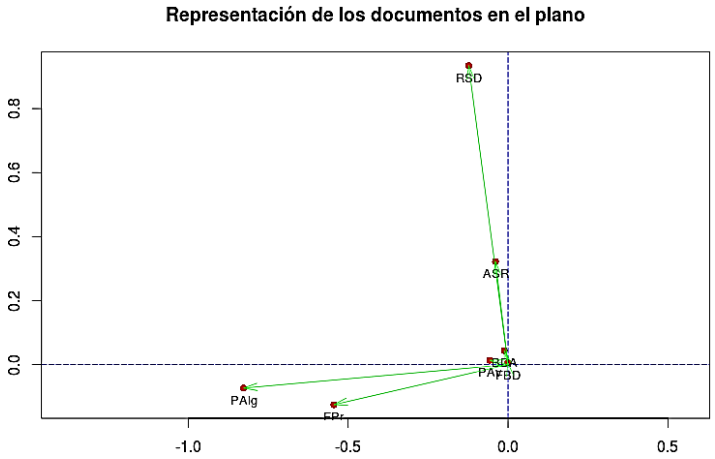


Figura 92. Plot para representar gráficamente los planes docentes en el plano. Elaboración: propia.

5.4.5. Pruebas con el algoritmo LSI.

Para comprobar la eficiencia del algoritmo de indexación semántica latente (LSI) se ejecutan pruebas con un número determinado de planes docentes con el fin de analizar los resultados. Las pruebas que se realizan en la sección son con los mismos planes docentes y vocabularios de términos empleados en las pruebas con el algoritmo k-means. Al final se realiza el análisis y comparación de los resultados obtenidos con cada algoritmo.

5.4.5.1. Prueba número uno.

Los planes docentes utilizados en la prueba número uno tanto para el algoritmo k-means y LSI pertenecen a las áreas de conocimiento de bases de datos, programación y redes.

- Arquitectura y Seguridad de Redes.
- Bases de Datos Avanzadas.
- Fundamento de Bases de Datos.
- Fundamentos de Programación.
- Programación de Algoritmos.
- Programación Avanzada.
- Redes y Sistemas Distribuidos.

Al igual que con el algoritmo k-means a los documentos se les aplica el proceso de limpieza y pre-procesamiento de datos. En el pre procesamiento de datos se crear la matriz términos por documentos (Ver Anexo 6) a partir del vocabulario de términos (Ver Tabla 3).

Una vez creada la matriz términos por documentos se la descompone en valores singulares, obteniendo como resultado tres nuevas matrices: U, V y D. La Figura 93 presenta las matrices U, V y D resultado de aplicar SVD.

```
$d
[1] 20.168740 13.188242 12.923350  9.144576  5.738050  4.787434  3.436749

$u
             [,1]       [,2]       [,3]        [,4]         [,5]        [,6]       [,7]
[1,] -0.0089460567  0.05480037 -0.12998157  0.001765388 -0.0338640098 -0.046177208  0.98830130
[2,] -0.0040848101  0.33664733  0.05465284  0.939062352  0.0008811544  0.041228866 -0.01123667
[3,] -0.0006657582  0.86406266  0.37861623 -0.331718661 -0.0008533360 -0.001151367  0.00238770
[4,] -0.6497413222 -0.04968338  0.11938488  0.010091044 -0.7480543584 -0.034810050 -0.01470147
[5,] -0.7422048096 -0.03792581  0.08775886  0.001108315  0.6563658061  0.089696060  0.03360586
[6,] -0.0479225510  0.01783278 -0.01089810  0.036999477  0.0863251798 -0.993124256 -0.04636665
[7,] -0.1567625692  0.36446729 -0.90260427 -0.081544432 -0.0315957756  0.024771139 -0.14011862

$v
              [,1]         [,2]         [,3]         [,4]         [,5]         [,6]         [,7]
 [1,] -9.902823e-05  0.1965529656  0.0878911940 -0.1088247320 -4.461460e-04 -0.0007214930  0.002084266
 [2,] -4.175496e-01  0.0753440062 -0.1948848764 -0.0304065887  2.572233e-01  0.1227623800 -0.111744944
 [3,] -3.109021e-02  0.1105431034 -0.2793716006 -0.0356689851 -2.202545e-02  0.0206967993 -0.163082728
 [4,] -2.355412e-04  0.0910439792  0.0335260641  0.0664157335  4.848054e-06  0.0083713949 -0.002574808
 [5,] -4.831052e-01 -0.0465008426  0.1122004857  0.0085728981 -1.118533e-01  0.0802521929  0.038504605
 [6,] -6.075953e-04  0.0765789721  0.0126869984  0.3080719324  4.606902e-04  0.0258356778 -0.009808691
 [7,] -3.016685e-03  0.1434487996  0.0411407752  0.2758430791  1.535632e-02 -0.1818487839 -0.022605361
 [8,] -4.050635e-04  0.0510526481  0.0084579989  0.2053812883  3.071268e-04  0.0172237852 -0.006539127
 [9,] -9.902823e-05  0.1965529656  0.0878911940 -0.1088247320 -4.461460e-04 -0.0007214930  0.002084266
[10,] -1.554510e-02  0.0552715517 -0.1396858003 -0.0178344925 -1.101272e-02  0.0103483996 -0.081541364
[11,] -2.331765e-02  0.0829073275 -0.2095287004 -0.0267517388 -1.651908e-02  0.0155225995 -0.122312046
```

Figura 93. Matrices U, V y D resultado de descomponer en valores singulares a la matriz términos por documentos de la prueba número uno.
Elaboración: propia.

Una vez que se obtiene las tres matrices (u, v y d) se reduce la dimensión del espacio a la matriz U que en nuestro caso simboliza a los documentos del corpus, la matriz está formada por siete columnas y siete filas. Se hace con la finalidad de trabajar solo con las columnas que tiene una alta concentración de valores, para saber con cuentas columnas trabajar se toma como referencia los valores más altos de la matriz D. Se observa en la Figura 93 que la matriz D tiene siete columnas de las cuales las tres primeras poseen valores altos, por lo que a la matriz U se la redice a tres dimensiones.

El código que se utiliza para reducir la dimensión de la matriz U es el siguiente:

```
matrizu <- algsvd$u[,1:3] #Reducción de la dimensión del espacio a la matriz U.
```

La Figura 94 presenta la matriz U reducida a tres columnas.

```
> matrizu
              [,1]        [,2]        [,3]
[1,] -0.0089460567  0.05480037 -0.12998157
[2,] -0.0040848101  0.33664733  0.05465284
[3,] -0.0006657582  0.86406266  0.37861623
[4,] -0.6497413222 -0.04968338  0.11938488
[5,] -0.7422048096 -0.03792581  0.08775886
[6,] -0.0479225510  0.01783278 -0.01089810
[7,] -0.1567625692  0.36446729 -0.90260427
```

Figura 94. Matriz U reducida su dimensión a tres columnas.
Elaboración: propia.

Se procede a graficar los documentos en el plano para identificar a través de los vectores la similitud que existe entre los contenidos de los planes docentes. La Figura 95 presenta los documentos graficados en el plano mediante vectores, donde cada vector posee el nombre del plan docente al que simbolizan.

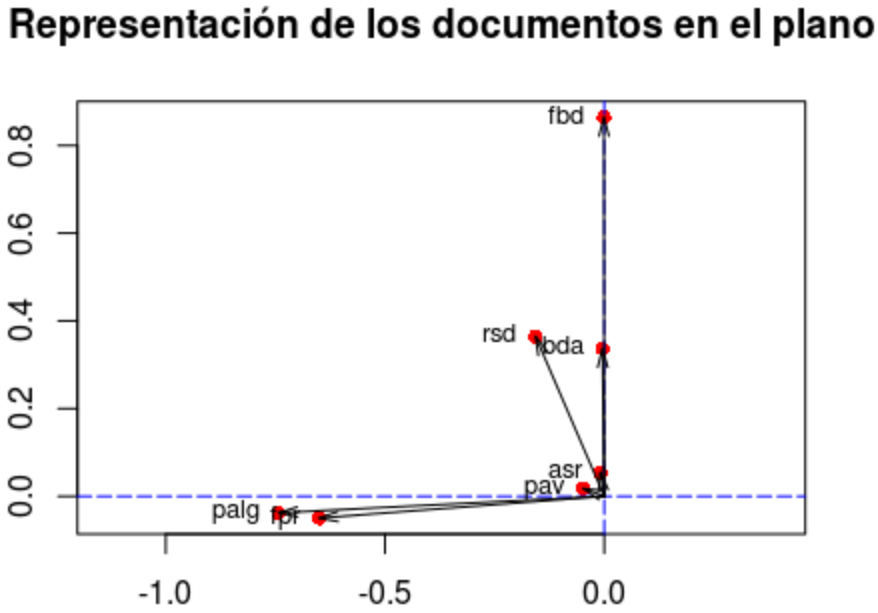


Figura 95. Representación de los siete documentos en el plano, que permite observar los documentos con contenidos similares.
Elaboración: propia.

La tabla 10 presenta a detalle los resultados obtenidos en la prueba número uno con LSI.

Tabla 10. Detalles de los resultados obtenidos al aplicar LSI.

Plan docente	Términos de cada documento	Descripción del vector
Arquitectura y Seguridad de Redes	- direccion ipv - vlans - conexión - internet - enrut dinam - red	El vector del plan docente está alineado con el vector del plan docente de Redes y Sistemas Distribuidos.
Redes y Sistemas Distribuidos	- algoritm enrut - udp	El vector que representa al plan docente está alineado con el vector del

Plan docente	Términos de cada documento	Descripción del vector
	- protocol transaccion dat - tcp - cap red - cap transport - datagram - enrut intern - multiplex - demultiplex - red	plan docente de Arquitectura y Seguridad de Redes, es decir, el vector de este plan docente pasa por el punto que representa al otro plan docente.
Fundamento de Bases de Datos	- arquitectur sgbd - model entid relacion - normaliz - sql - model relacional - algebr relacional - calcul relacional	El vector del plan docente está alineado con el vector del plan docente de Bases de Datos Avanzadas, es decir, el vector de este plan docente pasa por el punto que representa al otro plan docente.
Bases de Datos Avanzadas	- sgbd - bas dat relacional - replic bas dat - bas dat movil - bas dat semant - segur - oracl	El vector que representa al plan docente está alineado con el vector del plan docente de Fundamentos de Bases de Datos.
Programación de Algoritmos	- program - clas - program orient objet - metod - herenci	El vector del plan docente está alineado con el vector del plan docente de Fundamentos de Programación.
Programación Avanzada	- uml - cas uso - diagram clas - expresion regular	El vector que representa al plan docente no se alinea con ningún vector de otro plan docente, pero se ubica muy cerca del vector del plan docente de Programación de Algoritmos.
Fundamentos de Programación	- program - do whil - for - arregl - clas - algoritm - program orient objet - metod	El vector del plan docente está alineado con el vector del plan docente de Programación de Algoritmos.

Elaboración: propia.

5.4.5.2. Prueba número dos.

Los planes docentes utilizados en la prueba número dos pertenecen a las áreas de conocimiento de física, programación y bases de datos.

- Bases de Datos Avanzadas.
- Física.
- Fundamentos de Bases de Datos.
- Fundamentos de Programación.

- ❖ Programación de Algoritmos.
- ❖ Programación Avanzada.

Resultado de aplicar el proceso de limpieza y pre procesamiento de datos se obtiene la matriz términos por documentos (Ver Anexo 7) que es construida a partir del vocabulario de términos (Ver Tabla 4).

Una vez ejecutada la limpieza y pre-procesamiento de datos se procede aplicar el proceso del algoritmo LSI, y se inicia descomponiendo en valores singulares a la matriz términos por documentos, obteniendo como resultado tres nuevas matrices: U, V y D. La Figura 96 presenta las matrices U, V y D resultado de aplicar SVD sobre la matriz términos por documentos.

```
$d
[1] 10.704606 10.537092  8.602325  7.719113  5.472560  3.853240

$u
             [,1]         [,2]         [,3]         [,4]         [,5]         [,6]
[1,] -7.658933e-01  6.662278e-02  3.887805e-16  6.374482e-01 -5.125461e-02 -1.263958e-03
[2,] -6.358940e-01  5.841215e-02 -3.949656e-17 -7.695261e-01  7.519898e-03  2.787512e-05
[3,]  2.096408e-17  2.220339e-16  1.000000e+00  2.809766e-17  7.993591e-15 -1.006142e-16
[4,] -4.524850e-02 -5.291867e-01  4.141189e-16 -3.095182e-03 -2.932594e-02 -8.467847e-01
[5,] -7.129448e-02 -8.375458e-01  9.961001e-16 -5.689642e-03 -1.071830e-01  5.309554e-01
[6,] -4.377245e-02 -1.030919e-01 -7.887371e-15  3.804534e-02  9.924556e-01  3.225498e-02

$v
              [,1]         [,2]         [,3]         [,4]         [,5]         [,6]
 [1,]  4.489092e-18  1.731031e-16  4.649906e-01 -1.081987e-17  2.373903e-15  2.183327e-17
 [2,] -1.782113e-01  1.663044e-02 -1.371643e-17 -2.990730e-01  4.122329e-03  2.170261e-05
 [3,] -2.146440e-01  1.896807e-02  2.151731e-16  2.477415e-01 -2.809724e-02 -9.840743e-04
 [4,] -2.781369e-01  1.472784e-02 -7.549295e-16  1.529792e-01  1.546281e-01  7.394033e-03
 [5,] -1.430960e-01  1.264538e-02  7.405982e-17  1.651610e-01 -1.873149e-02 -6.560495e-04
 [6,] -1.782113e-01  1.663044e-02 -2.065532e-17 -2.990730e-01  4.122329e-03  2.170261e-05
 [7,]  1.973274e-17  1.833190e-16  4.649906e-01  1.296980e-17  2.371585e-15  2.051736e-18
 [8,]  1.479956e-17  1.374893e-16  3.487429e-01  9.727353e-18  1.778688e-15  1.538802e-18
 [9,] -4.089123e-03 -9.783710e-03 -8.513890e-16  4.928719e-03  1.813513e-01  8.370873e-03
[10,]  9.866370e-18  9.165950e-17  2.324953e-01  6.484902e-18  1.185792e-15  1.025868e-18
[11,] -6.761835e-02 -7.670673e-01  1.308427e-16 -1.834780e-03  2.281764e-02  9.389599e-02
[12,]  9.866370e-18  9.165950e-17  2.324953e-01  6.484902e-18  1.185792e-15  1.025868e-18
[13,] -8.316135e-03 -6.000503e-02 -8.097342e-16  4.527743e-03  1.759925e-01 -2.113883e-01
```

Figura 96. Matrices U, V y D resultado de descomponer en valores singulares a la matriz términos por documentos de la prueba número dos.
Elaboración: propia.

Una vez que se obtiene las tres matrices (u, v y d) se reduce la dimensión del espacio a la matriz U que representa a los documentos de la prueba, la misma que está formada por seis columnas y seis filas. Se realiza con el objetivo de trabajar solo con las columnas que tiene una alta concentración de valores, para determinar en cuentas columnas reducir se toma como referencia los valores más altos de la matriz D. Se observa en la Figura 96 que la matriz D tiene seis columnas de las cuales las tres primeras poseen valores altos, por ende, a la matriz U se la redice a tres columnas. La Figura 97 presenta a la matriz U reducida su dimensión.

```
> matrizu
              [,1]          [,2]          [,3]
[1,] -7.658933e-01  6.662278e-02  3.887805e-16
[2,] -6.358940e-01  5.841215e-02 -3.949656e-17
[3,]  2.096408e-17  2.220339e-16  1.000000e+00
[4,] -4.524850e-02 -5.291867e-01  4.141189e-16
[5,] -7.129448e-02 -8.375458e-01  9.961001e-16
[6,] -4.377245e-02 -1.030919e-01 -7.887371e-15
```

Figura 97. Matriz U reducida su dimensión a tres columnas.
Elaboración: propia.

Con los valores de la matriz U se procede a graficar los planes docentes en el plano para identificar a través de los vectores que los simbolizan la similitud que existe en sus contenidos. La Figura 98 muestra los vectores que representan a los documentos en el plano, donde cada vector posee el nombre del plan docente al que simbolizan.

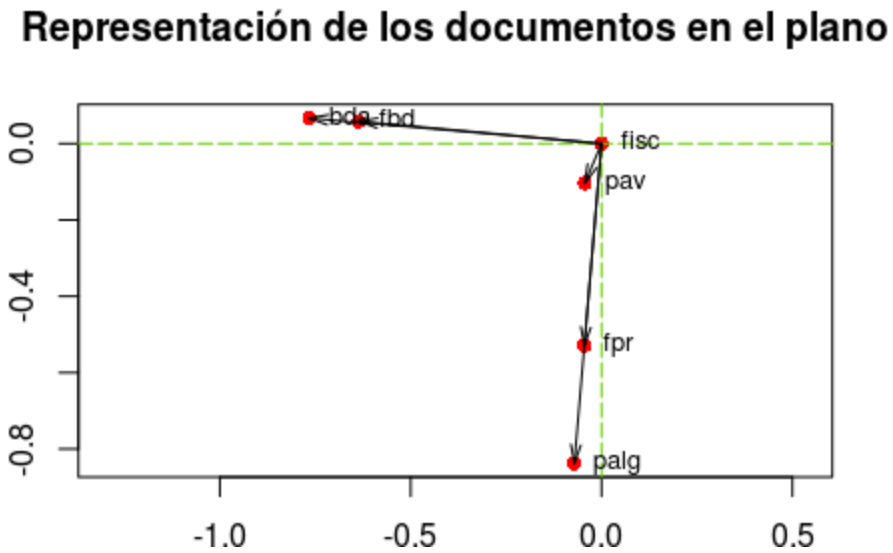


Figura 98. Representación de los seis planes docentes en el plano, indicando cuales son los que presentan contenidos similares.
Elaboración: propia.

La tabla 11 presenta a detalle los resultados obtenidos al aplicar LSI.

Tabla 11. Descripción a detalle de los resultados obtenidos al aplicar LSI en la prueba número dos.

Plan docente	Términos de cada documento	Descripción del vector
Bases de Datos Avanzadas	- sgbd - bas dat relacional - replic bas dat - bas dat movil - bas dat semant - oracl	El vector que representa al plan docente está alineado con el vector del plan docente de Fundamentos de Bases de Datos, es decir, el vector pasa por el punto que representa al otro plan docente.
Fundamentos de Bases de Datos	- entorn bas dat - model relacional - algebr relacional - calcul relacional	El vector que representa al plan docente está alineado con el vector del plan docente de Bases de Datos Avanzadas.

Plan docente	Términos de cada documento	Descripción del vector
	- model entid relacion - normaliz - segur	
Física	- cinemat - veloc - rapidez - aceler - energ cinet - carg electr - camp electr - electr diferent - energ potencial - corrient electr	El vector que representa al plan docente no se alinea con ningún vector de otro plan docente, debido a que está ubicado en la posición (0,0) del plano.
Fundamentos de Programación	- program orient objet - clas - do whil - herenci	El vector que representa al plan docente está alineado con el vector del plan docente de Programación de Algoritmos.
Programación de Algoritmos	- program orient objet - clas - herenci	El vector que representa al plan docente está alineado con el vector del plan docente de Fundamentos de Programación, es decir, el vector pasa por el punto que representa al otro plan docente.
Programación Avanzada	- uml - cas uso - expresion regular - diagram clas - subproces	El vector que representa al plan docente no se alinea con ningún vector de otro plan docente, pero se ubica muy cerca de los vectores de los planes docentes de Programación de Algoritmos y Fundamentos de Programación.

Elaboración: propia.

5.4.5.3. Prueba número tres.

Para la prueba número tres se utiliza los siete planes docentes pertenecientes a las áreas de conocimiento redes, inteligencia artificial, administración empresarial y desarrollo de software. Los documentos empleados en la prueba son los mismos para los dos algoritmos.

- Arquitectura y Seguridad de Redes.
- Redes y Sistemas Distribuidos.
- Organización y Administración Empresarial.
- Fundamentos de Ingeniería de Software.
- Ingeniería de Requisitos.
- Ingeniería Web.
- Inteligencia Artificial Avanzada.

Se aplica el proceso de limpieza y pre procesamiento de datos a los documentos con el fin de obtener la matriz términos por documentos (Ver Anexo 8), la misma que es creada a partir del vocabulario de términos (Ver Tabla 5).

Se procede aplicar el proceso del algoritmo LSI y se empieza a descomponer en valores singulares a la matriz términos por documentos, obteniendo como resultado tres nuevas matrices: U, V y D. La Figura 99 presenta las matrices U, V y D resultado de aplicar SVD.

```
$d
[1] 9.670124 7.614238 6.735137 6.480741 6.124364 3.525654 3.195613

$u
             [,1]        [,2]         [,3]         [,4]         [,5]         [,6]         [,7]
[1,] -0.0972267321  0.003460494  9.066781e-04  6.935732e-19 -3.426458e-04 -3.219441e-03 -0.9952505631
[2,] -0.0007690664 -0.036297958 -2.049703e-01 -5.933449e-16  7.092877e-02 -9.755151e-01  0.0028933745
[3,]  0.0000000000  0.000000000  3.053113e-16  1.000000e+00  6.730727e-16 -1.769418e-16  0.0000000000
[4,] -0.0087356377 -0.228037365 -9.183278e-01  1.633799e-16  2.381974e-01  2.187608e-01 -0.0015657533
[5,] -0.1421665878 -0.962481830  2.276385e-01  3.189625e-17 -3.566553e-02 -1.446632e-02  0.0108082526
[6,] -0.0003147337 -0.022830423 -2.494754e-01  6.314393e-16 -9.679607e-01 -1.711090e-02  0.0001126925
[7,] -0.9850170668  0.140630516 -2.456039e-02 -4.940694e-18  3.322841e-03  1.232718e-03  0.0966884817

$v
              [,1]          [,2]         [,3]         [,4]         [,5]         [,6]         [,7]
 [1,] -2.940326e-02 -2.528111e-01  6.759727e-02  2.753887e-18 -1.164710e-02 -8.206316e-03  6.764432e-03
 [2,]  3.264120e-17 -5.740350e-18  8.658799e-17  3.086067e-01  1.574507e-16 -3.011373e-17 -9.377436e-18
 [3,] -4.074476e-01  7.387766e-02 -1.458642e-02 -1.259402e-18  2.170244e-03  1.398570e-03  1.210265e-01
 [4,]  5.526908e-17 -4.531005e-18  1.771101e-16  6.172134e-01  3.212883e-16 -6.152858e-17 -1.572919e-17
 [5,]  1.272778e-18 -1.333826e-18  1.312837e-16  4.629100e-01  2.346860e-16 -4.636496e-17  9.981160e-20
 [6,] -2.037238e-01  3.693883e-02 -7.293211e-03 -6.297008e-19  1.085122e-03  6.992849e-04  6.051327e-02
 [7,] -3.055857e-01  5.540824e-02 -1.093982e-02 -9.445512e-19  1.627683e-03  1.048927e-03  9.076990e-02
 [8,] -1.806727e-03 -5.989762e-02 -2.726976e-01  6.107754e-17  7.778682e-02  1.240966e-01 -9.799393e-04
 [9,] -6.509404e-05 -5.996772e-03 -7.408175e-02  1.742606e-16 -3.161016e-01 -9.706510e-03  7.052953e-05
[10,] -5.340654e-01 -3.360399e-02  1.570023e-02 -9.350286e-20 -3.166691e-03 -3.268093e-03 -1.567774e-01
[11,] -3.055857e-01  5.540824e-02 -1.093982e-02 -9.445512e-19  1.627683e-03  1.048927e-03  9.076990e-02
[12,] -9.764107e-05 -8.995157e-03 -1.111226e-01  2.613909e-16 -4.741524e-01 -1.455977e-02  1.057943e-04
[13,] -2.940326e-02 -2.528111e-01  6.759727e-02  2.753887e-18 -1.164710e-02 -8.206316e-03  6.764432e-03
```

Figura 99. Matrices U, V y D resultado de descomponer en valores singulares a la matriz términos por documentos de la prueba número tres.
Elaboración: propia.

Se reduce la dimensión del espacio a la matriz U que en este caso está formada por siete columnas y siete filas. Se hace con el fin de trabajar solo con las columnas que tiene una alta concentración de valores, para identifica en cuentas columnas reducir a la matriz se toma como referencia los valores más altos de la matriz D. Se observa en la Figura 99 que la matriz D tiene siete columnas de las cuales las cinco primeras poseen valores altos por lo que a la matriz U se la redice a cinco columnas. La Figura 100 presenta a la matriz U reducida su dimensión a cinco columnas.

```
> matrizu
             [,1]        [,2]         [,3]         [,4]         [,5]
[1,] -0.0972267321  0.003460494  9.066781e-04  6.935732e-19 -3.426458e-04
[2,] -0.0007690664 -0.036297958 -2.049703e-01 -5.933449e-16  7.092877e-02
[3,]  0.0000000000  0.000000000  3.053113e-16  1.000000e+00  6.730727e-16
[4,] -0.0087356377 -0.228037365 -9.183278e-01  1.633799e-16  2.381974e-01
[5,] -0.1421665878 -0.962481830  2.276385e-01  3.189625e-17 -3.566553e-02
[6,] -0.0003147337 -0.022830423 -2.494754e-01  6.314393e-16 -9.679607e-01
[7,] -0.9850170668  0.140630516 -2.456039e-02 -4.940694e-18  3.322841e-03
```

Figura 100. Matriz U reducida su dimensión a cinco columnas.
Elaboración: propia.

Con la nueva matriz U se procede a graficar los documentos en el plano y así identificar a través de los vectores los planes docentes con contenidos similares. La Figura 101 muestra los vectores que simbolizan a los documentos en el plano.

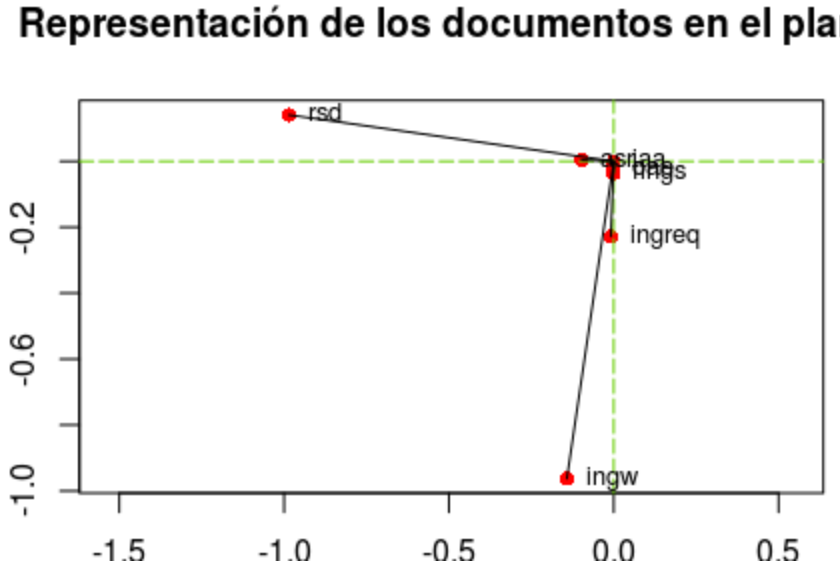


Figura 101. Representación de los siete planes docentes en el plano, permitiendo observar cuales poseen contenidos semejantes.
Elaboración: propia.

La tabla 12 muestra a detalle los resultados obtenidos en la prueba al aplicar LSI.

Tabla 12. Detalle de los resultados obtenidos al emplear el algoritmo LSI.

Plan docente	Términos de cada documento	Descripción del vector
Arquitectura y Seguridad de Redes.	- direccion ipv - vlans - conexión - enrut dinam - internet	El vector que representa al plan docente está alineado con el vector del plan docente de Redes y Sistemas Distribuidos.
Redes y Sistemas Distribuidos.	- algoritm enrut - udp - protocol transaccion dat - tcp - cap red - cap transport - datagram	El vector que representa al plan docente está alineado con el vector del plan docente de Arquitectura y Seguridad de Redes, es decir, el vector pasa por el punto que representa al otro plan docente.
Organización y Administración Empresarial.	- polit - premis - reingeni - factor human - tecnolog inform - departamentaliz - comerci	El vector que representa al plan docente está alineado con el vector del plan docente de Fundamentos de Ingeniería de Software.
Fundamentos de Ingeniería de Software.	- ingeni softwar - ingeni requer - prueb desarroll	El vector que personifica al plan docente está alineado con el vector del plan docente de Ingeniería de Requisitos.
Ingeniería de Requisitos.	- tar - proces requer	El vector que simboliza al plan docente está alineado con el

Plan docente	Términos de cada documento	Descripción del vector
	- prototip - document especif requer - captur requer - especif requer softwar - document requer - especif	vector del plan docente de Fundamentos de Ingeniería de Software.
Ingeniería Web.	- php - instal - laravel - desarroll modul - gestion usuari - maquet - metodolog desarroll web - accesibil - web app	El vector que representa al plan docente no se alinea con ningún vector de otro plan docente.
Inteligencia Artificial Avanzada.	- arbol clasif - regl neuronal - ajust pes - model ocult markov - analisis clust	El vector que simboliza al plan docente no se alinea con ningún vector de otro plan docente, debido a que está ubicado en la posición (0,0) del plano.

Elaboración: propia.

5.4.5.4. Prueba número cuatro.

Los planes docentes utilizados en la prueba número cuatro para los dos algoritmos pertenecen a las áreas de conocimiento de teoría de autómatas, desarrollo de software, gestión de proyectos y programación.

- Teoría de Autómatas y Compiladores.
- Fundamentos de Ingeniería de Software.
- Fundamentos de Programación.
- Gestión de Proyectos.
- Lógica de la Programación.
- Procesos de Ingeniería de Software.
- Programación de Algoritmos.
- Programación Avanzada.

A los planes docentes primero se les aplica el proceso de limpieza datos con el propósito de obtener un conjunto de documentos libres de caracteres incensarios. Enseguida se ejecuta el pre procesamiento de datos con el fin de conseguir el vocabulario de términos (Ver Tabla 6) para construir la matriz términos por documentos (Ver Anexo 9).

Una vez construida la matriz se ejecuta el siguiente paso del algoritmo LSI que consiste en descomponer en valores singulares a la matriz términos por documentos, consiguiendo como resultado de la descomposición tres nuevas matrices: U, V y D. La Figura 102 presenta las tres matrices U, V y D resultado de aplicar SVD.

```
$d
[1] 14.789659 13.158738 11.846741 10.427569  6.287701  4.487111  3.790177  3.464009

$u
              [,1]         [,2]         [,3]         [,4]         [,5]         [,6]         [,7]         [,8]
[1,] -9.980747e-01  0.0138062173 -0.004166970 -0.0001002085  0.002496827 -0.058324614 -0.014640495  0.004079934
[2,] -1.030190e-03 -0.0477885981  0.024888819  0.1055889561 -0.057903794  0.250267325 -0.889613818  0.358535046
[3,] -6.710243e-03 -0.5866362217  0.253326304 -0.0115635256  0.768954940 -0.005692165 -0.012944045  0.003651870
[4,] -9.245763e-05 -0.0072679209  0.007578189  0.9908240095  0.008286565 -0.026113921  0.063768934 -0.115500827
[5,] -1.688960e-03 -0.3781228391 -0.925559119  0.0047534657  0.016492769  0.007193553 -0.003466182  0.001488220
[6,] -8.427232e-03 -0.7125246622  0.278826999 -0.0103007149 -0.635313653 -0.086407597  0.053046742 -0.021984539
[7,] -6.104918e-02 -0.0515979803  0.026965413  0.0009124468 -0.037132987  0.961704030  0.248442561 -0.070039093
[8,] -3.595230e-06 -0.0003630502  0.000476072  0.0827979264  0.002498288 -0.029283217  0.373616392  0.923412794

$v
               [,1]         [,2]         [,3]         [,4]         [,5]         [,6]         [,7]         [,8]
 [1,] -1.709417e-03 -1.624452e-01  0.0706085330 -2.963504e-03 -3.031221e-01 -0.057770529  0.041987544 -0.019039679
 [2,] -4.567948e-04 -1.149420e-01 -0.3125109688  1.823422e-03  1.049208e-02  0.006412636 -0.003658069  0.001718494
 [3,] -4.049078e-01  6.295231e-03 -0.0021104388 -5.765974e-05  2.382582e-03 -0.077989527 -0.023176482  0.007066843
 [4,] -6.073617e-01  9.442847e-03 -0.0031656582 -8.648961e-05  3.573873e-03 -0.116984291 -0.034764723  0.010600265
 [5,] -1.709417e-03 -1.624452e-01  0.0706085330 -2.963504e-03 -3.031221e-01 -0.057770529  0.041987544 -0.019039679
 [6,] -6.073617e-01  9.442847e-03 -0.0031656582 -8.648961e-05  3.573873e-03 -0.116984291 -0.034764723  0.010600265
 [7,] -6.494596e-06 -5.799166e-04  0.0006798715  1.029599e-01  1.715230e-03 -0.012345836  0.115399709  0.233230345
 [8,] -4.197485e-03 -7.552896e-03  0.0043770884  1.021345e-02 -1.511471e-02  0.270100581 -0.169166573  0.083283840
 [9,] -1.250301e-05 -1.104653e-03  0.0012793711  1.900393e-01  2.635801e-03 -0.011639525  0.033649581 -0.066686220
[10,] -1.000097e-02 -5.649177e-01  0.2546648451 -1.137149e-03 -2.332192e-01  0.130228444 -0.084856259  0.043074850
[11,] -2.500602e-05 -2.209306e-03  0.0025587423  3.800786e-01  5.271603e-03 -0.023279050  0.067299162 -0.133372441
[12,] -1.250301e-05 -1.104653e-03  0.0012793711  1.900393e-01  2.635801e-03 -0.011639525  0.033649581 -0.066686220
```

Figura 102. Matrices U, V y D resultado de descomponer en valores singulares a la matriz términos por documentos de la prueba número cuatro.
Elaboración: propia.

La matriz U que simboliza a los documentos de la prueba se le reduce su dimensión del espacio con la finalidad de trabajar solo con las columnas que tiene una alta concentración de valores, para identificar con cuentas columnas trabajar se toma como referencia los valores más altos de la matriz D. En la Figura 102 se observa que la matriz D tiene ocho columnas de las cuales las cinco primeras poseen valores altos, por lo que a la matriz U se la redice a cinco columnas. La Figura 103 presenta a la matriz U reducida su dimensión a cinco columnas.

```
> matrizu
              [,1]         [,2]         [,3]         [,4]         [,5]
[1,] -9.980747e-01  0.0138062173 -0.004166970 -0.0001002085  0.002496827
[2,] -1.030190e-03 -0.0477885981  0.024888819  0.1055889561 -0.057903794
[3,] -6.710243e-03 -0.5866362217  0.253326304 -0.0115635256  0.768954940
[4,] -9.245763e-05 -0.0072679209  0.007578189  0.9908240095  0.008286565
[5,] -1.688960e-03 -0.3781228391 -0.925559119  0.0047534657  0.016492769
[6,] -8.427232e-03 -0.7125246622  0.278826999 -0.0103007149 -0.635313653
[7,] -6.104918e-02 -0.0515979803  0.026965413  0.0009124468 -0.037132987
[8,] -3.595230e-06 -0.0003630502  0.000476072  0.0827979264  0.002498288
```

Figura 103. Matriz U reducida su dimensión a cinco columnas.
Elaboración: propia.

Se procede a graficar los documentos en el plano para identificar a través de los vectores la similitud que existe en los contenidos de los planes docentes, para una visualización adecuada de los documentos en el plano se reemplaza el nombre de cada uno de ellos por números.

- Teoría de Autómatas y Compiladores=1

- ❖ Fundamentos de Ingeniería de Software=2
- ❖ Fundamentos de Programación = 3.
- ❖ Gestión de Proyectos = 4.
- ❖ Lógica de la Programación = 5.
- ❖ Programación de Algoritmos = 6.
- ❖ Programación Avanzada = 7.
- ❖ Procesos de Ingeniería de Software = 8.

La Figura 104 presenta el plot con los vectores que representan a los documentos. Además, se observa cuáles son los documentos que presentan contenidos similares.

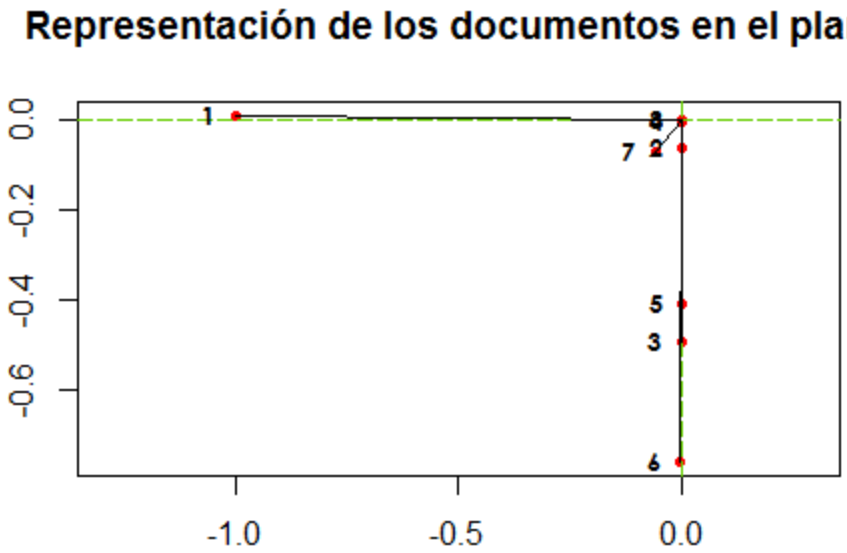


Figura 104. Representación de los ocho documentos en el plano.
Elaboración: propia.

La tabla 13 muestra a detalle los resultados obtenidos en la prueba.

Tabla 13. Descripción de los resultados obtenidos al aplicar LSI en los ocho planes docentes.

Plan docente	Términos de cada documento	Descripción del vector
Teoría de Autómatas y Compiladores	- automat finit - teor automat - analisis lexic - analisis sintact	El vector que representa al plan docente no se alinea con ningún vector de otro plan docente.
Fundamentos de Ingeniería de Software	- ingeni softwar - ingeni requer - prueb desarroll - clas - program orient objet	El vector del plan docente está alineado con los vectores de los planes docentes de: Fundamentos de Programación, Lógica de la Programación y Programación de Algoritmos.
Fundamentos de Programación	- do whil - for - clas - program orient objet	El vector del plan docente está alineado con los vectores de los planes docentes de: Fundamentos de Ingeniería de Software, Lógica de la Programación y Programación de Algoritmos.
Gestión de Proyectos.	- codig etic - cierr proyect - pmi - calendariz proyect - control proyect	El vector del plan docente está muy cerca del vector del plan docente de Procesos de Ingeniería de Software.

Plan docente	Términos de cada documento	Descripción del vector
Lógica de la Programación	- logic program - silog - proposicion - regl procedent - algoritm orient dat - diagram fluj	El vector del plan docente está alineado con los vectores de los planes docentes de: Fundamentos de Ingeniería de Software, Fundamentos de Programación y Programación de Algoritmos.
Procesos de Ingeniería de Software	- calendariz proyect - recurs human - gestion proyect - gestion riesg - gestion calid - mejor proces	El vector del plan docente está muy cerca del vector del plan docente de Gestión de Proyectos.
Programación de Algoritmos	- clas - program orient objet - herenci - api jav - program ficher jav	El vector del plan docente está alineado con los vectores de los planes docentes de: Fundamentos de Ingeniería de Software, Fundamentos de Programación y Lógica de la Programación.
Programación Avanzada	- uml - cas uso - diagram clas - expresion regular	El vector que representa al plan docente no se alinea con ningún vector de otro plan docente, pero se ubica muy cerca del vector del plan docente de Lógica de Programación.

Elaboración: propia.

5.4.5.5. Prueba número cinco.

La prueba número cinco al igual que las pruebas anteriores emplea los mismos planes docentes utilizados con el algoritmo k-means. Los documentos pertenecen a las áreas de conocimiento de bases de datos y redes

- Arquitectura y Seguridad de Redes.
- Bases de Datos Avanzadas.
- Fundamentos de Base de Datos.
- Redes y Sistemas Distribuidos.

Se ejecuta sobre los documentos el proceso de limpieza y pre procesamiento de datos con el fin de obtener el vocabulario de términos (Ver Tabla 7) y construir la matriz términos por documentos (Ver Anexo 10).

Una vez construida la matriz se aplica el paso de descomponer en valores singulares a la matriz, dando como resultado tres nuevas matrices: U, V y D. La Figura 105 presenta las matrices U, V y D resultado de descomponer en valores singulares a la matriz términos por documentos.

```
$d
[1] 10.698554 10.231836  7.716278  3.362964

$u
           [,1]       [,2]       [,3]       [,4]
[1,]  0.0000000 -0.08598908  0.0000000  0.99629608
[2,] -0.7687943  0.00000000  0.6394962  0.00000000
[3,] -0.6394962  0.00000000 -0.7687943  0.00000000
[4,]  0.0000000 -0.99629608  0.0000000 -0.08598908

$v
              [,1]          [,2]          [,3]          [,4]
 [1,] -1.793222e-01 -8.053628e-17 -2.988984e-01 -2.114841e-17
 [2,]  2.855828e-17 -3.894887e-01  4.760160e-17 -1.022777e-01
 [3,] -2.155789e-01  0.000000e+00  2.486288e-01  0.000000e+00
 [4,] -2.753530e-01  0.000000e+00  1.489960e-01  0.000000e+00
 [5,] -1.437193e-01  0.000000e+00  1.657525e-01  0.000000e+00
 [6,] -1.793222e-01  0.000000e+00 -2.988984e-01  0.000000e+00
 [7,]  0.000000e+00 -1.947443e-01  0.000000e+00 -5.113886e-02
 [8,]  0.000000e+00 -2.921165e-01  0.000000e+00 -7.670829e-02
 [9,] -2.141871e-17 -4.952649e-01 -3.570120e-17  1.684082e-01
[10,]  0.000000e+00 -2.921165e-01  0.000000e+00 -7.670829e-02
[11,]  0.000000e+00 -1.947443e-01  0.000000e+00 -5.113886e-02
[12,] -2.141871e-17 -1.057762e-01 -3.570120e-17  2.706859e-01
[13,] -4.283743e-17 -1.680814e-02 -7.140240e-17  5.925106e-01
```

Figura 105. Matrices U, V y D resultado de descomponer en valores singulares a la matriz términos por documentos de la prueba número cinco.
Elaboración: propia.

Se procede a reducir la dimensión del espacio a la matriz U que está formada por cuatro columnas y cuatro filas, con el fin de trabajar solo con las columnas que tiene una alta concentración de valores, para reducir el número de columnas se toma como referencia los valores más altos de la matriz D. La Figura 105 muestra que matriz D posee cuatro columnas y de las cuales las dos primeras poseen valores altos, por ende, a la matriz U se la redice a dos columnas. La Figura 106 presenta la matriz U reducida su dimensión a dos columnas.

```
> matrizu
           [,1]       [,2]
[1,]  0.0000000 -0.08598908
[2,] -0.7687943  0.00000000
[3,] -0.6394962  0.00000000
[4,]  0.0000000 -0.99629608
```

Figura 106. Matriz U reducida su dimensión a dos columnas.
Elaboración: propia.

A partir de la matriz U se procede a graficar los documentos en el plano para identificar a través de los vectores cuales son los documentos con contenidos similares. La Figura 107 muestra los documentos que poseen contenidos similares.

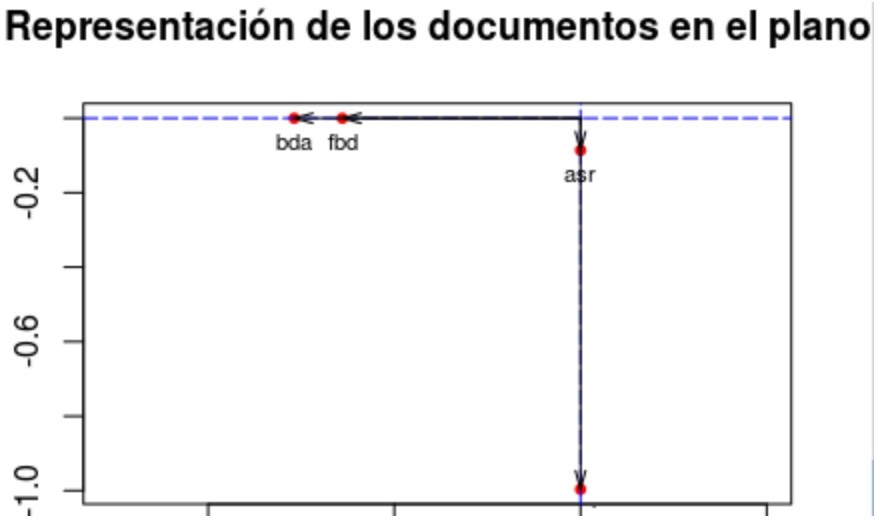


Figura 107. Representación de los cuatro documentos en el plano, los cuales están agrupados de acuerdo a la semejanza en sus contenidos.
Elaboración: propia.

La tabla 14 presenta a detalle los resultados obtenidos en la prueba.

Tabla 14. Detalle de los resultados conseguidos al aplicar LSI en los documentos de la prueba número cinco.

Plan docente	Términos de cada documento	Descripción del vector
Arquitectura y Seguridad de Redes	- direccion ipv - vlans - conexion - internet - enrut dinam - rip	El vector del plan docente está alineado con el vector del plan docente de Redes y Sistemas Distribuidos, es decir, el vector del otro plan docente pasa por el punto que representa al plan docente.
Bases de Datos Avanzadas	- bas dat relacional - segur - oracl - sgbd - web - replic bas dat - bas dat movil - bas dat semant	El vector del plan docente está alineado con el vector del plan docente de Fundamentos de Bases de Datos, es decir, el vector del otro plan docente pasa por el punto que representa a este plan docente.
Fundamentos de Bases de Datos	- entorn bas dat - sgbd - model relacional - algebr relacional - calcul relacional - sql - segur - model entid relacion - normaliz	El vector del plan docente está alineado con el vector del plan docente de Bases de Datos Avanzadas, es decir, el vector pasa por el punto que representa al otro plan docente.
Redes y Sistemas Distribuidos	- algoritm enrut - udp	El vector que simboliza al plan docente está alineado con el

Plan docente	Términos de cada documento	Descripción del vector
	- protocol transaccion dat - tcp - cap red - cap transport - datagram - enrut intern - multiplex - demultiplex	vector del plan docente de Arquitectura y Seguridad de Redes, es decir, el vector pasa por el punto que representa al otro plan docente.

Elaboración: propia.

5.4.5.6. Prueba número seis.

Los planes docentes utilizados en la prueba número seis tanto para el algoritmo de indexación semántica latente (LSI) y k-means corresponden a las áreas de conocimiento de redes, bases de datos e inteligencia artificial.

- Arquitectura y Seguridad de Redes.
- Bases de Datos Avanzadas.
- Fundamentos de Bases de Datos.
- Inteligencia Artificial Avanzada.
- Redes y Sistemas Distribuidos.

Resultado de ejecutar el proceso de limpieza y pre procesamiento de datos se obtiene el vocabulario de términos (Ver Tabla 8) que permite crear la matriz términos por documentos (Ver Anexo 11), que es fundamental para aplicar los pasos del algoritmo LSI. Con la matriz construida se aplica el siguiente paso de LSI que es descomponer en valores singulares a la matriz términos por documentos, obteniendo de resultado tres nuevas matrices: U, V y D. La Figura 108 muestra las tres nuevas matrices U, V y D resultado de aplicar SVD.

```
$d
[1] 10.698554  9.835974  7.716278  6.480741  3.202126

$u
              [,1]          [,2]          [,3]          [,4]          [,5]
[1,]  0.000000e+00 -9.289496e-02  0.000000e+00 -2.345960e-18  9.956759e-01
[2,] -7.687943e-01  0.000000e+00  6.394962e-01  7.585933e-16  0.000000e+00
[3,] -6.394962e-01  0.000000e+00 -7.687943e-01 -1.238593e-15  0.000000e+00
[4,]  2.224477e-16 -2.993818e-18  1.401605e-15 -1.000000e+00 -2.635466e-18
[5,]  0.000000e+00 -9.956759e-01  0.000000e+00  3.225694e-18 -9.289496e-02

$v
              [,1]          [,2]          [,3]          [,4]          [,5]
 [1,]  1.079600e-16 -2.808375e-17  4.964820e-16 -3.086067e-01 -8.048375e-18
 [2,] -1.793222e-01  5.702566e-17 -2.988984e-01 -4.347518e-16  1.634268e-17
 [3,]  9.352780e-17 -4.049120e-01 -9.352780e-17  1.990942e-18 -1.160416e-01
 [4,]  2.159200e-16  2.589224e-18  9.929641e-16 -6.172134e-01  7.420320e-19
 [5,]  1.619400e-16 -5.594736e-19  7.447230e-16 -4.629100e-01 -1.603366e-19
 [6,] -2.155789e-01 -2.218625e-17  2.486288e-01  3.146283e-16 -6.358242e-18
 [7,] -2.753530e-01  4.495254e-18  1.489960e-01  1.697110e-16  1.288271e-18
 [8,] -1.437193e-01 -8.120459e-18  1.657525e-01  2.097522e-16 -2.327200e-18
 [9,] -1.793222e-01  5.336301e-17 -2.988984e-01 -4.347518e-16  1.529303e-17
[10,]  4.676390e-17 -2.024560e-01 -4.676390e-17  9.954709e-19 -5.802080e-02
```

Figura 108. Matrices U, V y D resultado de descomponer en valores singulares a la matriz términos por documentos de la prueba número seis.
Elaboración: propia.

Se procede a reducir la dimensión del espacio a la matriz U que representa a los documentos de la prueba, con el objetivo de trabajar solo con las columnas que poseen una alta concentración de valores, para la reducción se toma como referencia los valores más altos de la matriz D. Se observa en la Figura 108 que la matriz D tiene cinco columnas de las cuales las tres primeras poseen valores elevados, por lo que se acorta a tres filas a la matriz U. La Figura 109 presenta la matriz U una vez reducida su extensión a tres.

```
> matrizu
              [,1]          [,2]          [,3]
[1,]  0.000000e+00 -9.289496e-02  0.000000e+00
[2,] -7.687943e-01  0.000000e+00  6.394962e-01
[3,] -6.394962e-01  0.000000e+00 -7.687943e-01
[4,]  2.224477e-16 -2.993818e-18  1.401605e-15
[5,]  0.000000e+00 -9.956759e-01  0.000000e+00
```

Figura 109. Matriz U una vez reducida su superficie a tres columnas.
Elaboración: propia.

A partir de los valores de la matriz U se grafica los documentos en el plano para identificar cuales contienen contenidos similares. La Figura 110 muestra el plot con los documentos que poseen contenidos semejantes.

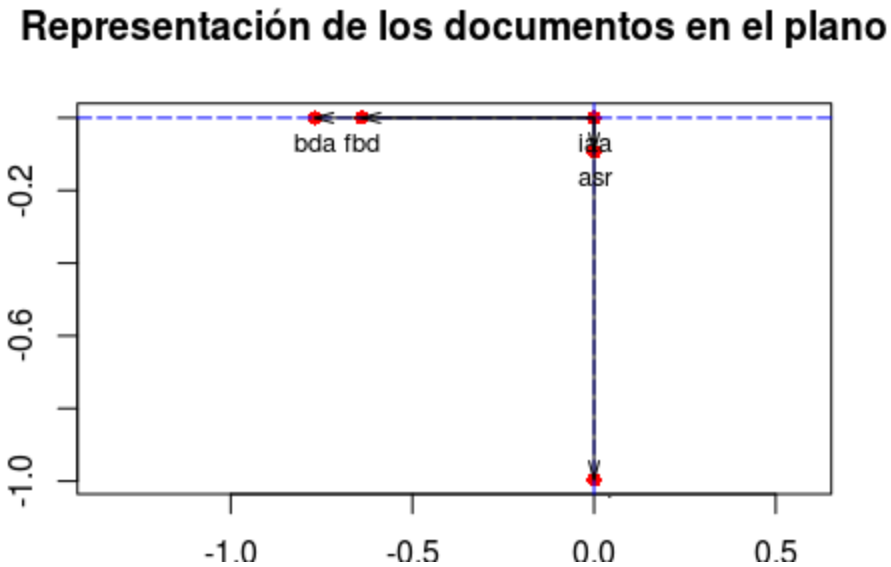


Figura 110. Representación de los cinco planes docentes en el plano, en donde se observa cuáles son los que poseen contenidos similares.
Elaboración: propia.

La Tabla 15 detalla los resultados derivados de aplicar el algoritmo LSI en los documentos de la prueba.

Tabla 15. Detalle de los resultados conseguidos al aplicar LSI en los documentos de la prueba número seis.

Plan docente	Términos de cada documento	Descripción del vector
Arquitectura y Seguridad de Redes	- conexión - internet	El vector que representa al plan docente está alineado con el vector

Plan docente	Términos de cada documento	Descripción del vector
	- direccion ipv - enrut dinam - vlans	del plan docente de Redes y Sistemas Distribuidos, ubicándose muy cerca del otro vector.
Bases de Datos Avanzadas	- sgbd - bas dat relacional - replic bas dat - bas dat movil - bas dat semant - oracl	El vector del plan docente está alineado con el vector del plan docente de Fundamentos de Bases de Datos, es decir, se encuentra muy cerca al vector del otro plan docente.
Fundamentos de Bases de Datos	- sgbd - entorn bas dat - model relacional - algebr relacional - calcul relacional - model entid relacion - normaliz - segur	El vector del plan docente está alineado con el vector del plan docente de Bases de Datos Avanzadas, es decir, se encuentra muy cerca al vector del otro plan docente.
Inteligencia Artificial Avanzada	- arbol clasif - regl neuronal - ajust pes - analisis clust - model ocult markov	El vector del plan docente no se alinea con ningún vector de otro plan docente, debido a que está ubicado muy cerca a la posición (0,0) del plano.
Redes y Sistemas Distribuidos	- algoritm enrut - udp - protocol transaccion dat - tcp - cap red - cap transport - datagram - red - enrut intern	El vector del plan docente está alineado con el vector del plan docente de Arquitectura y Seguridad de Redes, es decir, el vector del otro plan docente pasa cerca del punto que representa a este plan docente.

Elaboración: propia.

5.4.6. Diseño y desarrollo del prototipo para visualización de los resultados

Se diseña y desarrolla una aplicación Web para presentar los resultados obtenidos de aplicar las técnicas de Minería de Texto en los planes docentes de la titulación de Ingeniería de Sistemas Informáticos y Computación.

Como el proceso de Minería de Texto se desarrolla en R Project, se busca y analiza un framework que se adapte al proceso desarrollado. Una vez realizada la búsqueda se decide utilizar el framework de aplicaciones Web Shiny[30] elaborado por RStudio. Para empezar a desarrollar el prototipo en RStudio se descarga e instala el paquete shiny. El siguiente código instala la librería en RStudio.

```
install.packages("shiny") #Instalar shiny
```

[30] Shiny: http://shiny.rstudio.com/

5.4.6.1. Estructura de la aplicación Shiny

La aplicación shiny tiene dos componentes los cuales se construyen automáticamente al momento de crear una aplicación Web Shiny. Los componentes son los siguientes:

- ❖ Un script de interfaz de usuario (user-interface script).
- ❖ Un script servidor (server script).

La interfaz de usuario (ui): controla el diseño y el aspecto de la aplicación. Se define en un script de origen denominado ui.R.
El script server.R: contiene las instrucciones que el ordenador necesita para construir la aplicación. La Figura 111 presenta los dos componentes en la herramienta RStudio.

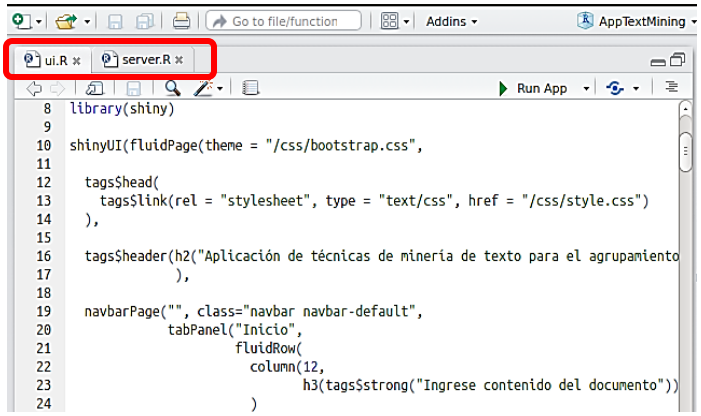


Figura 111. Script ui.R y Script server.R componentes de una aplicación Web Shiny.
Elaboración: propia.

4.4.6.2. Estilos en Shiny.

Para que la interfaz de usuario de la aplicación Web tenga una apariencia agradable y amigable shiny permite la vinculación de hojas de estilo (CSS) en el script ui.R. Para nuestro prototipo se utiliza la hoja de estilo (bootstrap.css) de un tema de la página Bootswatch[31], que contiene un sinnúmero de temas gratis de Bootstrap.

Para hacer uso de los temas disponibles en Bootswatch se descarga el archivo que contiene los estilos del tema seleccionado, y se coloca en el directorio del proyecto para llamarlo desde el archivo ui.R. La Figura 112 presenta el diseño del tema seleccionado para nuestro prototipo.

[31] Bootswatch: http://bootswatch.com/

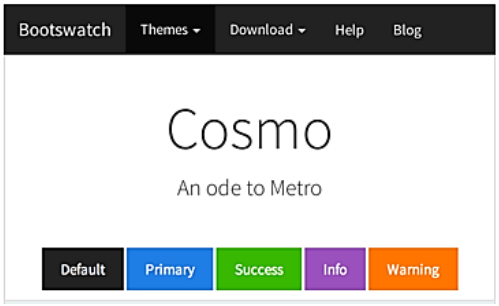


Figura 112. Tema a utilizar en el prototipo.
Elaboración: propia.
Fuente: http://bootswatch.com/

4.4.6.3. Desarrollo de la aplicación

Instalado el paquete shiny y conociendo la estructura que presenta, se procede a crear nuestra aplicación Web, que está formada por cinco secciones para facilitar la presentación de los contenidos. Las secciones son las siguientes:

- *Sección 1:* ingresar el contenido de un nuevo plan docente.
- *Sección 2:* presentar la matriz términos por documentos.
- *Sección 3:* frecuencia de los términos de la matriz términos por documentos.
- *Sección 4:* resultados del algoritmo k-means.
- *Sección 5: r*esultados del algoritmo de indexación semántica latente.

Sección 1. Ingresar contenido de un nuevo plan docente.

La sección permite al usuario ingresar texto (contenido de un plan docente) a la aplicación Web, con el propósito de almacenar el nuevo documento en el corpus y aplicar las técnicas de Minería de Texto, para analizar los resultados obtenidos al procesar el nuevo plan docente. Para facilitar incorporar el contenido de un plan docente en la aplicación Web se crea un *widget para ingresar texto* en el script ui.R, mediante la función *textInput* del paquete shiny. El widget admite que se copie el contenido de un plan docente y se pegue en el mismo para ser almacenado en el corpus y procesado. Además, se crea un botón que actualiza la matriz términos por documentos una vez que se ha ingresado el contenido del plan docente. El código para construir el widget es el siguiente:

```
textInput (inputId ="texto", label = "Pegue el contenido")
actionButton (inputId = "procesamiento", label = "Procesar", class="btn btn-
primary")
```

En el script server.R mediante la función renderPrint se captura el texto ingresado en el widget y se lo guarda en el corpus en un documento de texto/plano. A continuación, se presenta el código:

```
output$savecontenido = renderPrint ({·····})
```

Una vez añadidos los estilos a la aplicación Web, la sección de *Ingresar contenido de un nuevo plan docente* se visualiza en la Figura 113.

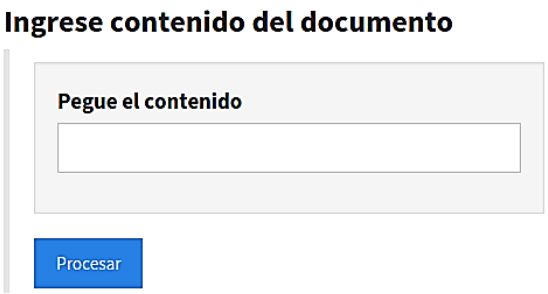


Figura 113. Sección de Ingresar contenido de un nuevo plan docente. Elaboración: propia.

Sección 2. Presentar la matriz términos por documentos.

La sección presenta la matriz términos por documentos generada a partir del vocabulario de términos obtenido de los planes docentes que conforman el corpus. En el script ui.R que es donde se controla el diseño de la aplicación se presenta la matriz términos por documentos mediante función *tableOutput.* El código es el siguiente:

```
tags$div (class = "tmtd", tableOutput ("mtd"))
```

En el script server.R en la función *renderTable* se genera la matriz términos por documentos al aplicar el proceso de limpieza y pre procesamiento de datos. El código empleado para generar la matriz es el siguiente:

```
output$mtd = renderTable ({·····})
```

La Figura 114 presenta la interfaz gráfica de la sección que muestra la MTD.

Matriz términos por documentos

	accesibil	aceler	ajust pes	algebr boolean	algebr relacional	algoritm	algoritm enrut	algoritm orden	algoritm orient dat
asr	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
bda	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
fbd	0.00	0.00	0.00	0.00	3.00	0.00	0.00	0.00	0.00
fpr	0.00	0.00	0.00	0.00	0.00	4.00	0.00	0.00	0.00
palg	0.00	0.00	0.00	0.00	0.00	7.00	0.00	3.00	0.00
pav	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
plan.txt	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
rsd	0.00	0.00	0.00	0.00	0.00	4.00	4.00	0.00	0.00

Figura 114. Sección para presentar la Matriz Términos por Documentos. Elaboración: propia.

Sección 3. Presentar la frecuencia de los términos

La sección presenta el número de veces que aparece una palabra, bigrama o trigrama en cada uno de los planes docentes, permitiendo observar cual es el término que mayor número de veces aparece y cuál es el que menor número de veces aparece.

En el script ui.R mediante la función *plotOutput* se presenta el histograma con la frecuencia de los términos generados en la función *output$frecuenciawords* del script server.R. Adicionalmente se crea un botón que permite actualizar el histograma con la frecuencia de los términos una vez que se ha ingresado el contenido de un nuevo plan docente. El código para construir el histograma se presenta a continuación:

```
plotOutput ("frecuenciawords")
actionButton (inputId = "frecuenciapalabras", label = "Frecuencia de palabras", class= "btn btn-primary")
```

En el script server.R en la función *renderPlot* se genera el histograma con el número de veces que aparecen los términos en los planes docentes. El código para generar la frecuencia de los términos es el siguiente:

```
output$frecuenciawords = renderPlot ({…})
```

La Figura 115 muestra el histograma con la frecuencia de los términos.

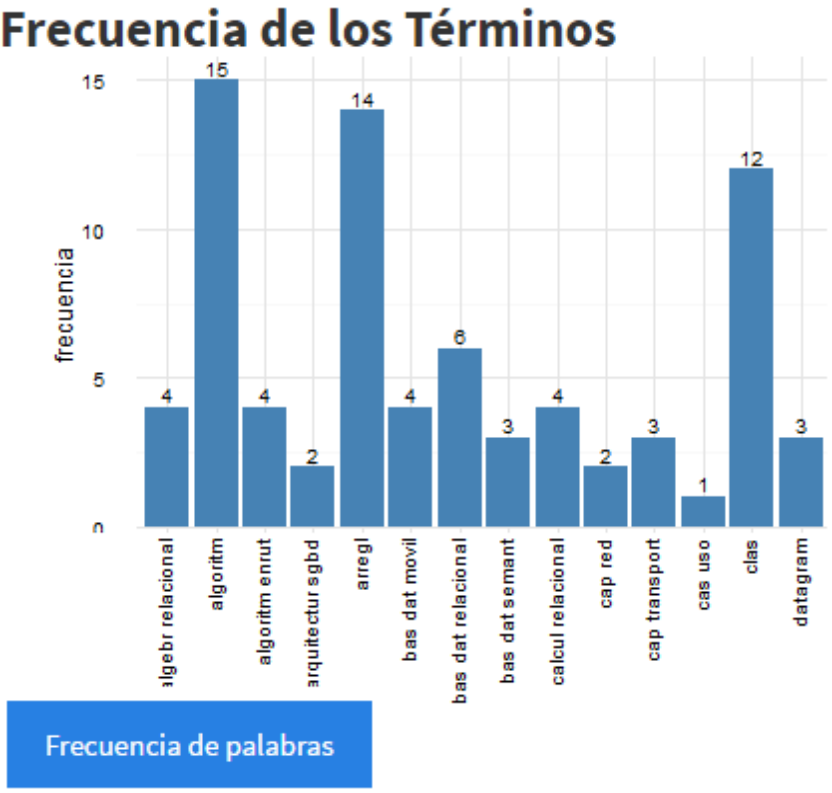


Figura 115. Sección para presentar la frecuencia de los términos. Elaboración: propia.

Sección 4: Resultados del algoritmo k-means.

La sección muestra gráficamente el número de clusters obtenidos al aplicar el algoritmo k-means. Adicional se presenta un widget que admite elegir el número de grupos en los que se quiere agrupar los contenidos, para actualizar la gráfica del algoritmo k-means se crea un botón.

En el script ui.R se crea un widget para introducir números mediante la función *numericInput* y poder elegir el número de grupos requeridos para el agrupamiento. Además, se crea un botón mediante la función *actionButton* para actualizar el número de grupos en la gráfica del algoritmo, y mediante la función *ploOutput* se presenta y visualiza la gráfica con los resultados del agrupamiento. El código para crear la interfaz de la sección es el siguiente:

```
numericInput (inputId = "numclusters", label = "Número de grupos", value =
3, min = 1, max = 10, width = "50%")
actionButton (inputId = "Kmeans", label = "Algoritmo K-means", class= "btn
btn-primary")
plotOutput ("plotkmeans")
```

En el script server.R en la función *renderPlot* se aplica el proceso del algoritmo k-means y se genera la gráfica con los resultados obtenidos del agrupamiento. El código empleado para llevar a cabo el agrupamiento es el siguiente:

```
output$plotkmeans = renderPlot ({……})
```

La Figura 116 muestra la interfaz gráfica de la sección del algoritmo k-means.

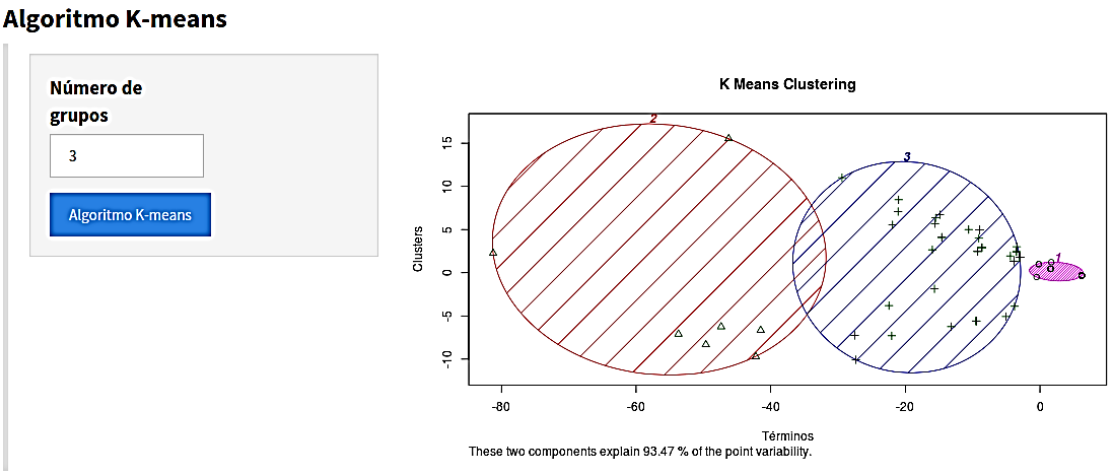


Figura 116. Sección para visualizar los resultados del algoritmo k-means.
Elaboración: propia.

Sección 5. Resultados del algoritmo LSI.

La sección muestra los resultados obtenidos al aplicar el algoritmo LSI en la matriz términos por documentos, para determinar los planes docentes con contenidos semejantes. En el script ui.R con la función *plotOutput* se muestra la gráfica con los resultados obtenidos al aplicar la indexación semántica latente al conjunto de planes docentes. Además, se crea un botón que permite actualizar los resultados del algoritmo. El código en el script ui.R es el siguiente:

```
plotOutput ("plotlsi", height = "400px", width = "100%")
actionButton (inputId = "btnlsi", label = "Indexación semántica latente",
class="btn btn-primary")
```

En el script server.R con la función *renderPlot* se aplica el proceso de la indexación semántica latente al corpus de datos y se genera el gráfico con los resultados obtenidos. El código en el script server.R que permite desarrollar el algoritmo LSI es el siguiente:

```
output$plotsvd = renderPlot ({
input$btnsvd
    svdmatriz = isolate ({…})
})
```

La Figura 117 presenta la gráfica con los vectores que simbolizan a los planes docentes.

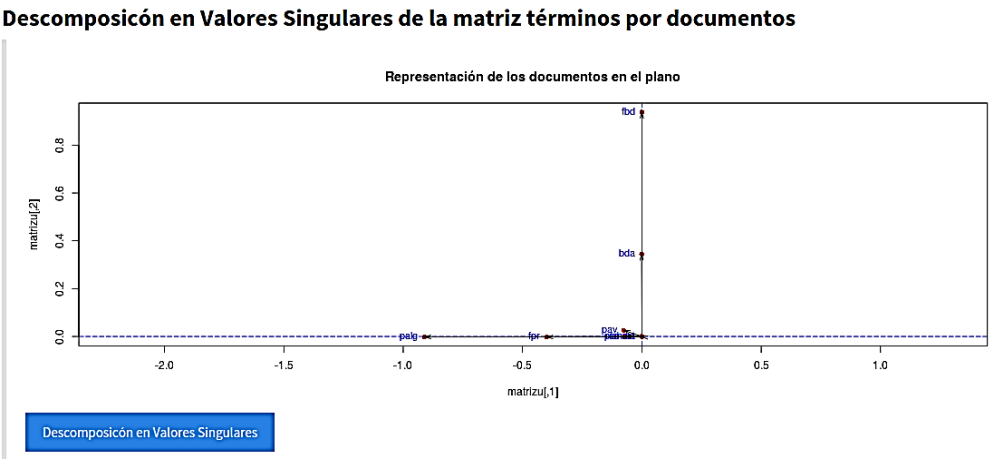


Figura 117. Sección para visualizar los resultados de la indexación semántica latente. Elaboración: propia.

CAPÍTULO 6: ANÁLISIS DE RESULTADOS

El presente capítulo muestra el análisis de los resultados obtenidos al aplicar el algoritmo k-means y el algoritmo de indexación semántica latente (LSI) en la herramienta R, sobre los contenidos de los planes docentes de los componentes académicos de la titulación de Ingeniería en Sistemas Informáticos y Computación de la UTPL.

Los resultados obtenidos en cada una de las seis pruebas ejecutadas con el algoritmo k-means son óptimos, debido a que la mayor parte de los grupos constituidos en cada una de las pruebas, están formados por contenidos que representan a determinadas áreas de conocimiento. Los resultados dependen del valor que se le asigne a k, por lo cual se debe encontrar el valor óptimo para el número de clústeres.

En lo que concierne a los resultados conseguidos por el algoritmo de indexación semántica latente (LSI) se considera que son eficientes, debido a que los vectores que representan a cada uno de los planes docentes de acuerdo a sus contenidos se hallan alineados o muy cerca entre ellos, y los vectores de los planes docentes que no tiene contenidos iguales se encuentran distantes entre sí, pero surge el inconveniente de que, si existe un solo plan docente sobre una determinada área de conocimiento, este se ubica en la posición (0,0) del plano y no se alinea con ningún vector de otro plan docente.

Cabe destacar que los algoritmos no son 100 % exactos, debido a que poseen un mínimo porcentaje de error y esto se observa en los resultados obtenidos al aplicar los algoritmos.

Los planes docentes que se utilizan en cada una de las pruebas son los mismos tanto para el algoritmo k-means como para la indexación semántica latente. A continuación, se analiza a detalle cada una de las pruebas realizadas con los algoritmos.

6.1. Análisis de los resultados de la prueba número uno

Algoritmo K-means: en la prueba k = 3 y se puede recalcar lo siguiente de cada uno de los tres grupos:

- *Cluster 1:* está formado solo por contenidos de planes docentes que contienen temas sobre programación, por esta razón se considera que esta agrupado de manera correcta.
- *Clúster 2:* está construido por contenidos de planes docentes que contienen temas sobre bases de datos, redes y programación, en el cual la mayor parte de términos pertenecen al área de bases de datos. Los términos que corresponden al área de

programación son: *cas uso, diagram clas, do while, expresión regular, herenci* y *uml*. Los contenidos del área de redes son: *enrut dinam, vlans.*

- *Clúster 3:* está formado solo por contenidos de planes docentes que contienen temas sobre redes, y es por esta razón que se considera que esta construido de manera correcta.

Una vez que se analizado a cada uno de los grupos formados por el algoritmo k-means se considera que son óptimos, debido a que están formados por términos que representan a determinadas áreas de conocimiento.

LSI: se recalca lo siguiente de cada uno de los vectores que representan a los planes docentes:

- Los vectores de los planes docentes de *Fundamentos de Bases de Datos* y *Bases de Datos Avanzadas* se encuentran alineados en el plano, indicando que este grupo de planes docentes presentan contenidos similares.
- Los vectores de los planes docentes de *Redes y Sistemas Distribuidos* y *Arquitectura y Seguridad de Redes* se ubican muy cerca entre sí en el plano, permitiendo identificar fácilmente este grupo de planes docentes con contenidos similares.
- Los vectores de los planes docentes de *Fundamentos de Programación* y *Programación de Algoritmos* se ubican muy cerca en el plano, revelando que presentan contenidos similares.
- El vector del plan docente de Programación Avanzada no está alineado con ningún otro vector, pero se encuentra muy cercar del vector del plan docente de *Programación de Algoritmos* lo que nos indica que contienen una baja cantidad de contenidos similares.

Una vez que se analizado a cada uno de los vectores que simbolizan a los planes docentes en el plano, se puede decir que los resultados generados por la indexación semántica latente en la prueba son correctos, debido a que los documentos con contenidos similares se encuentran agrupados en un mismo lugar, y por ende los que contienen contenidos diferentes se encentran distantes entre sí.

La Tabla 16 presenta los resultados a detalle de la prueba número uno tanto del algoritmo k-means como LSI.

Tabla 16. Descripción a detalle de los resultados obtenidos en la prueba uno con los algoritmos k-means y LSI

N° prueba	Planes docentes	K-means N° clúster asignado	LSI
Prueba 1	Arquitectura y Seguridad de Redes	3 y 2	Los vectores de los planes docentes que contienen temas sobre redes, se encuentran muy cerca en el plano lo que indica que presentan contenidos similares.
	Redes y Sistemas Distribuidos	3	
	Bases de Datos Avanzadas	2	Los vectores de los dos documentos que contienen temas relacionados a bases de datos, están ubicados muy cerca en el plano, indicando que existe una semejanza en sus contenidos.
	Fundamentos de Bases de Datos	2	
	Fundamentos de Programación	1 y 2	Los vectores de los planes docentes están muy cerca en el plano, e indican que se asemeja su contenido.
	Programación de Algoritmos	1 y 2	
	Programación Avanzada	1	El vector del plan docente posee una extensión muy corta y no se alinea con ningún vector de otro plan docente, pero está cerca del vector del plan docente de Programación de Algoritmos.

Elaboración: propia.

6.2. Análisis de los resultados de la prueba número dos

Algoritmo K-means: en la prueba k = 3 y se recalca lo siguiente de cada uno de los tres grupos:

- *Clúster 1:* está formado solo por los contenidos de planes docentes que contienen temas sobre Física, y es por esta razón que se considera que esta agrupado de manera correcta.
- *Clúster 2:* está formado solo por contenidos de los planes docentes que contienen temas sobre Programación, y es por esto que se considera que está constituido de manera correcta.
- *Clúster 3:* está formado en su mayor parte por contenidos de planes docentes que contienen temas sobre bases de datos y por una mínima cantidad (cinco términos) de contenidos que representan a planes docentes de programación. Los dos términos que pertenecen a los documentos de programación son: *cas uso, herencia, uml, subproces* y *diagram clas*, los cuales deben estar agrupados en el clúster número dos que contiene los contenidos de programación.

Una vez realizado el análisis a cada uno de los grupos formados por el algoritmo k-means se considera que son óptimos, debido a que solo uno de los grupos posee una mínima cantidad de términos de otra área de conocimiento.

LSI: se describe lo siguiente de cada uno de los vectores que representan a los planes docentes en la prueba:

- Los vectores de los planes docentes de *Fundamentos de Bases de Datos* y *Bases de Datos Avanzadas* se ubican muy cerca entre sí en el plano, lo que permite identificar fácilmente este grupo de planes docentes con contenidos semejantes.
- Los vectores de los planes docentes de *Fundamentos de Programación* y *Programación de Algoritmos* están cerca entre sí en el plano, indicando que poseen contenidos similares.
- El vector del plan docente de *Física* está ubicado en la posición (0,0) del plano y por ende no está alineado con ningún otro plan docente. Esto se debe a que no existen otros planes docentes con contenidos relacionados al área de conocimiento de física, para realizar la comparación y poder agrupar dichos documentos.
- El vector del plan docente de *Programación Avanzada* posee un vector con una dimensión corta y no está alineado ni agrupado con ningún otro plan docente, pero está próximo a los vectores de los planes docentes de *Fundamentos de Programación* y *Programación de Algoritmos*.

Una vez que se analizado a cada uno de los vectores que simbolizan a los planes docentes en el plano, se concluye que los resultados son eficientes debido a que los vectores con contenidos similares se encuentran agrupados en un mismo lugar.

La Tabla 17 describe a detalle los resultados conseguidos por los dos algoritmos en la prueba.

Tabla 17. Detalle de los resultados obtenidos en la prueba dos con los algoritmos k-means y LSI.

N° prueba	Planes docentes	K-means N° clúster asignado	LSI
Prueba 2	Bases de Datos Avanzadas.	3	Los vectores de los dos planes docentes se encuentran alineados en el plano, indicando que hay una semejanza en sus contenidos.
	Fundamentos de Bases de Datos.	3	
	Física.	1	El vector del plan docente está ubicado en la posición (0,0) del plano, y por ende no está alineado con ningún otro plan docente. Esto se debe a que existe un solo

N° prueba	Planes docentes	K-means N° clúster asignado	LSI
			plan docente con contenidos relacionados a la física.
	Fundamentos de Programación.	2 y 3	Los vectores de los documentos se encuentran alineados en el plano pasando el uno por el otro, indicando que hay una semejanza en sus contenidos.
	Programación de Algoritmos.	2 y 3	
	Programación Avanzada.	2 y 3	El vector del plan docente posee una extensión muy corta y no se encuentra alineado con ningún vector de otro plan docente.

Elaboración: propia.

6.3. Análisis de los resultados de la prueba número tres

Algoritmo K-means: en la prueba k = 6 y se resalta lo siguiente de cada uno de los seis grupos:

- *Clúster 1:* está formado solo por los contenidos de los planes docentes que contienen temas solo sobre programación, y es por esta razón que se considera que está formado de manera correcta.
- *Clúster 2:* formado solo por los contenidos del plan docente de inteligencia artificial y se considera que está formado de manera correcta.
- *Clúster 3:* está conformado solo por los contenidos de los planes docentes que contienen temas sobre organización y gestión empresarial, y por esta razón se considera que el clúster está formando de manera correcta.
- *Clúster 4:* está conformado solo por los contenidos de los planes docentes que contienen temas sobre redes, y es por esto que se considera que está establecido de manera óptima.
- *Clúster 5:* está construido solo por los contenidos de los planes docentes que contienen temas sobre desarrollo web, y por esta razón se considera que está organizado de manera correcta.
- *Clúster 6:* está formado en su mayor parte por contenidos de planes docentes que contienen temas sobre redes, y por una mínima cantidad (dos términos) de contenidos que representan a planes docentes de programación. Los dos términos que pertenecen a los planes docentes de programación son: *ingeni requer* y *prueb desarroll*, los cuales deben estar agrupados en el clúster número uno que contiene los contenidos de esta área de conocimiento.

Una vez analizados los seis grupos formados por el algoritmo se concluye que su estructura es óptima, debido a que todos los grupos están formados por términos que representan a determinadas áreas de conocimiento.

LSI: se resalta lo siguiente de cada uno de los vectores que representan a los planes docentes de la prueba:

- Los vectores de los planes docentes de *Redes y Sistemas Distribuidos* y *Arquitectura y Seguridad de Redes* se ubican muy cerca entre sí, y además se dirigen hacia la misma dirección en el plano, permitiendo deducir que los planes docentes poseen contenidos similares.
- Los vectores de los planes docentes de *Organización y Administración Empresarial* y de *Fundamentos de Ingeniería de Software* se ubican muy cerca entre sí en el plano, permitiendo identificar fácilmente a este grupo de documentos con contenidos semejantes.
- Los vectores de los planes docentes de *Ingeniería de Requisitos y de Ingeniería Web* poseen contenidos semejantes, debido a que se ubican muy cerca entre sí en el plano.
- El vector del plan docente de *Inteligencia Artificial* está ubicado en la posición (0,0) del plano y por ende no está alineado con ningún otro plan docente. Esto se debe a que no existen otros planes docentes con contenidos relacionados a esta área de conocimiento, para realizar la comparación y poder agrupar los planes docentes.

Una vez analizados los vectores se concluye que los resultados son óptimos, porque los vectores con contenidos similares están agrupados entre sí. La Tabla 18 presenta a detalle los resultados generados por los dos algoritmos.

Tabla 18. Análisis de los resultados obtenidos en la prueba tres con los algoritmos k-means y LSI.

N° prueba	Planes docentes	K-means N° clúster asignado	LSI
Prueba 3	Arquitectura y Seguridad de Redes	4 y 6	Los vectores de los dos planes docentes están ubicados muy cerca en el plano, por ende, sus contenidos se asemejan.
	Redes y Sistemas Distribuidos	4 y 6	
	Organización y Administración Empresarial	3 y 6	Los vectores de los documentos están ubicados muy cerca de la posición (0,0) en el plano. Además, están muy cerca entre sí indicando que contienen contenidos similares.
	Fundamentos de Ingeniería de Software	1	
	Ingeniería de Requisitos	1 y 6	Los vectores de los dos planes docentes se encuentran

N° prueba	Planes docentes	K-means N° clúster asignado	LSI
	Ingeniería Web	5	alineados, es decir, contienen contenidos similares.
	Inteligencia Artificial Avanzada	2	El vector del documento está ubicado en la posición (0,0) del plano, y por ende no está alineado con ningún otro plan docente. Esto se debe a que existe un solo plan docente con contenidos relacionados a inteligencia artificial.

Elaboración: propia.

6.4. Análisis de los resultados de la prueba número cuatro

Algoritmo K-means: en la prueba k = 4 y se recalca lo siguiente de cada uno de los cuatro grupos:

- *Clúster 1:* está formado solo por los contenidos de los planes docentes que contienen temas sobre programación, y es por esta razón que se considera que está formado de manera correcta.
- *Clúster 2:* está formado por los contenidos de los planes docentes de *programación*, *teoría de autómatas* y de *gestión de proyectos*, la combinación de todos estos términos en el mismo grupo nos indica que los planes docentes presentan semejanza en sus contenidos. Los términos que pertenecen a los planes docentes de programación son: *diagram clas* y *do whil*. El término que pertenece al plan docente de teoría de autómatas y compiladores es: *teor automat*, y los términos que perteneces a los planes docentes de gestión de proyectos son: *calendariz proyect, cierr proyect, control proyect, gestión calid, gestión riesg, prueb desarroll, y recurs human*.
- *Clúster 3:* está formado solo por los contenidos de los planes docentes que contienen temas sobre autómatas y compiladores, por lo que se considera que está formado de manera correcta.
- *Clúster 4:* está formado en su mayor parte por los contenidos de planes docentes que contienen temas sobre programación, y se considera que esta formado de manera correcta.

Los resultados generados por el algoritmo k-means son eficientes, debido a que cada uno de los grupos está estructurado solo por contenidos de determinadas áreas de conocimiento.

LSI: con los resultados obtenidos se puede recalcar lo siguiente de cada uno de los vectores:

- El vector del plan docente de *Teoría de Autómatas y Compiladores* no está alineado ni cerca de ningún vector de otro plan docente, esto indica que posee contenidos distintos a los de los otros documentos.
- El vector del plan docente del componente académico de *Programación Avanzada* posee un vector con una dimensión corta, y no está alineado ni agrupado con ningún otro plan docente, indicando que posee contenidos diferentes.
- Los vectores de los planes docentes de los componentes de *Gestión de Proyectos* y *Procesos de Ingeniería de Software* se ubican muy cerca entre sí, indicando que poseen contenidos semejantes.
- Los vectores de los planes docentes de *Fundamentos de Ingeniería de Software*, *Fundamentos de Programación*, *Lógica de la Programación* y *Programación de Algoritmos* poseen contenidos semejantes porque se ubican muy cerca entre sí en el plano.

Los resultados generados por el algoritmo LSI son eficientes debido a que los vectores con contenidos similares se encuentran agrupados en un mismo lugar y distantes de los vectores con contenidos diferentes. La Tabla 19 presenta a detalle los resultados obtenidos.

Tabla 19. Detalle de los resultados obtenidos en la prueba cuatro con los algoritmos k-means y LSI.

N° prueba	Planes docentes	K-means N° clúster asignado	LSI
Prueba 4	Teoría de Autómatas y Compiladores.	2 y 3	El vector del plan docente no está alineado a ningún vector de otro plan docente. Esto indica que no existe otro plan docente con contenidos iguales.
	Fundamentos de Ingeniería de Software.	1 y 4	Los vectores de los cuatro planes docentes se encuentran alineados en el plano, indicando que contienen contenidos similares.
	Fundamentos de Programación.	1 y 4	
	Lógica de la Programación.	1 y 4	
	Programación de Algoritmos.	1 y 4	
	Gestión de Proyectos.	2	Los vectores de los dos documentos están ubicados muy cerca en el plano, indicando que poseen contenidos iguales.
	Procesos de Ingeniería de Software.	2	
	Programación Avanzada.	1 y 4	El vector del plan docente posee una extensión muy corta y no se encuentra cerca ni alineado a ningún vector de documento.

Elaboración: propia.

6.5. Análisis de los resultados de la prueba número cinco

Algoritmo K-means: en la prueba k = 2 y se resalta lo siguiente de cada uno de los dos grupos:

- *Clúster 1:* está constituido en su mayor parte por los contenidos de los planes docentes que contienen temas sobre redes, y por tres términos que pertenecen a los contenidos de los documentos de bases de datos, los mismos que debes estar agrupados en el clúster número dos que contiene los contenidos sobre ese tema.
- *Clúster 2:* está formado solo por los contenidos de los planes docentes que contienen temas sobre Bases de datos, y es por esta razón que se considera que está formado de manera correcta.

Una vez que se analizado a cada uno de los dos grupos se considera que son óptimos, porque los términos que forman a cada uno de los grupos pertenecen a determinadas áreas de conocimiento.

LSI: con los resultados obtenidos se puede recalcar lo siguiente de cada uno de los vectores:

- Los vectores de los planes docentes de los componentes académicos de *Arquitectura y Seguridad de Redes* y de *Redes y Sistemas Distribuidos* se encuentra alineados en el plano, permitiendo identificar fácilmente este grupo de planes docentes con contenidos similares.
- Los vectores de los planes docentes de *Fundamentos de Base de Datos* y *Bases de Datos Avanzadas* contienen contenidos semejantes porque se encuentran alineados en el plano.

Una vez analizados los vectores de la prueba se concluye que los resultados son eficientes, debido que los planes docentes con contenidos similares se encuentran agrupados en un mismo lugar, y los que contienen contenidos diferentes se encentran distantes entre sí. La Tabla 20 muestra detalles de los resultados obtenidos en la prueba al aplicar los algoritmos.

Tabla 20. Detalles de los resultados obtenidos en la prueba cinco con los algoritmos k-means y LSI.

N° prueba	Planes docentes	K-means N° cluster asignado	LSI
Prueba 5	Arquitectura y Seguridad de Redes	1 y 2	Los vectores de los dos documentos se encuentran alineados en el plano,
	Redes y Sistemas Distribuidos	1	

N° prueba	Planes docentes	K-means N° cluster asignado	LSI
			indicando que poseen contenidos similares.
	Fundamentos de Base de Datos	2	Los vectores de los dos planes docentes se encuentran alineados en el plano, por lo que sus contenidos son similares.
	Bases de Datos Avanzadas	2	

Elaboración: propia.

6.6. Análisis de los resultados de la prueba número seis

Algoritmo K-means: en la prueba k = 3 y se recalca lo siguiente de cada uno de los cuatro grupos:

- *Clúster 1:* está formado solo por los contenidos de los planes docentes que contienen temas sobre Redes, y por esta razón se considera que está formado de manera correcta.
- *Clúster 2:* está formado solo por los contenidos de los planes docentes que contienen temas sobre inteligencia artificial, y por esta razón se considera que está formado de manera correcta.
- *Clúster 3:* está constituido en su mayor parte por los contenidos de los planes docentes que contienen temas sobre bases de datos, y por dos contenidos que pertenecen a los documentos de redes.

Una vez que se analizado a cada uno de los cuatro grupos se considera que son óptimos, debido a que están formados por términos que representan a determinadas áreas de conocimiento.

LSI: se recalca lo siguiente de cada uno de los vectores que representan a los planes docentes en la prueba:

- Los vectores de los planes docentes de los componentes académicos de *Arquitectura y Seguridad de Redes* y de *Redes y Sistemas Distribuidos* se encuentran alineados en el plano, demostrando que poseen contenidos similares.
- Los vectores de los planes docentes de los componentes académicos de *Fundamentos de Base de Datos* y *Bases de Datos Avanzadas* están alineados en el plano, permitiendo identificar fácilmente que poseen contenidos similares.
- El vector del plan docente del componente académico de *Inteligencia Artificial* está ubicado en la posición (0,0) del plano y por ende no está alineado con ningún otro

documento. Esto se debe a que no existen otros documentos con contenidos relacionados a este tema para realizar la comparación y así agrupar los planes docentes.

Una vez analizados los vectores se concluye que los resultados generados por el algoritmo LSI son correctos, debido a que los documentos con contenidos similares se encuentran agrupados en un mismo lugar distantes de los que contienen contenidos diferentes. La Tabla 21 presenta los resultados generados por los dos algoritmos.

Tabla 21. Detalles de los resultados obtenidos en la prueba seis con los algoritmos k-means y LSI.

N° prueba	Planes docentes	K-means N° clúster asignado	LSI
Prueba 6	Arquitectura y Seguridad de Redes	1 y 3	Los vectores de los dos planes docentes se encuentran alineados en el plano, revelando que poseen contenidos similares.
	Redes y Sistemas Distribuidos	1 y 3	
	Bases de Datos Avanzadas	3	Los vectores de los dos documentos se encuentran alineados en el plano, mostrando que presentan contenidos similares.
	Fundamentos de Bases de Datos	3	
	Inteligencia Artificial Avanzada	2	El vector del plan docente no esta alineado a ningún vector de otro documento, esto indica que no existe otro plan docente que tenga contenidos semejantes.

Elaboración: propia.

CONCLUSIONES

Una vez completadas las fases del trabajo de titulación y basados en los objetivos planteados para el proyecto, se presenta las conclusiones en base a las contribuciones alcanzadas:

- La aplicación de técnicas de Minería de Texto al contenido de los planes docentes de los componentes académicos de la titulación de Ingeniería en Sistemas Informáticos y Computación, permite identificar cuáles son los componentes académicos que contienen contenidos similares correspondientes a las áreas de estudio, por ejemplo, redes, programación, etc.

- La utilización de un correcto Procesamiento del Lenguaje Natural es elemental en la Minería de Texto, debido a que el proceso permite remover del contenido de los planes docentes caracteres innecesarios, lematizar las palabras, identificar bigramas y trigramas y crear el vocabulario de términos para construir la matriz términos por documentos, la cual es de suma importancia para aplicar los algoritmos de clustering.

- La generación de un vocabulario de términos eficiente ayuda a que los resultados obtenidos por los algoritmos sean efectivos, debido a que a partir de este se crea la matriz términos por documentos, a la cual los algoritmos reciben como parámetro de entrada para emplear sus procesos.

- La aplicación del algoritmo k-means en el contenido de los documentos genera resultados óptimos y eficientes, debido a que la mayor parte de los grupos formados en cada una de las pruebas llevadas a cabo, están constituidos por contenidos que guardan relación dentro de la misma área de conocimiento, permitiendo identificar fácilmente cuales son los componentes académicos que poseen planes docentes con contenidos similares.

- La utilización del algoritmo de indexación semántica latente (LSI) en el texto de los planes docentes genera resultados excelentes y eficaces, debido a que los vectores que representan a cada uno de los planes docentes en las pruebas realizadas se encuentran agrupados entre sí de acuerdo a la semejanza de sus contenidos, facilitando visualizar cuales son los componentes académicos que poseen planes docentes con contenidos similares.

- El análisis y comparación de los resultados obtenidos en las pruebas realizadas tanto con el algoritmo de agrupamiento k-means como con el algoritmo LSI, permite concluir que cada una de las técnicas posee un margen de error al momento de su ejecución. En el caso del algoritmo k-means, por ejemplo, algunos términos que pertenecen al área de conocimiento de redes se encuentran agrupados en el cluster que posee los contenidos de programación. En cambio, en LSI, si solo se tiene un plan docente que representa a un área de conocimiento específica, este no se agrupa con ningún otro plan docente, debido a que se ubica en la posición (0,0) del plano.

- El prototipo desarrollado en el framework Shiny con las técnicas de Minería de Texto empleadas en el presente trabajo, permite la visualización de los resultados generados en cada fase del proyecto en una interfaz web, la cual facilita el análisis a los usuarios.

- Los resultados obtenidos en las pruebas llevadas a cabo con los algoritmos, demuestran que los grupos están formados por contenidos de planes docentes que pertenecen a la misma área de estudio, por lo que no existe un solapamiento de contenidos con otras áreas de estudio.

RECOMENDACIONES

A partir de los resultados del presente trabajo de titulación, se realiza las recomendaciones siguientes con el fin de dar continuidad y mejoras a trabajos fututos sobre este dominio de investigación.

- Revisar los planes docentes correctamente por los profesores antes de ser cargados a la base de datos de la Universidad Técnica Particular de Loja, para que no tengan faltas de ortografía o caracteres especiales, con el fin de aplicar de forma rápida las diferentes técnicas de Minería de Texto.

- Aplicar otros algoritmos de clasificación no supervisada (clustering) al corpus de planes docentes de la universidad, para contrastar los resultados que se obtengan de estos, con los resultados obtenidos por los algoritmos k-means y LSI aplicados en el presente proyecto.

- Para seleccionar las diferentes técnicas de Minería de Texto (clasificación supervisada, clasificación no supervisada, etc.), se debe tener en cuenta el dominio del proyecto, los datos a manipular y la aplicación que se le dará al mismo, debido a que los resultados que se quiere conseguir al ejecutar el proyecto dependen de estos.

- Aplicar la técnica de agrupamiento a otros grupos de planes docentes de otras titulaciones, con el propósito de poder evaluar si los contenidos de los documentos están estructurados de acuerdo a las áreas de conocimiento de cada carrera.

BIBLIOGRAFÍA

- Aguirre Pérez, Y. (n.d.). Sistema para la detección de plagio y validación de estructura en los documentos científicos emitidos en el centro de información científico técnica (CICT) del ISMMM. Revisado en Marzo 21, 2016, disponible en http://xn--caribea-9za.eumed.net/wp-content/uploads/deteccion-plagio.pdf
- Barbara, R. (2000). Latent Semantic Indexing : An overview. *Infosys 240*, 1–16.
- Bedregal, C. (2008). Agrupamiento de Datos utilizando técnicas MAM-SOM, 77.
- Benítez, I. J., & Díez, L. J. (2005). Trabajo de Investigacíon Técnicas de Agrupamiento para el Análisis de Datos Cuantitativos y Cualitativos.
- Berzal, F. (n.d.). Clustering.
- Blázquez, M. (2012). Técnicas avanzadas de recuperación de información: Frecuencias y pesos. Revisado en Marzo 21, 2016, disponible en http://ccdoc-tecnicasrecuperacioninformacion.blogspot.com/search/label/05.- Frecuencias y pesos
- Blázquez, M. (2012). Técnicas avanzadas de recuperación de información: Modelo Vectorial. Revisado en Agosto 26, 2015, disponible en http://ccdoc-tecnicasrecuperacioninformacion.blogspot.com/2012/12/modelo-vectorial.html
- Botana, G. (2010). La técnica del Análisis de la Semántica Latente (LSA / LSI) como modelo informático de la comprensión del texto y el discurso.
- Bouchet, M. (2015). Package " SnowballC ," 1–5. Revisado en Enero 14, 2016, disponible en https://r-forge.r-project.org/projects/r-temis/
- Brun, E., & Senso, J. (2004). Minería textual. *El Profesional de la Informacion*, pp. 11–27. http://doi.org/10.1076/epri.13.1.11.29021
- Camarena Ibarrola, J. A. (n.d.). El Algoritmo E-M.
- Cardoso, C., & Pérez, A. (2010). Minería de texto para la categorización automática de documentos, 11–45.
- Chali, Y., Joty, S. R., & Hasan, S. a. (2009). Complex question answering: Unsupervised learning approaches and experiments. *Journal of Artificial Intelligence Research*, *35*, 1–47. http://doi.org/10.1613/jair.2784
- Cheng, W., Wang, W., & Batista, S. (n.d.). Data Clustering: Algorithms and Applications.
- Cortez A., Vega, H., & Quispe, J. (2009). Procesamiento de lenguaje natural.
- Davitkov, M. (2011). The BIRCH Algorithm.
- Deepak, P., & Roy, S. (2010). OPTICS on Sequential Data: Experiments and Test Results. *International Journal of Computer Applications*. http://doi.org/10.5120/1582-2119
- El-Sharkawi, E., & El-Zawawy, A. (2009). Algorithm for Spatial Clustering with

Obstacles.

- Fan, W., Wallage, L., & Rich, S. (2006). Tipping the power of text mining.pdf. Revisado en Enero 14, 2016, disponible en http://dml.cs.byu.edu/~cgc/docs/mldm_tools/Reading/Power of Text Mining.pdf
- Feinerer, I., & Hornik, K. (2015). Package "tm" Title Text Mining Package. Revisado en Enero 16, 2016, disponible en https://cran.r-project.org/web/packages/tm/tm.pdf
- Feinerer, I., Hornik, K., & Meyer, D. (2008). Text Mining Infrastructure in R. *JSS Journal of Statistical Software*, *25*(5). Revisado en Enero 16, 2016, disponible en http://www.jstatsoft.org/
- Figuerola, C., Alonso, J. L., & Zazo, A. (2000). Categorización automática de documentos en español: algunos resultados experimentales. *Jornadas de Bibliotecas Digitales*, *1*. Retrieved from http://imhotep.unizar.es/jbidi/jbidi2000/14_2000.pdf
- Gallardo, M. (2009). Aplicación de Técnicas de clustering para la mejora del aprendizaje.
- Gálvez, C. (2007). Identificación de nombres personales por medio de sistemas de codificación fonética 10.5007/1518-2924.2006v11n22p105. *Encontros Bibli: Revista Eletrônica de Biblioteconomia E Ciência Da Informação*. http://doi.org/10.5007/1518-2924.2006v11n22p105
- Gálvez, C. (2008). Minería de textos: la nueva generación de análisis de literatura científica en biología molecular y genómica 10.5007/1518-2924.2008v13n25p1. *Encontros Bibli: Revista Eletrônica de Biblioteconomia E Ciência Da Informação*. http://doi.org/10.5007/1518-2924.2008v13n25p1
- García, C., & Gómez, I. (n.d.). Algoritmos de aprendizaje: knn & kmeans.
- Garre, M., Cuadrado, J., & Sicilia, M. (2007). Comparación de diferentes algoritmos de clustering en la estimación de coste en el desarrollo de software.
- González, D. (2010). Algoritmos de Agrupamiento basados en densidad y Validación de clusters Tesis Doctoral.
- Gupta, V., & Lehal, G. S. (2009). A survey of text mining techniques and applications. *Journal of Emerging Technologies in Web Intelligence*.
- Hearst, M. (2009). Information Visualization for Text Analysis (Ch 11) | Search User Interfaces | Marti Hearst | Cambridge University Press 2009. Revisado en Agosto 9, 2015, disponible en http://searchuserinterfaces.com/book/sui_ch11_text_analysis_visualization.html
- Hornik, K., Buchta, C., Hothorn, T., Karatzoglou, A., Meyer, D., & Zeileis, A. (2015). Package "RWeka" Title R/Weka interface. Revisado en Enero 15, 2016, disponible en https://cran.r-project.org/web/packages/RWeka/index.html
- Hueso, A., & Cascant, J. (2012). Cuadernos docentes en procesos de desarrollo

Metodología y Técnicas Cuantitativas de Investigación.

- Karanikas, H., Tjortjis, C., & Theodoulidis, B. (n.d.). An Approach to Text Mining using Information Extraction. Revisado en Diciembre 11, 2015, disponible en http://www.crim.org.uk
- Kaur, K., & Gupta, V. (2012). A Survey of Topic Tracking Techniques. *International Journal of Advanced Research in Computer Science and Software Engineering*. Revisado en Diciembre 15, 2015, disponible en http://www.ijarcsse.com/docs/papers/May2012/Volum2_issue5/V2I500532.pdf
- Lama, P. (2013). Clustering system based on text mining using the k-means algorithm.
- Larsen, B., & Aone, C. (n.d.). Fast and Effective Text Mining Using Linear-time Document Clustering.
- Lavengood, K., & Kiser, P. (2007). Profesionales de la Información en la mina de texto. Revisado en Agosto 9, 2015, disponible en http://www.infotoday.com/online/may07/Lavengood_Kiser.shtml
- Liu, J. (2008). clique.
- López, A. M. (n.d.). Análisis de Conglomerados (Cluster Analysis). Revisado en Enero 27, 2016, disponible en http://personal.us.es/analopez/ac.pdf
- Macmanus, R. (2010). Top Topic Trackers (Updated List) - ReadWrite. Revisado en Agosto 7, 2015, disponible en http://readwrite.com/2010/01/24/top_topic_trackers_updated
- Maechler, M., Rousseeuw, P., Struyf, A., Hubert, M., Hornik, K., Studer, M., & Roudier, P. (2015). "Finding Groups in Data": Cluster Analysis Extended Rousseeuw et al. Revisado en Agosto 20, 2015, disponible en https://cran.r-project.org/web/packages/cluster/index.html
- Microsoft Azure. (2015). Agrupación en clústeres k-means. Revisado en Enero 26, 2016, disponible en https://msdn.microsoft.com/es-es/library/azure/dn905944.aspx
- Montes, M. C. (2013). clustering: Clasificación no Supervisada Gráficas estadística y minería de datos con python.
- Morales, E. (2012). Algoritmo EM. Revisado en Agosto 12, 2015, disponible en http://ccc.inaoep.mx/~emorales/Cursos/NvoAprend/node81.html
- Morales, L., Canessa, F., Mattar, C., Orrego, R., & Matus, F. (2006). Caracterización y zonificación edáfica y climática de la Región de Coquimbo, Chile. *R.C.Suelo Nutr. Veg.*, 52–74. http://doi.org/10.4067/S0718-27912006000300005
- Navarro, J. (2001). EXP-MAX (Expectation-Maximization, algoritmo para reconocimiento de patrones). Revisado en Agosto 12, 2015, disponible en https://sites.google.com/site/jenavarrob/home/projects/support-vector-machines-para-reconocimiento-de-patrones

- Ng, R. T., & Han, J. (2002). CLARANS: A method for clustering objects for spatial data mining. *IEEE Transactions on Knowledge and Data Engineering*. http://doi.org/10.1109/TKDE.2002.1033770
- Ochando, M. B. (2011). Aplicaciones Documentales de la Recuperación de Información: 09.- Sistemas de clustering. Revisado en Agosto 9, 2015, disponible en http://ccdoc-appsrecuperacioninformacion.blogspot.com/2011/10/09-sistemas-de-clustering.html
- Pascual, D., Pla, F., & Sánchez, S. (2007). Algoritmos de agrupamiento.
- Patel, R., & Sharma, G. (2014). A Survey on Text Mining in Clustering, *3*(0976), 111–116.
- Paulsen, J. R., & Ramampiaro, H. (2009). Combining Latent Semantic Indexing and Clustering to Retrieve and Cluster Biomedical Information : A 2-step Approach Jon Rune Paulsen and Heri Ramampiaro Department of Computer and Information Science Norwegian University of Science and Technology (NTNU. *Science And Technology*.
- Pérez, J., Cruz, L., Reyes, G., & Mexicano, A. (2007). Mejora al algoritmo de agrupamiento K-means mediante un nuevo criterio de convergencia y su aplicación a bases de datos poblacionales de cáncer. *Liver- 2do Taller Latino Iberoamericano de Investigacion de Operaciones "la IO Aplicada a la Solución de Problemas Regionales".*, (August 2015), 1–7. Revisado en Diciembre, 15, 2015, disponible en http://www.tlaio.org.mx/DOCS/T2_5_A27iJPO.pdf
- Pérez, R. (n.d.). Los textos explicativos, 1–18.
- Possamai, L. (2006). Cure, Clustering Algorithm. Revisado en Agosto 11, 2015, disponible en http://es.slideshare.net/ellepiu/cure-clustering-algorithm
- Puldón, J., Espín, R., & Jiménez, S. (2012). Modelo clustering para el análisis en la ejecución de procesos de negocio.
- R Core Team and contributors worldwide. (2016). R: The R Stats Package. Revisado en Enero 27, 2016, disponible en https://stat.ethz.ch/R-manual/R-devel/library/stats/html/00Index.html
- R Core Team and contributors, W. (2016). R: The R Base Package. Revisado en Enero 30, 2016, disponible en https://stat.ethz.ch/R-manual/R-devel/library/base/html/00Index.html
- Ruiz, M. B., & González, I. (n.d.). Tema 5: Organización de los Datos.
- Sierra Araujo, B. (2006). *Aprendizaje Automático: conceptos básicos y avanzados*. Madrid.
- Thilagavathi, K., & Shanmuga, V. (2014). International Journal of Research in Computer Applications and Robotics Issn 2320-7345 Detection of Tumor Region Using

Fast, *2*(4), 145–149.

- Vallez, M., & Pedraza, R. (2007). El Procesamiento del Lenguaje Natural en la Recuperación de Información Textual y áreas afines. *Hipertext. net*. Universitat Pompeu Fabra. Revisado Febrero 10, 2016, disponible en http://www.upf.edu/hipertextnet/numero-5/pln.html#procesamiento-linguistico-lenguaje-natural
- Van, N., & Waltman, L. (2011). Text mining and visualization using VOSviewer, 1–5. Revisado Febrero 10, 2016, disponible en http://arxiv.org/abs/1109.2058
- Vidal, J. (2014). Big data: Gestión de datos no estructurados | Dataprix TI. Revisado en Agosto 4, 2015, disponible en http://www.dataprix.com/blog-it/big-data/big-data-gestion-datos-no-estructurados
- Wickham, H., & Chang, W. (2015). ggplot2. Revisado Enero 30, 2016, disponible en https://cran.r-project.org/web/packages/ggplot2/index.html

ANEXOS

ANEXO 1

Contenido del archivo spanish.data

a
al
algo
algunas
análisis
algunos
ante
antes
aspectos
básicos
básicas
como
cargas
cargo
casos
cargado
código
códigos
cómo
con
componente
contra
componentes
cual
control
controles
cuando
construcción
de
defensa
del
desde
estés
definición
donde
distribuciones
durante
e
el
él
ella
ellas
ellos
en
entre
er
estándar
era
erais
éramos
eran
explicación
eras
entrada
evolución
eres
es
esa
esas
ese
eso
esos
esta
está
estaba
estuvieron
estabais
estábamos
estaban
estabas
estada
estadas
estado
estado
estados
estáis
estamos
están
estando
estar
estará
estarán
estarás
estaré
estaréis
estaremos
estaría
estaríais
estaríamos
estarían
estarías
estas
estás
este
esté
estéis
estemos
estén
fue
fuesen
fueses
fui
fuimos
fuiste
fuisteis
gestión
ha
habéis
había
habíais
habíamos
habían
habías
habida
habidas
habido
habidos
habiendo
habrá
habrán
habrás
habré
habréis
habremos
habría
habríais
habríamos
habrían
habrías
han
has
hasta

esto
estos
estoy
estuve
estuviera
estuvierais
estuviéramos
estuvieran
estuvieras
hubiera
hubierais
hubiéramos
hubieran
hubieras
hubieron
hubiese
hubieseis
hubiésemos
hubiesen
hubieses
hubimos
hubiste
hubisteis
hubo
inicios
inicio
la
las
le
les
línea
líneas
lo
los
más
me
estuviese
estuvieseis
estuviésemos
estuviesen
estuvieses
estuvimos
estuviste
estuvisteis
estuvo
nuestra
nuestras
nuestro
nuestros
o
os
otra
otras
otro
otros
para
pero
prácticas
presentación
propósito
poco
planificación
por
porque
que
qué
quien
quienes
representación
revisión
relación
rutas
fuera
fuerais
fuéramos
fueran
fueras
fueron
fuese
fueseis
fuésemos
sí
sido
siendo
sin
sobre
sois
somos
son
soy
su
sus
suya
suyas
suyo
suyos
también
tanto
te
tendrá
tendrán
tendrás
tendré
tendréis
tendremos
tendría
tendríais
tendríamos
hay
haya
hayáis
hayamos
hayan
hayas
he
hemos
hube
tenidos
teniendo
ti
tiene
tienen
tienes
todo
todos
tú
tu
tus
tuve
tuviera
tuvierais
tuviéramos
tuvieran
tuvieras
tuvieron
tuviese
tuvieseis
tuviésemos
tuviesen
tuvieses
tuvimos
tuviste
tuvisteis
tuvo

método
métodos
mí
mi
mía
mías
mío
míos
mis
mucho
muchos
modelo
muy
nada
ni
no
nos
nosotras
nosotros
carga
clasificación
clasificaciones
componente
organización
se
sea
seáis
seamos
sean
seas
será
serán
serás
seré
seréis
seremos
selección
sería
seríais
seríamos
salida
serían
serías
creación
descripción
evaluación
introducción
áreas
tendrían
tendrías
tened
tenéis
tenemos
tenga
tengáis
tengamos
tengan
tengas
tengo
tenía
teníais
teníamos
tenían
tenías
tenida
tenidas
tenido
tuya
tuyas
tuyo
tuyos
un
una
uno
unos
uso
vosotras
vosotros
vuestra
vuestras
vuestro
vuestros
y
ya
yo
académico

ANEXO 2

Lista de palabras vacías

abajo	aquello	ciertos	cuánto
acá	aquellos	cinco	cuantos
actualmente	aquéllos	claro	cuántos
acuerdo	aquí	comentó	cuatro
adelante	arriba	conmigo	cuenta
además	arriba	conocer	d
adrede	aseguró	conseguimos	da
afirmó	así	conseguir	dado
agregó	atrás	considera	dan
ahí	aun	consideró	dar
ahora	aún	consigo	debajo
ajena	aunque	consigue	debe
ajenas	ayer	consiguen	deben
ajeno	b	consigues	debido
ajenos	bajo	contigo	debidos
algún	base	control	decir
alguna	bastante	cosas	dejar
alguno	bien	creo	dejó
allá	breve	cuál	delante
allí	buen	cuales	demás
alrededor	buena	cuáles	demasiada
ambos	buenas	cualquier	demasiadas
añadió	bueno	cualquiera	demasiado
antaño	buenos	cualquieras	demasiados
anterior	c	cuan	dentro
apenas	cabe	cuán	
aquel	cada	cuándo	
aquél	casi	cuanta	
aquella	cerca	cuánta	
aquélla	cierta	cuantas	
aquellas	ciertas	cuántas	
aquéllas	cierto	cuanto	
deprisa	eras	hacemos	llegó

despacio
después
detrás
día
días
dice
dicen
dicho
dieron
diferente
diferentes
dijeron
dijo
dio
dónde
dos
él
ella
ellas
ellos
embargo
empleáis
emplean
emplear
empleas
empleo
encima
encuentra
enfrente
enseguida
entonces
era
éramos
eran
muchísimo
muchísimos
eres
es
ésa
ésas
ése
ésos
ésta
están
éstas
éste
éstos
etc
ex
excepto
existe
existen
explicó
expresó
f
fin
final
fueron
fui
g
general
gran
grandes
h
haber
había
habla
hablan
hace
hacéis
pocas
pocos
hacen
hacer
hacerlo
haces
hacia
haciendo
hago
hecho
hicieron
hizo
hoy
i
iv
igual
incluso
indicó
informo
informó
intenta
intentáis
intentamos
intentan
intentar
intentas
intento
ir
j
jamás
junto
juntos
l
lado
largo
lejos
quedó
queremos
lleva
llevar
lo
luego
lugar
m
mal
manera
manifestó
mas
mayor
me
mediante
medio
mejor
mencionó
menos
menudo
mi
mía
mías
mientras
mío
míos
misma
mismas
mismo
mismos
modo
momento
mucha
muchas
muchísima
muchísimas
será
sería

n
nadie
ningún
ninguna
ningunas
ninguno
ningunos
nivel
nueva
nuevas
nuevo
nuevos
null
nulo
número
nunca
o
ocho
otra
otras
otro
otros
p
país
parece
parecer
parte
partir
pasada
pasado
peor
pesar
poca
podéis
podemos
poder
podrá
podrán
podría
podríais
podríamos
podrían
podrían
podrías
poner
por
porque
posible
primer
primera
primero
primeros
pronto
propia
propias
propio
propios
próximo
próximos
pudo
pueda
puede
pueden
puedo
pues
q
querer
quien
quién
quienes
quiénes
quiera
quiera
quiere
quizá
quizás
r
raras
realizado
realizar
realizó
repente
respecto
ruta
s
sabe
sabéis
sabemos
saben
saber
sabes
salvo
sé
según
segunda
segundo
seis
señaló
ser
si
siempre
siendo
siete
sigue
siguiente
sin
sino
so
sola
solamente
solas
solo
sólo
solos
sr
sra
sres
sta
supuesto
t
tabla
tal
tales
tampoco
tan
tanta
tantas
tantos
tarde
te
temprano
tenéis
tener
tenido
usar
usas

tercera
ti
tiempo
tipo
toda
todas
todavía
tomar
total
trabaja
trabajáis
trabajamos
trabajan
trabajar
trabajas
trabajo
tras
trata
través
tres
u
última
últimas
ultimo
último
últimos
un
una
unos
usa
usáis
usamos
usan
usted
ustedes
v
va
vais
valor
vamos
van
varias
varios
vaya
veces
ver
verdad
verdadera
verdadero
vez
volcado
voy
vs
w
x
y
ya
z

ANEXO 3

Lista de bigramas y trigramas

program orient objet
tecnolog distribu
patron creacional
patron relacional
patron estructural
patron comport
teor automat
expresion regular
cuent activ
circuit logic
flip flops
distribu frecuenci
distribu probabil discret
distribu probabil continu
distribu probabil normal
teorem limit central
energ cinet
carg electr
electr diferent
energ potencial
corrient electr
codig etic
gestion proyect
cierr proyect
ea cub framework
gestion tecnolog inform
teor grafic
model red
recurs human
vocabulari abiert
sistem oper
gestion memori
sistem distribu
arquitectur sgbd
bas dat relacional
control intern
departament auditor
segur fisic
auditor segur
desarroll modul
gestion usuari
metodolog desarroll web
web app
regl neuronal
model ocult markov
analisis clust
teor grafic
algebr boolean
circuit combinatori
replic bas dat
bas dat movil
ajust pes

ANEXO 4

Vocabulario

direccion
dma
microarquitectur
cicl instruccion
memori
unix
memori compart
comun proces
cerroj
lector escritor
jav
program orient objet
tecnolog distribu
patron creacional
patron relacional
patron estructural
patron comport
rest
cobit
control intern
departament auditor
outsourcing
segur fisic
auditor segur
direccion ipv
vlans
conexion
internet
enrut dinam
rip
bas dat relacional
segur
oracl
web
replic bas dat
bas dat movil
bas dat semant
automat finit
teor automat
expresion regular
analisis lexic
analisis sintact
cuent activ
caj
capital
ingres
egres
contabil
empres
plan cuent
cuent ingres
circuit logic
flip flops
disposit
multiplexor
estadist
distribu frecuenci
distribu probabil discret
med
distribu probabil continu
distribu probabil normal
teorem limit central
muestre
entorn bas dat
sgbd
model relacional
algebr relacional
calcul relacional
sql
model entid relacion
normaliz
cienci comput
informat
logic proposicional
arquitectur von neumann
desarroll sistem
web semant
sistem numer
inteligent artificial
cinemat
veloc
rapidez
aceler
energ cinet
carg electr
camp electr
electr diferent
energ potencial
corrient electr
ingeni softwar
diagram
ingeni requer
prueb desarroll
cas uso
program
do whil
for
arregl
matriz

clas
metod
etic
codig etic
gestion proyect
cost
calendariz proyect
control proyect
gestion riesg
gestion calid
cierr proyect
pmi
ea cub framework
riesg
gestion tecnolog inform
negoci
document arquitectur empresarial
cultur empresarial
arbol clasif
algoritm
regl neuronal
ajust pes
model ocult markov
analisis clust
tar
proces requer
prototip
document especif requer
captur requer
analisis requer
especif requer softwar
requer softwar
document requer
php
laravel
desarroll modul
gestion usuari
maquet
metodolog desarroll web
accesibil
web app
logic program
silog
premis
domini
regl procedent
algoritm orient dat
miniespecif
prueb escritori
diagram fluj
proposicion
teor grafic
algebr boolean
circuit combinatori
isomorf
model red
polit
reingeni
departamentaliz
tecnolog inform
comerci
uml
diagram clas
subproces
recurs human
mejor proces
constructor
herenci
api jav
polimorf
excepcion
program ficher jav
algoritm orden
algoritm enrut
udp
protocol transaccion dat
tcp
cap red
cap transport
datagram
enrut intern
multiplex
demultiplex
vocabulari abiert
ontolog
consult
dat rdf
sistem oper
gestion memori
interfaz sistem archiv

ANEXO 5

Matriz términos por documentos de la prueba número uno

Terms														
Docs	algebr	relacional	algoritm	algoritm	enrut	arquitectur	sgbd	arregl	bas dat	movil	bas dat	relacional	bas dat	semant
asr	0	0	0		0	0	0	0	0		0	0	0	
bda	0	0	0		1	0	3		3		2			
fbd	3	0	0		1	0	0		1		0			
fpr	0	4	0		0	7	0		0		0			
palg	0	7	0		0	7	0		0		0			
pav	0	0	0		0	0	0		1		0			
rsd	0	4	4		0	0	0		0		0			

Terms																
Docs	calcul	relacional	cap red	cap transport	cas uso	clas	conexion	datagram	demultiplex	diagram	clas	direccion	ipv do	whil	enrut	dinam
asr	0	0	0	0	0	1	0	0	0		1	0		0		2
bda	0	0	0	0	0	0	0	0	0		0	0		0		0
fbd	3	0	0	0	0	0	0	0	0		0	0		0		0
fpr	0	0	0	0	4	0	0	0	1		0	4		0		
palg	0	0	0	0	7	0	0	0	0		0	0		0		
pav	0	0	0	1	1	1	0	0	1		0	0		0		
rsd	0	2	3	0	0	5	3	2	0		1	0		0		

Terms																
Docs	enrut	intern	expresion	regular	for	herenci	internet	metod	model	entid	relacion	model	relacional	multiplex	normaliz	oracl	program	
asr	0		0	0	0	1	0		0		0		0	0	0	0		
bda	0		0	0	0	0	0		0		0		0	0	2	0		
fbd	0		0	0	0	0	0		2		4		0	5	0	0		
fpr	0		0	5	0	0	5		0		0		0	0	0	6		
palg	0		0	1	3	0	3		0		0		0	0	0	7		
pav	0		3	0	0	0	0		0		0		0	0	0	1		
rsd	2		0	0	0	2	1		0		0		2	0	0	0		

Terms																
Docs	program	orient	objet	protocol	transaccion	dat	red	replic	bas dat	segur	sgbd	sql	tcp	udp	uml	vlans
asr	0				0	2		0	0	0	0	0	0	0	2	
bda	0				0	1		5	4	5	0	0	0	0	0	
fbd	0				0	0		0	1	4	9	0	0	0	0	
fpr	3				0	0		0	0	0	0	0	0	0	0	
palg	5				0	0		0	0	0	0	0	0	0	0	
pav	0				0	0		0	0	0	0	0	0	3	0	
rsd	0				2	7		0	0	0	0	4	2	0	0	

ANEXO 6

Matriz términos por documentos de la prueba número dos

Terms / Docs	aceler	algebr	relacional bas dat	movil bas dat	relacional bas dat	semant calcul	relacional camp	electr carg	electr cas	uso cinemat	clas
bda	0	0	3	3	2	0	0	0	0	0	0
fbd	0	3	0	1	0	3	0	0	0	0	0
fisc	4	0	0	0	0	0	4	3	0	2	0
fpr	0	0	0	0	0	0	0	0	0	0	4
palg	0	0	0	0	0	0	0	0	0	0	7
pav	0	0	0	1	0	0	0	0	1	0	1

Terms / Docs	corrient electr	diagram clas	do whil	electr diferent	energ cinet	energ potencial	entorn bas dat	expresion regular	herenci
bda	0	0	0	0	0	0	0	0	0
fbd	0	0	0	0	0	0	1	0	0
fisc	2	0	0	2	2	2	0	0	0
fpr	0	1	4	0	0	0	0	0	0
palg	0	0	0	0	0	0	0	0	3
pav	0	1	0	0	0	0	0	3	0

Terms / Docs	model entid	relacion model	relacional normaliz	oracl program	orient objet	rapidez replic	bas dat segur	sgbd subproces	uml veloc
bda	0	0	0	2	0	0	5	4	5	0	0	0
fbd	2	4	5	0	0	0	0	1	4	0	0	0
fisc	0	0	0	0	0	2	0	0	0	0	0	3
fpr	0	0	0	0	3	0	0	0	0	0	0	0
palg	0	0	0	0	5	0	0	0	0	0	0	0
pav	0	0	0	0	0	0	0	0	0	3	3	0

ANEXO 7

Matriz términos por documentos de la prueba número tres

Terms / Docs	accesibil	ajust	pes	algoritm	enrut	analisis	clust	arbol	clasif	cap	red	cap	transport	captur	requer	comerci	conexion	datagram	departamentaliz
asr	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
fings	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
iaa	0	2	0	4	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0
ingreq	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0
ingw	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
oae	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	3
rsd	0	0	4	0	0	2	3	0	0	5	3	0							

Terms / Docs	desarroll	modul	direccion	ipv	document	especif	requer	document	requer	enrut	dinam	especif	especif	requer	softwar	factor	human	gestion	usuari
asr	0	1	0	0	0	2	0												
fings	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0
iaa	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
ingreq	0	0	2	2	0	3	2	0	0	0	0	0	0	0	0	0	0	0	0
ingw	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3
oae	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0
rsd	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Terms / Docs	ingeni	requer	ingeni	softwar	instal	internet	laravel	maquet	metodolog	desarroll	web	model	ocult	markov	php	polit	premis	proces	requer
asr	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
fings	2	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
iaa	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0
ingreq	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	2
ingw	0	0	3	0	2	2	0	0	0	3	0	3	0	0	0	0	0	0	0
oae	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	2	0	0
rsd	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Terms / Docs	protocol	transaccion	dat	prototip	prueb	desarroll	regl	neuronal	reingeni	tar	tcp	tecnolog	inform	udp	vlans	web	app
asr	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0		
fings	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0		
iaa	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0		
ingreq	0	2	0	0	0	3	0	0	0	0	0	0	0	0	0		
ingw	0	0	0	0	0	1	0	0	0	0	0	0	0	0	2		
oae	0	0	0	0	2	0	0	0	0	0	0	3	0	0	0		
rsd	2	0	0	0	0	0	4	0	2	0							

ANEXO 8

Matriz términos por documentos de la prueba número cuatro

Terms																	
Docs	algoritm	orden	algoritm	orient	dat	analisis	lexic	analisis	sintact	api	jav	automat	finit	calendariz	proyect	cas	uso	cierr	proyect	clas
1	0		0		6		9		0		9		0		0		0		0	
2	0		0		0		0		0		0		0		1		0		1	
3	0		0		0		0		0		0		0		0		0		4	
4	0		0		0		0		0		0		1		0		2		0	
5	0		4		0		0		0		0		0		0		0		0	
6	3		0		0		3		0		0		0		0		0		7	
7	0		0		0		0		0		0		0		1		0		1	
8	0		0		0		0		0		0		1		0		0		0	

Terms																					
Docs	codig	etic	control	proyect	cost	diagram	clas	diagram	fluj	do	whil	etic	expresion	regular	for	gestion	calid	gestion	proyect	gestion	riesg	herenci
1	0		0		0		0		0		0		4		0		0		0		0	
2	0		0		1		0		0		1		0		0		0		0		0	
3	0		0		0		1		0		4		0		5		0		0		0	
4	4		2		1		0		0		0		7		0		1		5		1	
5	0		0		0		0		6		0		0		0		0		0		0	
6	0		0		0		0		0		0		0		1		0		0		0	
7	0		0		0		1		0		0		3		0		0		0		0	
8	0		0		0		0		0		0		0		0		1		1		1	

Terms																					
Docs	ingeni	requer	ingeni	softwar	logic	program	mejor	proces	metod	miniespecif	pmi	program	ficher	jav	program	orient	objet	proposicion			
1	0		0		0		0		0		0		0		0		0				
2	2		2		0		0		0		0		0		0		0				
3	0		0		0		0		5		0		0		3		0				
4	0		1		0		0		0		2		0		0		0				
5	0		0		6		0		0		6		0		2		2				
6	0		0		0		0		3		0		2		5		0				
7	0		0		0		0		0		0		0		0		0				
8	0		0		0		3		0		0		0		0		0				

Terms									
Docs	prueb	desarroll	recurs	human	regl	procedent	silog	teor	automat	uml
1	0		0		0		0		2	0
2	2		0		0		0		0	0
3	0		0		0		0		0	0
4	0		0		0		0		0	0
5	0		0		3		2		0	0
6	0		0		0		0		0	0
7	0		0		0		0		0	3
8	0		0		0		0		0	0

ANEXO 9

Matriz términos por documentos de la prueba número cinco

Terms																	
Docs	algebr	relacional	algoritm	enrut	bas dat	movil	bas dat	relacional	bas dat	semant	calcul	relacional	cap red	cap transport	conexion		
asr	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1		
bda	0	0	3	3	2	0	0	0	0								
fbd	3	0	0	1	0	3	0	0	0								
rsd	0	4	0	0	0	0	2	3	5								

Terms																	
Docs	datagram	demultiplex	direccion	ipv	enrut	dinam	enrut	intern	entorn	bas dat	internet	model	entid	relacion	model	relacional	multiplex
asr	0	0	1	2	0	0	1	0	0	0							
bda	0	0	0	0	0	0	0	0	0	0							
fbd	0	0	0	0	0	1	0	2	4	0							
rsd	3	2	1	0	2	0	2	0	0	2							

Terms																	
Docs	normaliz	oracl	protocol	transaccion	dat	replic	bas dat	rip	segur	sgbd	tcp	udp	vlans				
asr	0	0	0	0	1	0	0	0	2								
bda	0	2	0	5	0	4	5	0	0	0							
fbd	5	0	0	0	0	1	4	0	0	0							
rsd	0	0	2	0	0	0	0	4	2	0							

ANEXO 10

Matriz términos por documentos de la prueba número seis

Terms														
Docs	ajust	pes	algebr	relacional	algoritm	enrut	analisis	clust	arbol	clasif	bas dat	movil	bas dat	relacional	bas dat	semant
asr	0		0		0	0	0	0	0	0		0		0		0
bda	0		0		0	0	0	0	0	3		3		2		
fbd	0		3		0	0	0	0	0	0		1		0		
iaa	2		0		0	4	3	0	0	0		0		0		
rsd	0		0		4	0	0	0	0	0		0		0		

Terms																
Docs	calcul	relacional	cap	red	cap	transport	conexion	datagram	direccion	ipv	enrut	dinam	enrut	intern	entorn	bas dat	internet
asr	0		0		0		1	0	1		2	0		0	0	1
bda	0		0		0		0	0	0		0	0		0	0	0
fbd	3		0		0		0	0	0		0	0		1	0	
iaa	0		0		0		0	0	0		0	0		0	0	
rsd	0		2		3		5	3	1		0	2		0	2	

Terms																	
Docs	model	entid	relacion	model	ocult	markov	model	relacional	normaliz	oracl	protocol	transaccion	dat	regl	neuronal	replic	bas dat	segur	
asr	0		0		0		0		0	0		0		0		0	0	0	
bda	0		0		0		0		0	2		0		0		5	4		
fbd	2		0		4		5		0	0		0		0		0	1		
iaa	0		2		0		0		0	0		0		3		0	0		
rsd	0		0		0		0		0	0		2		0		0	0		

Terms				
Docs	sgbd	tcp	udp	vlans
asr	0	0	0	2
bda	5	0	0	0
fbd	4	0	0	0
iaa	0	0	0	0
rsd	0	4	2	0

ANEXO 11

El código del prototipo esta disponible en: https://github.com/rwcaraguay/TextMiningPlanesDocentes