APLICACIÓN DE TÉCNICAS DE MINERÍA DE DATOS PARA EXTRAER INFORMACIÓN DE FUENTES ORGANIZACIONALES, EN LA EDUCCIÓN DE REQUISITOS

AUTOR:
Ing. JUAN SEBASTIÁN MORALES PÉREZ

DIRECTORES:
Dra. Bell Manrique Losada
Dr. Juan Bernardo Quintero

LÍNEA TEMÁTICA:
Minería de Datos

MAESTRÍA EN INGENIERÍA DE SOFTWARE
UNIVERSIDAD DE MEDELLÍN

MEDELLÍN
2020

AGRADECIMIENTOS

Quisiera dar mis más sinceros agradecimientos a la Dra. Bell Manrique Losada, docente y directora de este trabajo investigativo, quien desde el inicio me brindó la guía a seguir en esta línea de trabajo, desde la Especialización en Ingeniería de Software. También quiero agradecer, al Dr. Juan Bernardo Quintero, docente y co-director. Los dos me apoyaron con constancia y rigurosidad en la elaboración de este trabajo investigativo.

A las docentes María Clara Gómez y Luisa Fernanda Villa, quienes me brindaron las bases y guías que fundamentaron la propuesta inicial de este trabajo, partiendo del planteamiento de la revisión sistemática de literatura.

A los ingenieros que participaron del experimento, en especial a Diego Alejandro Marín, colega que admiro mucho por su tenacidad y acompañamiento constante en este proyecto, considerando que él se encontraba en una situación académica similar a la mía, adicional al desgaste laboral del día a día.

TABLA DE CONTENIDO

LISTA DE FIGURAS

LISTA DE TABLAS

PARTE I:

INTRODUCCIÓN

CAPÍTULO 1: INTRODUCCIÓN

En la ingeniería de software la primera actividad del ciclo de vida del desarrollo de software es la ingeniería de requisitos (Javed, 2010) la cual es una disciplina enfocada en la educción, especificación, validación, modelado y análisis de la necesidades y restricciones que tendrá un producto de software (Unterkalmsteiner, Gorschek, Feldt, & Klotins, 2015). Partiendo de esto, es evidente que la ingeniería de requisitos juega un rol importante en el éxito o fracaso de un producto de software (Javed, 2010). Por ello, entidades como IEEE, *European Space Agency*(ESA) y la NASA han definido estándares que ayudan a la definición de requisitos, los cuales aportan restricciones semánticas que reducen las irregularidades como ambigüedad, polisemia o vaguedad al momento de definirlos, garantizando así la consistencia, trazabilidad, veracidad y modificabilidad de los mismos (Toussaint, 1995).

La ingeniería de requisitos se relaciona con las propiedades y restricciones a alcanzar en entornos de software, basándose en fases tales como definición, calidad, negociación y liberación del producto. Estas fases en porcentaje de costos corresponden normalmente al 40% del costo total del proyecto, por ende, la mala ejecución de las mismas conlleva a requerimientos de baja calidad, una de las principales razones para el fracaso en los proyectos de software (Ninaus et al., 2014).

Dentro del proceso de ingeniería de requisitos, existe una fase relevante llamada educción de requisitos. Lo que trae consigo un conjunto de actividades que permiten la comunicación, la priorización, la negociación y la colaboración entre todas las partes interesadas de acuerdo con Zowghi y Coulin (2005). Según los autores, la educción de requisitos debe proveer bases sólidas para el nacimiento, descubrimiento e identificación de requisitos como parte de un proceso de captura interactivo. En la fase de educción de requisitos, los requisitos funcionales (RF) son especificados en su mayoría por los usuarios interesados que cuentan con conocimiento del dominio. Dichos RF, son definidos comúnmente en lenguaje natural, el cual se encuentra en documentos, emails y notas que se propagan a lo largo del ciclo de vida de desarrollo del software. Por ello, la habilidad de capturar y clasificar dicho conocimiento semántico evidencia la necesidad de implementar procesos automáticos (Mahmoud & Williams, 2016).

De acuerdo con Ninaus y otros (2014), los métodos y herramientas utilizados en la educción de requisitos, fallan al momento de proveer información adicional, como las relaciones ocultas entre los requisitos y el estado de calidad de éstos, debido a la creciente complejidad de los sistemas de software y de las fuentes de información que van desde repositorios de información, bases de datos, usuarios, documentos escritos en lenguaje natural hasta sistemas legados. Esto ha generado una creciente demanda de enfoques basados en inteligencia artificial, en aras de mejorar la calidad de los procesos en ingeniería de requisitos, a través de la automatización de estos (Ninaus et al., 2014).

Actualmente, la creciente complejidad de los sistemas de software ha llevado a las organizaciones a ser cada vez más competitivas al momento de desarrollar nuevos productos que satisfagan las necesidades de un determinado dominio. Impulsándolas a mejorar su proceso de ingeniería de requisitos desde la fase

de educción. Sin embargo, en este camino de mejoras se evidencia la dificultad en la tarea de extraer información significativa de un conjunto de datos de una manera automatizada (Kumar, Rath, Nadaf, & Simha, 2015). Según estos autores (Kumar et al., 2015), los datos se encuentran en línea y en directorios locales (pdf, xml, doc y html) de una manera no estructurada, en donde palabras claves, siglas y formatos de las tablas varían de una fuente a otra, lo cual conlleva a una difícil y gigantesca tarea de extracción de información (Kumar et al., 2015).

No obstante, la fase de educción de requisitos cuenta con un trasfondo muy relevante que es la identificación, captura y especificación de los RF y los requisitos no funcionales (RNF). Sin embargo, a medida que incrementa la complejidad y escala de los sistemas de software, no es suficiente con centrarse en la funcionalidad del sistema. Por el contrario, los RNF deben ser tratados con cautela y con el debido detalle. En la ingeniería de requisitos, los RNF son descritos como restricciones que un sistema debe respetar, aparte del comportamiento funcional del sistema. Algunos RNF como la carga, la seguridad y facilidad de uso son fundamentales para el éxito de un sistema (Yin & Jin, 2012).

Mahmoud y Williams (2016) definen los RNF como un conjunto de atributos de calidad que un sistema de software debe exhibir. Dichos atributos imponen restricciones operativas en diferentes aspectos del comportamiento del sistema, como su facilidad de uso, seguridad, confiabilidad, rendimiento y portabilidad. Por ello, es fundamental la identificación de los RNF al inicio de proyecto de software. Sin embargo, los autores afirman que los RNF a menudo se pasan por alto durante la fase de educción de requisitos, donde el énfasis principal es lograr que las características funcionales del sistema se definan de manera explícita y formal. Parte de este fenómeno puede atribuirse al poco entendimiento de lo que realmente son los RNF y la falta de métodos efectivos de obtención, modelado y documentación de estos (Mahmoud & Williams, 2016).

Infortunadamente, es común que los RNF se descubran relativamente tarde en el proceso de desarrollo, de acuerdo con Cleland-Huang y otros (2006). La definición tardía de los RNF es debido a la baja calidad presente en el proceso de educción de requisitos. Ya que estos, se documentan en múltiples artefactos que incluyen anotaciones, notas de entrevistas y actas de reuniones. Esto genera incertidumbre en los analistas de requisitos, quienes no logran, obtener una perspectiva clara sobre los RNF de todo el sistema. Las especificaciones de requisitos resultantes a menudo se organizan por funcionalidad (RF), a diferencia de los RNF que se encuentran dispersos en toda la especificación (Cleland-Huang et al., 2006). Por tanto, es importante tener en cuenta que cuando un usuario no sabe lo que quiere en su sistema, mezcla información de RNF con RF. Por ende, es necesario explotar a nivel semántico los RF, para extraer RNF de acuerdo con Mahmoud y Williams (2016). Sin embargo, dado que el conocimiento de los RNF puede tener estructuras y reglas potencialmente complejas, resulta difícil capturar y reutilizar los patrones RNF cuando se representan solo de forma textual (Supakkul & Chung, 2010).

Todo lo anterior denota el poco avance que actualmente presenta la fase de educción de requisitos al ser tan dependiente de la labor humana y de la experticia de los ingenieros para la identificación y clasificación de los RNF, convirtiéndose en un proceso lento, tedioso y costoso.

Considerando la evidencia anteriormente descrita, este proyecto de grado propone el uso de técnicas de minería de datos, para semi-automatizar el proceso de extracción de información desde documentos organizacionales (documentos de requisitos y texto sin estructura escritos en lenguaje natural), lo cual contribuye a la disminución de esfuerzo, costo, y asegura la calidad de los resultados obtenidos, con respecto a la labor manual realizada por analistas de requisitos en la fase de educción.

Este trabajo tiene como objetivo proponer un método para la extracción de información desde documentos organizacionales escritos en lenguaje natural aplicando técnicas de minería de datos, para disminuir esfuerzo, costo y asegurar la calidad en la educción de requisitos. Para lo cual fue necesario clasificar y aplicar técnicas de minería de datos para el procesamiento de documentos escritos en lenguaje natural, diseñar e implementar un método para el proceso de extracción de información desde fuentes organizacionales y evaluar el método diseñado aplicándolo en un caso de estudio real, que permite comparar la información extraída en términos de esfuerzo, costo y calidad con respecto a la labor manual realizada por profesionales de ingeniería de requisitos en la fase de educción.

El presente trabajo se estructurado de la siguiente manera. El Capítulo 2 contiene el contexto general de la investigación a nivel del problema, pregunta investigativa, hipótesis y objetivos. En el Capítulo 3 se detalla el marco teórico con la fundamentación de la investigación. En el Capítulo 4 se presenta la revisión de literatura, donde se halla el estado del arte del presente trabajo. Luego, en el Capítulo 5 se define el contexto del método propuesto, a través de la caracterización de técnica, algoritmo y método en el ámbito de Minería de Datos. En el Capítulo 6 se justifica mediante fundamentos el por qué la información a extraer de los documentos de requisitos será RNF; e igualmente se presenta el detalle de la experimentación inicial realizada para seleccionar los algoritmos. Después, en el Capítulo 7 se plantea el método de extracción de información propuesto. En el Capítulo 8 se evalúa el método propuesto. Para finalizar, en el Capítulo 9 se encuentran las conclusiones y el trabajo futuro, producto de este trabajo investigativo.

CAPÍTULO 2: CONTEXTO DE LA INVESTIGACIÓN

2.1. CONTEXTO DEL PROBLEMA

2.1.1. Definición

La ingeniería de requisitos tiene un papel importante en el éxito de un proyecto de software (Javed, 2010) a través de la educción, especificación, modelado y análisis de las necesidades planteadas por los *Stakeholders* sobre un producto de software (Unterkalmsteiner et al., 2015).

La educción de requisitos dentro de la ingeniería de requisitos abarca el aprendizaje y la comprensión de las necesidades de los usuarios y los *Stakeholders* del proyecto, en aras de transmitirlas de una manera clara y concisa a los desarrolladores de software (Zowghi & Coulin, 2005). Sin embargo, es importante resaltar que un usuario comúnmente se centra en los RF del producto de software, dejando por fuera los RNF que imponen restricciones operativas en diferentes aspectos del comportamiento del sistema, según Mahmoud y Williams (2016).

Por tanto, es vital definir desde el inicio de un proyecto de software en la educción de requisitos los RNF, ya que van más allá de la funcionalidad de un producto de software y fundamentan el éxito de un sistema (Yin & Jin, 2012).

2.1.2. Problema a tratar en la propuesta investigativa

Los RNF describen un conjunto de atributos de calidad que un sistema de software debe exhibir. Dichos atributos imponen restricciones operativas en diferentes aspectos del comportamiento del sistema, como su facilidad de uso, seguridad, confiabilidad, rendimiento y portabilidad. Por ello, es fundamental la identificación de los RNF al inicio de un proyecto de software en la ingeniería de requisitos, en aras de tomar decisiones óptimas al diseñar el modelo de arquitectura para evitar arquitecturas anómalas (Mahmoud & Williams, 2016). Infortunadamente, es común que los RNF se descubran relativamente tarde en el proceso de desarrollo. Esto debido a las falencias de calidad presentadas en las partes interesadas, provocadas durante el proceso de educción de requisitos (Cleland-Huang et al., 2006).

La implementación de técnicas de Minería de Datos basadas en la clasificación de información podría aportar significativamente a la labor manual realizada por los ingenieros de requisitos, al momento de clasificar los RNF. En esta instancia, se puede hablar de semi-automatizar la clasificación de los RNF que son la base de los diseños de arquitectura (Casamayor et al., 2010).

2.2. PREGUNTA DE INVESTIGACIÓN

¿De qué manera se podrían implementar técnicas de minería de datos para optimizar el proceso de extracción de información del negocio desde documentos organizacionales escritos en lenguaje natural?

2.3. HIPÓTESIS

La implementación de técnicas de minería de datos, para semi-automatizar el proceso de extracción de información desde documentos organizacionales (documentos de requisitos y texto sin estructura escritos en lenguaje natural), contribuye a la disminución de esfuerzo y costo, y asegura la calidad de los resultados obtenidos, con respecto a la labor manual realizada por ingenieros de requisitos en el proceso de educción.

2.4. OBJETIVOS

2.4.1. Objetivo general

Proponer un método de extracción de información desde documentos organizacionales escritos en lenguaje natural aplicando técnicas de minería de datos, para disminuir esfuerzo, costo y asegurar la calidad en la educción de requisitos.

2.4.2. Objetivos específicos

1. Identificar las diversas problemáticas presentadas actualmente en el proceso de extracción de información, en la educción de requisitos.
2. Clasificar las técnicas de minería de datos que pueden ser más efectivas para el proceso de extracción de información desde documentos escritos en lenguaje natural, en términos de esfuerzo, costo y calidad.
3. Aplicar técnicas de minería de datos en el procesamiento de documentos organizacionales escritos en lenguaje natural, para analizar la información extraída e identificar patrones. Dichos documentos serán procesados uno a la vez.
4. Diseñar un método para el proceso de extracción de información de fuentes organizacionales, a partir de la integración de técnicas de minería de datos identificadas.
5. Evaluar el método diseñado por medio de su aplicación en un caso de estudio, que permita comparar la efectividad de la información extraída (con respecto al esfuerzo, costo y calidad), en comparación con la labor manual realizada por profesionales de ingeniería de requisitos en la fase de educción.

2.5. ALCANCES DEL TRABAJO

En este trabajo se espera procesar documentos de requisitos escritos en lenguaje natural, en aras de clasificar RNF en la fase de educción de requisitos en la ingeniería de requisitos. Para ello, se propone utilizar técnicas de clasificación de minería de datos, basadas en algoritmos del mismo tipo. De esta manera, mediante el mapeo de características de los tipos de RNF, es

posible identificar secciones, asociadas a este tipo de requisitos que se encuentran en los documentos de requisitos. Esto, podría ser de gran ayuda para los ingenieros de requisitos, considerando que el proceso de clasificación de RNF es una labor desgastante, porque estos últimos se encuentran en su mayoría dispersos en la documentación y ocultos dentro de los RF (Casamayor et al., 2010).

PARTE II:

FUNDAMENTOS TEÓRICOS

CAPÍTULO 3. MARCO TEÓRICO

Un método de extracción de información desde documentos organizacionales escritos en lenguaje natural (documentos de requisitos y textos sin estructura), debe contar con los ejes conceptuales necesarios, que faciliten la compresión e interpretación del área temática en la que se enmarca el trabajo. En este apartado se encuentra la definición de áreas de inteligencia artificial como minería de datos y procesamiento del lenguaje natural. Seguido por otros conceptos que hacen parte de este trabajo investigativo como lo son ingeniería de software e ingeniería de requisitos.

3.1. MINERÍA DE DATOS

De acuerdo con Fayyad y otros (1996), la noción de encontrar patrones útiles en los datos ha recibido una variedad de nombres, que incluyen minería de datos, extracción de conocimiento, descubrimiento de información, recolección de información, arqueología de datos y procesamiento de patrones de datos. Según los autores, el término minería de datos ha sido utilizado principalmente por los estadísticos, los analistas de datos y las comunidades de administración de sistemas de información. El termino *Knowledge Discovery in Databases* (KDD) se acuñó en el primer taller de KDD en 1989 para enfatizar que el conocimiento es el producto final de un descubrimiento basado en datos, dicho término se ha popularizado en los campos de inteligencia artificial y de aprendizaje automático.

Según Gorunescu (2011), minería de datos se refiere a extraer conocimientos de grandes cantidades de datos. Según este autor el término de minería de datos es incorrecto, ya que la extracción de oro de las rocas o la arena se conoce como minería de oro en vez de minería de roca o de arena. El autor citado anteriormente sugiere, que la minería de datos debería ser llamada "minería del conocimiento a partir de datos", que lamentablemente es un poco largo. La minería de datos hace alusión a un pequeño conjunto de piedras preciosas, provenientes de una gran cantidad de materia prima. Esto se refiere al proceso de extracción de datos relevantes de grandes cantidades de datos. Muchos otros términos tienen un significado similar o diferente a la minería de datos, como lo son: extracción de conocimiento a partir de datos, extracción de conocimiento, análisis de datos/patrones, arqueología de datos y dragado de datos. De acuerdo con Jiawei y otros (2012), muchos investigadores tratan la minería de datos como sinónimo de otro término comúnmente usado como lo es el KDD por sus siglas en ingles. Mientras que otros ven la minería de datos como un simple paso esencial en el proceso de descubrimiento de conocimiento. Este proceso es definido según los autores citados como una secuencia de iteraciones, la cual está estructurada por los siguientes pasos:

Limpiar datos:
En este paso se remueven datos que generan inconsistencias en el proceso.

Integración de datos:
En este paso múltiples fuentes de datos pueden combinarse.

Selección de datos:
En este paso se obtiene los datos relevantes desde una base de datos de información para un posterior análisis.

Transformación de la data:
En este paso la data es transformada y consolidada en un formato conveniente, para hacer minería de datos mediante operaciones de agregación.

Minería de datos:
En este paso se lleva a cabo un proceso de *Maching Learning*, haciendo uso de métodos inteligentes en miras de extraer patrones de datos.

Evaluación del patrón:
En esta instancia se identifican patrones relevantes, que representan conocimiento valioso basado en las métricas de interés.

Presentación del conocimiento:
Este es el paso final y tiene como objetivo hacer uso de técnicas de representación y visualización, para mostrar al usuario final el resultado del conocimiento obtenido.

Como se puede evidenciar en los pasos descritos anteriormente, la minería de datos se sitúa como un paso en el proceso de descubrimiento de conocimiento. Sin embargo, la minería de datos es esencial al momento de identificar patrones ocultos para la evaluación según los autores. Sin embargo, en la industria, en los medios y en el ámbito de investigación, el término minería de datos con frecuencia se usa para referirse a todo proceso enfocado en hallar nuevo conocimiento (Jiawei et al., 2012). Por lo tanto, una definición acotada de minería de datos podría ser: la minería de datos es el proceso de descubrir patrones y conocimiento interesante a partir de grandes cantidades de datos. Estos datos pueden surgir de diversos orígenes como bases de datos, repositorios de datos, la web u otras fuentes de información que se transmiten dinámicamente (Jiawei et al., 2012).

Autores como Das y otros (2011), definen minería de datos como una rama de la informática encargada de identificar patrones de grandes conjuntos de datos, combinando métodos estadísticos e inteligencia artificial con la gestión de bases de datos. Por consiguiente, la minería de datos es vista como una herramienta cada vez más importante por los negocios modernos, para transformar los datos en inteligencia de negocios dando una ventaja informativa.

3.2. PROCESAMIENTO DE LENGUAJE NATURAL

Otra técnica de Inteligencia Artificial analizada en esta propuesta es el Procesamiento de Lenguaje Natural (PLN), debido a que mucha información de gran relevancia se encuentra en documentos escritos en lenguaje natural. De acuerdo con Henderson (2003), PLN es un campo multidisciplinario que se basa en la lingüística y la informática, particularmente en la inteligencia artificial. En términos de lingüística, un programa debe ser capaz de tratar con palabras que tienen múltiples significados ("dar cuerda al reloj" y "el viento está frío hoy"), así como ambigüedades gramaticales (en la frase "escuela de la niña pequeña") (Henderson, 2003). Según el autor citado, los programas pueden usar varias estrategias para tratar estos problemas, incluido el uso de modelos estadísticos para predecir el significado probable de una frase dada, basada en un "corpus" de texto existente en ese idioma. A pesar de lo formidable que puede ser la tarea de extraer el significado correcto del texto, en realidad es solo el primer

nivel del procesamiento del lenguaje natural. Si un programa debe resumir con éxito o sacar conclusiones sobre un informe de noticias de Corea del Norte, por ejemplo, también debería tener una base de conocimientos de hechos sobre ese país y / o un conjunto de "marcos", sobre cómo interpretar varias situaciones como amenaza, engaño o compromiso (Henderson, 2003).

Hay una variedad de aplicaciones emergentes para PLN (Henderson, 2003), que incluyen las siguientes:

- Interfaces de computadora controladas por voz (como en las cabinas de los aviones).
- Programas que pueden ayudar con la planificación u otras tareas.
- Interacciones más realistas con personajes de juegos controlados por computadora.
- Robots que interactúan con humanos en diversos entornos, como hospitales.
- Análisis automático o resumen de noticias y otros textos.
- Aplicaciones de inteligencia y vigilancia (análisis de comunicación, etc.).
- Minería de datos, creación de perfiles de consumidores y otras aplicaciones de comercio electrónico.
- Mejoras en el motor de búsqueda, como la determinación de la relevancia.

Según Chopra y otros (2013), el PLN es un campo de la informática, la inteligencia artificial y la lingüística, que se enfoca principalmente en las interacciones entre las computadoras y los lenguajes humanos, es decir, los lenguajes naturales. La necesidad de procesar el lenguaje natural también fue evidente, debido a la gran cantidad de información almacenada en lenguaje natural que podría ser accesible a través de computadoras. Esta información se genera constantemente en forma de libros, noticias, informes empresariales y gubernamentales, y documentos científicos, muchos de los cuales están disponibles en línea o en informes. El procesamiento del lenguaje natural es un campo interesante y complejo en el que es necesario desarrollar, evaluar o analizar teorías de representación y razonamiento. De acuerdo con Chopra y otros (2013), todos los problemas de inteligencia artificial surgen en este campo, ya que, resolver "el problema del lenguaje natural" es tan difícil como resolver "el problema de la inteligencia artificial", porque cualquier campo puede ser expresado o puede ser representado en lenguaje natural; de aquí radica la dificultad de extraer información de fuentes organizacionales.

3.3. INGENIERÍA DE SOFTWARE

Autores como Sommerville (2010), definen la ingeniería de software como una disciplina de ingeniería que se ocupa de todos los aspectos de la producción de software desde las primeras etapas de la especificación del sistema hasta el mantenimiento del sistema una vez que se ha puesto en uso. De acuerdo con el autor hay dos frases que componen esta definición:

- Disciplina de ingeniería: Los ingenieros hacen que las cosas funcionen. Aplican teorías, métodos y herramientas donde son apropiados. Sin embargo, los usan de forma selectiva y siempre intentan descubrir soluciones a problemas incluso cuando no hay teorías y métodos aplicables.
- Todos los aspectos de la producción de software: La ingeniería de software no se trata solo de los procesos técnicos de desarrollo de software. También incluye actividades tales como la gestión de proyectos de software y el

desarrollo de herramientas, métodos y teorías para respaldar la producción de software.

La ingeniería de software es críticamente importante para el futuro de la humanidad. Por ello, es importante educar a los ingenieros de software y desarrollar la disciplina aún más para crear sistemas de software más complejos. Por supuesto, todavía hay problemas con los proyectos de software. Sin embargo, no se puede permitir que estos problemas oculten los verdaderos éxitos en ingeniería de software así como los métodos y tecnologías que se han desarrollado (Sommerville, 2010).

Uno de los aportes más grandes de la ingeniería de software según Henderson (2003), ha sido definir y mejorar el proceso mediante el cual se desarrollan los programas. Los pasos generales para desarrollar un programa son:

- Especificación detallada de lo que el programa deberá hacer. Esto puede incluir desarrollar un prototipo y obtener la reacción del usuario.
- Creación de algoritmos de arquitectura de programas adecuados y tipos de datos, objetos u otras estructuras necesarias para implementarlos.
- Codificación: Escritura de las declaraciones de lenguaje de programa que implementan la estructura.
- Verificación y prueba del programa usando datos realistas y pruebas de campo.
- Mantenimiento, o la corrección de errores y la adición de características menores solicitadas.

3.4. INGENIERÍA DE REQUISITOS

Autores como Ninaus y otros (2014), definen la ingeniería de requisitos como una de las fases más críticas del proceso de desarrollo de software. De acuerdo con esta propuesta, los requisitos de baja calidad son una de las principales razones para el fracaso de un proyecto en ingeniería de software.

La educción en la ingeniería de requisitos es la fase que se pretende apoyar a través de un método de extracción de información aplicando técnicas de minería de datos. De acuerdo con Zowghi y Coulin (2005), actualmente hay poca investigación y práctica de educción de requisitos en relación con una definición estándar para la obtención de estos. Según los autores, una parte sustancial de la educción de requisitos se dedica a descubrir, extraer e identificar las necesidades de las partes interesadas. Por ende, la priorización y la negociación de requisitos son parte fundamental en la educción de requisitos, ya que una sobrecarga de grandes cantidades de requisitos es un problema serio que puede ser mitigado mediante esta propuesta investigativa, al extraer información concisa acerca de las necesidades planteadas por los usuarios y los *Stakeholders*.

3.5. CONCEPTOS TRANSVERSALES

A continuación, se brinda la definición de otros términos que a nivel general son nombrados a lo largo de este trabajo investigativo.

Técnica:
De acuerdo con Işman (2012), el termino Técnica hace referencia a tener conocimientos especiales y generalmente prácticos, de un tema particular ya sea mecánico o científico. Esto indica que el sentido más general de la palabra técnica es tener algún conocimiento propio sobre un tema en particular. En otras palabras, el conocimiento técnico está "puntualmente" pero no necesariamente asociado con temas mecánicos o científicos.

Método:
Según Kosterec (2016), método abarca un conjunto de instrucciones que conducen a un objetivo específico. Sobre la base de esta definición, cualquier método que conduzca a un objetivo científicamente comprobado, puede considerarse científico. Normalmente, el uso de un método está motivado por algún tipo de problema formulado sobre la base de un fondo teórico y fáctico. Esto indica que un método siempre está expuesto a cambios que modifiquen su implementación. Es decir, un método se modifica cada vez que se obtienen nuevos conocimientos teóricos o fácticos siguiendo las instrucciones del método dado.

Algoritmo:
De acuerdo con Seaver (2019), el término algoritmo se refiere a cualquier procedimiento computacional bien definido que toma algún valor, o conjunto de valores, como entrada y produce algún valor, o conjunto de valores, como salida. Un algoritmo es, por lo tanto, una secuencia de pasos computacionales que transforman la entrada en la salida. Seaver (2019) sugiere que los algoritmos deberían considerarse como el hardware de una computadora, es decir como una tecnología.

Requisito funcional – RF:
Los RF son declaraciones de servicios que el sistema debe proporcionar, cómo debe reaccionar el sistema a insumos particulares y cómo debe comportarse el sistema en situaciones particulares. En algunos casos, los RF también pueden indicar explícitamente lo que el sistema no debe hacer (Sommerville, 2010).

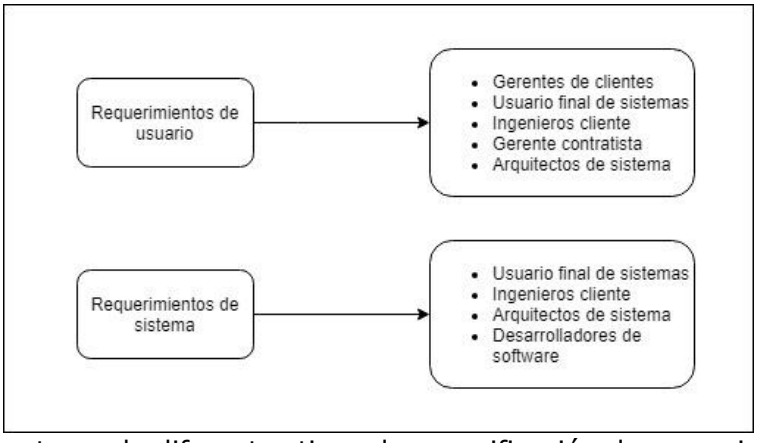


Figura 1. Lectores de diferentes tipos de especificación de requerimientos (Sommerville, 2010).

Requisito no funcional – RNF:

Según Cleland-Huang y otros (2006), los RNF describen restricciones importantes sobre el desarrollo y el comportamiento de un sistema de software y se clasifican en categorías como seguridad, rendimiento, disponibilidad, extensibilidad y portabilidad. Estas cualidades cumplen un papel fundamental en el diseño arquitectónico en la ingeniería de software. Por ello, es fundamental considerarlos y especificarlos lo antes posible durante la fase de educción de requisitos.

De acuerdo con Yin y Jin (2012), los RNF se clasifican en un grupo de categorías, las que a su vez encapsulan características. La siguiente imagen evidencia en detalle lo anteriormente descrito.

Número	Características	RNF
1	**Transmisión de red**	Exactitud
		Tiempo de respuesta
2	**Datos virtuales**	Exactitud
		Seguridad
3	**Datos masivos**	Rendimiento
4	**Información personal**	Seguridad
		Privacidad
5	**Acceso simultáneo**	Tiempo de respuesta
		Carga
6	**Respuesta instantánea**	Tiempo de respuesta
7	**Experiencia de usuario**	Internacionalización
		Robustez
		Usabilidad
8	**Estético**	Estético
9	**Trabajo continuo**	Carga
		Confiabilidad
		Disponibilidad
10	**Equipo peligroso**	Restricción operacional
		Seguridad
11	**Servicio atómico**	Atomicidad
12	**Ambientes distribuidos**	Seguridad
		Compatibilidad
		Consistencia

Figura 2. Relación entre características y RNF (Yin & Jin, 2012)

StopWords:
Con el fin de reducir la dimensionalidad en los textos, no todas las palabras de un idioma determinado deben incluirse en el contenido. En el idioma inglés, es común eliminar palabras tales como, *a, of, and, to*, y otros artículos que probablemente no contribuyan a la comprensión semántica. Estas palabras comúnmente son llamadas *StopWords*. El listado de estas palabras, están disponibles en varios idiomas para automatizar la identificación de las mismas (Gessler & Shrivastava, 2015).

POS Tagging:
De acuerdo con Gessler y Shrivastava (2015), el objetivo del *POS Tagging* es construir un modelo cuya entrada sea una oración y cuya salida es una secuencia de etiquetas donde cada una marca el POS (*part-of-speech*) para la palabra correspondiente, ejemplo: etiqueta <pronombre></pronombre>, <verbo></verbo>, <adjetivo></adjetivo>. Por lo tanto, las cuatro palabras se asignan a pronombre (personal), verbo (tiempo pasado), determinante y sustantivo (singular), respectivamente.

Los ejes conceptuales anteriormente descritos se convierten en el marco teórico- referencial de esta propuesta investigativa, como un punto de partida en cuanto a términos y descripciones conceptuales de cada una de las áreas involucradas.

3.6. METODOLOGÍA DE INVESTIGACIÓN

El diseño metodológico utilizado es el propuesto por Hevner y otros (2004). El cual tiene como objetivo describir el desempeño de la Ciencia Basada en el Diseño de sistemas de información, a través de un marco conceptual con directrices claras para la compresión, ejecución y evaluación de la propuesta. En las siguientes fases se estructuran las actividades a realizar para llevar a cabo esta propuesta investigativa:

3.6.1. Fase 1. Relevancia

- Identificación de propuestas de investigación que involucren problemáticas en la extracción de información.
- Clasificación de las problemáticas halladas.
- Selección de problemáticas a tratar con la propuesta investigativa.

3.6.2. Fase 2. Rigor

- Identificación de propuestas de solución que involucren técnicas de minería de datos.
- Identificación de técnicas de minería de datos.
- Identificación de algoritmos a utilizar en las técnicas de minería de datos.
- Recopilación de propuestas de solución fundamentadas en términos de esfuerzo, costo y calidad.
- Comparación de propuestas en términos de esfuerzo, costo y calidad.

3.6.3. Fase 3. Diseño y validación

- Identificación de la o las técnicas de minería de datos a utilizar.
- Recopilación de los lenguajes de programación más utilizados en minería de datos.
- Clasificación de los lenguajes de programación por simplicidad. Es decir, que su implementación y uso sea sencillo y óptimo a la vez.
- Recopilación de las herramientas *open source* de minería de datos a utilizar.
- Clasificación de las herramientas *open source* de minería de datos a utilizar.
- Identificación y selección los tipos de fuentes de datos organizacionales.
- Definición del formato de representación del resultado de educción de requisitos (formato de documento de requisito).
- Estructuración del experimento de prueba, para la priorización del tipo de técnica, método y algoritmo a utilizar en método propuesto.
- Ejecución del experimento de prueba planteado.
- Identificación de patrones a partir de los resultados generados sobre fuentes organizacionales.
- Priorización de los tipos de métodos, técnicas y algoritmos con base en el resultado obtenido en la ejecución del experimento.
- Definición de las fases y actividades del método de extracción de información propuesto, a partir del tipo de método, técnica y algoritmo priorizados. Dichas fases y actividades están relacionas con procesos de entrenamiento, procesamiento y obtención del resultado.
- Recopilación de documentos escritos en lenguaje natural desde diferentes sectores empresariales.
- Identificación de los profesionales de ingeniería de requisitos en términos de experiencia y conocimiento del dominio (negocio).
- Identificación de métricas a utilizar para medir los resultados en términos de esfuerzo, costo y calidad.
- Distribución de profesionales de ingeniería de requisitos en dos grupos (primer grupo realiza extracción de información de forma manual, segundo grupo utiliza nueva propuesta de solución semi-automatizada).
- Comparación y evaluación de los resultados obtenidos en el proceso de educción de requisitos en términos de esfuerzo, costo y calidad, para los procesos manual y semi-automatizado.
- Recopilación de los aportes realizados por la propuesta.
- Identificación del trabajo a futuro.

Muchas de las actividades descritas en el presente diseño metodológico, no fueron especificadas explícitamente. Es decir, que muchas de estas actividades se encuentran de manera implícita dentro de los capítulos y secciones de esta propuesta investigativa.

CAPÍTULO 4. REVISIÓN DE LITERATURA

4.1. ESTADO DEL ARTE

La construcción de sistemas de software que soporten las necesidades de las organizaciones es una labor demandante que exige día a día acciones de mejora, a través de la implementación de técnicas como las de inteligencia artificial. Estas técnicas han sido aplicadas para aportar hacia la automatización de actividades en la fase de educción de requisitos, apoyando así la labor de los profesionales de ingeniería de requisitos al momento de extraer información valiosa desde diferentes fuentes de datos organizacionales, las cuales van desde documentos organizacionales como documentos de requisitos y textos sin estructura, hasta la experticia humana.

La labor de los profesionales de ingeniería de requisitos en la fase de educción de requisitos se centra primordialmente en la identificación y estructuración de los RF y RNF. Estos últimos son los de mayor relevancia al diseñar las bases de arquitectura de un producto de software de acuerdo con Kurtanovic y Maalej (2017). Por ende, en campos de *Machine Learning* como lo es la minería de datos, cobra tanto valor en el proceso de clasificación de los RNF (Kurtanovic & Maalej, 2017). Este proceso se basa en los diferentes tipos de RNF expresados a través de atributos de calidad como son: usabilidad, rendimiento, seguridad, confiabilidad y sus características relacionadas.

En la actualidad, han surgido diversos enfoques que proponen implementar técnicas, algoritmos y métodos de minería de datos y procesamiento de lenguaje natural para apoyar actividades de clasificación de los RNF en la fase de educción, disminuyendo costo, esfuerzo y manteniendo o mejorando la calidad del resultado obtenido. Por tanto, en la presente propuesta investigativa se formularon preguntas de investigación relacionadas con los dominios anteriormente descritos, las cuales son:

- ¿Qué tipo de técnica, algoritmo y método de minería de datos podría utilizarse para diseñar un método de clasificación de RNF que apoye la fase de educción de requisitos?
- ¿Un método de clasificación de RNF basado en minería de datos, puede semi-automatizar la labor manual realizada por los analistas de requisitos en la fase de educción de requisitos?
- ¿La minería de datos contribuye a la disminución de esfuerzo, costo, y asegura la calidad de los resultados obtenidos en la fase de educción de requisitos?

Las preguntas de investigación anteriormente nombradas, son resueltas en el desarrollo de esta propuesta investigativa.

La búsqueda de los artículos primarios se llevó a cabo mediante las siguientes cadenas de búsqueda y bases de datos documentales especificadas en la **Tabla 1** y **Tabla 2**:

Tabla 1. Cadenas de búsqueda, bases de datos documentales y cantidad de estudios hallados

Cadena de búsqueda	Springer	IEEE	Science Direct	Engineering Village	ACM	Scopus
("text analytics" or "requirements reuse") or ("text analytics" and "functional and non-functional requirements") or ("non-functional requirements patterns") or ("structure of non-functional requirements") or ("NO FUNCTIONAL requirements identification")	37	35	0	0	40	8
("Data Mining" AND ("Requirement" or "engineering")) or ("Data Mining" AND ("extract information"))	16	4	6	2	2	1
("Requirements engineering" AND "artificial intelligence") or ("Requirements engineering" AND " Data Mining") or ("Requirements engineering" AND "extract information")	1	5	1	1	0	0
("Text Mining" AND ("Requirement" or "engineering")) or ("Natural Language Processing" AND "Requirements")	6	3	2	2	0	0

Tomando como punto de partida la problemática previamente definida, se realizó un proceso de revisión literaria considerando los siguientes criterios de inclusión y exclusión, para la selección de los artículos primarios que componen el presente estado del arte:

Criterios de inclusión:

- Los artículos seleccionados deben ser estudios primarios.
- Los artículos primarios seleccionados deben datar entre los años 2010 y 2019.
- El resultado de las soluciones planteadas en los artículos debe apoyar el proceso de educción de requisitos.
- Los artículos deben especificar la técnica, algoritmo y método utilizados para brindar solución a la problemática planteada.
- Las propuestas de solución planteadas en los artículos deben hacer uso de técnicas de minería de datos, para la extracción de información desde documentos de organizacionales como documentos de requisitos y textos sin estructura, hasta la experticia humana.

Criterios de exclusión:

- Los artículos de estudio secundarios deben ser excluidos.
- Los artículos cuya solución planteada sea diferente a las temáticas de minería de datos y educción de requisitos deben ser descartados.
- Los artículos cuya solución sea diferente a la extracción de información mediante técnicas, métodos y algoritmos de minería de datos deben ser descartados.

Tabla 2. Cadenas de búsqueda, bases de datos documentales y cantidad de estudios primarios seleccionados

Cadena de búsqueda	Springer	IEEE	Science Direct	Engineering Village	ACM	Scopus
("text analytics" or "requirements reuse") or ("text analytics" and "functional and non-functional requirements") or ("non-functional requirements patterns") or ("structure of non-functional requirements") or ("NO FUNCTIONAL requirements identification")	8	5	0	0	0	2
("Data Mining" AND ("Requirement" or "engineering")) or ("Data Mining" AND ("extract information"))	10	1	2	2	1	0
("Requirements engineering" AND "artificial intelligence") or ("Requirements engineering" AND " Data Mining") or ("Requirements engineering" AND "extract information")	1	4	0	0	0	0
("Text Mining" AND ("Requirement" or "engineering")) or ("Natural Language Processing" AND "Requirements")	3	1	2	2	0	0

Adicionalmente, algunos artículos fueron seleccionados desde las citas de los artículos primarios identificados.

18 estudios primarios fueron seleccionados de la **Tabla 2** para elaborar el estado del arte del presente proyecto investigativo; estos estudios se hallan agrupados por categorías. En la **Tabla 3** se presentan los estudios seleccionados agrupados por categorías y se describen a continuación.

Tabla 3. Estudios primarios agrupados por categoría

Artículos/Categorías	Extracción de información con Minería de Datos	Extracción de RF/RNF con técnicas de Minería de Datos	Extracción de información con otras técnicas
(Raharja & Siahaan, 2019)		x	
(Alzu'Bi et al., 2018)		x	
(Aysolmaz et al., 2017)			x
(Kurtanovic & Maalej, 2017)		x	
(Mahmoud & Williams, 2016)		x	
(Minku, Mendes, & Turhan, 2016)	x		
(Liang et al., 2015)	x		
(Arora, 2014)	x		
(Landhäußer et al., 2014)			x
(Génova et al., 2013)			x
(Naz et al., 2013)	x		
(Slankas & Williams, 2013)		x	
(Das et al., 2011)	x		
(Zhang et al., 2011)		x	
(Casamayor et al., 2010)		x	
(Xie et al., 2009)	x		
(Cleland-Huang et al., 2006)		x	
(Alvarez et al., 2004)	x		

4.2. EXTRACCIÓN DE INFORMACIÓN CON MINERÍA DE DATOS

Según Minku y otros (2016), la minería de datos es una técnica de inteligencia artificial que está siendo asumida en los últimos años por los expertos en ingeniería de software, incrementando así su participación en el diseño de modelos que apoyen los procesos de desarrollo de software. Estos autores aseguran que la aceptación incremental de enfoques de minería de datos en el proceso de desarrollo de software facilita la integración del conocimiento obtenido de manera automática de las diferentes fuentes de datos organizacionales, con el conocimiento de los expertos en los procesos de ingeniería de software. Adicionalmente Minku y otros (2016), afirman que involucrar el rol del experto en ingeniería de software en la minería de datos no solo puede mejorar la aceptación de esta técnica por la ingeniería de software en la práctica, sino que también es posible mejorar los modelos de datos resultantes. Por ende, el conocimiento adquirido de los modelos datos podría ayudar a superar los posibles errores cometidos por expertos en ingeniería de software y proporcionar información útil.

De acuerdo con Xie y otros (2009), para mejorar la productividad y la calidad del software, los ingenieros de software aplican cada vez más algoritmos de minería de datos a diversas tareas de ingeniería de software, debido a que el software desempeña un papel fundamental en las organizaciones, los gobiernos y las sociedades. Por ende, mejorar la productividad y la calidad del software es un objetivo importante de la ingeniería del software. Según los autores, la minería de datos en la ingeniería de software ha surgido recientemente como un medio prometedor para cumplir este objetivo debido a dos tendencias principales: la creciente abundancia de datos y su utilidad demostrada para resolver inconvenientes. Aunque los autores Xie y otros (2009) no especifican cuales algoritmos de minería de datos pueden apoyar las tareas del ciclo de vida de la ingeniería de software. Si está claro las posibles fuentes de datos, los tipos de algoritmos de minería de datos y las tareas a apoyar dentro ciclo de vida de ingeniería de software como se muestra en la **Tabla 4**:

Tabla 4. Datos de ingeniería de software, algoritmos de minería y tareas de ingeniería de software (Xie et al., 2009).

Datos de ingeniería de software	**Algoritmos de ingeniería de datos**	**Tareas ingeniería de software**
Secuencias: ejecución / rastros estáticos, co-cambios	Conjunto de elementos frecuentes / secuencia / minería de orden parcial, coincidencia de secuencia / agrupación / clasificación	Programación, mantenimiento, detección de errores, depuración
Gráficos: gráficos de llamadas dinámicas / estáticas, gráficos de dependencia del programa	Minería de subgrafos frecuente, correspondencia de gráficos / agrupación / clasificación	Detección de errores, depuración
Text: bug reports, e-mails, code comments, documentation	Texto coincidente / agrupamiento / clasificación	Mantenimiento, detección de errores, depuración

Según los autores citados anteriormente, los ingenieros de software pueden optimizar la productividad y calidad de las tareas en los proyectos de software a través de la minería de datos. En la actualidad herramientas para el desarrollo de software como Eclipse y Microsoft Visual Studio implementan dentro de sus *frameworks* librerías que facilitan la implementación de algoritmos de minería de datos. En este proceso de implementación se ha logrado evidenciar que los algoritmos clasificadores son más precisos para las tareas de ingeniería de software al momento de identificar patrones y coincidencias en las diversas fuentes de datos (Xie et al., 2009).

En el enfoque propuesto por Naz y otros (2013), se comenta que en el mundo de ingeniería de software existen diferentes enfoques de acuerdo con el contexto en el cual se esté desarrollando una solución de software; este es el caso de la ingeniería web, que busca a través de los principios de la ingeniería moderna y la ciencia, obtener desarrollos exitosos, mejorando la calidad, minimizando los riesgos de escalabilidad y asegurando el mantenimiento de las aplicaciones web. Para lograr esto, es necesario la reutilización del conocimiento previo del sistema a través de técnicas como *Case Based Reasoning*. Según los autores, dicha técnica consiste en cuatro pasos los cuales son: recuperar casos similares, reutilizar la solución del caso anterior, revisar la solución y retener el caso que se puede utilizar en el futuro. Gracias a esto, es posible mejorar la deficiencia de las prácticas web utilizadas actualmente por los desarrolladores web.

De acuerdo con Das y otros (2011), los datos del proceso de ingeniería de software (como bases de código, trazas de ejecución, cambios de código históricos, listas de correo y bases de datos de errores) contienen gran cantidad de información sobre el estado, el progreso y la evolución de un

proyecto; sin embargo, muchos de éstos no aportan a su proceso de desarrollo. Según los autores, esto se logra al utilizar técnicas de minería de datos para explorar el potencial de éstos, en pro de un mejor procesamiento de las etapas del proceso de desarrollo de software para construir sistemas de alta calidad, entregados a tiempo y bajo el presupuesto establecido. Para lograrlo, es necesario implementar algoritmos de minería de datos, los cuales se encuentran distribuidos en las siguientes categorías:

- La minería de patrones frecuentes: Se encuentran patrones que ocurren comúnmente.
- Coincidencia de patrones: Encontrar instancias de datos para patrones dados.
- *Clustering*: Agrupar datos en *clusters*.
- Clasificación: Etiquetas de predicción de datos basadas en datos ya etiquetados.

Gracias a esto, es posible depurar los datos para obtener información precisa que apoye los procesos de ingeniería de software.

Según Arora (2014), la información histórica de las organizaciones aporta a los procesos de gerencia en la construcción de sistemas de software confiables, aumentando la predicción de bugs y disminuyendo esfuerzos. Por esta razón, los ingenieros de software están aplicando cada vez más algoritmos de minería de datos a las tareas de ingeniería de software. Estos algoritmos pueden mejorar la productividad y la calidad del software en términos de escalabilidad y mantenimiento, es posible determinar qué segmentos de código deben cambiarse cuando otros segmentos de código son modificados o cambiados de ubicación dentro de un sistema de software, aportando así a la detección temprana de líneas de código erradas que pueden causar futuros fallos en el rendimiento del software.

Según los autores Alvarez y Riquelme (2004), la simulación de procesos de desarrollo de software a través de modelos dinámicos ha logrado avances significativos en la gestión de proyectos de desarrollo de software, ya que emulan los posibles riesgos y resultados de un proyecto antes de emprenderlo. De esta manera, el gerente de proyecto decide los valores de los atributos y simula el proceso para verificar su comportamiento. No obstante, la cantidad de posibilidades existentes no permite una verificación exhaustiva del proceso. Por ende, la tarea principal de la minería de datos en este caso es brindar al gerente de proyectos un conocimiento adicional para estimar cuales son los valores adecuados que deben llevar las variables en cuanto a tiempo, costo y calidad de un proyecto. Gracias a esto, el gerente de proyecto cuenta con información clave en caso de iniciar un nuevo proyecto, hacerle seguimiento a un proyecto existente y/o analizar un proyecto ya terminado. Se evidencia así la efectividad de las técnicas de minería de datos en modelos emulados para un proyecto de desarrollo de software.

En el planteamiento realizado por Liang y otros (2015), las aplicaciones móviles en el mundo de los sistemas de software son constantemente actualizadas debido a los requisitos generados por los usuarios, obligando a sus fabricantes a liberar actualizaciones continuas con el único objetivo de satisfacer sus necesidades. En este punto, los autores plantean la eficiencia de las técnicas de minería al momento de obtener requisitos de usuario. Esta propuesta busca identificar tendencias en comportamientos, gustos y costumbres que denotan

los usuarios al hacer uso de sus dispositivos móviles, utilizando el algoritmo *Apriori-M* para extraer los requisitos de los usuarios a partir de los datos móviles aportados por la multitud. Posteriormente, los datos se clasifican con respecto al tiempo, la ubicación y el estado de movimientos instantáneos del usuario. Para terminar, se identifican los patrones de comportamiento del usuario a partir de los datos obtenidos con el algoritmo, el cual hace parte de un nuevo método no guiado que automáticamente encuentra frecuencias de patrones en los datos, e indica hábitos de usuarios entre la multitud. Así se logra una actualización constante de los requisitos de los usuarios y conlleva a la sostenibilidad de los sistemas de software en los dispositivos móviles.

En la **Tabla 5** se presenta una abstracción para esta sección del total de estudios primarios especificados en la **Tabla 3**. Cinco de siete propuestas hacen uso de técnicas de minería de datos en pro de extraer información relevante de fuentes datos organizacionales, de procesos de ingeniería de software e incluso utilizan información de antecedentes de proyectos de software, en aras de emular el desarrollo de un proyecto de principio a fin, y/o optimizar la ejecución de un proyecto actual en términos de esfuerzo, costo y calidad. Sin embargo, cinco de siete propuestas, no proponen una nueva herramienta de minería de datos, y la propuesta que lo hace, reutiliza herramientas implementadas por otros estudios. Dejando así un gran campo de investigación para nuevos estudios que incluyan herramientas nuevas, basadas en técnicas de minería de datos que apoyen el proceso de desarrollo software desde la educción de requisitos.

Tabla 5. Herramientas y técnicas de minería de datos

Referencia	**Herramienta**	**Técnicas de Minería de Datos**	**Procesa documentos en lenguaje natural**
(Minku et al., 2016)	No especificada	No especificada	No
(Naz et al., 2013)	No especificada	*Case Based Reasoning (Text mining)*	Si
(Das et al., 2011)	*Alitheia Core Tool*	• Muestreo frecuente de patrones; • búsqueda de concordancia de instancias de datos; • Agrupación de datos; • Clasificación-predicción de etiquetas de datos basadas en datos ya etiquetados.	Si
(Arora, 2014)	No especificada	• *Association rules and frequent patterns* • *Clasificación* • *Clustering*	Si
(Alvarez et al., 2004)	• Simulación SDP • GAR • ELLIPSES	No especificadas	No
(Liang et al., 2015)	No especificada	Algoritmo AprioriM	No
(Xie et al., 2009)	No especificada	*frequent itemset/sequence/ partial-order mining, sequence matching/ clustering/classification frequent subgraph mining, graph matching/clustering/classification Text matching/clustering/ classification*	No

De los trabajos priorizados que aparecen en la **Tabla 5**, solo tres procesan documentos escritos en lenguaje natural como punto de partida, para extraer información relevante que apoye los procesos de ingeniería de software. Por lo anterior, es posible evidenciar la necesidad de nuevos enfoques que procesen documentos escritos en lenguaje natural, con la finalidad de extraer información relevante que apoye procesos de ingeniería de software, específicamente en la fase de educción de requisitos. Adicionalmente, seis de siete trabajos priorizados en la **Tabla 5** abordan soluciones a problemas

relacionados con el proceso de ingeniería de software a nivel general; solo el trabajo de Liang y otros (2015), se enfoca en la educción de requisitos a través del algoritmo *AprioriM*. Por ende, son necesarios enfoques basados en mejoras y optimizaciones a procesos específicos, como es el caso del presente trabajo investigativo, el cual se centra en la fase de educción de requisitos mediante una nueva herramienta llamada Clasificador RNF, el cual es un sistema desarrollado en *Python* que aporta al proceso de identificación y clasificación de RNF en la ingeniería de requisitos.

4.3. EXTRACCIÓN DE RNF CON TÉCNICAS DE MINERÍA DE DATOS

En la fase de educción de requisitos es claro que el foco se encuentra en la identificación y estructuración de los RF y RNF. Estos últimos toman mayor importancia, como se evidencia en los siguientes enfoques basados en la clasificación de RNF.

De acuerdo con Mahmoud y Williams (2016), los enfoques no supervisados han demostrado su efectividad para detectar, clasificar y rastrear RNF. En muchas ocasiones los RNF tienden a ser implementados implícitamente por los RF del sistema de software. Los autores afirman que algoritmos de *clustering* como el *average linkage*, son más efectivos en la generación de grupos temáticos de palabras de RF que los algoritmos de partición (*k-medoids*).

Cleland-Huang y otros (2006) proponen un nuevo enfoque basado en métodos de recuperación de información para detectar y clasificar RNF de especificaciones de requisitos estructurados, así como el texto de forma libre. Aunque se requiere que un analista evalúe la exactitud de los RNF identificados, es menos esfuerzo que los métodos manuales utilizados para la clasificación de los RNF. Poder clasificar los RNF de esta manera es útil, según estos autores, para los ingenieros de requisitos en la fase de educción, a medida que diseñan, construyen y analizan un sistema de software. Esto puede ayudar a entender las necesidades de las partes interesadas y ver puntos de vista coherentes de varias restricciones del sistema, así se ahorran costos en los que se habría incurrido en esfuerzos de refactorización posteriores.

El enfoque de Zhang y otros (2011), propone un estudio empírico para automatizar la clasificación de los RNF. La descripción textual de los RNF se representa utilizando diferentes tipos de *index terms* como *n-grams*, *individual words* y MWE (*multi-word expressions*), respectivamente. Luego, se transfiere la descripción textual de cada RNF a vectores numéricos en diferentes espacios de características definidas para cada tipo de *index terms*. Por último, utiliza el algoritmo *Support Vector Machine* (SVM) para llevar a cabo la clasificación de los RNF. De acuerdo con estos autores, los resultados demuestran que la utilización del *index term* llamado *invidual words*, presenta el mejor rendimiento al clasificar los RNF automáticamente. De acuerdo con Zhang y otros (2011), a mayor cantidad de muestras en el conjunto de datos, mayor será el rendimiento que el clasificador automático de RNF producirá.

El enfoque anteriormente descrito demuestra la efectividad que puede tener el algoritmo SVM en la clasificación de RNF en documentos de requisitos y la importancia de categorizar los RNF al momento de diseñar el sistema.

Como se ha evidenciado en múltiples enfoques y propuestas de investigación, el objetivo común, es apoyar la labor manual realizada por los ingenieros de requisitos en la fase de educción al momento de identificar y clasificar los RNF. Este es el caso de Slankas y Williams (2013), quienes implementaron un enfoque basado en herramientas llamado *NFR Locator*. Dicha herramienta tiene como finalidad clasificar y extraer oraciones en textos de lenguaje natural existentes en las categorías de RNF. *NFR Locator* es una herramienta para ayudar a los ingenieros de requisitos a extraer los RNF relevantes de una amplia variedad de documentos. Al analizar categorías específicas de RNF, se pueden identificar características específicas únicas para cada categoría. Las características pueden ser usadas por los programas para identificar otros RNF en la misma categoría, ampliando así la posibilidad de captura e identificación de RNF.

La detección y clasificación de los RNF es posible realizarse mediante la implementación de enfoques semi-supervisados, apoyados en algoritmos de clasificación como *Expectation Maximization* (EM) según Casamayor y otros (2010). En este enfoque, la clasificación se basa en un número reducido de requisitos previamente categorizados, aprovechando el conocimiento proporcionado por los requisitos no categorizados, así como propiedades de texto. El método de aprendizaje de este enfoque también explota los comentarios realizados por los usuarios para mejorar el rendimiento de la clasificación (Casamayor et al., 2010). En este enfoque, se evidenció que la implementación de este algoritmo de clasificación requiere menos esfuerzo humano en el proceso de identificación y clasificación de los RNF que los métodos totalmente supervisados, y se puede mejorar aún más en función de los comentarios proporcionados por los analistas.

El aporte de Kurtanovic y Maalej (2017) se orienta hacia la precisión con la cual se pueden clasificar automáticamente los requisitos RNF mediante el aprendizaje automático supervisado. Con *Support Vector Machine* y utilizando características léxicas, se alcanza según los autores una recuperación y precisión del 92% para ambas clases. Adicionalmente, se evaluó el clasificador binario y multiclase de RNF para identificar los RNF de usabilidad, seguridad, funcionamiento y rendimiento. Se logra así una precisión del 93% y una recuperación del 90%. El objetivo principal fue evaluar si un conjunto de datos adicional derivado de los comentarios de los usuarios puede ayudar a la identificación y estructuración de los RNF.

Los autores Raharja y Siahaan (2019), basan su aporte en brindar una posible solución al problema de identificación y clasificación de RNF en la educción de requisitos. Por esto razón, un método de clasificación de RNF es necesario para facilitar a los desarrolladores de software la clasificación de RNF en los documentos de requisitos. Por consiguiente, los autores Raharja y Siahaan (2019), proponen un modelo para clasificar los RNF en atributos de calidad basado en la norma ISO/IEE 25010, utilizando *Fuzzy Similarity KNN* (FSKNN) para construir el modelo de clasificación basado en frases. El proceso de entrenamiento en este algoritmo utiliza un enfoque de medida de similitud difusa para calcular la similitud entre los términos, documentos y categorías de atributos de calidad para obtener un patrón de entrenamiento, luego el patrón de entrenamiento se agrupa en ciertos valores de umbral. En la prueba realizada con seis diferentes sets de datos que contienen 1432 frases, los mejores resultados de clasificación medidos en exactitud, precisión y valor

usando FSKNN fueron 42,8%, 68,1% y 55,9% respectivamente. Demostrando así la efectividad del algoritmo FSKNN en el modelo planteado.

El advenimiento de nuevos métodos de extracción de datos, trae consigo muchas oportunidades de mejora en la fase de educción de requisitos, una de estas mejoras es el uso de sistemas de recomendación. Los cuales en la ingeniería de requisitos pueden ser de gran utilidad para los ingenieros de requisitos, al proporcionar información adecuada en el momento oportuno. Por tanto, autores como Alzu'Bi y otros(2018), proponen un sistema de recomendación para las necesidades de los usuarios llamado *Recommendation systems in software engineering* (RSSEs). RSSEs utiliza el algoritmo *Apriori* para extraer reglas de los requisitos de los usuarios. Estas reglas pueden utilizarse para sugerir nuevos requisitos a los usuarios. Según los autores Alzu'Bi y otros(2018), el objetivo de este enfoque es aumentar la precisión y la completitud del proceso de ingeniería de requisitos mediante el uso de RSSEs. Dicho sistema fue puesto a prueba con un conjunto de datos conformado por 4000 registros que contienen RF y RNF; en el resultado de esta prueba se demostró la eficiencia del RSSEs en términos de tiempo de ejecución, ya que solo tardo de 11 a 21 segundos para procesar 4000 registros. Siendo capaz de extraer reglas de requisitos muy rápidamente, apoyando así la labor manual de los ingenieros de requisitos en la fase de educción.

En la **Tabla 6** se presenta la síntesis de los métodos, técnicas y algoritmos identificados para esta categoría. Se puede evidenciar que cuatro de ocho trabajos implementaron métodos en sus propuestas. En el caso de la técnica, seis de ocho enfoques optan por la clasificación en sus soluciones propuestas. Dando a entender, que se podrían obtener mejores resultados mediante el uso de técnicas y algoritmos de clasificación al momento de identificar RNF y RF. A nivel de algoritmos SVM, NB y KNN son los más utilizados en este sondeo para la clasificación e identificación de RNF, en comparación con los tipos algoritmos restantes.

Tabla 6. Método, técnica y Algoritmo de minería de datos.

Referencia	**Método**	**Técnica**	**Algoritmo**
(Mahmoud & Williams, 2016)	NA	*Clustering*	*Average linkage*(AL), *k-medoids*
(Cleland-Huang et al., 2006)	*NFR-Classifier*	Clasificación	Algoritmo propuesto
(Zhang et al., 2011)	NA	Clasificación	*Support Vector Machine*
(Slankas & Williams, 2013)	*NFR Locator*	Clasificación	*Naïve Bayes, Support Vector Machine, k-nearest neighbor classifier (k-NN)*
(Casamayor et al., 2010)	NA	Clasificación	*Expectation Maximization (EM), TF-IDF, k-NN, Naïve Bayes*
(Kurtanovic & Maalej, 2017)	NA	Clasificación	*Support Vector Machine*
(Raharja & Siahaan, 2019)	*Fuzzy Similarity KNN*	Clasificación	*KNN*
(Alzu'Bi et al., 2018)	*RSSEs*	*Clustering*	*AprioriM*

Los trabajos priorizados en la **Tabla 6** abordan múltiples enfoques relacionados con el proceso de extraer, identificar y clasificar RF y RNF; siendo estos últimos los de mayor relevancia en los trabajos priorizados. Diversos enfoques han planteado soluciones basadas en minería de datos, haciendo uso de algoritmos supervisados de clasificación (EM, SVM, KNN, FSKNN) y no supervisados de *clustering* (*Average linkage*(AL), *k-medoids*, *Apriori*). El objetivo común de los enfoques presentados por los trabajos priorizados en la **Tabla 6** es optimizar el

esfuerzo, el costo traducido en tiempo y la calidad en la fase de educción de requisitos. Todos estos enfoques presentan una gran variedad de soluciones la cuales se sintetizan a continuación:

- Los autores Mahmoud y Williams (2016), proponen un enfoque basado en algoritmos de *clustering* como el *average linkage*, para la generación de grupos temáticos de palabras.
- Cleland-Huang y otros (2006), proponen un enfoque basado en un método de recuperación de información para clasificar RNF de documentos de requisitos y de texto en libre formato.
- Zhang y otros (2011), propone un enfoque empírico para automatizar la clasificación de los RNF. Según los autores, a mayor cantidad de datos de prueba, mayor será el rendimiento que el clasificador automático de RNF producirá.
- Slankas y Williams (2013), Proponen un enfoque basado en la herramienta llamada NFR Locator.
- Casamayor y otros (2010), proponen un enfoque semi-supervisado basado en el algoritmo de clasificación *Expectation Maximization* (EM). Este enfoque se apoya en conocimiento proporcionado por los requisitos no categorizados.
- Kurtanovic y Maalej (2017) se orientan hacia la precisión con la cual se pueden clasificar automáticamente los RNF mediante el aprendizaje automático supervisado, apoyándose en los comentarios de los usuarios para la identificación y estructuración de los RNF.
- Raharja y Siahaan (2019), proponen un modelo para clasificar los RNF en atributos de calidad basado en la norma ISO/IEE 25010, utilizando *Fuzzy Similarity KNN* (FSKNN) para construir el modelo de clasificación basado en frases.
- Alzu'Bi y otros(2018), proponen un sistema de recomendación para las necesidades de los usuarios llamado *Recommendation systems in software engineering* (RSSEs). RSSEs utiliza el algoritmo *A priori* como la base lógica de su solución.

Los enfoques y propuestas de solución anteriormente citados y detallados, tienen como objetivo extraer, identificar y clasificar RNF, a través de diferentes medios de solución que incluyen herramientas, algoritmos de minería de datos y modelos de clasificación de RNF que utilizan algoritmos supervisados, no supervisados y semi supervisados. Estas soluciones, utilizan fuentes de datos como documentos de requisitos e información no estructurada (correos electrónicos y comentarios de usuarios). Lo anterior, no dista mucho del método de propuesto en este trabajo investigativo. Sin embargo, la diferencia principal radica en el modelo clasificador generado para el sistema clasificador de RNF; el cual se construye a partir de un corpus de entrenamiento que puede ser enriquecido constantemente en miras de optimizar la identificación y clasificación de RNF. Adicionalmente, este modelo clasificador puede ser importado por otras herramientas de extracción y clasificación de RNF en aras de potenciar el resultado obtenido en la fase de educción de requisitos.

4.4. EXTRACCIÓN DE INFORMACIÓN CON OTRAS TÉCNICAS

De acuerdo a Aysolmaz y otros (2017), los modelos de procesos de negocio aportan información de gran valor al proceso de educción de requisitos, sin embargo, dicha información no es aprovechada en el proceso de ingeniería de requisitos, por consiguiente, los autores proponen un nuevo enfoque semi-automatizado cuyo resultado son documentos de requisitos que integran el modelo de proceso del negocio y los documentos escritos en lenguaje natural de una manera comprensible para su posterior interpretación. En este mismo orden, las organizaciones pueden compartir la información contenida en los modelos de procesos de negocio con actividades del ciclo de vida del desarrollo de software. En este enfoque semi-automático los autores proponen tres pasos: En el primer paso, se analizan los modelos de proceso que son relevantes para el sistema a ser desarrollado, identificando el conjunto de actividades automatizables. En el segundo paso, se capturan los datos relacionados con la ejecución, como responsabilidades, sistemas de aplicación, necesidades de datos y restricciones adicionales en un modelo de requisitos. En el tercer paso, se obtienen automáticamente documentos de requisitos a partir de los modelos creados mediante un algoritmo de generación de lenguaje natural basado en plantillas. Finalmente, es posible disminuir la incoherencia entre los modelos de procesos de negocio y el lenguaje natural en la educción de requisitos.

Según Génova y otros (2013), la mejora de la calidad del software demanda el uso de controles de calidad desde el comienzo mismo del desarrollo, es decir, desde la educción de requisitos. Como consecuencia de esto, los autores proponen un enfoque plasmado en un *framework* que obtiene indicadores de calidad a partir de requisitos textuales, tales como el tamaño del requisito, numero de términos ambiguos, formas verbales imperativas y requisitos superpuestos. Este *framework* ha sido implementado en una herramienta que calcula métricas de calidad de una manera automatizada, llamado *Requirements quality analyzer* (RQA). En esta propuesta, los autores se centraron en el análisis de la calidad de los requisitos, utilizando documentos de requisitos escritos en lenguaje natural como datos de entrada. El propósito de los autores en este enfoque ha sido proporcionar a los profesionales (ingenieros de requisitos, gerentes de calidad, etc.) una herramienta de gran utilidad, que abstrae la complejidad del procesamiento del lenguaje natural y les brinda sugerencias acerca de mejores prácticas.

Actualmente, los ingenieros de requisitos dedican mucho tiempo a mejorar las especificaciones de los requisitos. La detección de defectos antes de que surjan en etapas posteriores de producción de software es de suma importancia. Landhäußer y otros (2014), presentan un enfoque de automatización del proceso de ingeniería de requisitos, el cual hace hincapié en las especificaciones del lenguaje natural y apoya el proceso de producción de software, desde el aseguramiento de la calidad de los requisitos, a través de la generación automática de modelos UML (diagramas de clases, diagramas de estado y diagramas de actividad).

De los enfoques analizados, en la **Tabla 7** se evidencia que una de tres estudios plantea nuevas técnicas de procesamiento del lenguaje natural y a nivel de herramientas, sólo dos propuestas utilizan herramientas de automatización, en miras de apoyar la labor manual de los profesionales de ingeniería de requisitos. Así, se puede deducir que en la actualidad muchas

propuestas utilizan herramientas y técnicas ya implementadas, para obtener resultados en términos de educción y calidad de requisitos, dejando por fuera la posibilidad de proponer nuevos métodos, técnicas y/o herramientas personalizadas que brinden soluciones a dominios de negocio determinados.

Tabla 7. Herramientas y técnicas de procesamiento del lenguaje natural.

Referencia	**Herramienta**	**Técnica procesamiento del lenguaje natural**	**Resultado**
(Aysolmaz et al., 2017)	No especificada	algoritmo de generación de lenguaje natural basado en plantillas.	documentos de requisitos
(Génova et al., 2013)	*Requirements quality analyzer* RQA	No especificada	sugerencias acerca de mejores prácticas
(Landhäußer et al., 2014)	*RESI AUTOANNOTATOR REFS*	*Requirements engineering complete automation approach.*	Automatización de tareas manuales y propensas a errores en desarrollo de software.

Como se evidencia en las propuestas anteriores, el foco es la identificación y clasificación de RNF, los cuales son comúnmente ignorados y descuidados en el proceso de educción de requisitos. De aquí, radica la importancia de apoyar la labor realizada por los ingenieros de requisitos en la fase de educción. Por tanto, se evidencia que estos enfoques buscan apoyar a través de técnicas de minería de datos el proceso de clasificación de RNF mediante métodos automáticos y semi-automáticos utilizando algoritmos de minería de datos supervisados como es el caso *Support Vector Machine*, *Naive Bayes* y *Expectation Maximization*. Así se demuestra que, mediante esta variedad de enfoques y propuestas, es posible apoyar la labor manual realizada por los ingenieros de requisitos, así como disminuir el esfuerzo y desgaste humano en el proceso de identificación y clasificación.

En conclusión, los trabajos anteriormente descritos no pretenden en ningún momento reemplazar la labor manual realizada por los profesionales de ingeniería de requisitos, por el contrario, se pretende apoyar la labor de los profesionales, brindando herramientas semi-automatizadas que sirvan de apoyo en los diferentes procesos de ingeniería de requisitos, con el objetivo común de disminuir costo y esfuerzo; mantener o mejorar la calidad actual de los requisitos detallados en el ciclo de vida de desarrollo de software y de esta manera, obtener sistemas de software confiables, sostenibles y escalables.

PARTE III:

CARACTERIZACIÓN Y CLASIFICACIÓN DE TÉCNICAS POTENCIALES

CAPÍTULO 5. CARACTERIZACIÓN Y CLASIFICACIÓN DE TÉCNICAS POTENCIALES

5.1. CONTEXTO DEL MÉTODO PROPUESTO

La minería de datos como el proceso de extracción de datos, patrones y tendencias útiles desde grandes cantidades de datos, usa técnicas como *clustering,* clasificación*,* asociación y regresión. Actualmente, hay gran variedad de aplicabilidad de dicho proceso en la vida cotidiana. Varias herramientas disponibles en el mercado se encuentran soportadas por múltiples algoritmos de minería de datos (Gera & Goel, 2015).

De acuerdo con Gera y Goel (2015), la minería de datos es parte de un marco más amplio, denominado *knowledge discovery in databases* (KDD) que abarca un proceso complejo desde la preparación de datos hasta el modelado de conocimiento. Las tareas principales de minería de datos son clasificación, agrupación y asociación. La clasificación tiene como objetivo principal asignar cada registro de una base de datos a una de las clases predefinidas. La agrupación funciona encontrando grupos de datos como registros, términos y/o palabras que hacen parte de un contexto común previamente definido. Por último, la asociación define las reglas de implicación sobre la base de un subconjunto de atributos de registro.

El método que se propone en este trabajo de grado plantea llevar a cabo un proceso de extracción, identificación y clasificación de requisitos no funcionales*,* comúnmente reconocidos por sus siglas en español RNF, a través de técnicas de minería de datos, en miras de semi-automatizar la labor manual realizada por los ingenieros de requisitos en la fase de educción, tomando como insumo documentos organizacionales escritos en lenguaje natural, como: documentos de requisitos y textos sin estructura.

En miras de llevar a cabo una explicación más detallada acerca de este método, es importante abordar cada uno de los términos que hacen parte de la solución. Por ello, es relevante tener claro conceptos como técnica, algoritmo y método en el ámbito de la minería de datos.

5.2. TÉCNICAS DE MINERÍA DE DATOS

La minería de datos en la actualidad utiliza herramientas de compilación y procesamiento, para obtener patrones y secuencias útiles de tendencias ocultas. Esto en aras de generar predicciones con la data analizada a través del uso de técnicas de minería de datos (Rehman, 2017). Según Gera y Goel (2015), la minería de datos es el principal paso en KDD, ya que KDD convierte datos de bajo nivel a alto nivel, la minería entrega un resultado de un previo análisis de un gran conjunto de datos, haciendo uso de herramientas ya construidas en miras de identificar patrones, tendencias y predicciones. Dichas herramientas se fundamentan en el uso de las técnicas de clasificación*, clustering y* regresión, que se describen a continuación*.*

5.2.1. Clasificación

De acuerdo con Gera y Goel (2015), clasificación es la técnica de minería de datos más usada, que hace uso de un conjunto de ejemplos preclasificados para desarrollar un modelo que puede clasificar poblaciones de registros en general. Esta técnica es un tipo de aprendizaje automático supervisado en el que se cuenta con insumos de datos previamente etiquetados (Baudier, 2000). Este enfoque emplea con frecuencia "Clasificación por árbol de decisión de inducción" basados en redes neuronales. El proceso de clasificación de datos implica aprendizaje y clasificación según Baudier (2000). Algunos tipos de métodos de clasificación son los siguientes:

- Árbol de decisión de inducción
- Clasificación basada en reglas
- Clasificación por *backpropagation*
- *Lazy learners*
- Redes neuronales
- *Support Vector Machines (SVM)*

5.2.2. *Clustering*

Autores como Gera y Goel (2015) definen c*lustering* como una técnica no supervisada que se centra en llevar a cabo análisis exploratorio de datos en el que no hay datos previamente etiquetados. El objetivo principal de esta técnica es separar el conjunto de datos sin etiquetar en un conjunto finito y discreto de estructuras de datos naturales. No hay ninguna posibilidad de proporcionar una caracterización previa de muestras no observadas que se generen a partir de la misma distribución de probabilidad. c*lustering* se puede definir como la identificación de clases similares de objetos. Mediante el uso de técnicas de *clustering* es posible identificar regiones densas y dispersas en el espacio de objetos y descubrir patrones de distribución general y las correlaciones entre los atributos de los datos (Baudier, 2000). Estos autores afirman que esta técnica es usada comúnmente como un enfoque de preprocesamiento para la selección y clasificación de subconjuntos de atributos. Los tipos de métodos de c*lustering* que se pueden encontrar, son:

- Métodos de partición
- Métodos aglomerados jerárquicos (divisorios)
- Métodos basados en la densidad
- *Grid-based methods*
- Métodos basados en modelos

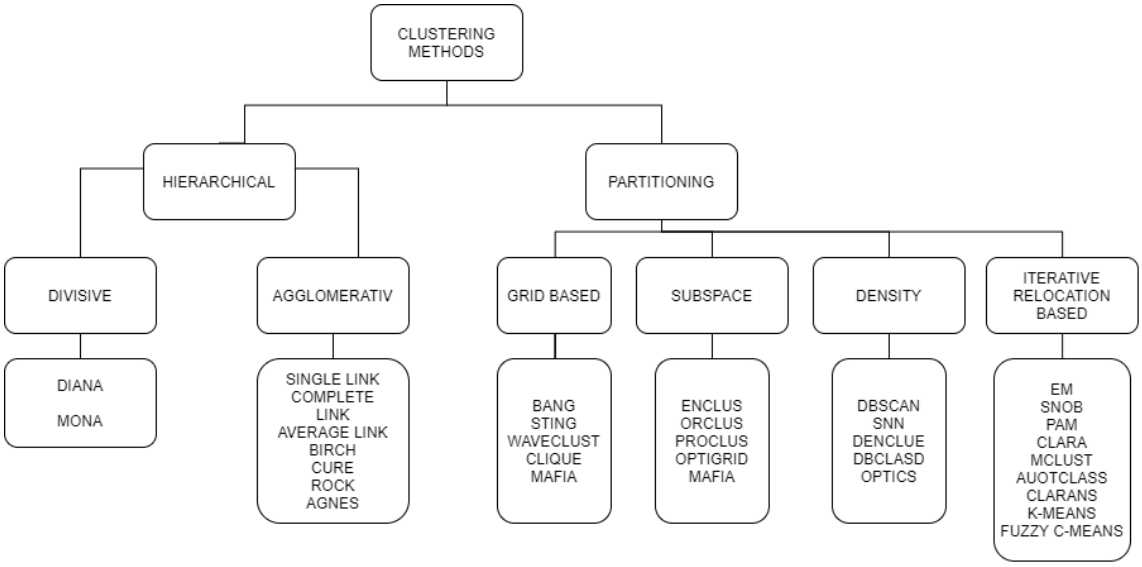


Figura 3. Categorización de métodos y algoritmos *clustering* (Gera & Goel, 2015).

5.2.3. Regresión

Regresión es una técnica de minería de datos que se basa en el aprendizaje supervisado y se usa para predecir un objetivo continuo y numérico según Gera y Goel (2015). Según el autor, regresión predice números, ventas, ganancias, pies cuadrados, temperatura o tasas hipotecarias. Todo esto se puede predecir mediante el uso de técnicas de regresión, iniciando con un conjunto de datos ya conocido, y estima el valor comparando valores ya conocidos y predichos. La técnica de regresión se puede adaptar para la predicción (Baudier, 2000). El análisis de regresión se puede usar para modelar la relación entre una o más variables independientes y variables dependientes. En la minería de datos, las variables independientes son atributos ya conocidos y las variables dependientes son lo se quiere predecir (Baudier, 2000).

Entre los tipos de métodos de regresión están:
- Regresión lineal
- Regresión lineal multivariante
- Regresión no lineal
- Regresión multivariante no lineal

5.3. MÉTODO EN MINERÍA DE DATOS

Kosterec (2016), define método como un conjunto de instrucciones que conducen a un objetivo específico. Sobre la base de esta definición, cualquier método que conduzca a un objetivo científicamente comprobado, puede considerarse científico. Normalmente, el uso de un método está motivado por algún tipo de problema formulado sobre la base de un fondo teórico y fáctico. Esto indica que un método siempre está expuesto a cambios que modifiquen su implementación. Es decir, un método se modifica cada vez que se obtienen nuevos conocimientos teóricos o fácticos siguiendo las instrucciones del método dado.
En el ámbito de minería de datos, los métodos se encuentran estrechamente relacionados con las técnicas de clasificación*, clustering y* regresión*.* Cada una

de estas técnicas se puede implementar a través de una variedad de métodos específicos de minería de datos, que se especifican a continuación.

5.3.1. Métodos de clasificación

Árbol de decisión de inducción:
Es una estructura de tipo árbol en la que hay un nodo interno, una rama y un nodo hoja. El nodo interno especifica la prueba en el atributo, la rama representa el resultado de la prueba y el nodo de hoja representa la etiqueta de la clase (Gera & Goel, 2015).

Clasificación basada en reglas:
Este método está representado por un conjunto de reglas *IF-THEN*. En primer lugar, cuántas de estas reglas se examinan y lo que sigue es cómo se construyen estas reglas y cómo se pueden generar a partir del árbol de decisión o se pueden generar a partir de datos de entrenamiento utilizando un algoritmo de cobertura secuencial (Gera & Goel, 2015).

Clasificación por *backpropagation*:
Backpropagation es un algoritmo de aprendizaje de redes neuronales y el más utilizado en el segmento. El aprendizaje de redes neuronales se suele denominar aprendizaje conexionista ya que crea conexiones. Este algoritmo tiene un nivel de complejidad alto debido a que requiere mucho tiempo de entrenamiento para poder ser utilizado (Gera & Goel, 2015).

Lazy learners:
Este método se basa en la forma en que se está desarrollando el modelo de generalización antes de que se reciba la nueva tupla para la clasificación. En el enfoque de *Lazy Learners*, cuando se le da una tupla de entrenamiento, simplemente la almacena y espera hasta que se le da una tupla de prueba (Gera & Goel, 2015).

Redes neuronales:
Según Gorunescu (2011), son modelos computacionales compuestos de muchos elementos de procesamiento no lineal, organizados en un patrón similar a las redes neuronales biológicas. Una red neuronal típica tiene un valor de activación asociado con cada nodo y un valor de peso asociado con cada conexión. Una función de activación gobierna la activación de nodos y la propagación de datos a través de conexiones de red en un paralelismo masivo.

Support Vector Machines (SVM):
Los SVM son considerados máquinas, algoritmos y métodos de aprendizaje que pueden realizar tareas de estimación de clasificación y regresión binaria (Joseph, Hlomani, & Letsholo, 2018). Los SVM son cada vez más utilizados como un nuevo paradigma de clasificación y aprendizaje debido a dos factores importantes. En primera instancia, los SVM minimizan el error esperado en lugar de minimizar el error de clasificación. En segunda instancia, emplean la teoría de la dualidad de la programación matemática para obtener un problema dual que admite métodos computacionales eficientes (Joseph et al., 2018).

5.3.2. Métodos de *clustering*

Métodos de *clustering* suelen dividirse en dos categorías principales que tienen varias instancias. Sobre esta base existen métodos jerárquicos y métodos de partición (Gera & Goel, 2015).

Métodos de partición:
Métodos de partición, simplemente crea *k* particiones de *N* elementos del conjunto de datos. Existen diferentes tipos de enfoques de este método como: *Grid based, Subspace based, Density based* y *Relocation based* (Gera & Goel, 2015).

Métodos jerárquicos*:*
En este método, los conjuntos de datos de *n* elementos se dividen en jerarquía de grupos que tiene una estructura similar a un árbol en este método. Hay dos enfoques que son aglomerado y divisorio (Gera & Goel, 2015).

5.3.3. Métodos de regresión

Para la técnica de regresión, existen dos tipos de métodos conocidos como regresión lineal y regresión no lineal (Gera & Goel, 2015).

Regresión lineal:
Este método se utiliza cuando la relación entre el objetivo y el predictor se puede representar en línea recta.

Regresión no lineal:
En este método se presenta cuando puede haber una relación no lineal y esto no se puede representar como una línea recta. Esto puede representarse como reacción lineal mediante el preprocesamiento de los datos.

5.4. ALGORITMO EN MINERÍA DE DATOS

Un algoritmo de minería de datos es un conjunto de heurísticas y cálculos que crea un modelo de minería de datos a partir de datos.

De acuerdo con Joseph y otros (2018), puede ser complicado elegir el algoritmo adecuado para resolver un problema determinado. Aunque es posible hacer uso de diferentes algoritmos para brindar solución a un problema, no es práctico usar diferentes algoritmos para la misma tarea, ya que cada uno fue diseñado para un objetivo en específico. Es decir, algunos algoritmos pueden realizar el proceso de clasificación, prediciendo una o más variables, en función de los demás atributos del conjunto de datos. Otros algoritmos realizan propósitos de regresión, pueden predecir variables continuas según los otros atributos del conjunto de datos. Algunos algoritmos pueden realizar segmentación, es decir *clustering* dividiendo los datos en grupos que tienen propiedades similares.

Todos estos algoritmos se dividen en dos grandes categorías: aprendizaje supervisado y aprendizaje no supervisado.

5.4.1. Algoritmos de aprendizaje supervisado

Los algoritmos de aprendizaje supervisado son aquellos para los cuales se conocen los valores de atributos de clase para el conjunto de datos antes de ejecutar el algoritmo. Estos datos se denominan datos etiquetados o datos de entrenamiento (Joseph et al., 2018).

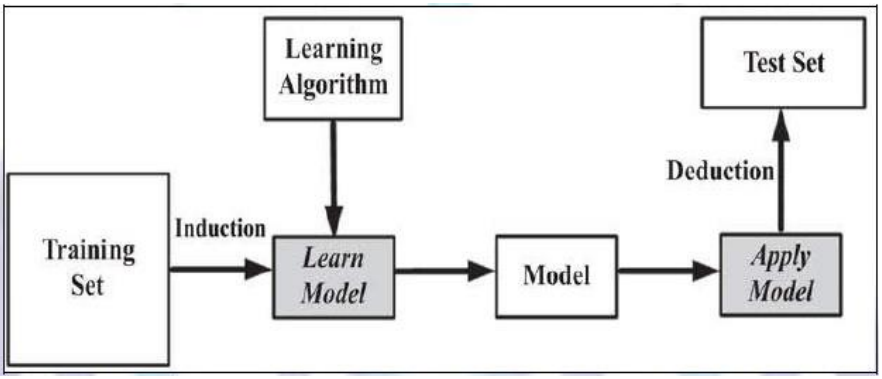


Figura 4. *Supervised learning process* (Joseph et al., 2018).

Algunos de los algoritmos más relevantes de esta categoría, son:

Aprendizaje del árbol de decisiones (C4.5, ID3, CART):
Se basan en el entrenamiento a partir de un conjunto de datos, donde cada nodo en un árbol representa una entidad y cada rama representa un valor que la entidad puede tomar. Este proceso es una clasificación que sigue una ruta que comienza en el nodo raíz y termina en una hoja siguiendo ramas basadas en valores de características de instancia. Este proceso se basa principalmente en heurísticas (Joseph et al., 2018).

Naive Bayes Classifiers(NB):
Representa un modelo que codifica relaciones probabilísticas entre variables de interés. En el aprendizaje automático, los clasificadores de Bayes son una familia de clasificadores probabilísticos simples basados en la aplicación del teorema de Bayes con supuestos de independencia fuertes entre las características (Joseph et al., 2018).

K-Nearest Neighbour (KNN):
KNN es un algoritmo simple que almacena todos los casos disponibles y clasifica los nuevos casos en función de una medida de similitud (funciones de distancia). KNN es comúnmente utilizado en estadística y en el reconocimiento de patrones como una técnica no paramétrica (Joseph et al., 2018).

Support Vector Machine (SVM):
Se centra en asignar vectores de entidades de entrada a un espacio de entidades de mayor dimensión a través de algún mapeo no lineal. Los SVM se desarrollan según el principio de minimización del riesgo estructural. La minimización del riesgo estructural busca encontrar una hipótesis, para lo cual se puede encontrar la menor probabilidad de error, mientras que las técnicas de aprendizaje tradicionales para el reconocimiento de patrones se basan en la minimización del riesgo empírico, que intenta optimizar el rendimiento del conjunto de aprendizaje (Joseph et al., 2018).

Regresión lineal:
Es un procedimiento estadístico para predecir el valor de una variable dependiente, a partir de una variable independiente cuando la relación entre las variables se puede describir con un modelo lineal (Joseph et al., 2018).

Regresión logística:
Es una clasificación análoga a la regresión lineal. Es preferible al aprendizaje del árbol de decisiones en las mismas situaciones que la regresión lineal, es preferible a la regresión de los efectos cuando los efectos son pequeños y cuando los predictores contribuyen de manera aditiva (sin interacciones) (Joseph et al., 2018).

5.4.2. Algoritmos de aprendizaje no supervisado

Cuando se habla de aprendizaje no supervisado, la mayoría de los autores se refieren a agrupar datos. En la agrupación, los datos a menudo no se encuentran etiquetados, es decir sin un contexto previo. Por lo tanto, la etiqueta para cada instancia no es conocida por el algoritmo de agrupamiento. Esta es la principal diferencia entre aprendizaje no supervisado y aprendizaje supervisado. Cualquier algoritmo de agrupamiento requiere una medida de distancia. La medida de distancia más popular para características continuas es la distancia euclidiana (Joseph et al., 2018). Joseph y otros (2018), señalan los siguientes algoritmos como los más relevantes de esta categoría:

K-Means:
Es un algoritmo basado en una técnica de agrupamiento en la que el usuario comienza con una recopilación de muestras e intenta agruparlas en "K". El número de agrupaciones se encuentran basadas en ciertas medidas de distancia específicas. Las agrupaciones generadas por el algoritmo *K-Means* son un número puntual de agrupaciones desunidas, planas (no jerárquicas).

Density Based:
Es un algoritmo de agrupamiento que juega un papel vital en la búsqueda de estructuras de formas no lineales basadas en la densidad, haciendo uso del concepto de *density reachability* y *density connectivity*.

Apriori:
Es un algoritmo influyente para extraer conjuntos de elementos frecuentes por medio de reglas de asociación booleanas. *Apriori* utiliza un enfoque "*bottom up*", donde se extienden subconjuntos frecuentes.

Como se puede evidenciar, en la actualidad existe una gran variedad de algoritmos en minería de datos para procesar y analizar la información de acuerdo a la necesidad y/o problemática a tratar. Dichos algoritmos pueden procesar datos según sea el caso, teniendo o no contexto previo. Esto es gracias a las dos grandes categorías de aprendizaje que clasifican la orientación de estos, es decir aprendizaje no supervisado y aprendizaje supervisado.

5.5. CRITERIOS PARA LA SELECCIÓN DEL ALGORITMO DE MINERÍA DE DATOS

Una vez se tiene un contexto de los términos técnica, método, algoritmo y sus componentes, es necesario abordar los múltiples criterios considerados para la selección de los algoritmos a utilizar en el desarrollo del método como objetivo de esta tesis. Por tanto, en esta sección se brinda una contextualización de cada uno de estos criterios, en aras de justificar su uso para la selección de los algoritmos.

Los elementos que se definen a continuación fueron producto de una revisión de literatura en el ámbito de la minería de datos. En dicha revisión fue posible identificar algunos métodos, algoritmos y características que fueron analizadas en los trabajos seleccionados, para finalmente definir cuáles algoritmos de Minería de Datos eran adecuados para cumplir el objetivo planteado en este trabajo. La descripción de cada uno de estos elementos se presenta a continuación:

5.5.1. Método

Agrupar las palabras en grupos semánticamente coherentes*:*
Según Bekkerman y otros (2003) y Slonim y Tishby (2000), este método hace parte de una estrategia conformada por un grupo de algoritmos que tienen como finalidad agrupar palabras de una manera coherente según el contexto – *Requirement Word Clustering*. Dicho método podría ser de gran ayuda en el proceso de identificación de RNF en este método propuesto.

Término de indicador:
De acuerdo con Cleland-Huang y otros (2006), el método clasificador RNF está compuesto por dos etapas. Durante la primera etapa, se identifica un conjunto de términos de indicador para cada categoría de RNF. Este paso asume la existencia de un conjunto de requisitos preclasificados correctamente que se pueden utilizar para el entrenamiento del algoritmo a utilizar. Los requisitos en el conjunto de entrenamiento se utilizan para calcular un peso probabilístico para cada término de indicador potencial con respecto a cada tipo de RNF. El peso mide la fuerza con la que un término de indicador representa un tipo de requisito. Por ejemplo, términos como "autenticar" y "acceso" que aparecen con frecuencia en los requisitos de seguridad y con poca frecuencia en otros tipos de requisitos, representan términos de indicador sólidos para los RNF de seguridad, mientras que otros términos como "asegurar" que ocurren con menos frecuencia en los requisitos de seguridad, se encuentran en varios tipos diferentes de requisitos, representando indicadores mucho más débiles.

5.5.2. Algoritmos

LSA (Latent semantic analysis):
En las técnicas de *clustering*, para el agrupamiento de palabras por contexto o similitud semántica es muy común hacer uso del algoritmo LSA, para la representación y extracción de relaciones entre palabras y documentos en grandes cuerpos de texto (Deerwester, Furnas, Landauer, & Harshman, 1990). LSA en *clustering* es usado como un algoritmo auxiliar según los autores.

Similitud semántica:
Según Mahmoud y Williams (2016), han surgido varios métodos en la literatura para estimar el proceso de similitud semántica entre las palabras del lenguaje natural. Tales métodos, explotan un amplio espectro de esquemas semánticos, en diferentes niveles de complejidad computacional, para capturar relaciones de asociación entre palabras en un corpus.

5.5.3. Características

Words occur/ Weight Term:
Según Zhang y otros (2011), esta característica se haya embebida en la ejecución del algoritmo *Support Vector Machine*. Ya que, según la ocurrencia de palabras en un texto de requerimiento, se le brinda un peso a una palabra lo cual determina su importancia en el contexto del requerimiento.

Index Term:
Esta característica hace referencia a palabras clave que se pueden identificar en el proceso de clasificación de los RNF. Por ejemplo seguridad, usabilidad y *Look And Feel* (Zhang et al., 2011). Es importante aclarar que este criterio puede ser implementado en diferentes algoritmos de minería de datos, ya que hace parte del proceso de la clasificación texto.

Word decompose:
Este característica es la base del proceso de factorización llamado *singular-value decomposition* y según Mahmoud y Williams (2016), hace parte de las técnicas de clasificación y extracción de información a partir de documentos escritos en lenguaje natural y es aplicado para descomponer palabras por documento (Demmel & Kahan, 2005).

Menor esfuerzo humano:
En la fase de educción de requisitos, es común que los analistas de requisitos lleven a cabo tareas tediosas que requieren gran esfuerzo y consumo de tiempo. Sin embargo, autores como Casamayor y otros (2010), afirman que el uso de algoritmos de clasificación trae consigo la característica "Menor esfuerzo humano" en los requisitos etiquetados, haciendo uso de métodos semi-supervisados en comparación con los métodos totalmente supervisados. Dichos métodos semi-supervisados pueden mejorar aún más apoyándose en los comentarios de los analistas, una vez integrados dentro de un sistema de soporte de decisiones para la educción de requisitos.

En la **Tabla 8** se sintetizan los anteriores elementos como criterios de comparación de los potenciales algoritmos a implementar en el método a proponer. Con base en los resultados obtenidos de la experimentación con los algoritmos seleccionados, se demostrará su efectividad.

Tabla 8. Aplicación de criterios sobre algoritmos (Joseph et al., 2018).

Criterios (Métodos, características y algoritmos)	**SVM (Support Vector Machine)**	**Naive Bayes**	**K-NN**	**Expectation Maximization (EM)**	**Apriori**	**K-Means**	**Density Based**
LSA (Latent semantic analysis)			**x**	**x**			
Word decompose	**x**	**x**	**x**	**x**			
Similitud semántica					**x**	**x**	**x**
Agrupar las palabras en grupos semánticamente coherentes					**x**	**x**	**x**
Clustering – Métodos de partición					**x**	**x**	**x**
Clustering – Métodos jerárquicos					**x**	**x**	**x**
Término de indicador	**x**	**x**	**x**	**x**			
Training set Data		**x**	**x**	**x**			
Words occur	**x**	**x**					
Weight Term	**x**	**x**					
Index Term	**x**			**x**			
Menor esfuerzo humano	**x**	**x**	**x**	**x**			
Tipo de algoritmo: Supervisado= 1 No supervisado= 0	**1**	**1**	**1**	**N/A**	**0**	**0**	**0**

Como se puede observar en la **Tabla 8**, de los 12 criterios conformados por métodos, características y algoritmos, solo cuatro criterios encajan con los algoritmos *clustering*, los cuales son *A priori, K-Means y Density Based*. Caso contrario ocurre con los algoritmos de clasificación, los cuales involucran en su proceso entre cinco y seis criterios cada uno. Se demuestra así una ventaja de nivel de clasificación de texto, lo cual brinda una orientación evidente sobre cuál (es) algoritmo(s) implementar en el método propuesto. Para este caso, tomando en cuenta la **Tabla 5** y la evidencia identificada en la literatura de otros enfoques descritos previamente, se seleccionaron los algoritmos *Support Vector Machine* y *Naives Bayes,* para llevar a cabo el proceso de extracción, identificación y clasificación de RNF.

PARTE IV:

APLICACIÓN DE TÉCNICAS A DOCUMENTOS ORGANIZACIONALES

CAPÍTULO 6. APLICACIÓN DE TÉCNICAS A DOCUMENTOS ORGANIZACIONALES

En la búsqueda de estudios primarios que apoyaran el objetivo de esta tesis, se halló evidencia bibliográfica que demuestra una probable ayuda en la fase de educción de requisitos; la cual, podría darse en la extracción de información desde documentos organizacionales (documentos de requisitos y textos sin estructura) para la clasificación de RF y RNF.

A continuación, se describen algunas de las razones fundamentadas en estudios primarios, que sustentan que la identificación y clasificación de los RNF tienen un peso relevante en la fase de educción de requisitos:

6.1. IMPORTANCIA DE LA EXTRACCIÓN DE RNF EN LA EDUCCIÓN DE REQUISITOS

Según Mahmoud y Williams (2016), los RNF describen un conjunto de atributos de calidad que un sistema de software debe exhibir. Dichos atributos imponen restricciones operativas en diferentes aspectos del comportamiento del sistema, como su facilidad de uso, seguridad, confiabilidad, rendimiento y portabilidad.

Mahmoud y Williams (2016), afirman la importancia que conlleva la identificación de los RNF al inicio de un proyecto de software en aras de tomar decisiones óptimas al diseñar el modelo de arquitectura para evitar arquitecturas anómalas. Sin embargo, los RNF a menudo se pasan por alto durante la fase de educción de requisitos, donde el énfasis principal es lograr que las características funcionales del sistema se definan de manera explícita y formal. Parte de este fenómeno puede atribuirse al poco entendimiento de lo que realmente son los RNF en la ingeniería de software y la falta de métodos efectivos de obtención, modelado y documentación (Mahmoud & Williams, 2016).

A pesar de la importancia de los RNF en la educción de requisitos, es común que éstos se descubran relativamente tarde en el proceso de desarrollo de software, debido a las falencias de calidad presentadas en las partes interesadas, provocadas durante el proceso de recopilación de requisitos. La información recopilada en la educción de requisitos se documenta en una variedad de artefactos que incluyen textos sin estructura, notas de entrevistas y actas de reuniones (Cleland-Huang et al., 2006). Según estos autores, ello genera incertidumbre en los analistas, quienes con frecuencia no logran obtener una perspectiva clara sobre los RNF de todo el sistema. En la mayoría de los casos, los RNF tienden a estar integrados en los RF del sistema. Es el caso cuando se supone que un sistema con una función de inicio de sesión es seguro (Mahmoud & Williams, 2016). Esto puede conllevar a conflictos que no detectan soluciones arquitectónicas que satisfagan las necesidades reales de los interesados. A pesar de estas dificultades presentadas en el proceso de identificación y clasificación de RNF, existen métodos que apoyan la clasificación de RNF como:

- Técnicas de elicitación
- Técnicas de detección
- Métodos de clasificación de RNF partiendo de palabras clave.

Sin embargo, estos métodos no son suficientes para revisar todas las posibles fuentes de RNF, ya que es un proceso tedioso que requiere esfuerzo y consume mucho tiempo (Slankas & Williams, 2013).

Los RNF cumplen un papel transcendental en el diseño arquitectónico del sistema, en aras de satisfacer las necesidades de las partes interesadas. Sin embargo, cuando los RNF son definidos por profesionales que no cuentan con la experiencia necesaria para ello, se obtienen RNF con vaguedad y redundancia. Esto a su vez genera retrocesos y esfuerzos adicionales en la ejecución de un proyecto (Zhang et al., 2011). Lo anterior sugiere que los desarrolladores no brindan suficiente importancia a la tarea de evaluar los RNF. Ayudarlos a identificar y administrar los RNF podría reducir este riesgo que conlleva a esfuerzos adicionales en tiempo, costo y rediseño (Kurtanovic & Maalej, 2017).

Los métodos para la obtención de RNF a menudo implican el uso de cuestionarios, *CheckList* o plantillas para preguntar a las partes interesadas sobre temas de calidad (atributos de calidad), de acuerdo con Casamayor y otros (2010). Una clasificación adicional de los requisitos obtenidos podría ser asistida por un conocimiento específico del dominio o en su defecto por ontologías que representan los atributos de calidad a considerar. En un escenario real, un analista de requisitos debe revisar todos los documentos de requisitos recopilados por un equipo de educción de requisitos para decidir si especifican RF y/o RNF, así como las categorías o clases de RNF (seguridad, disponibilidad, escalabilidad, etc.), con el objetivo de priorizarlos y mapearlos en diseños de arquitectura. Esta es una tarea que consume mucho tiempo y requiere un gran esfuerzo manual por parte de los analistas, según Casamayor y otros (2010).

Según Slankas y Williams (2013), una práctica que viene siendo muy descuidada en el proceso de educción de requisitos es la documentación de software, a pesar de los esfuerzos realizados por educadores y profesionales. Siendo los correos, archivos y listas de mensajes, la única fuente de información para llevar a cabo la educción de requisitos. Esto abre la puerta a posibles soluciones basadas en el procesamiento de documentos organizacionales en la fase de educción de requisitos.

Tomando como referencia lo anteriormente descrito, es posible concluir que existen fundamentos y criterios para afirmar que los RNF hacen parte esencial de la fase de educción de requisitos; lo cual, ratifica el enfoque del presente trabajo investigativo basado en la extracción y clasificación de RNF desde documentos de requisitos.

6.2. EXPERIMENTACIÓN Y SELECCIÓN DE TÉCNICA Y ALGORITMO

Con base en la evidencia bibliográfica anteriormente citada, se define que los algoritmos a considerar para diseñar un método de extracción de RNF desde documentos de requisitos se encuentran fundamentados en técnicas de clasificación de texto. Los algoritmos seleccionados para dicha tarea de clasificación son *Support Vector Machine* y *Naive Bayes,* debido a la evidencia de casos de éxito descrita en los estudios primarios. No obstante, para justificar la decisión de seleccionarlos, fue necesario llevar a cabo múltiples pruebas tipo diagnóstico, las cuales tienen como objetivo evidenciar el comportamiento de un determinado algoritmo al ser puesto a prueba con un

documento de entrada. Este documento para nuestro caso es un archivo con oraciones diferenciadas por características de categorías pertenecientes a atributos de calidad, simulando un apartado/extracto de un documento de requisitos (Ver **Figura 2).**

A continuación, se describe el detalle de los algoritmos seleccionados *Support Vector Machine* y *Naive Bayes*, así como el contexto documental referente a su objetivo de éstos, frente al proceso de clasificación.

6.2.1. Algoritmos y técnica Seleccionados

En el proceso de selección de la técnica, algoritmo y método de minería de datos para realizar el desarrollo y estructuración del método propuesto, fue necesario llevar a cabo un análisis detallado de la evidencia bibliográfica existente a nivel de estudios primarios, relacionados con la extracción de información seguida de la identificación y clasificación de RF y RNF, haciendo uso de técnicas, algoritmos y métodos. En los estudios primarios citados en el **Capítulo 4** (Sección **4.1 Estado del arte**), se logra evidenciar cómo las técnicas y algoritmos de clasificación son los más utilizados al momento de extraer y clasificar información relevante en la ingeniería de requisitos, específicamente en la fase de educción.

Es evidente, que la clasificación es el tipo de técnica y algoritmo que prima en múltiples estudios primarios, como se puede evidenciar en la **Tabla 6**, en donde cinco de seis estudios hacen uso de técnicas y algoritmos de clasificación. Esto permite evidenciar, que los algoritmos de clasificación SVM, NB y KNN, son los más utilizados en gran parte de los estudios primarios. Dichos algoritmos citados son utilizados en conjunto para lograr un resultado más preciso en métodos de extracción y clasificación de texto. Como es el caso de Casamayor y otros (2010) y Slankas y Williams (2013), quienes utilizan en sus métodos propuestos los algoritmos NB, SVM y KNN. Estos autores, resaltan la utilidad de los algoritmos NB y SVM por su popularidad entro los investigadores y su simplicidad al momento de implementarlos a través de *frameworks* y lenguajes de programación.

Con base en lo anteriormente descrito, se llevará a cabo un experimento de prueba, haciendo uso de los algoritmos NB y SVM en miras de validar su efectividad, para seleccionar el algoritmo de clasificación sobre el cual se construirá el método propuesto.

6.2.2. Estructuración del experimento de prueba con los algoritmos seleccionados

En la implementación de los algoritmos *Support Vector Machine* y *Naive Bayes*, fue necesario indagar en los múltiples lenguajes de programación, que son comúnmente utilizados en minería de datos como Java, R, *Python*, C++ y C#; así como en las herramientas que hacen uso de dichos lenguajes como *Weka, Spyder, Orange, Eclipse y Visual Studio*. Una vez se tuvo claro las herramientas y lenguajes de programación más utilizados, se prosiguió con la selección de un lenguaje *open source* que fuese de fácil aprendizaje; *Python* es uno de los lenguajes que cumple con dichas características. Adicionalmente, *Python* cuenta con uno de los *framework* más completos en minería de datos *y Data analysis*, llamado *Scikit-learn*, que tiene una gran variedad de componentes que usan técnicas, algoritmos y métodos. Es decir, que no es necesario conocer

a fondo las operaciones matemáticas y/o de inteligencia artificial de cada algoritmo aquí implementado; sólo basta con configurar previamente los algoritmos a través de la importación de librerías, y suministrar los insumos que van a ser procesados por los algoritmos. Esto sería suficiente para iniciar su ejecución y evaluar los resultados.

Una vez se tiene claro que el lenguaje a utilizar es *Python* y que el *framework* es *Scikit-learn*, se eligió la herramienta, la cual en el mundo de desarrollo de software es conocida como un IDE (*Integrated Development Environment*). El IDE seleccionado para este proceso fue *Spider (Python* 3.6*),* el cual, es un IDE básico portable que no requiere de instalación en el sistema operativo del equipo de cómputo y se le pueden agregar los *frameworks* que se requieran para una determinada implementación.

En la implementación de los algoritmos seleccionados, fue necesario crear una instancia de *Python* en *Spyder* como se muestra en las **Figuras 5 y 6**:

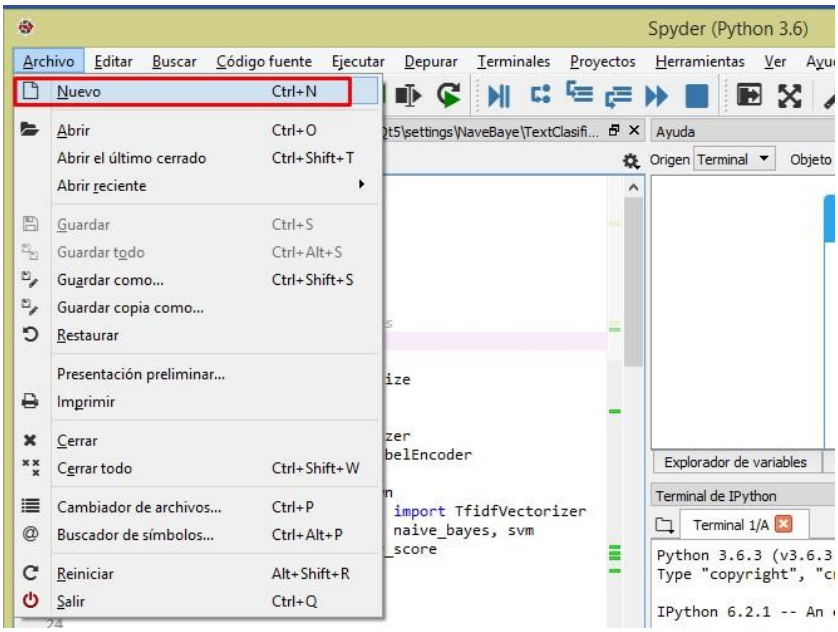


Figura 5. Creación de una instancia Python.

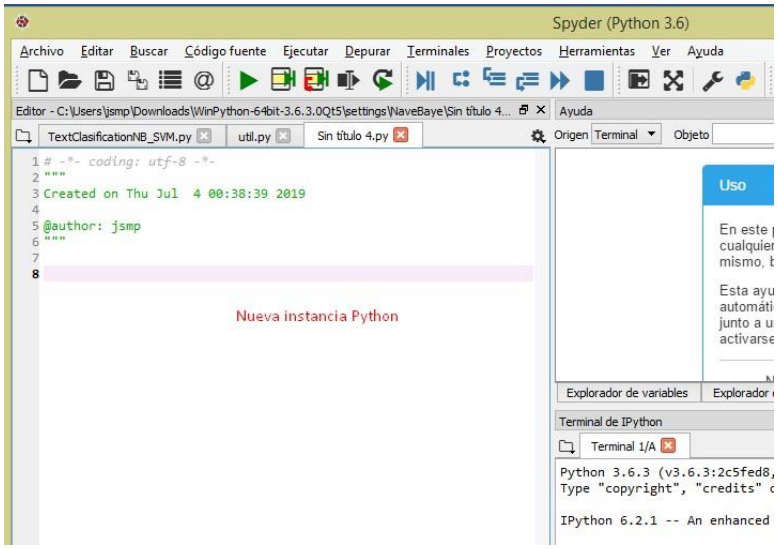


Figura 6. Nueva instancia Python.

Una vez se creó la instancia de *Python* en el IDE *Spyder,* se procedió con la codificación de ambos algoritmos de clasificación (SVM y *Naive Bayes*). En este proceso de codificación, fue necesario llevar a cabo la importación de las librerías necesarias, como se muestra en la **Figura 7**, para utilizar las funcionalidades requeridas en la implementación.

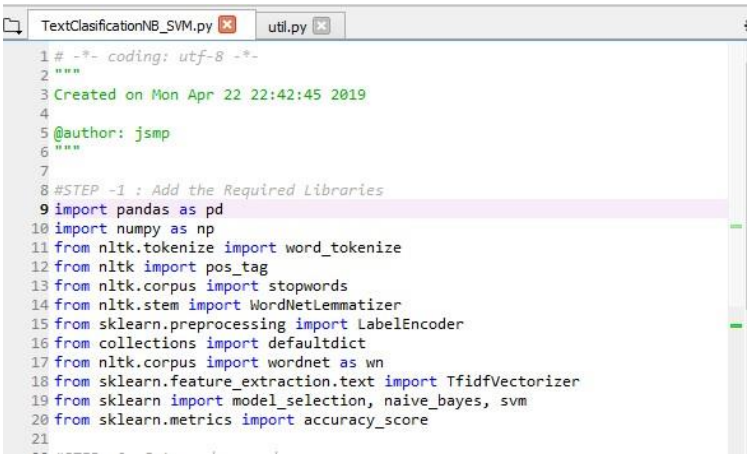


Figura 7. Importación de librerías.

Luego de la importación de las librerías, se inicia la codificación para la implementación de los algoritmos mencionados. Dicha codificación se encuentra compuesta por las siguientes actividades:

Carga de insumos a procesar:
En este primer paso, se carga en memoria el documento *Input* a procesar por los algoritmos.

Preprocesamiento del texto contenido en el insumo:
En este paso se depura el texto retirando *StopWords*, espacios en blanco y puntuación. Adicionalmente, se hace uso del *Pos Tagging* para identificar si una palabra es sustantivo, verbo o adjetivo.

Entrenamiento de los algoritmos SVM y *Naive Bayes*:
El set de datos es distribuido en porcentajes de datos: de entrenamiento (70%) y de testeo (30%).

Ejecución de clasificación de los SVM y *Naive Bayes*:
En este paso se evaluó comparando la efectividad de los algoritmos nombrados en cuanto a precisión, traducida en porcentaje (%). A mayor porcentaje, mayor es la precisión del algoritmo evaluado.

Es relevante aclarar que esta sección tiene como objetivo brindar un contexto a alto nivel de la implementación de los algoritmos SVM y *Naive Bayes*.

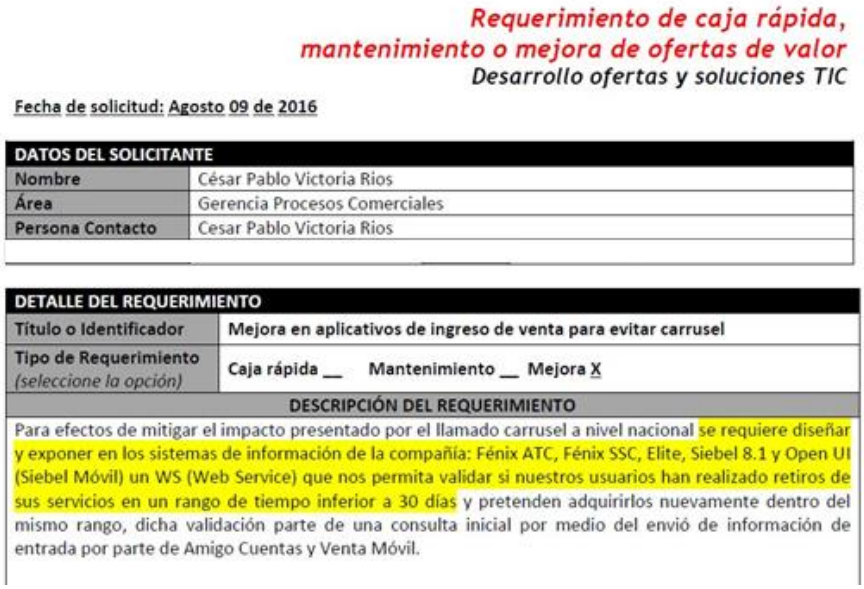
6.2.3. Documentos de prueba utilizados

Los documentos utilizados en la implementación de los algoritmos SVM y NB, son documentos de requisitos de una empresa de telecomunicaciones. En total son 31 documentos de requisitos en formato *Word,* escritos en texto español, los cuales cuentan con un nivel de complejidad variable según la necesidad planteada. En esta implementación se usaron los documentos que tuvieran mayor complejidad en aras de captar una mayor cantidad de posibles RNF, debido a que en estos documentos comúnmente priman los RF, de acuerdo con Mahmoud y Williams (2016).

6.2.4. Ejecución del experimento de prueba planteado

Los artefactos procesados por los algoritmos SVM y *Naive Bayes* en *Python,* fueron archivos .csv producto de un procesamiento manual de documentos de requisitos de mayor complejidad, seleccionados entre los 31 documentos nombrados anteriormente. Este proceso manual se llevó a cabo mediante las siguientes actividades:

Lectura de documento de requisitos:
En esta actividad se realiza la lectura de documentos de requisitos *Word* de mayor complejidad es decir los que presentaban mayor volumen de texto y detalle de necesidades planteadas a través de requisitos. Los documentos *Word* que fueron procesados manualmente, cuentan con una estructura definida por la empresa de telecomunicaciones. En las **Figuras 8 y 9**, se muestran imágenes de dichos documentos:



Requerimiento de caja rápida, mantenimiento o mejora de ofertas de valor
Desarrollo ofertas y soluciones TIC

Fecha de solicitud: Agosto 09 de 2016

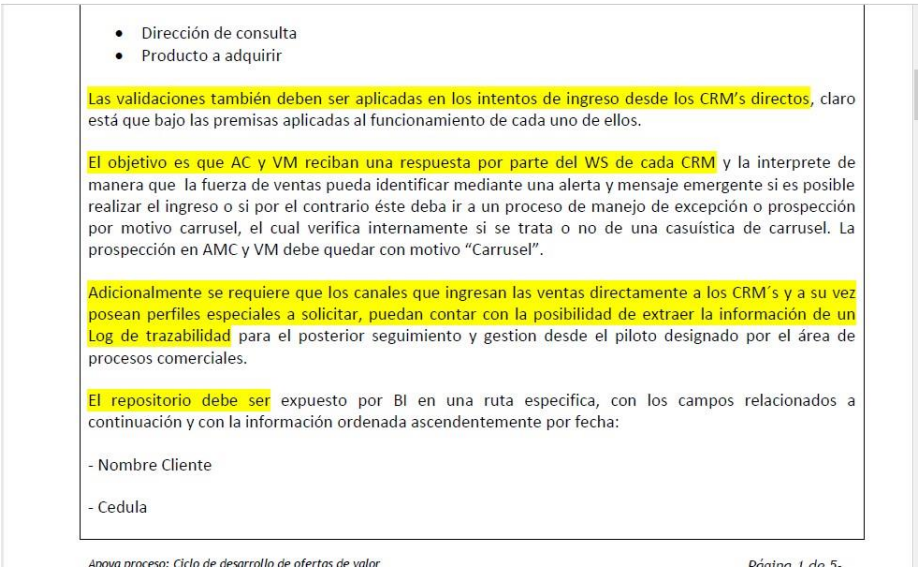
DATOS DEL SOLICITANTE	
Nombre	César Pablo Victoria Rios
Área	Gerencia Procesos Comerciales
Persona Contacto	Cesar Pablo Victoria Rios

DETALLE DEL REQUERIMIENTO	
Título o Identificador	Mejora en aplicativos de ingreso de venta para evitar carrusel
Tipo de Requerimiento *(seleccione la opción)*	Caja rápida __ Mantenimiento __ Mejora X

DESCRIPCIÓN DEL REQUERIMIENTO

Para efectos de mitigar el impacto presentado por el llamado carrusel a nivel nacional se requiere diseñar y exponer en los sistemas de información de la compañía: Fénix ATC, Fénix SSC, Elite, Siebel 8.1 y Open UI (Siebel Móvil) un WS (Web Service) que nos permita validar si nuestros usuarios han realizado retiros de sus servicios en un rango de tiempo inferior a 30 días y pretenden adquirirlos nuevamente dentro del mismo rango, dicha validación parte de una consulta inicial por medio del envió de información de entrada por parte de Amigo Cuentas y Venta Móvil.

Figura 8. Documento de requerimiento "**[Requerimiento]Carrusel con Control de Cambios.docx**" página 1



- Dirección de consulta
- Producto a adquirir

Las validaciones también deben ser aplicadas en los intentos de ingreso desde los CRM's directos, claro está que bajo las premisas aplicadas al funcionamiento de cada uno de ellos.

El objetivo es que AC y VM reciban una respuesta por parte del WS de cada CRM y la interprete de manera que la fuerza de ventas pueda identificar mediante una alerta y mensaje emergente si es posible realizar el ingreso o si por el contrario éste deba ir a un proceso de manejo de excepción o prospección por motivo carrusel, el cual verifica internamente si se trata o no de una casuística de carrusel. La prospección en AMC y VM debe quedar con motivo "Carrusel".

Adicionalmente se requiere que los canales que ingresan las ventas directamente a los CRM´s y a su vez posean perfiles especiales a solicitar, puedan contar con la posibilidad de extraer la información de un Log de trazabilidad para el posterior seguimiento y gestion desde el piloto designado por el área de procesos comerciales.

El repositorio debe ser expuesto por BI en una ruta especifica, con los campos relacionados a continuación y con la información ordenada ascendentemente por fecha:

- Nombre Cliente

- Cedula

Apoyo proceso: Ciclo de desarrollo de ofertas de valor *Página 1 de 5*

Figura 9. Documento de requerimiento "**[Requerimiento]Carrusel con Control de Cambios.docx**" página 2.

Identificación de párrafos que, por su estructura y composición, tienen alta probabilidad de ser RNF:
En esta actividad se identificaron manualmente varios párrafos que contenían textos que hacían alusión directa o indirecta a características de RNF (Ver **Figura 2**). Cuando es indirecta se hace alusión a RF que pueden tener embebidos RNF.

Asignación de tipos de RNF a los párrafos previamente identificados:
Como se puede ver en la **Figura 2,** los tipos de RNF pueden ser agrupados por características. Por tanto, en esta actividad se realizó una asignación individual

de tipo de RNF por párrafo, luego de identificar posibles características que se relacionaban con uno o varios tipos de RNF.

Luego de realizar las actividades previamente descritas, la información capturada y clasificada fue posteriormente registrada en un archivo .csv como se muestra en la **Figura 10**:

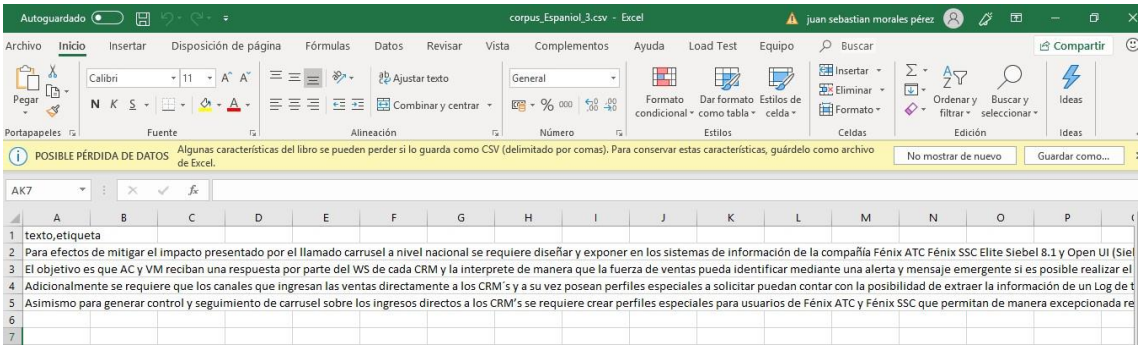


Figura 10. Insumo para la implementación de algoritmos (**"corpus_espaniol.csv"**)

La estructura del archivo **"corpus_espaniol.csv"**, está compuesta de la siguiente manera:

- Una hoja llamada "corpus_espaniol".
- Una fila con el *Header* "texto,etiqueta".
- Registros conformados por una o múltiples filas con una única columna por registro. Ejemplo: "Para efectos de mitigar el impacto presentado por el llamado carrusel a nivel nacional., __exactitud__". Como se puede observar en el ejemplo anterior, el texto y la etiqueta se encuentran separadas por ",". Es decir, el texto y la etiqueta del *Header.*

El archivo **"corpus_espaniol.csv"**, fue producto de un proceso de clasificación manual de RNF y fue insumo de procesamiento para los algoritmos SVM y *Naive Bayes*. El objetivo de este procesamiento fue evaluar la precisión medida en porcentaje (%) de ambos algoritmos. Es decir, a mayor porcentaje mayor precisión del algoritmo al momento de identificar los párrafos RNF que corresponden a un tipo determinado de RNF. A continuación, se muestra el resultado del procesamiento de tres archivos **"corpus_espaniol.csv"**, con diferente cantidad de registros y tipos de RNF. El detalle de los archivos se muestra de la **Figura 11** a la **15**.

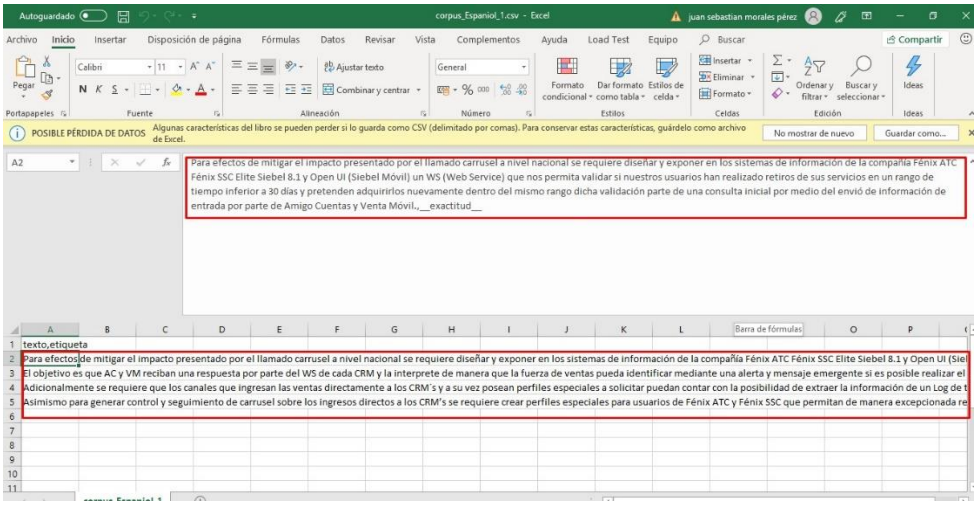


Figura 11. Insumo para la implementación de algoritmos (**"corpus_espaniol.csv"**) de 4 registros

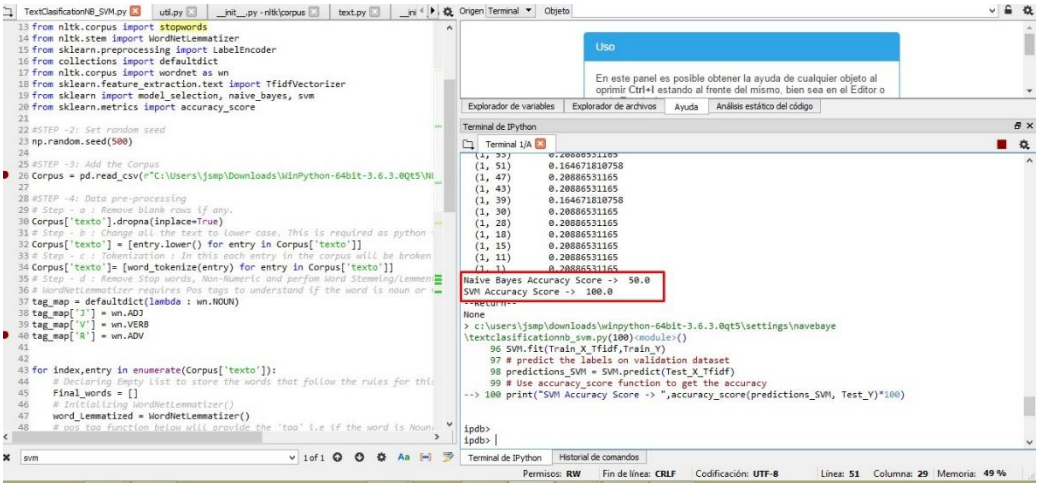


Figura 12. Resultado del procesamiento del archivo **"corpus_espaniol.csv"** de 4 registros

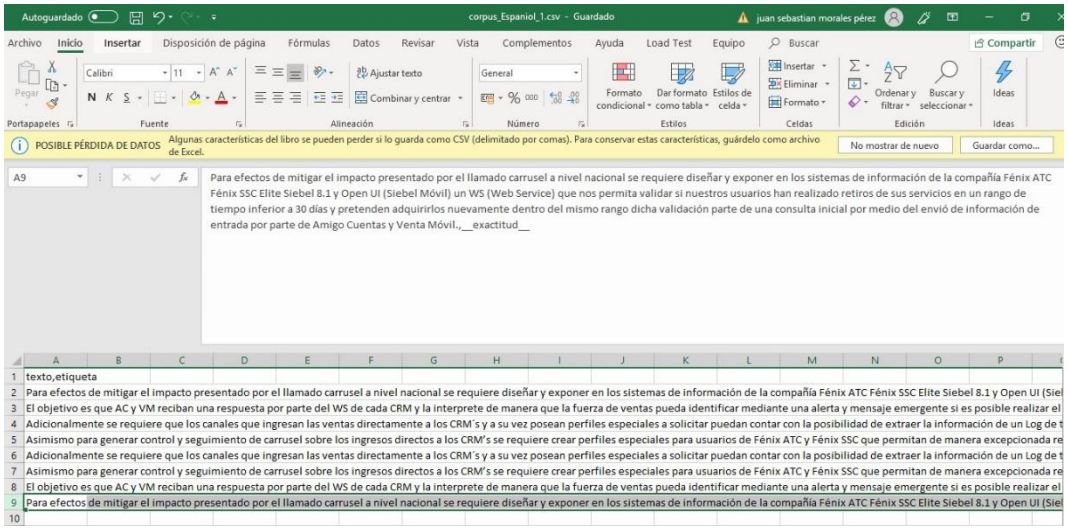


Figura 13. Insumo para la implementación de algoritmos (**"corpus_espaniol.csv"**) de 8 registros

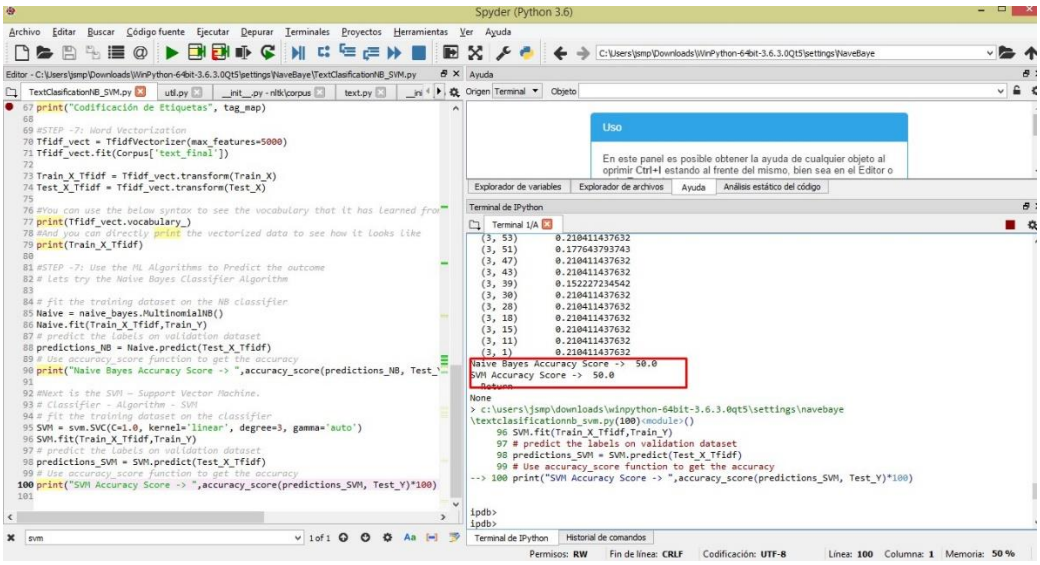


Figura 14. Resultado del procesamiento del archivo **"corpus_espaniol.csv"** de 8 registros

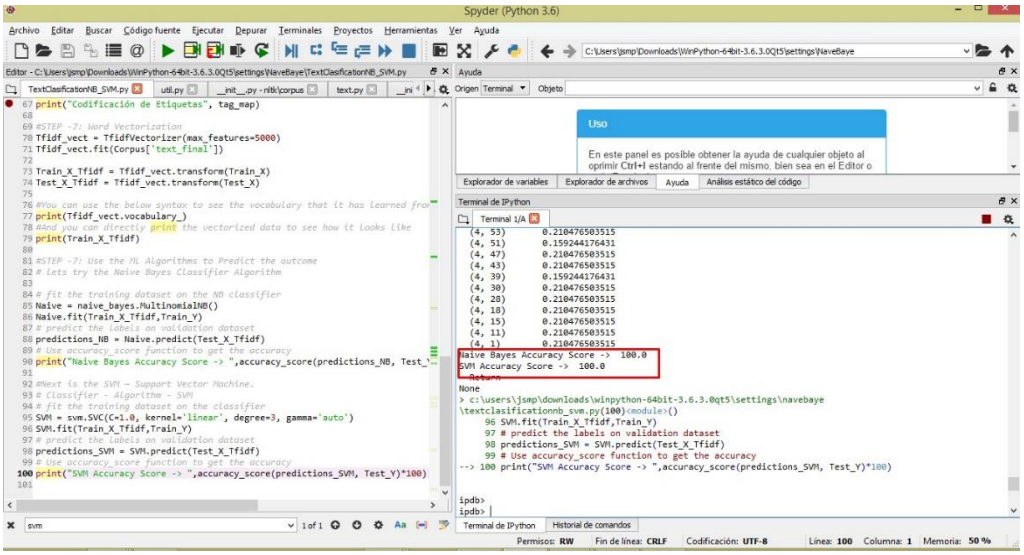


Figura 15. Resultado del procesamiento del archivo **"corpus_espaniol.csv"** de 8 registros, segundo procesamiento de datos.

Las **Figuras 11, 12, 13, 14 y 15** muestran tanto los insumos de pruebas para la implementación de los algoritmos SVM y NB, como sus resultados. Se evidencia así, el proceso ejecutado para hacer esta implementación, desde la preparación de los insumos, hasta su procesamiento por los algoritmos nombrados. En la siguiente sección se brindará mayor detalle del análisis efectuado.

6.2.5. Análisis y toma de decisiones

Luego de llevar a cabo el experimento se obtuvieron resultados relacionados con la precisión de clasificación hecha por los algoritmos. Se logró evidenciar que luego de procesar 4 registros en español (ver **Figura 11**), se obtuvo una clasificación a nivel de características de RNF, más precisa para el algoritmo

SVM ubicándose por encima de *Naive Bayes*. En términos exactos, SVM obtuvo una precisión en su clasificación del 100%, a diferencia de *Naive Bayes*, el cual obtuvo una precisión del 50% (ver **Figura 12**). Sin embargo, también se logró evidenciar que mientras más registros de textos se tengan para procesar, mayor es la precisión de clasificación del *Naive Bayes* (ver **Figura 14)**. En esta prueba se utilizaron 8 registros en el archivo **.csv**, dejando claro que, a mayor cantidad de registros por procesar, mayor es la efectividad de ambos algoritmos. Así, quedan los dos algoritmos en situaciones muy similares cuando se habla de precisión en la clasificación de RNF.

De acuerdo con el análisis hecho previamente, se tomó la decisión de implementar el método de extracción con el algoritmo supervisado de clasificación llamado *Support Vector Machine*.

PARTE V:

MÉTODO DE EXTRACCIÓN DE INFORMACIÓN

CAPÍTULO 7. MÉTODO DE EXTRACCIÓN DE INFORMACIÓN

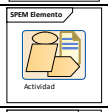
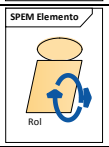
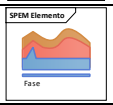
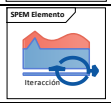
Con el objetivo de brindar una representación visual del método propuesto de manera formal y detallada, se siguió la implementación del estándar *Software Process Engineering Metamodel* (SPEM). A continuación, se brinda el contexto del estándar SPEM, utilizado como modelo de representación para el método propuesto.

7.1. MODELO DE REPRESENTACIÓN UML – SPEM

De acuerdo con Menendez y Castellanos (2008), SPEM fue creado como un estándar de alto nivel para definir procesos utilizados en el desarrollo de software orientado a objetos. Según los autores, inicialmente se desarrolló como un metamodelo independiente pero más tarde fue rediseñado como un modelo de tipo UML.

SPEM se fundamenta en la idea de que un proceso de desarrollo de software es una colaboración entre componentes, procesos y roles para llevar a cabo actividades previamente definidas con la finalidad de obtener un producto tangible y/o resultado. En la **Tabla 9**, se encuentran los elementos que hacen parte del estándar SPEM y se describen a continuación.

Tabla 9. Elementos SPEM

SPEM 2.0	Elemento
	Artefacto
	Actividad
	Producto de trabajo
	Herramienta
	Rol
	Fase
	Iteración

Es importante aclarar que los elementos SPEM contenidos en la **Tabla 9**, son una abstracción del estándar, es decir, que dichos elementos son solo los utilizados en el método propuesto.

Artefacto:
Puede considerarse como cualquier insumo ya sea corpus de información, documento, base de datos y/o archivos planos que puedan ser utilizados por una determinada fase dentro del método propuesto, para llevar a cabo actividades.

Fase:
La fase es un conjunto de actividades destinadas a ser ejecutadas por un determinado rol dentro de un proceso, cuyo objetivo principal es la generación de un producto de trabajo. Las fases comúnmente se encuentran relacionadas entre sí por restricciones de secuencialidad relacionadas con tiempo de ejecución, fechas e iteraciones.

Actividad:
Una actividad puede ser definida como un conjunto de tareas y/o pasos específicos dentro del método propuesto. La ejecución de una o varias actividades pueden generar un producto de trabajo al finalizar una fase durante la ejecución de un proceso.

Producto de trabajo:
El producto de trabajo puede ser cualquier elemento producido, consumido, o modificado por un proceso. Esto puede ser información, un documento, un modelo, código fuente y demás. También se podría definir como un producto tangible, resultado de la ejecución de una fase.

Herramienta:
Las herramientas en este caso, son aquellos aplicativos de software utilizados en la implementación de este método propuesto. Dichas herramientas podrían ser *Spyder*, *Word*, *Excel, Sklearn, Enterprise Architect* y *Python Frameworks*.

Rol:
El rol define las responsabilidades sobre el producto de trabajo y las actividades relacionadas a este. Para este método propuesto, los roles principales son: el científico de datos, quien es el encargado del pre procesamiento del Corpus a utilizar en el entrenamiento del algoritmo SVM y la generación del modelo de clasificación y el analista de requisitos, quien es el encargado de seleccionar el archivo de requisitos a procesar, desencadenar el procesamiento del documento de requisitos y evaluar los resultados obtenidos mediante los artefactos generados como producto de ejecución del método propuesto.

Iteración:
La iteración es una tarea y/o actividades que se puede repetir N veces para obtener un resultado esperado. Para este caso, son muy comunes las iteraciones en las actividades dentro de las fases de entrenamiento, procesamiento y resultado, ya sea para depurar textos, entrenar algoritmos y/o procesar resultados.

Representación del método:

El método propuesto es representado por medio de un diagrama SPEM, como se puede observar en la **Figura 16.** El orden de ejecución de cada una de las fases y sub-fases es estricto y obligatorio para obtener el resultado esperado, posterior al proceso de ejecución. Las actividades también deben ejecutarse en el orden especificado, ya que este método presenta una ejecución evolutiva en donde una actividad es el incremento de la otra con los insumos provistos por esta. Es importante aclarar que los roles que intervienen en este método son:

(i) *Científico de datos*, que se encarga de la definición y construcción de la fase de entrenamiento y sus actividades. Esta fase permite la generación del modelo de clasificación que es utilizado por las fases posteriores, y

(ii) *Analista de requisitos*, que se encarga de ejecutar las fases de procesamiento y resultado, así como las actividades pertenecientes a dichas fases en miras de obtener los artefactos resultantes del proceso de clasificación de RNF.

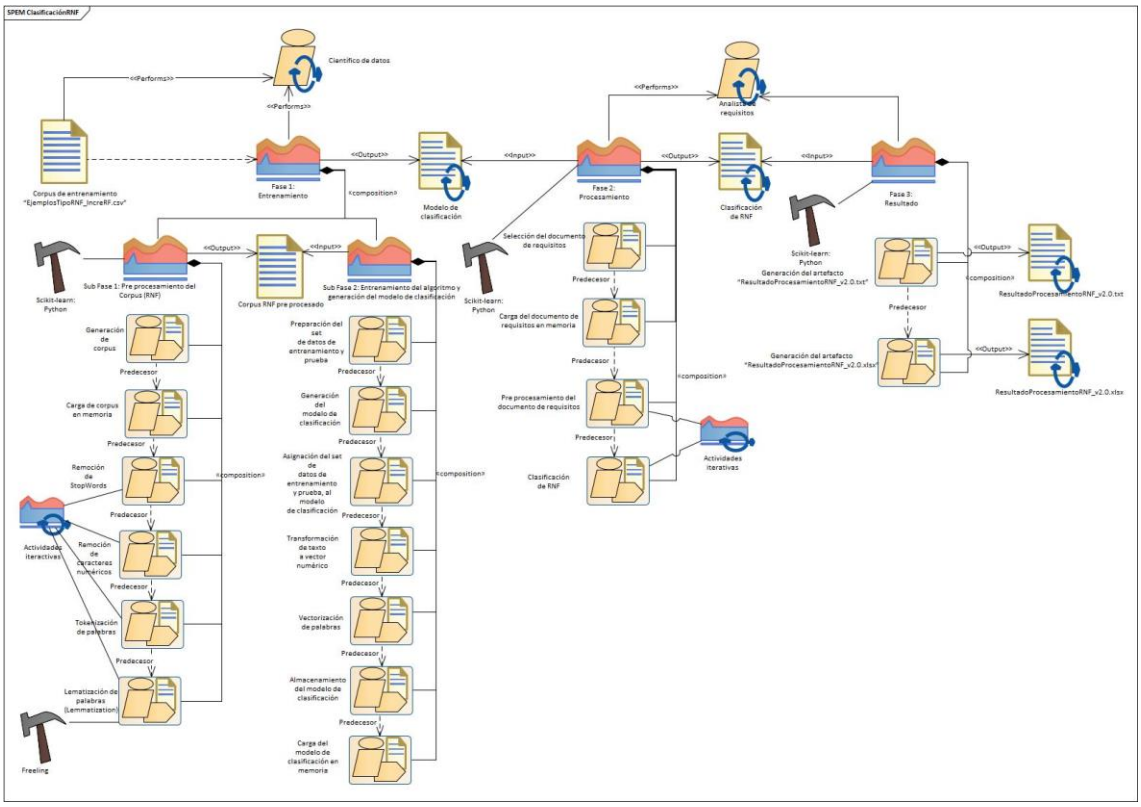


Figura 16. Método propuesto para la extracción de información (RNF) desde documentos de requisitos.

7.2. PASO A PASO DEL MÉTODO PROPUESTO

Partiendo de un contexto claro del método propuesto y su finalidad de llevar a cabo una clasificación de RNF a partir de documentos de requisitos, a continuación, se esboza el detalle de cada una de las fases, sub fases y actividades que lo conforman.

7.2.1. Fase 1: Entrenamiento

En la **Figura 17** se muestra la secuencia de actividades que son ejecutadas en esta fase:

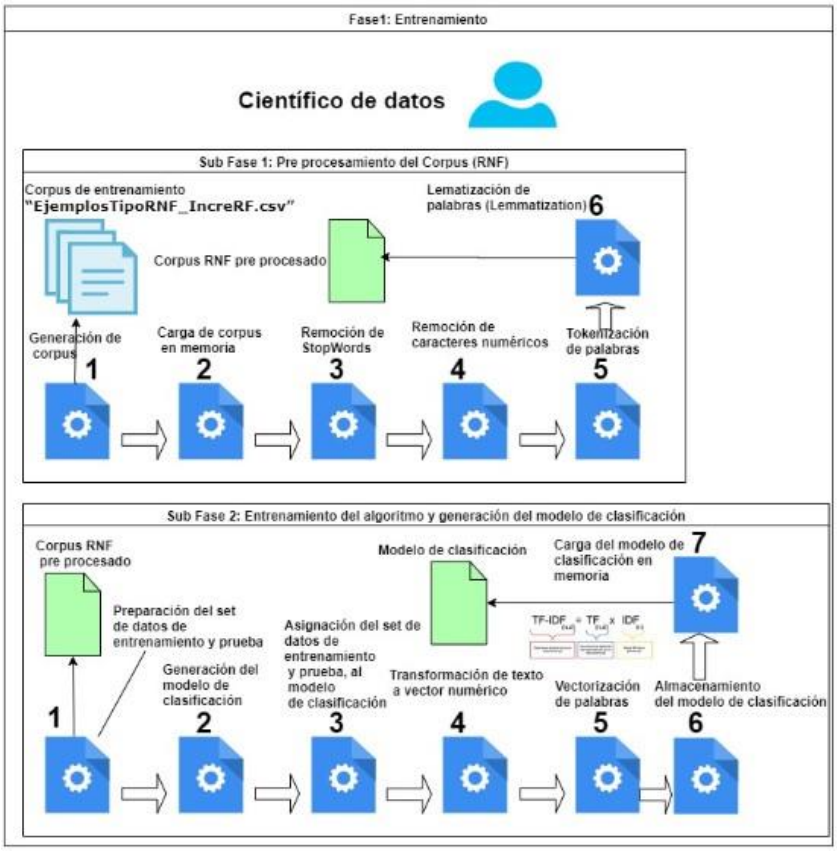


Figura 17. Fase 1: Entrenamiento

7.2.1.1. Sub Fase 1: Pre-procesamiento del Corpus (RNF)

Generación de corpus:

Esta actividad se realiza una sola vez durante la ejecución del proyecto y tiene como objetivo generar el corpus con el cual se va a entrenar el algoritmo de clasificación SVM. El artefacto .csv generado en esta actividad es llamado **"EjemplosTipoRNF_IncreRF.csv"**. Este artefacto contiene múltiples ejemplos de los tipos de RNF performance, disponibilidad, usabilidad, exactitud, seguridad y portabilidad. Adicional a los RNF, este artefacto también contiene múltiples registros de requisitos funcionales (RF), los cuales son considerados en el entrenamiento del algoritmo SVM en aras de descartar en el proceso de clasificación las frases, líneas y/o párrafos que se encuentran por fuera del ámbito de RNF anteriormente descrito.

En miras de brindar aún más detalle acerca de la generación del corpus .csv llamado **"EjemplosTipoRNF_IncreRF.csv"**, es importante resaltar que el presente artefacto se construyó con una gran variedad de registros de RF y RNF de los tipos anteriormente citados.

Los RNF y RF que componen el artefacto nombrado, fueron obtenidos de las siguientes fuentes:

- Documentos de requisitos de organizaciones de múltiples sectores empresariales.
- RNF y RF obtenidos de la web. Tanto de documentos de requisitos, como de ejemplos propuestos por los internautas.
- RNF y RF de autoría propia. Es decir, requisitos ficticios generados por el autor de la presente propuesta investigativa.

El único objetivo de tomar diversos ejemplos de RNF y RF, es alimentar el corpus llamado **"EjemplosTipoRNF_IncreRF.csv"**. El cual será utilizado posteriormente como insumo y base de conocimiento para el modelo clasificador utilizado en el método propuesto. Gracias a esto, será posible comparar el resultado del proceso de clasificación del método propuesto (grupo experimental), con el resultado del proceso de clasificación realizado por los analistas de requisitos (grupo de control).

Nota: La cantidad de registros de los RF incluidos dentro del artefacto .csv **"EjemplosTipoRNF_IncreRF.csv"**, no debe sobrepasar la cantidad de registros de los tipos RNF. Ya que esto podría generar inconsistencias y poca precisión en el proceso de clasificación de los RNF.

Carga de corpus en memoria:

En esta actividad se lleva a cabo la carga en memoria del corpus que se encuentra en el artefacto **"EjemplosTipoRNF_IncreRF.csv".** Este proceso de carga en memoria se realiza a través de la función de *Python* llamada *read_csv()*. Gracias a esto se obtiene un objeto llamado *DataFrame* el cual representa un archivo .csv en memoria. En este proceso de carga en memoria se realiza la eliminación de líneas en blanco y se cambia a minúscula todo el texto, en aras de facilitar el procesamiento.

Remoción de *StopWords:*

De acuerdo con Gessler y Shrivastava (2015), en el idioma inglés, es común eliminar *StopWords* tales como *a, of, and, to,* y otros artículos que probablemente no contribuyan a la comprensión semántica. En nuestro caso el texto a procesar está en idioma español. Por tanto, las *StopWords* a eliminar serían a, acá, ahí, como, con, sin, te, ti, tu entre otras *StopWords*, que entorpecen el procesamiento del algoritmo de clasificación. Las listas de *StopWords* están disponibles en varios idiomas para automatizar su identificación. Por tanto, en esta actividad es indispensable retirar las *StopWords* para llevar a cabo un proceso de clasificación de RNF más eficaz y preciso.

Nltk.corpus de *Python,* es la librería a utilizar para acceder al listado de *StopWords* en español. Una vez se tiene el listado de *StopWords*, es posible eliminarlas de las sentencias de texto mediante la función *words()*.

Remoción de caracteres numéricos:

En esta actividad simplemente se eliminan los caracteres que son diferentes al texto, es decir, los valores numéricos.

Tokenización de palabras:
Según Gessler y Shrivastava (2015), esta actividad se usa para separar palabras del cuerpo del texto. El texto sin formato se convierte en colecciones de tokens después de la tokenización, donde cada token generalmente es una palabra, frase, símbolo u otro elemento. La librería a utilizar en *Phyton* se llama *nltk.tokenize* y provee los métodos *word_tokenize* y *sent_tokenize* para dividir cadenas de texto en listas de palabras u oraciones.

Lematización de palabras (Lemmatization):
El objetivo de esta actividad es reducir las formas de inflexión de cada palabra en una base o raíz común según Gessler y Shrivastava (2015). La *Lemmatization* y *Stemming* está estrechamente relacionada con la derivación. La diferencia es que un *Stemming* opera con una sola palabra sin conocimiento del contexto y, por lo tanto, no puede discriminar entre palabras que tienen diferentes significados según la parte del habla. Por ende, se tomó la decisión de procesar las palabras con *Lemmantization* ya que *Stemming* sacrifica la precisión requerida en el proceso.

En esta actividad de lematización de palabras en *Python,* se utiliza la librería de *POS Tagging* llamada *Freeling*, la cual retorna un XML por cada palabra procesada. Este XML contiene los atributos de la palabra, lo cual permite obtener la categoría (verbo, adverbio, sustantivo, etc.) y el lema asociado. La decisión de utilizar *Freeling* como herramienta para la lematización se basó en la efectividad de la misma en idioma español comprobada por Marín (2019). El autor comprueba la efectividad de dicha herramienta para el idioma español, superando por mucho a la librería *nltk.stem – WordNetLemmatizer* de *Python*. Esta, a pesar de su precisión en el idioma inglés, no se encuentra lo suficientemente madura para procesar textos en idioma español.

7.2.1.2. Sub Fase 2: Entrenamiento del algoritmo y generación del modelo de clasificación

Preparación del set de datos de entrenamiento y prueba:
El Corpus que se encuentra compuesto por frases, líneas y/o párrafos de texto pertenecientes al artefacto .csv **"EjemplosTipoRNF_IncreRF.csv"**, se divide en dos conjuntos de datos: entrenamiento y prueba. El conjunto de datos de entrenamiento se utiliza para ajustarse al modelo y el conjunto de datos de prueba son usados para realizar las clasificaciones. Esto se hace a través del método *train_test_split* de la librería de *Python* llamada *sklearn.model_selection*. Los datos de entrenamiento tienen el 70% del corpus y los datos de prueba el 30% restante, esta distribución se obtiene al establecer el parámetro *test_size = 0.3.*

Generación del modelo de clasificación:
Con el objetivo de lograr una mayor precisión en el proceso de extracción y clasificación de RNF, se optó por crear un modelo de clasificación en la implementación, a través de la clase *Pipeline* de la librería de *Python* llamada *sklearn.pipeline* en miras de generar un modelo de entrenamiento más eficiente que genere mayor exactitud al momento de clasificar los RNF de documentos de requisitos. En esta actividad de generación del modelo se lleva a cabo la parametrización

del algoritmo de clasificación SVM, del método para análisis de texto llamado *Term Frequency - Inverse Document Frequency* (TFIDF) y por último concepto *OneVsRest Classifier,* el cual se centra en la clasificación de un texto contra un corpus de entrenamiento.

Nota: El modelo de clasificación es generado y almacenado en cada ejecución del proyecto.

Asignación del set de datos de entrenamiento y prueba, al modelo de clasificación:

Una vez se tiene el modelo debidamente creado y parametrizado, se procede con la asignación del set de datos de entrenamiento y prueba a través de la función *fit()* de la librería de *Python* llamada *sklearn.pipeline.* En este punto se cuenta con el modelo de clasificación listo para ser almacenado y posteriormente utilizado en el proceso de clasificación de RNF.

Transformación de texto a vector numérico:

Esta actividad se lleva a cabo dentro del modelo de clasificación una vez se asignan los datos de entrenamiento y prueba; en aras de transformar los datos categóricos (etiqueta) de tipo de cadena (*String*), en el conjunto de datos de tipo numérico que el modelo puede interpretar. Como se muestra en la **Tabla 10**, cada tipo de RNF que aparece en la columna "***Enconded Labels***" cuenta con un homónimo numérico que se ubica en la columna "***Raw Labels***". Una vez que el modelo internamente haga esta transformación de valores de tipo cadena a numéricos, se encontrará listo para iniciar con el proceso de extracción y clasificación de RNF, como se especifica en las siguientes actividades.

Como se puede observar en la **tabla 10**, los tipos de RNF que se busca identificar con este método son: seguridad, exactitud, disponibilidad, performance, portabilidad y usabilidad. Estos tipos de RNF son una abstracción de la propuesta de Kurtanovic y Maalej (2017), quienes especifican el listado de tipos de RNF que comúnmente son identificados en el proceso de educción de requisitos.

Tabla 10. Codificación de etiquetas.

Raw Labels	Enconded Labels
0	__seguridad__
1	__exactitud__
2	__disponibilidad__
3	__performance__
4	__portabilidad__
5	__usabilidad__

Es importante aclarar que para la clasificación de los diferentes tipos de RNF que puede tener un documento de requisitos, es necesario descomponer los tipos de RNF en las características que los conforman. Existen características que comparten el mismo tipo de RFN. Por ello es importante realizar una discriminación detallada de características por tipo RNF en aras de disminuir probabilidades de error. Aquí es donde el modelo de clasificación se hace responsable de esta primordial tarea. La

relación entre características y tipo de RNF, es posible evidenciarla en la **Figura 2**.

Vectorización de palabras:

Esta actividad se fundamenta en el uso del método llamado *Term Frequency - Inverse Document Frequency* (TFIDF) y se lleva a cabo dentro del modelo de clasificación, generado y parametrizado en pasos anteriores. Los autores Gessler y Shrivastava (2015) definen TFIDF como una medida ampliamente utilizada en la recuperación de información y el análisis de texto. En lugar de utilizar un corpus tradicional como base de conocimiento, TFIDF trabaja directamente sobre los documentos extraídos y trata estos documentos como el "corpus". TFIDF es robusto y eficiente en contenido dinámico, porque los cambios de documentos requieren solo la actualización de los conteos de frecuencia. En la **Figura 18** se muestra el esquema de funcionamiento de TFIDF.

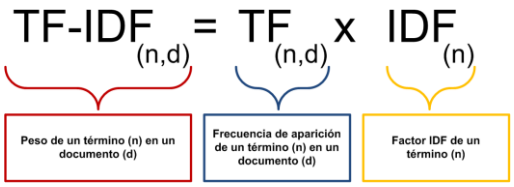


Figura 18. Esquema de Funcionamiento TFIDF (*Term frequency – Inverse document frequency*).

En esta fase TDIDF es usado para convertir una colección de documentos de texto en vectores de características numéricas.

- *Term Frequency*: resume la frecuencia con la que aparece una palabra en un documento.
- *Inverse Document Frequency*: Esta escala reduce las palabras que aparecen mucho en los documentos.

En términos generales, TF-IDF son puntuaciones de frecuencia de palabras que intentan resaltar palabras que son más interesantes. Es decir, más frecuente en un documento, pero no en todos los documentos.

Almacenamiento del modelo de clasificación:

En esta actividad se lleva a cabo la creación del artefacto .pkl llamado **"modelo_clasificacion_RNF.pkl".** Este artefacto representa el modelo de clasificación que será utilizado en la ejecución del método propuesto. En la creación de este modelo se utiliza el método *dump()*; el cual pertenece a la librería de *Python* llamada *sklearn.externals*.

Carga del modelo de clasificación en memoria:

En esta actividad se realiza la carga en memoria del modelo de clasificación (**"modelo_clasificacion_RNF.pkl"**) mediante el método *load(),* el cual pertenece a la librería de *Python* llamado *sklearn.externals*. Una vez se tiene el modelo de clasificación en memoria, este sirve como insumo para la siguiente fase de procesamiento.

7.2.2. Fase 2: Procesamiento

En la **Figura 19** se muestra la secuencia de actividades que son ejecutadas en esta fase:

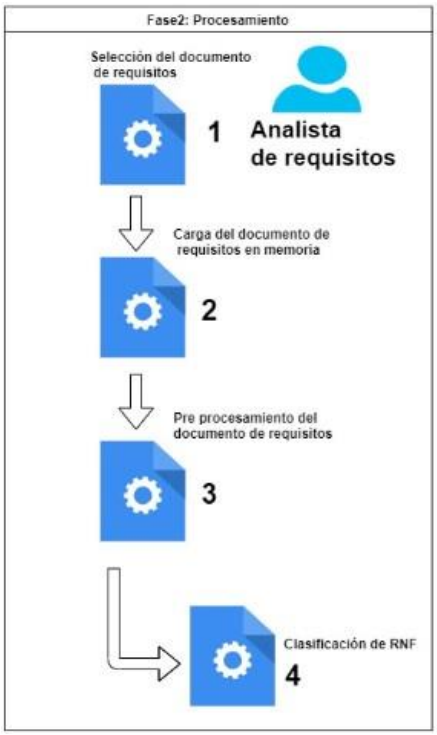


Figura 19. Fase 2: Procesamiento

Selección del documento de requisitos:

Una vez se encuentre en ejecución el aplicativo para la extracción y clasificación de RNF, se solicita al analista de requisitos a través de una ventana emergente, que seleccione el documento de requisitos en formato *Word .docx* a procesar. Como se logra observar en la **Figura 20** se evidencia el inicio de la ejecución del proyecto y la **Figura 21** solicita analista de requisitos que elija el documento de requisitos a procesar.

Figura 20. Mensaje de bienvenida al proyecto *Python* "Clasificación de RNF"

Figura 21. Mensaje para la selección del documento de requisitos a procesar

Carga del documento de requisitos en memoria:

En esta actividad se lleva a cabo la carga en memoria del documento de requisitos en formato *Word* .docx a procesar. Este proceso de carga en memoria se realiza a través del método de *Python* llamado *process()*, el cual pertenece a la librería *docx2txt*. Gracias a esto se obtiene un objeto de tipo *String,* el cual representa un archivo .docx en memoria. En este proceso de carga en memoria se realiza la eliminación de líneas en blanco y se cambia a minúscula todo el texto, en aras de facilitar su procesamiento. En la **Figura 22** se muestra el momento en el que el analista de requisitos debe seleccionar el documento de requisitos, para realizar su carga.

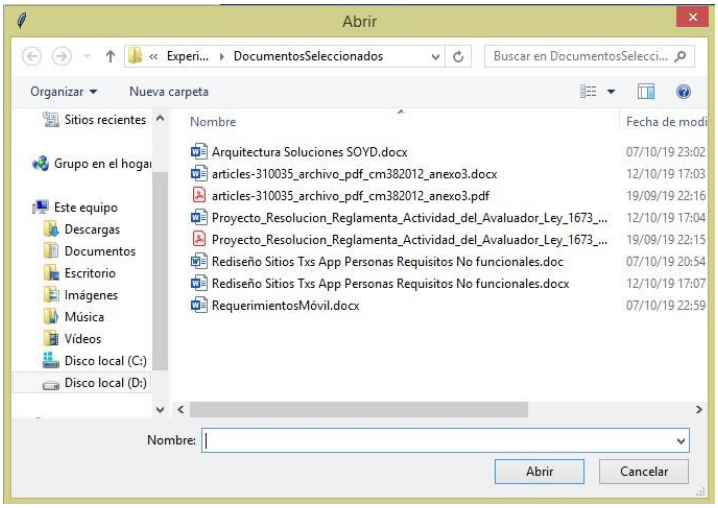


Figura 22. Explorador de Windows

Pre procesamiento del documento de requisitos:

En esta actividad se procede con el procesamiento del texto por cada salto de línea, es decir, tomando frases, líneas o párrafos completos. Esto se realiza por medio de la función *splitlines()* de *Python.* Cada frase, línea o párrafo es posteriormente tokenizada a través de la función *word_tokenize()*. Luego, se verifica que el vector generado por la ejecución de dicha función se encuentre en el rango de 10 a 200 caracteres, para evaluarlo y clasificar el tipo de RNF.

Clasificación de RNF:

Ya en esta instancia, se le entrega una cadena de texto al modelo de clasificación. El cual, mediante el método *predict()* de la librería *sklearn.pipeline* y haciendo uso del algoritmo de clasificación SVM, lleva a cabo la clasificación del tipo RNF (performance, usabilidad, disponibilidad, seguridad, exactitud y portabilidad) de una manera automática. Se entrega como resultado una clasificación de RNF que tienen alta probabilidad de ser requisitos no funcionales.

7.2.3. Fase 3: Resultado

Una vez el texto del documento de requisitos es procesado por el algoritmo de clasificación SVM, el aplicativo genera dos artefactos los cuales son un archivo plano .txt llamado **"ResultadoProcesamientoRNF_v2.0.txt"** y un archivo Excel llamado **"ResultadoProcesamientoRNF_v2.0.xlsx".** Ambos artefactos tienen la misma información, a nivel de párrafos, frases y/o líneas de textos que tienen alta probabilidad de ser RNF. La creación de dos artefactos en formato diferente se hace para facilitar el análisis e interpretación del analista de requisitos. La **Figura 23** indica el fin del proceso de ejecución del aplicativo mientras que la **Figura 24** muestra los artefactos generados como producto de la ejecución.

Figura 23. Finalización del proceso de clasificación de RNF

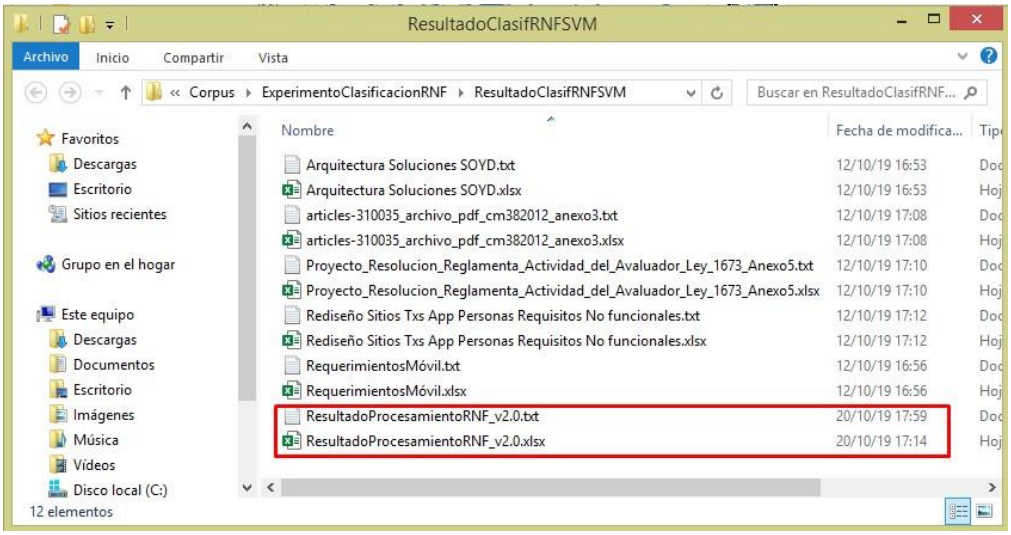


Figura 24. Artefactos generados como producto de la ejecución

Dicho resultado de la ejecución de las fases anteriores del método propuesto tiene como objetivo semi-automatizar la labor manual que llevan a cabo en la actualidad los analistas de requisitos en la fase de educción. En la **Figura 25** se especifican las actividades de esta fase:

Figura 25. Fase 3: Resultado

En esta última fase se ejecutan dos actividades encargadas de generar el resultado final del procesamiento del documento de requisitos.

Generación del artefacto "ResultadoProcesamientoRNF_v2.0.txt":
Una vez se tiene la información del procesamiento en memoria a través de una colección de datos en *Python*, se procede con la generación del artefacto llamado **"ResultadoProcesamientoRNF_v2.0.txt"**, a través del método *open()*. El cual cuenta con una línea con el potencial RNF, seguido de una etiqueta con el nombre del tipo de RNF. Esto se evidencia en la **Figura 26**.

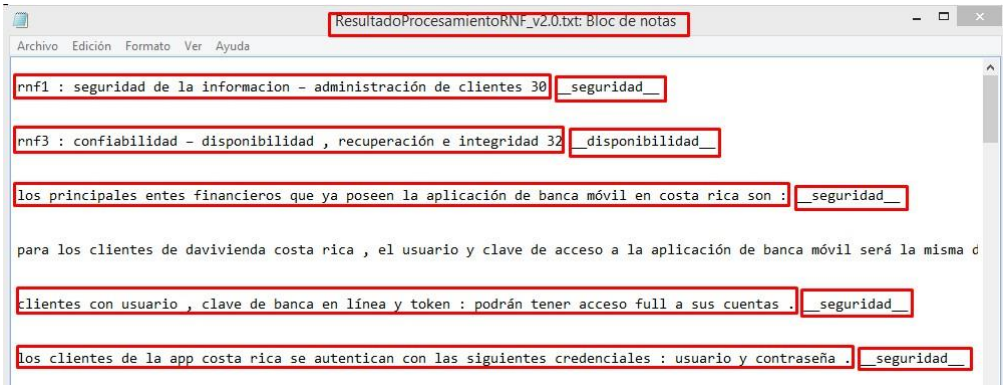


Figura 26. Artefacto generado "ResultadoProcesamientoRNF_v2.0.txt"

Generación del artefacto "ResultadoProcesamientoRNF_v2.0.xlsx":
En esta actividad se procede con la generación del artefacto llamado **"ResultadoProcesamientoRNF_v2.0.xlsx"**, a través del método *to_excel(),* el cual crea un Excel mediante un objeto *DataFrame*. Este artefacto cuenta con solo dos columnas, una llamada "Requisito No Funcional(RNF)" y la otra llamada "Tipo Requisito No Funcional", ambas columnas contenidas en una hoja

llamada "Requisito No Funcional(RNF)". Esto se puede evidenciar en la **Figura 27**.

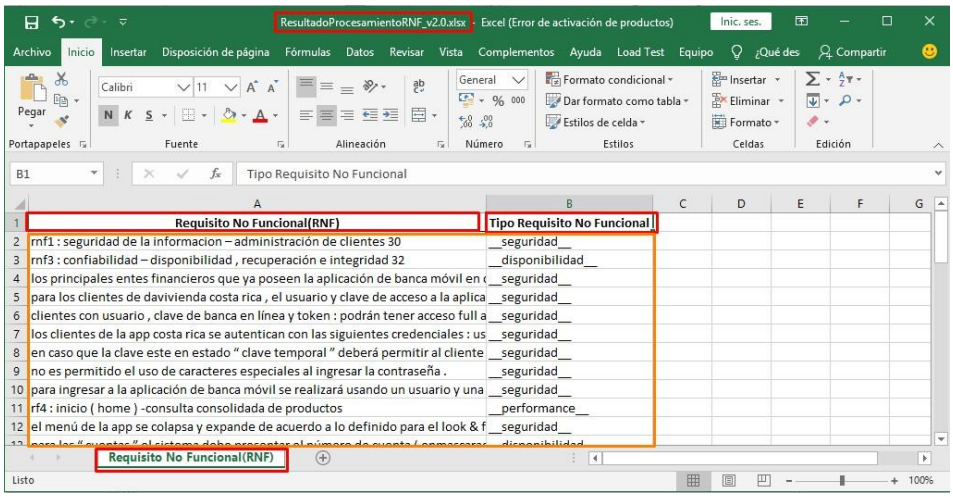


Figura 27. Artefacto generado "ResultadoProcesamientoRNF_v2.0.xlsx"

PARTE VI:

EVALUACIÓN DEL MÉTODO PROPUESTO

CAPÍTULO 8. EVALUACIÓN DEL MÉTODO PROPUESTO

8.1. DISEÑO EXPERIMENTAL

8.1.1. Método de validación propuesto: Método experimental

El objetivo de este método experimental es evaluar el enfoque propuesto para la extracción y clasificación de RNF desde documentos de requisitos escritos en lenguaje natural; dicho enfoque aplica técnicas, algoritmos y métodos de minería de datos, en miras de disminuir esfuerzo, costo y asegurar o mantener la calidad de los resultados obtenidos, en la fase de educción de requisitos.

8.1.2. Sujetos

- Área de Tecnología.
- Analistas de requisitos de una organización.
- Documentos de requisitos escritos en lenguaje natural.

8.1.3. Tratamientos

Se distribuyeron 9 profesionales de ingeniería de requisitos, en dos grupos, así:

- Grupo experimental: Conformado por un analista de requisitos, quien hará uso del método de propuesto para la extracción y clasificación de RNF, aplicando técnicas, algoritmos y métodos de minería de datos en la fase de educción de requisitos.
    - Analista de requisitos que procesa un grupo de documentos de requisitos escritos en lenguaje natural a través del método propuesto y diligencia un artefacto Excel por cada documento procesado.
    - El tiempo registrado equivale al tiempo de procesamiento del documento y tiempo de diligenciamiento del Excel.

- Grupo de control: Conformado por nueve analistas de requisitos, quienes ejecutan la labor manual de extracción y clasificación de RNF en la fase de educción de requisitos.
    - Analistas de requisitos que leen y procesan un grupo de documentos de requisitos escritos en lenguaje natural y diligencian un artefacto Excel por cada documento.
    - El tiempo registrado equivale al tiempo de lectura del documento y tiempo de diligenciamiento del Excel.

La manipulación de la variable independiente en términos de técnicas, algoritmos y métodos de minería de datos, hará que el grupo experimental obtenga resultados distintos en las variables dependientes esfuerzo, costo y calidad.

En este método experimental se cuenta con un grupo de control, para respaldar la validez de la propuesta, por tanto, se realiza una etapa de equivalencias de grupos en la cual, el grupo experimental lleva a cabo la ejecución del método propuesto, al procesar cinco documentos de requisitos de una manera semi automática. Luego, el grupo de control conformado por nueve analistas de requisitos procesa los cinco documentos de requisitos usados por el grupo

experimental. Vale la pena aclarar que cada analista procesa manualmente los cinco documentos de requisitos. Al final de esta etapa se comparan los resultados obtenidos por cada grupo a nivel esfuerzo, costo y calidad.

8.1.4. Variables

Independientes:
Técnicas, algoritmos y métodos de minería de datos.

Dependientes:
(i) Esfuerzo y costo, traducidos en tiempo de procesamiento (por un lado, tiempo dedicado en leer los documentos, y analizar, extraer y etiquetar los RNF; y por el otro lado, el tiempo que tarde el método propuesto en generar de manera semi-automática los RNF clasificados). Se evalúa tomando el tiempo en minutos desde que inicia el proceso hasta que finaliza, para realizar el comparativo.
(ii) Calidad, traducida en número de RNF identificados, los cuales se extraen y clasifican manualmente por cada analista de requisitos. Este resultado, se compara con la cantidad de RNF extraídos y clasificados por el método propuesto. Posteriormente, se evalúa el número de RNF recuperados para cada tipo de RNF.

8.1.5. Criterios de satisfacción

La hipótesis se validar en términos de costo, esfuerzo y calidad (Variables dependientes), de acuerdo a la técnica, método y algoritmo de minería de datos implementada (Variable independiente).

Los siguientes criterios segmentados, permiten medir los valores de las variables dependientes (costo, esfuerzo y calidad) en términos de satisfacción.

1. El esfuerzo y costo en términos de tiempo generado por el grupo experimental, debe ser inferior al generado por el grupo de control.

2. La calidad en términos de número de RNF recuperados por el grupo experimental, debe ser igual o superior al número de RNF recuperados por el grupo de control.

Se deben cumplir ambos criterios de aceptación, si se cumple sólo uno la hipótesis no puede ser validada.

8.2. EXAMEN Y MITIGACIÓN DE AMENAZAS A LA INTEGRIDAD

En la **Tabla 11** se muestran las posibles amenazas que se pueden presentar en esta propuesta y cómo mitigarlas.

Tabla 11. Amenazas y mitigación de las mismas

Amenaza	Mitigación
Desconocimiento del dominio del negocio por parte de los analistas de requisitos.	Evaluar previamente el conocimiento del dominio que deben tener los analistas de requisitos seleccionados.
Analistas de requisitos sin la experiencia necesaria para educir requisitos.	Evaluar y validar las habilidades de los analistas de requisitos seleccionados, para garantizar la efectividad de los resultados.
Poca disponibilidad de tiempo, al momento de ejecutar el método experimental.	Planear con antelación la ejecución del método experimental, para contar con la disponibilidad suficiente de los participantes.
Poca efectividad del enfoque propuesto.	Tomar los aspectos positivos del método propuesto y replantear nuevamente la propuesta investigativa.
Limitaciones para promover el nuevo enfoque.	Concienciar y promover las ventajas que podría conllevar la implementación de esta propuesta en el proceso de extracción y clasificación de RNF, desde documentos de requisitos en el proceso de educción.
Limitaciones tecnológicas para implementar la propuesta.	Identificar las herramientas tecnológicas necesarias-básicas, para llevar a cabo el método experimental.
Vulnerabilidad de los documentos organizacionales utilizados como insumos en el método experimental.	Definir los criterios mínimos de aceptación de los documentos de requisitos, que serán utilizados como insumos al momento de ejecutar el método experimental.

8.3. ESPECIFICACIONES DE LA VALIDACIÓN

En la validación se considera los siguientes 6 archivos. 5 archivos de documentos de requisitos y 1 archivo plano .csv de entrenamiento:

1- Documento de requisitos de caja de compensación: **"Arquitectura Soluciones SOYD.docx"**
2- Documento de requisitos del Ministerio de Educación Nacional: **"articles-310035_archivo_pdf_cm382012_anexo3.docx"**
3- Documento de requisitos de la Superintendencia de industria y comercio: **"Proyecto_Resolucion_Reglamenta_Actividad_del_Avaluador_Ley_1673_Anexo5.docx"**
4- Documento de requisitos para software bancario: **"Rediseño Sitios Txs App Personas Requisitos No funcionales.docx"**
5- Documento de requisitos para software bancario: **"RequerimientosMóvil.docx"**
6- Archivo plano .csv que contiene el corpus para el entrenamiento del algoritmo SVM: **"EjemplosTipoRNF_IncreRF.csv"**

Los archivos de documentos de requisitos, los cuales fueron enumerados previamente del 1 al 5, fueron obtenidos de diferentes sectores empresariales como el bancario, telecomunicaciones, cajas de comprensión y entes del estado. Estos documentos están escritos en lenguaje español y en su estructura y contenido cuentan con la definición de la necesidad y la especificación de requisitos funcionales y no funcionales. Son comúnmente usados por los analistas de requisitos, específicamente en la fase de educción de requisitos, para especificar la funcionalidad del sistema y sus restricciones de uso, traducidas en atributos de calidad o RNF.

Nota: Los documentos enumerados del 1 al 5, son los documentos que procesaran ambos grupos, el grupo de control y el grupo experimental. Donde el grupo de control hará uso de su conocimiento y experticia en la educción de requisitos para procesarlos; a diferencia del grupo experimental, el cual hará uso del artefacto 6 llamado **"EjemplosTipoRNF_IncreRF.csv"**, para llevar a cabo este procesamiento.

La finalidad de aplicación del método propuesto es semi-automatizar la labor manual realizada por los analistas de requisitos en la fase de educción de requisitos.

A continuación, se especifican los profesionales que hicieron parte del experimento. Sus nombres son resumidos con sus iniciales. Los Profesionales que realizaron el proceso manual de lectura, son:

- Juan Sebastián Salazar Mesa (JSSM)
- Diego Alejandro Marín (DAM)
- Jaiber Santiago Serna Varela (JSSV)
- Jhon Alexander Villada (JAV)
- Jorge Valencia (JV)
- Henry Muñoz(HM)
- Harbey Alirio Briceño (HAB)

Profesional que hace el proceso semi automatizado mediante el proyecto *Python*(Clasificador RNF) que representa el método propuesto:

- Nataly Ospina García (NOG)

Profesional quien realizó el proceso manual de lectura, para definir la línea base de clasificación de RNF:

- Juan Sebastián Morales Pérez (JSMP)

En la **Tabla 12** se muestra la estructura del artefacto Excel diligenciado por los grupos experimental y de control. El objetivo de dicho artefacto es recopilar los datos básicos del analista de requisitos, el tiempo y los tipos de RNF identificados en la ejecución del método experimental.

Tabla 12. Formato plantilla para la clasificación de RNF

Nombre Completo	
Cargo	
Empresa	
FechaDiligenciamiento	
Tiempo (minutos)	

ID-RNF	**Tipo RNF**	**Requisito no funcional**
RNF01	Performance	xxxxxxxxxxxxxxxxxx
RNF02	Seguridad	xxxxxxxxxxxxxxxxxx
RNF03	Usabilidad	xxxxxxxxxxxxxxxxxx

8.4. RESULTADOS

En la prueba realizada por los analistas de requisitos, se llevó a cabo el diligenciamiento del artefacto Excel llamado **"FormatoClasificacionRNF.xlsx".** Este artefacto tiene la estructura mostrada en la en **Tabla 12.** La cual se encontraba conformada por dos secciones: una sección que contiene la información básica del analista y el tiempo en minutos que éste tardó en leer el documento de requisitos y registrar los RNF hallados; y otra sección donde se registran los RNF por su ID, tipo de RNF y RNF.

Una vez se obtuvo toda la información de los analistas consolidada, se procedió con la abstracción de los diversos resultados con respecto a las variables dependientes, las cuales fueron tratadas de la siguiente forma:

- Esfuerzo y costo = Tiempo
- Calidad = Número de RNF identificados

Hablando en términos de tiempo, las **Tablas 13** y **14** consolidan el tiempo en minutos que tardaron los analistas de requisitos y el Clasificador RNF, en procesar los cinco documentos de requisitos.

Nota: Importante aclarar que el Clasificador RNF llevó a cabo un proceso semi automático de clasificación RNF, porque requirió de un analista que lo ejecutara.

Tabla 13. Consolidado de tiempo en minutos por analista de requisitos

Máx. de Tiempo Invertido	**DocumentoID**					
Analista de requisitos	**DOC1**	**DOC2**	**DOC3**	**DOC4**	**DOC5**	**Total**
DAM	22	14	10	11	39	96
HAB	22	21	33	25	44	145
HM	40	40	38	25	35	178
JSSV	35	25	23	45	35	163
JAV	44	31	15	20	21	131
JV	27	25	45	33	42	172
JSSM	55	45	52	40	58	250
PromedioTiempo_AnalistasXDoc	35	28.71	30.85	28.42	39.14	162.14

Tabla 14. Consolidado de tiempo del Clasificador RNF

Máx. de Tiempo Invertido	**DocumentoID**					
Analista de requisitos quien ejecuta proyecto *Python*	**DOC1**	**DOC2**	**DOC3**	**DOC4**	**DOC5**	
NOG	8	5	6.3	3.4	7	29.7
Porcentaje de tiempo	22.9%	17.4%	20.4%	12.0%	17.9%	18.3%

En la **Tabla 14**, se evidencia el porcentaje de tiempo que tardó el Clasificador RNF, con respecto al promedio de tiempo que tardaron los analistas de requisitos en procesar los documentos. Este porcentaje se obtuvo al dividir el tiempo que tardó el Clasificador RNF en procesar cada documento de requisitos entre el tiempo promedio que tardaron los analistas de requisitos en procesar cada documento. A continuación, se describe la ecuación utilizada para obtener el porcentaje:

- Porcentaje = TiempoProyectoPython/PromedioTiempo_AnalistasXDoc

Las Variables son las siguientes:

- Porcentaje = El porcentaje de tiempo que tarda el Clasificador RNF.
- TiempoClasificadorRNF = Es el tiempo en minutos que tarda el Clasificador RNF en procesar los documentos de requisitos.
- PromedioTiempo_AnalistasXDoc = Es el tiempo que tardaron los analistas de requisitos en procesar los documentos de requisitos.

En la **Figura 28,** se evidencia el consolidado del tiempo en minutos que tardo el clasificador RNF en procesar los documentos, en comparación con el tiempo promedio que tardaron los analistas. A simple vista es evidente la superioridad del clasificar RNF (grupo experimental), con respecto a la labor manual realizada por los analistas (grupo de control). Según lo explicado anteriormente.

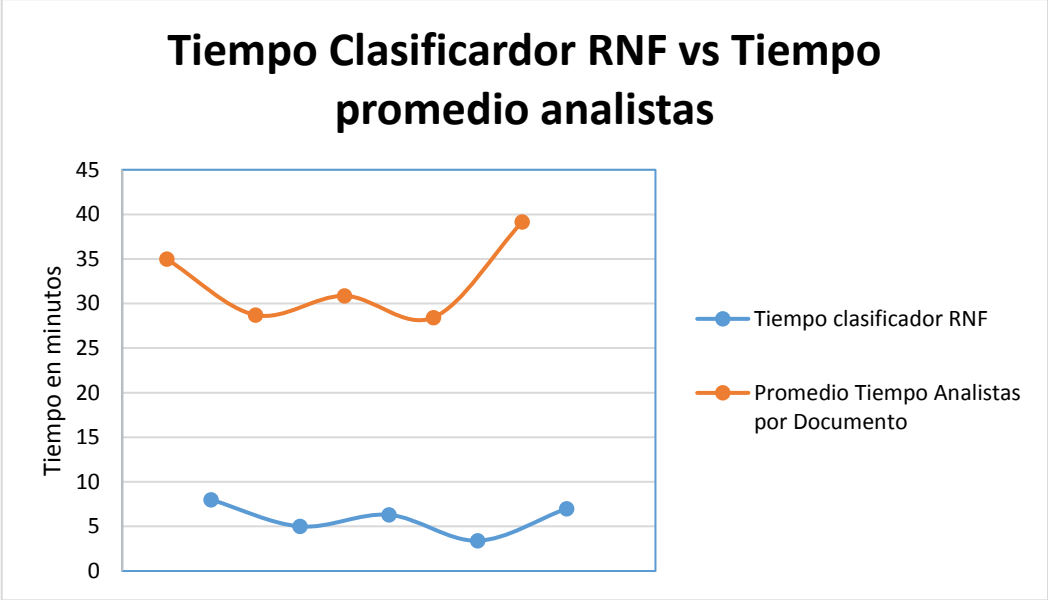


Figura 28. Tiempo Clasificador RNF vs tiempo promedio analistas

Continuando con la abstracción de resultados, en las **Tablas 15, 16** y **17** se consolida el número de RNF identificados por los analistas de requisitos, por el Clasificador RNF y por la línea base.

Tabla 15. Conteo de tipo RNF por analista de requisitos

Cuenta de Requisito no funcional		Tipo RNF						
Analista de requisitos	**DocumentoID**	**Disponibilidad**	**Exactitud**	**Performance**	**Portabilidad**	**Seguridad**	**Usabilidad**	**Total General**
DAM	DOC1	4	0	6	8	1	4	23
DAM	DOC2	4	3	3	8	9	4	31
DAM	DOC3	3	3	2	2	8	2	20
DAM	DOC4	6	3	4	3	3	1	20
DAM	DOC5	5	7	8	4	27	10	61
HAB	DOC1	0	0	5	1	3	1	10
HAB	DOC2	1	0	3	1	7	3	15

HAB	DOC3	2	0	2	1	14	3	22
HAB	DOC4	4	0	4	0	5	4	17
HAB	DOC5	5	0	3	0	10	6	24
HM	DOC1	1	15	8	0	4	3	31
HM	DOC2	2	8	2	3	10	7	32
HM	DOC3	3	7	2	1	8	2	23
HM	DOC4	10	4	7	0	6	5	32
HM	DOC5	8	14	0	0	8	7	37
JSSV	DOC1	1	4	5	3	3	5	21
JSSV	DOC2	1	1	4	6	9	4	25
JSSV	DOC3	3	3	2	1	8	3	20
JSSV	DOC4	6	1	6	1	8	1	23
JSSV	DOC5	2	1	1	1	1	1	7
JAV	DOC1	2	2	5	1	3	3	16
JAV	DOC2	2	1	3	1	6	3	16
JAV	DOC3	1	2	2	1	8	1	15
JAV	DOC4	2	1	3	0	1	0	7
JAV	DOC5	0	2	4	0	7	2	15
JV	DOC1	3	0	6	2	9	4	24
JV	DOC2	2	1	2	4	6	1	16
JV	DOC3	3	19	7	4	16	8	57
JV	DOC4	5	0	6	4	4	1	20
JV	DOC5	4	7	4	1	11	5	32
JSSM	DOC1	0	0	0	0	10	1	11
JSSM	DOC2	2	2	3	3	5	4	19
JSSM	DOC3	4	2	2	3	8	1	20
JSSM	DOC4	7	0	5	2	5	0	19
JSSM	DOC5	5	0	3	4	7	4	23
Total general		**113**	**113**	**132**	**74**	**258**	**114**	**804**

Tabla 16. Conteo de tipo RNF por el Clasificador RNF

Clasificador RNF								
Cuenta de Tipo RNF		**Tipo RNF**						
Analista de requisitos	**DocumentoID**	**Disponibilidad**	**Exactitud**	**Performance**	**Portabilidad**	**Seguridad**	**Usabilidad**	**Total general**
NOG	DOC1	1	0	11	0	13	2	27
NOG	DOC2	1	0	7	2	16	0	26
NOG	DOC3	5	1	8	1	26	3	44
NOG	DOC4	3	0	6	0	4	2	15
NOG	DOC5	6	2	15	3	20	3	49
Total NOG		**16**	**3**	**47**	**6**	**79**	**10**	**161**
Total general		**16**	**3**	**47**	**6**	**79**	**10**	**161**

Tabla 17. Conteo de tipo RNF por la línea base

Cuenta de Tipo RNF	Tipo RNF							
Analista de requisitos	**DocumentoID**	**Disponibilidad**	**Exactitud**	**Performance**	**Portabilidad**	**Seguridad**	**Usabilidad**	**Total general**
JSMP	DOC1	0	2	7	3	5	3	20
JSMP	DOC2	2	0	4	7	18	5	36
JSMP	DOC3	5	0	3	1	12	1	22
JSMP	DOC4	11	0	17	1	3	1	33
JSMP	DOC5	4	3	8	0	19	14	48
Total Juan Sebastián Morales		**22**	**5**	**39**	**12**	**57**	**24**	**159**
Total general		**22**	**5**	**39**	**12**	**57**	**24**	**159**

La **Figura 29,** muestra el consolidado de los tipos de RNF identificados por analistas de requisitos, Clasificador RNF y línea base. Esto en miras, que visualizar de una forma más resumida que tan lejos y/o cerca estuvo el proceso de clasificación del Clasificador RNF y los analistas de requisitos, con respecto a la línea base. La cual, representa el punto de referencia del presente análisis de resultados.

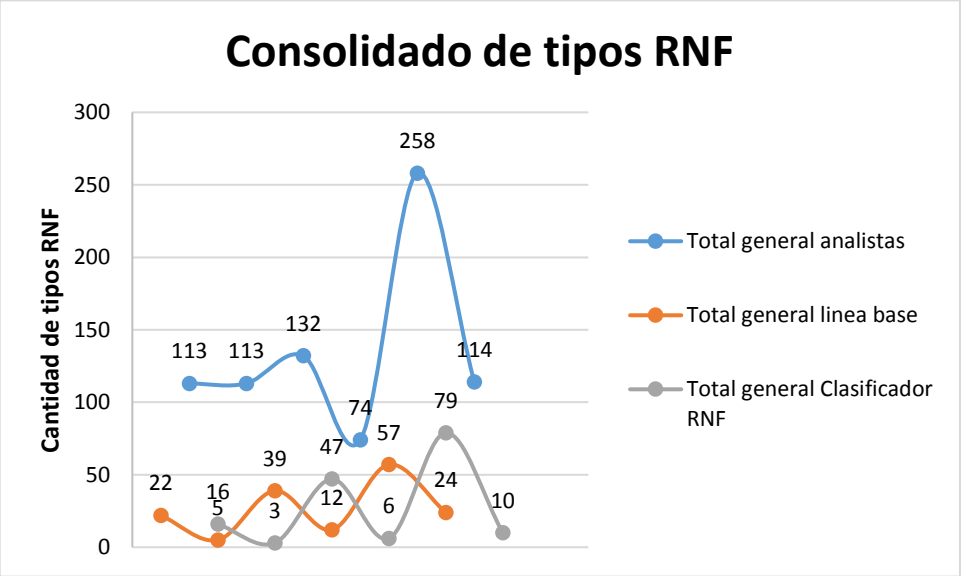


Figura 29. Consolidado total de RNF por analistas, línea base y Clasificador RNF

Como se puede observar en la **Figura 29,** en cuanto al total de tipos RNF clasificados, el proceso realizado por el Clasificar RNF es el que más se acerca al punto de referencia establecido por la línea base. Sin considerar, los tipos de RNF identificados por documento, ya que en esta instancia estamos hablando de cifras totales.

Una vez se tiene consolidado el número de RNF identificados por analistas de requisitos, por el Clasificador RNF y por la línea base, se realiza un promedio basado en el número de tipos de RNF identificados por los analistas de requisitos. Es decir, que se suman y promedian el número de RNF identificados

por analista, por documento y por tipo de RNF. De esta manera se obtiene como resultado el promedio de tipos RNF identificados por el total de analistas, al procesar cada documento de requisitos. Esto se puede evidenciar en la **Tabla 18**.

Tabla 18. Promedios de tipo RNF por documento de requisitos

Documento	Disponibilidad	Exactitud	Performance	Portabilidad	Seguridad	Usabilidad	Total
DOC1	1.571	3.000	5.000	2.143	4.714	3.000	19.429
DOC2	2.000	2.286	2.857	3.714	7.429	3.714	22.000
DOC3	2.714	5.143	2.714	1.857	10.000	2.857	25.286
DOC4	5.714	1.286	5.000	1.429	4.571	1.714	19.714
DOC5	4.143	4.429	3.286	1.429	10.143	5.000	28.429
Total -->	16.143	16.143	18.857	10.571	36.857	16.286	114.857

El promedio descrito en la **Tabla 18**, no se realizó para el número de tipos RNF identificados por el Clasificador RNF y por la línea base, ya que en ambos casos estamos se habla de un registro único. Por ende, no aplica promediar registros.

Luego de contar con el número de tipos RNF definidos y estructurados, se llevó a cabo la comparación de estos en dos segmentos, como se muestra en las **Tablas 19** y **20**.

En la **Tabla 19**, se realiza la comparación entre el número promedio de los tipos RNF identificados por los analistas de requisitos, y el número de tipos RNF identificados por la línea base. En este proceso de comparación se logra obtener el porcentaje de precisión y/o cercanía del proceso manual de identificación y clasificación de tipos de RNF realizado por los analistas, con respecto a la línea base. La ecuación para identificar el porcentaje de precisión es el siguiente:

- PorcentajeAnalistasVSLíneaBase=PromedioNumeroTipoRNF/NumeroTipoRNF

Las variables de la ecuación de validación son las siguientes:

- PorcentajeAnalistasVSLineaBase = Representa el porcentaje de exactitud, producto de la división del promedio del número de tipos RNF identificados y clasificados por los analistas de requisitos, entre el número de tipos RNF de la línea base.
- PromedioNumeroTipoRNF = Es el promedio resultante de la suma del número de tipos de RNF identificados y clasificados por los analistas de requisitos.
- NumeroTipoRNF = Es el número de tipos de RNF identificados y clasificados en la línea base.

Tabla 19. Porcentaje de precisión de la clasificación de tipos RNF de los analistas, con respecto a la línea base

Documento	Disponibilidad	Exactitud	Performance	Portabilidad	Seguridad	Usabilidad	Total
DOC1	0.0%	150.0%	71.4%	71.4%	94.3%	100.0%	97.1%
DOC2	100.0%	0.0%	71.4%	53.1%	41.3%	74.3%	61.1%
DOC3	54.3%	0.0%	90.5%	185.7%	83.3%	285.7%	114.9%
DOC4	51.9%	0.0%	29.4%	142.9%	152.4%	171.4%	59.7%
DOC5	103.6%	147.6%	41.1%	0.0%	53.4%	35.7%	59.2%
Total -->	73.4%	322.9%	48.4%	88.1%	64.7%	67.9%	72.2%

En la **Tabla 20** se realiza la comparación entre el número de tipos RNF identificados por el Clasificador RNF, y el número de tipos RNF identificados por la línea base. En este proceso de comparación se logra obtener el porcentaje de precisión y/o cercanía del proceso semi automático de identificación y clasificación de tipos de RNF realizado por el Clasificador RNF, con respecto a la línea base. La ecuación para identificar el porcentaje precisión fue la siguiente:

- PorcentajeClasRNFVSLineaBase = NumeroTipoRNFClas/NumeroTipoRNF

Las variables de la ecuación de validación son las siguientes:

- PorcentajeClasRNFVSLineaBase = Representa el porcentaje de exactitud, producto de la división del número de tipos RNF del Clasificador RNF entre el número de tipos RNF de la línea base.
- NumeroTipoRNFClas = Es el número de tipos de RNF identificados y clasificados por el Clasificador RNF.
- NumeroTipoRNF = Es el número de tipos de RNF identificados y clasificados en la línea base.

Tabla 20. Porcentaje de precisión de la clasificación de tipos RNF del Clasificador RNF, con respecto a la línea base

Documento	Disponibilidad	Exactitud	Performance	Portabilidad	Seguridad	Usabilidad	Total
DOC1	0.0%	0.0%	157.1%	0.0%	260.0%	66.7%	135.0%
DOC2	50.0%	0.0%	175.0%	28.6%	88.9%	0.0%	72.2%
DOC3	100.0%	0.0%	266.7%	100.0%	216.7%	300.0%	200.0%
DOC4	27.3%	0.0%	35.3%	0.0%	133.3%	200.0%	45.5%
DOC5	150.0%	66.7%	187.5%	0.0%	105.3%	21.4%	102.1%
Total -->	72.7%	60.0%	120.5%	50.0%	138.6%	41.7%	101.3%

En las **Tablas 19** y **20**, donde se encuentra el porcentaje de precisión de clasificación de tipos RNF, se logró evidenciar en ambos casos, tanto para los analistas de requisitos como para el Clasificador RNF, que el RNF que registra mayor cantidad de porcentajes de precisión en 0% por documento es el tipo RNF de exactitud. Esto permite concluir que este tipo de RNF podría tener un bajo impacto o ser poco común en los documentos de requisitos. Esto se puede evidenciar en la **Tabla 17** de la línea base, donde solo se encuentran 5 RNF de exactitud en los cinco documentos procesados, contando con un promedio de solo 1 RNF de exactitud por documento. Por ende, si dicho tipo RNF no es común en los documentos de requisitos, esto podría afectar de una u otra forma la definición del corpus de entrenamiento utilizado para generar el modelo de clasificación usado en el método propuesto.

Por otra parte, en la **Tabla 20** se puede observar que el documento "DOC3" presenta los porcentajes de precisión más desfasados en comparación con los demás documentos de requisitos. Esto podría deberse a que el "DOC3" es el que contiene la mayor cantidad de RF en su contenido, lo cual puede posiblemente generar ambigüedad en el proceso de clasificación de RNF. Adicionalmente, la sección de RNF dentro del documento de requisitos "DOC3", contiene múltiples RNF por párrafo y/o frase, es decir, que dichos RNF no se encuentran fragmentados y distribuidos por registro como en los demás documentos. Por consiguiente, esto afecta considerablemente el proceso de clasificación de RNF realizado por el Clasificador RNF, ya que este evalúa párrafos, frases y líneas de texto por salto de línea. El método propuesto, en su

implementación, no fue desarrollado para identificar múltiples RNF dentro de párrafos, frases y líneas.

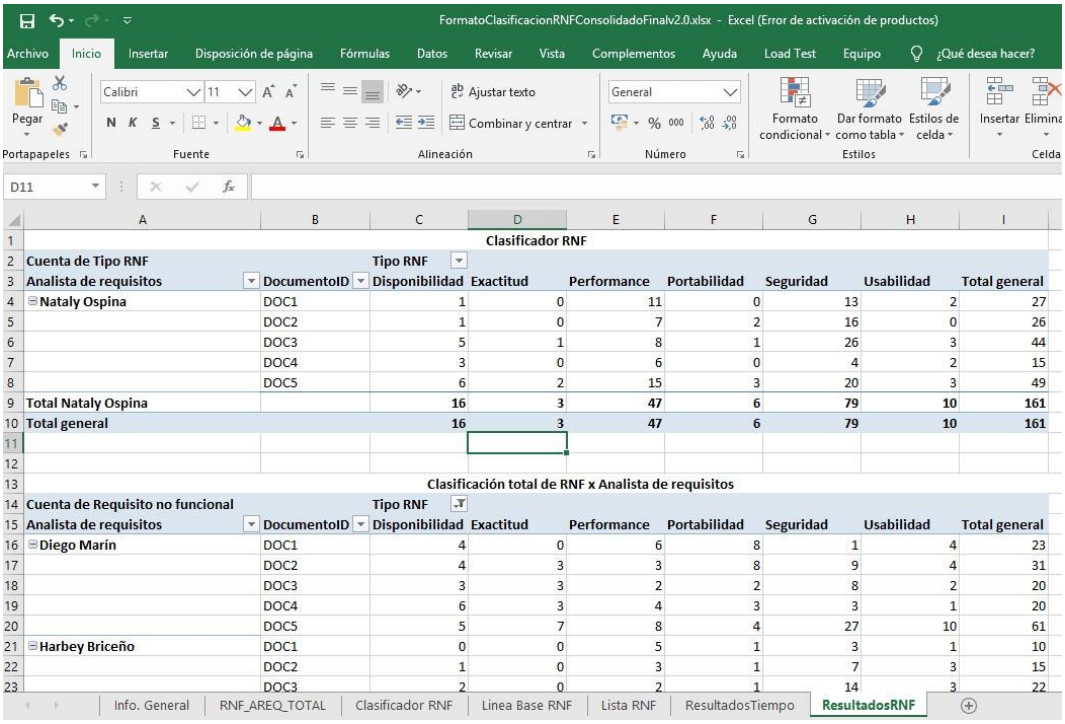
Para comprobar la hipótesis, se llevó a cabo un análisis de los resultados obtenidos en el experimento de la siguiente manera:

- Para este caso de estudio se procesaron 5 documentos de requisitos de diferentes áreas y organizaciones. Los 5 documentos fueron procesados por los grupos experimental y de control. Adicionalmente, se procesaron de forma manual los 5 documentos de requisitos en aras de obtener la línea base, la cual fue utilizada como punto de comparación y evaluación de los resultados obtenidos por los grupos.

- Las variables utilizadas para evaluar los criterios de satisfacción de la hipótesis son:
  - T01= Tiempo invertido por el grupo experimental en el procesamiento de los documentos de requisitos.
  - T02= Tiempo invertido por el Clasificador RNF en el procesamiento de los documentos de requisitos.
  - NUM_RNF01= Número de RNF identificados por el grupo de control.
  - NUM_RNF02= Número de RNF identificados por el grupo experimental.

- Las reglas establecidas para evaluar el cumplimiento de los criterios de satisfacción, son:
  - CRIT01=T01 < T02
  - CRIT02= NUM_RNF02 >= NUM_RNF01

- Este análisis de resultados se llevó a cabo tabulando la información generada por los grupos experimental y de control en la ejecución del experimento. La herramienta Microsoft Excel 2016 fue utilizada para recolectar esta información. Luego de tabular y distribuir la información en las hojas de cálculo, dentro del artefacto Excel llamado "**FormatoClasificacionRNFConsolidadoFinalv2.0.xlsx**", se realizó la creación de tablas dinámicas para obtener los resultados. En la **Figura 30** se evidencia lo descrito.



FormatoClasificacionRNFConsolidadoFinalv2.0.xlsx - Excel (Error de activación de productos)

Archivo | Inicio | Insertar | Disposición de página | Fórmulas | Datos | Revisar | Vista | Complementos | Ayuda | Load Test | Equipo | ¿Qué desea hacer?

Pegar | Portapapeles | Calibri | 11 | Fuente | Ajustar texto | Combinar y centrar | Alineación | General | Número | Formato condicional | Dar formato como tabla | Estilos de celda | Estilos | Insertar | Elimina | Celda

D11

	A	B	C	D	E	F	G	H	I
1				Clasificador RNF					
2	Cuenta de Tipo RNF		Tipo RNF						
3	Analista de requisitos	DocumentoID	Disponibilidad	Exactitud	Performance	Portabilidad	Seguridad	Usabilidad	Total general
4	Nataly Ospina	DOC1	1	0	11	0	13	2	27
5		DOC2	1	0	7	2	16	0	26
6		DOC3	5	1	8	1	26	3	44
7		DOC4	3	0	6	0	4	2	15
8		DOC5	6	2	15	3	20	3	49
9	Total Nataly Ospina		16	3	47	6	79	10	161
10	Total general		16	3	47	6	79	10	161
11									
12									
13				Clasificación total de RNF x Analista de requisitos					
14	Cuenta de Requisito no funcional		Tipo RNF						
15	Analista de requisitos	DocumentoID	Disponibilidad	Exactitud	Performance	Portabilidad	Seguridad	Usabilidad	Total general
16	Diego Marín	DOC1	4	0	6	8	1	4	23
17		DOC2	4	3	3	8	9	4	31
18		DOC3	3	3	2	2	8	2	20
19		DOC4	6	3	4	3	3	1	20
20		DOC5	5	7	8	4	27	10	61
21	Harbey Briceño	DOC1	0	0	5	1	3	1	10
22		DOC2	1	0	3	1	7	3	15
23		DOC3	2	0	2	1	14	3	22

Info. General | RNF_AREQ_TOTAL | Clasificador RNF | Linea Base RNF | Lista RNF | ResultadosTiempo | ResultadosRNF

Figura 30. Artefacto Excel para la consolidación de resultados

- En el proceso de validación del criterio CRIT01, se logró evidenciar en las **Tablas 13** y **14** del consolidado de tiempos, que el Clasificador RNF perteneciente al grupo experimental fue considerablemente más efectivo que el grupo de control. El Clasificador RNF, tardó solo el 18.3% del tiempo que consumió el grupo de control. Por tanto, se cumplió el CRIT01.
  La ecuación, reemplazando valores sería la siguiente:
  CRIT01= T01 < T02 → CRIT01= 29.7 minutos < 162.14 minutos.

- Es importante especificar que los tiempos de procesamiento especificados en la **Tabla 14**, corresponden al tiempo total invertido, desde el procesamiento de cada documento de requisitos por el Clasificador RNF, hasta el diligenciamiento del artefacto Excel por cada documento de requisito procesado por el analista. Por tanto, si se habla netamente del tiempo que tarda el Clasificador RNF en procesar cada documento de requisitos, se estaría hablando de un promedio de 22.8 segundos por documento. Lo cual, es un tiempo inmensamente bajo, si se compara con el tiempo en minutos que tarda el grupo de control en procesar cada documento de requisitos (Ver **Tabla 13**).

- En el proceso de validación del criterio CRIT02, se logró evidenciar como se muestra en las **Tablas 15** y **16,** que el grupo experimental fue más efectivo en calidad hablando en términos del número de RNF identificados y clasificados, con respecto al grupo de control. Cumpliendo así con la ecuación definida, la cual es "CRIT02= NUM_RNF02 >= NUM_RNF01". Sin embargo, en aras de darle mayor credibilidad y análisis a los resultados obtenidos, se generó una línea base. Esta línea base fue utilizada en miras de contar con un punto de referencia. Es decir, el número de RNF identificados y clasificados en la línea base permite determinar cuál grupo (experimental o de control) se encuentra más cerca de tener el número más acertado de RNF por tipo y por documento. Producto de esta validación

y haciendo uso de la línea base, se genera el análisis de las **Figuras 31** y **32**.

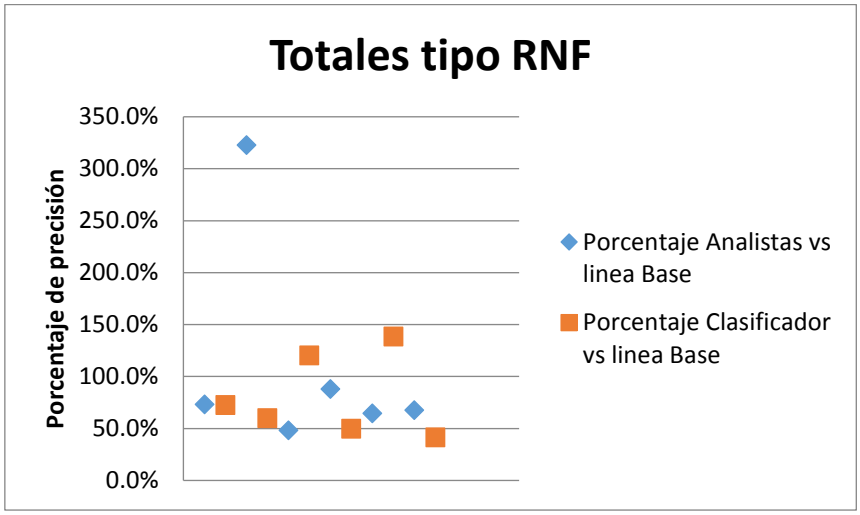


Figura 31. Porcentaje de precisión de analistas vs línea base y Clasificador vs línea base para el total de tipos RNF

- En la **Figura 31** se pueden observar dos símbolos de diferente color que representan el porcentaje de precisión de los analistas y Clasificador vs la línea base, con respecto al total de RNF identificados y clasificados por cada tipo RNF. Como se puede observar, los porcentajes de analistas, son los que más se acercan al 100% de precisión de la línea base, contando con 5 tipos de RNF cercanos al 100% de precisión; a pesar de esto, se logra observar en la parte superior izquierda de esta figura, que existe un porcentaje de precisión de los analistas que se encuentra muy por encima del definido por la línea base (100%). Este porcentaje de precisión corresponde al tipo RNF de exactitud y cuenta con un valor de 322.9% especificados (ver en **Tabla 19**)**;** este porcentaje de precisión se obtiene al dividir la suma de los promedios de RNF de exactitud identificados por los analistas de requisitos (ver **Tabla 18**), entre el número de RNF de exactitud identificados en la línea base (Ver **Tabla 17**). El resultado de este proceso permite concluir dos aspectos: (i) los documentos de requisitos no cuentan en su mayoría con RNF de exactitud y (ii) los analistas de requisitos no cuentan con el conocimiento necesario para identificar RNF de exactitud, lo cual se puede justificar a que este tipo de RNF es poco común en los documentos de requisitos.

  Nota: La **Figura 31** fue construida con los porcentajes totales contenidos en la fila "**Total -->**" de las **Tablas 19 y 20**.

- Lo anterior permite inferir una ventaja significativa en los porcentajes de precisión entregados por el Clasificador de RNF, ya que la tendencia de los porcentajes de precisión se acerca más al 100% de la línea base.

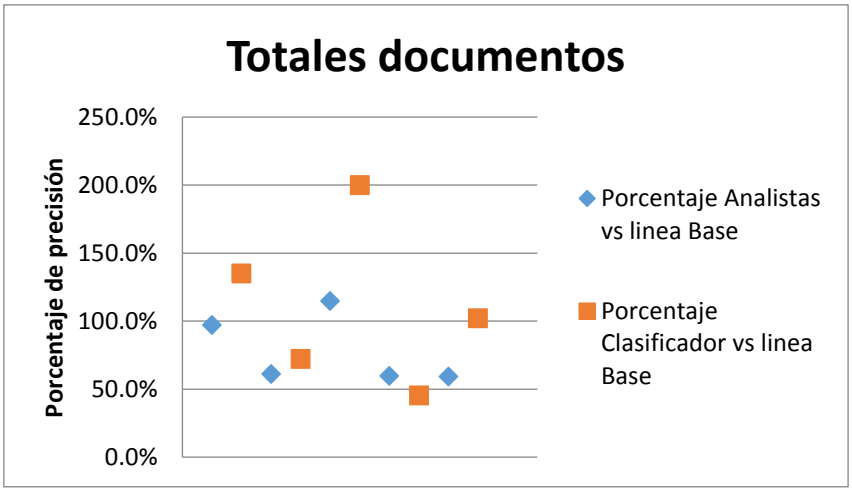


Figura 32. Porcentaje de precisión de analistas vs línea base y Clasificador vs línea base para el total de documentos

- En la **Figura 32** se evalúan los porcentajes de precisión de analistas y Clasificador RNF con respecto al definido por la línea base, considerando el total de RNF identificados y clasificados en cada documento procesado. Como se puede observar, los porcentajes de precisión de los analistas se acercan más a la totalidad de RNF identificados por documento en la línea base, con respecto a los porcentajes de precisión del Clasificador RNF. Brindando así una ventaja considerable en número de RNF identificados y clasificados por los analistas.

  Nota: La **Figura 32** fue construida con los porcentajes totales contenidos en la columna "**Total**" de las **Tablas 19 y 20**.

- Con base al análisis anteriormente descrito, se puede evidenciar que se cumple el CRIT01 en su totalidad. No obstante, el CRIT02 no se cumple en su totalidad, basándose en el resultado parcial que se obtiene al evaluar el porcentaje de precisión de los analistas y Clasificador RNF con respecto a la línea base, faltando un porcentaje de cumplimiento en la hipótesis planteada. No obstante, vale la pena resaltar que, si se compara el tiempo que se tarda en procesar los documentos de requisitos manualmente, con el tiempo que tarda el Clasificador RNF, es aceptable la aseveración de la hipótesis, ya que el margen de error en la identificación y clasificación de RNF puede ser controlable en miras de explotar el método aquí propuesto.

Observaciones adicionales:

Las observaciones especificadas a continuación, fueron detectadas durante la implementación del experimento y de alguna u otra forma influyeron en el resultado final de esta propuesta investigativa.

- La cantidad de RNF identificados por los profesionales fluctuó significativamente debido a la experiencia. Es decir, los analistas que tenían más años de experiencia (DAM, HM, JV) fueron los que más tipos de RNF identificaron y por ello, se acercaron más al número total de RNF identificados por la línea base. No obstante, el número total de tipos RNF identificados y clasificados por los analistas presentan fluctuaciones con respecto a la línea base.

- Los profesionales que desempeñan cargos de arquitectos, líderes técnicos y analistas desarrolladores experimentados (DAM, HB, JAV), fueron quienes tardaron la menor cantidad de tiempo en procesar manualmente los cinco documentos de requisitos. Vale la pena aclarar que, en esta instancia, no se comparó el tiempo de procesamiento de los analistas, con el tiempo de la línea base, ya que el objetivo era realizar la comparación con el tiempo que tardaría el Clasificador RNF.
- Los profesionales que desempeñan cargos de líderes técnicos y arquitectos (DAM, JSSV, HM), fueron quienes hicieron una identificación y clasificación de RNF más clara y concisa, en comparación con el resto de participantes. Es decir, que se acercaron más al número total de RNF identificados y clasificados por la línea base.
- Los profesionales que desempeñan cargos netamente técnicos como desarrolladores de software (JV, HB, JSSM, JAV), fueron quienes realizaron un proceso de identificación y clasificación de RNF más impreciso. Generando ambigüedad, imprecisión y vaguedad en los diferentes tipos RNF hallados, ya que en algunos casos asignaban más de un tipo RNF a un RNF, es decir un RNF fue registrado en dos ocasiones, una de ellas como tipo RNF seguridad y en otra como tipo RNF performance. Esta imprecisión radica en la diferencia evidenciada, entre el número total de RNF identificados por cada analista y el número total de RNF identificados en la línea base.
- La mayoría de los profesionales se limitaron a identificar los RNF que aparecen en la sección de RNF en cada documento de requisitos. Por tanto, no identificaron RNF que se encontraban embebidos dentro de los RF. Esto se evidencia al comparar el total de RNF identificados por los analistas, con los RNF identificados por la línea base. Teniendo mayor cantidad de RNF identificados y clasificados esta última.
- El Clasificador RNF en su resultado, producto del procesamiento de los documentos de requisitos, identifica y clasifica RNF no concluyentes, respecto a la línea base. Es decir, aunque el Clasificador RNF identifique y clasifique múltiples tipos de RNF que cumplen con la línea base, muchos de estos no aportan al proceso de clasificación de RNF, porque son párrafos, frases y líneas de texto que no aplican ninguna característica de tipos RNF.
- El corpus de entrenamiento utilizado por el Clasificador RNF, debería estar compuesto por igual número de registros, para cada tipo RNF. En miras de contar con un proceso de identificación y clasificación RNF más preciso y homogéneo. De lo contrario, se tendría un modelo de clasificación que se centraría en clasificar los tipos de RNF que contaran con mayor cantidad de registros en el entrenamiento del algoritmo SVM.

PARTE V:

CONCLUSIONES

CAPÍTULO 9. CONCLUSIONES Y TRABAJO FUTURO

9.1. CONCLUSIONES

El desarrollo de este trabajo de investigación permitió identificar en términos generales, las siguientes conclusiones:

- El campo de minería de datos ayuda considerablemente, a semi automatizar tareas de lectura y procesamiento de texto en lenguaje español, apoyando así la labor desempeñada por los analistas de requisitos, reduciendo esfuerzo, costo y manteniendo la calidad del proceso ejecutado.

- Las técnicas, algoritmos y métodos de tipo clasificación dentro del campo de minería de datos, son muy efectivos en el procesamiento de texto en lenguaje español. Esto se pudo evidenciar en la ejecución del experimento del método propuesto, con el procesamiento de documentos de más de 10 páginas en menos de 30 segundos.

En cuando al proceso de construcción del corpus, entrenamiento y experimentación con los algoritmos para el procesamiento de los documentos:

- La construcción de un corpus completo y estructurado es el insumo principal en la fase de entrenamiento de un algoritmo de clasificación. Gracias a esto es posible determinar el porcentaje de precisión en el proceso de clasificación de un algoritmo del mismo tipo.

- La eliminación de *StopWords*, espacios en blanco y puntuación en el corpus de entrenamiento. Disminuye la ambigüedad, vaguedad e inconsistencia al momento de realizar la clasificación de RNF en el método propuesto.

- La fase de entrenamiento en el método propuesto es fundamental y esencial para el éxito del proceso de clasificación de RNF. De no ser así, no se podría contar con un proceso de clasificación confiable con un margen de error aceptable por los ingenieros de requisitos.

- El algoritmo de clasificación SVM es comúnmente usado en el procesamiento de textos, dado que, gracias a su característica de aprendizaje automático, puede disminuir la probabilidad de error en las hipótesis que este plantee, al llevar a cabo la clasificación de RNF.

- Los módulos *NTLK* y *Sklearn* de *Python*, permiten acceder a todas las funcionalidades de minería de datos necesarias, para ejecutar y desarrollar el método propuesto.

Como resultado del análisis y procesamiento de documentos de requisitos, con el fin de encontrar potenciales RNF, se puede concluir que:

- La estructura de un documento de requisitos facilita o dificulta el proceso de identificación y clasificación de RNF.

- El tipo de RNF menos común en los documentos de requisitos, es el RNF de exactitud. Caso contrario ocurre con el tipo de RNF de seguridad, el cual es el más común entre los documentos de requisitos.

- Los profesionales con años de experiencia que desempañan cargos de líderes técnicos y arquitectos, son quienes mejor clasifican RNF desde documentos de requisitos, llegando al punto de segmentar un párrafo en varios RNF.

- La experiencia y el rol que desempeñe o haya desempeñado el analista de requisitos, influye enormemente en el proceso de identificación y clasificación de RNF, al procesar manualmente documentos de requisitos.

Finalmente, en cuanto al método semi-automático implementado, se puede concluir que:

- El Clasificador RNF es mucho más rápido que los analistas de requisitos, en función de procesar documentos de requisitos, identificando y clasificando RNF. Sin embargo, este puede ser impreciso al momento de clasificar RNF, ya que algunos de estos terminan siendo textos ambiguos y sin sentido que no aplican a ningún tipo de RNF.

- La eficiencia del Clasificador RNF al momento de clasificar e identificar tipos de RNF, depende principalmente del corpus de entrenamiento que sea usado en el algoritmo de clasificación SVM.

- A pesar de los buenos resultados obtenidos en el experimento por el Clasificador RNF, esta versión requiere mayor intervención para obtener 100% de confiabilidad en su procesamiento. Se requiere de la experticia del analista de requisitos para determinar la veracidad del proceso de clasificación realizado y mejoras en algunas reglas para el entrenamiento.

9.2.TRABAJO FUTURO

- Diseñar una interfaz visual intuitiva e interactiva para mejorar la experiencia del usuario final.

- Permitir el procesamiento de múltiples documentos de requisitos simultáneamente, en miras de optimizar el proceso de identificación y clasificación de RNF.

- Generar un ejecutable .exe que permita instalar el aplicativo Python en múltiples plataformas (Sistemas operativos). Esto facilitaría la portabilidad, implementación y uso del aplicativo desarrollado.

- Ampliar la utilidad del método propuesto a otros dominios, a través de la generación y utilización de diversos corpus de entrenamiento. Lo cual le permitiría al aplicativo poder ser explotado en otros sectores con el fin de semi automatizar y automatizar, laborales manuales que sean desgastantes y tediosas.

- Ampliar la gama de artefactos resultantes a mapas conceptuales y mapas mentales, que permitan un entendimiento más fácil y dinámico del proceso de clasificación realizado.

- Mejorar el proceso de identificación y clasificación de RNF, a través de la implementación integrada de otros algoritmos de clasificación como KNN, EM y NB. Esto con el fin de optimizar la precisión del proceso de clasificación de RNF.

- Ampliar la gama de clasificación de requisitos, pasando de clasificar RNF, a clasificar RNF y RF.

- Optimizar el pre-procesamiento de documentos de requisitos, buscando que el Clasificador RNF pase de clasificar texto por salto de línea, a desglosar párrafos, frases y líneas en múltiples RNF, siempre y cuando el texto cumpla con las características de RNF.

9.3. ACCIONES DE DIVULDACIÓN

- Escritura y aprobación de capítulo de libro de investigación "Analítica de Texto para procesar documentos de requisitos" del libro "Tecnologías del lenguaje humano: aplicaciones desde la Lingüística Computacional y de Corpus", editado por Sello Editorial Universidad de Medellín.

- Escritura de artículo "MÉTODO DE CLASIFICACÍON DE RNF DESDE DOCUMENTOS DE REQUISITOS" y sometimiento a INFONOR 2020.

REFERENCIAS BIBLIOGRÁFICAS

Alvarez, J. L., Mata, J., & Riquelme, J. C. (2004). Data mining for the management of software development process. *International Journal of Software Engineering and Knowledge Engineering*, *14*(6), 665–695. Retrieved from http://dx.doi.org/10.1142/S0218194004001841

Alzu'Bi, S., Hawashin, B., Eibes, M., & Al-Ayyoub, M. (2018). A Novel Recommender System Based on Apriori Algorithm for Requirements Engineering. *2018 5th International Conference on Social Networks Analysis, Management and Security, SNAMS 2018*, 323–327. https://doi.org/10.1109/SNAMS.2018.8554909

Arora, P. (2014). Application of Data Mining Techniques on Software Engineering Data for Software Quality, *3*(6), 6722–6725.

Aysolmaz, B., Leopold, H., Reijers, H. A., & Demirörs, O. (2017). A semi-automated approach for generating natural language requirements documents based on business process models. *Information and Software Technology*, *0*, 1–16. https://doi.org/10.1016/j.infsof.2017.08.009

Baudier, F. (2000). DATA MINING TECHNIQUES AND APPLICATIONS. *Sante Publique*, *12*(SPEC. ISS.), 5–10.

Bekkerman, R., El-Yaniv, R., Tishby, N., & Winter, Y. (2003). Distributional Word Clusters vs. Words for Text Categorization. *Journal of Machine Learning Research*, *3*(7–8), 1183–1208. Retrieved from http://www.crossref.org/jmlr_DOI.html

Casamayor, A., Godoy, D., & Campo, M. (2010). Identification of non-functional requirements in textual specifications: A semi-supervised learning approach. *Information and Software Technology*, *52*(4), 436–445. https://doi.org/10.1016/j.infsof.2009.10.010

Chopra, A., Prashar, A., & Chandresh, S. (2013). Natural Language Processing. *International Journal of Technology Enhancements and Emerging Engineering Research*, *1*(4), 131–134.

Cleland-Huang, J., Settimi, R., Zou, X., & Solc, P. (2006). The Detection and Classification of Non-Functional Requirements with Application to Early Aspects Bayesian Forecasting View project Architectually Significant Requirements View project The Detection and Classification of Non-Functional Requirements with A. https://doi.org/10.1109/RE.2006.65

Cleland-Huang, J., Settimi, R., Zou, X., & Sole, P. (2006). The detection and classification of non-functional requirements with application to early aspects. *Proceedings of the IEEE International Conference on Requirements Engineering*, 36–45. https://doi.org/10.1109/RE.2006.65

Das, M. A., Das, M. K., & Puthal, P. B. (2011). Improving Software Development Process through Data Mining Techniques Embedding Alitheia Core Tool, *2*(2), 629–632.

Deerwester, S., Furnas, G. W., Landauer, T. K., & Harshman, R. (1990). Indexing by Latent Semantic Analysis, 34. https://doi.org/10.1017/CBO9781107415324.004

Demmel, J., & Kahan, W. (2005). Accurate Singular Values of Bidiagonal Matrices. *SIAM Journal on Scientific and Statistical Computing*, *11*(5), 873–912. https://doi.org/10.1137/0911052

Fayyad, U., Piatetsky-Shapiro, G., & Smyth, P. (1996). From Data Mining to Knowledge Discovery in Databases. *AI Magazine*, *17*(3), 37. https://doi.org/10.1609/aimag.v17i3.1230

Génova, G., Fuentes, J. M., Llorens, J., Hurtado, O., & Moreno, V. (2013). A framework to measure and improve the quality of textual requirements. *Requirements Engineering*, *18*(1), 25–41. https://doi.org/10.1007/s00766-

011-0134-z
Gera, M., & Goel, S. (2015). Data Mining - Techniques, Methods and Algorithms: A Review on Tools and their Validity. *International Journal of Computer Applications*, *113*(18), 22–29. https://doi.org/10.5120/19926-2042
Gessler, N., & Shrivastava, A. (2015). *Data Science & Big Data Analytics: Discovering, Analyzing, Visualizing and Presenting Data*. (Kelly Talbot, Ed.). Indianapolis: John Wiley & Sons, Inc.
Gorunescu, F. (2011a). *Data Mining: Concepts and Techniques*. *Elsevier* (Vol. 12). https://doi.org/10.1007/978-3-642-19721-5
Gorunescu, F. (2011b). Introduction to data mining. *Data Mining: Concepts, Models and Techniques*, (August), 1–43. https://doi.org/10.1007/978-3-642-19721-5
Henderson, H. (2003). *Encyclopedia of Computer Science and Technology*. *Reference Reviews incorporating ASLIB Book Guide* (Vol. 17). https://doi.org/10.1108/09504120310503999
Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design science in information systems research. *MIS Quarterly: Management Information Systems*, *28*(1).
Işman, A. (2012). Technology and technique: An educational perspective. *Turkish Online Journal of Educational Technology*, *11*(2), 207–213. Retrieved from http://www.scopus.com/inward/record.url?eid=2-s2.0-84859405558&partnerID=40&md5=36cfa397738db04645f44a8a577c3cf6
Javed, B. (2010). Process Support for Requirements Engineering Activities in Global Software Development: A Literature Based Evaluation. *Engineering*, 6.
Jiawei, H., Micheline, K., & Jian, P. (2012). *Dm Concepts and Techniques Preface and Introduction*.
Joseph, S. R., Hlomani, H., & Letsholo, K. (2018). Data Mining Algorithms: An Overview. *International Journal of Computers & Technology*, *15*(6), 6806–6813. https://doi.org/10.24297/ijct.v15i6.1615
Kosterec, M. (2016). Methods of conceptual analysis. *Filozofia*, *71*(3), 220–230.
Kumar, A., Rath, H. K., Nadaf, S. M., & Simha, A. (2015). Novel Self-learning based Crawling and Data Mining for Automatic Information Extraction. *International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, 7.
Kurtanovic, Z., & Maalej, W. (2017). Automatically Classifying Functional and Non-functional Requirements Using Supervised Machine Learning. *Proceedings - 2017 IEEE 25th International Requirements Engineering Conference, RE 2017*, 490–495. https://doi.org/10.1109/RE.2017.82
Landhäußer, M., Körner, S. J., & Tichy, W. F. (2014). From requirements to UML models and back: How automatic processing of text can support requirements engineering. *Software Quality Journal*, *22*(1), 121–149. https://doi.org/10.1007/s11219-013-9210-6
Liang, W., Qian, W., Wu, Y., Peng, X., & Zhao, W. (2015). Mining Context-Aware User Requirements from Crowd Contributed Mobile Data. *Internetware*, 132–140. https://doi.org/10.1145/2875913.2875933
Mahmoud, A., & Williams, G. (2016). Detecting, classifying, and tracing non-functional software requirements. *Requirements Engineering*, *21*(3), 357–381. https://doi.org/10.1007/s00766-016-0252-8
Marín, D. (2019). EXTRACCION DE INFORMACIÓN DE DOCUMENTOS DE NEGOCIO ESCRITOS EN LENGUAJE NATURAL EN IDIOMA ESPAÑOL Y SU REPRESENTACIÓN EN UN MODELO CONCEPTUAL, *1*, 72.
Menendez, V. ., & Castellanos, M. . (2008). Software Process Engineering Metamodel (SPEM). *Revista Latinoamericana de Ingenieria de Software*, *3*(2), 92–100. https://doi.org/10.18294/relais.2015.92-100
Minku, L. L., Mendes, E., & Turhan, B. (2016). Data mining for software

engineering and humans in the loop. *Progress in Artificial Intelligence*, *5*(4), 307–314. https://doi.org/10.1007/s13748-016-0092-2

Naz, H., Motla, Y. H., Asghar, S., Ahmed, M., Shabbir Hassan, M., Mukhtar, M., & Javed, A. (2013). A systematic approach for web engineering practices by integrating data mining technique with requirement change management. *2013 IEEE 4th International Conference on Software Engineering and Service Science*, 373–376. https://doi.org/10.1109/ICSESS.2013.6615327

Ninaus, G., Felfernig, A., Stettinger, M., Reiterer, S., Leitner, G., Weninger, L., & Schanil, W. (2014). INTELLIREQ: Intelligent Techniques for Software Requirements Engineering. *European Conference on Artificial Intelligence*, 1161–1166. https://doi.org/10.3233/978-1-61499-419-0-1161

Raharja, I. M. S., & Siahaan, D. O. (2019). Classification of non-functional requirements using fuzzy similarity KNN Based on ISO / IEC 25010. *Proceedings of 2019 International Conference on Information and Communication Technology and Systems, ICTS 2019*, 264–269. https://doi.org/10.1109/ICTS.2019.8850944

Rehman, N. (2017). Data Mining Techniques Methods Algorithms and Tools. *International Journal of Computer Science and Mobile Computing*, *6*(7), 227–231. Retrieved from www.ijcsmc.com

Seaver, N. (2019). Knowing Algorithms. *DigitalSTS*, (February), 412–422. https://doi.org/10.2307/j.ctvc77mp9.30

Slankas, J., & Williams, L. (2013). Automated extraction of non-functional requirements in available documentation. *2013 1st International Workshop on Natural Language Analysis in Software Engineering, NaturaLiSE 2013 - Proceedings*, 9–16. https://doi.org/10.1109/NAturaLiSE.2013.6611715

Slonim, N., & Tishby, N. (2000). Document Clustering using Word Clusters via the Information Bottleneck Method. *Neural Computation*, 208–215.

Sommerville, I. (2010). *Software Engineering*. *Software Engineering*. https://doi.org/10.1111/j.1365-2362.2005.01463.x

Supakkul, S., & Chung, L. (2010). Visualizing non-functional requirements patterns. *2010 5th International Workshop on Requirements Engineering Visualization, REV 2010*, 25–34. https://doi.org/10.1109/REV.2010.5625663

Toussaint, M. B. & Y. (1995). Artificial intelligence tools for software engineering: Processing natural language requirements. *Communications*, *11*, 16.

Unterkalmsteiner, M., Gorschek, T., Feldt, R., & Klotins, E. (2015). Assessing requirements engineering and software test alignment - Five case studies. *Journal of Systems and Software*, *109*, 62–77. https://doi.org/10.1016/j.jss.2015.07.018

Xie, T., Thummalapenta, S., Lo, D., & Liu, C. (2009). Data Mining for Software Engineering. *Computer*, *42*(August), 55–62. https://doi.org/10.1109/MC.2009.256

Yin, B., & Jin, Z. (2012). Extending the problem frames approach for capturing non-functional requirements. *Proceedings - 2012 IEEE/ACIS 11th International Conference on Computer and Information Science, ICIS 2012*, 432–437. https://doi.org/10.1109/ICIS.2012.47

Zhang, W., Yang, Y., Wang, Q., & Shu, F. (2011). An empirical study on classification of non-functional requirements. *SEKE 2011 - Proceedings of the 23rd International Conference on Software Engineering and Knowledge Engineering*, 444–449. Retrieved from https://www.scopus.com/inward/record.uri?eid=2-s2.0-84862953188&partnerID=40&md5=5873bd4a0a3ca9752a1ffee19cd758ba

Zowghi, D., & Coulin, C. (2005). Requirements elicitation: A survey of techniques, approaches, and tools. *Engineering and Managing Software Requirements*, 19–46. https://doi.org/10.1007/3-540-28244-0_2

ANEXO 1: REQUERIMIENTO CARRUSEL CON CONTROL DE CAMBIOS

DETALLE DEL REQUERIMIENTO	
Título o Identificador	**Mejora en aplicativos de ingreso de venta para evitar carrusel**
Tipo de Requerimiento *(seleccione la opción)*	**Caja rápida __ Mantenimiento __ Mejora X**

DESCRIPCIÓN DEL REQUERIMIENTO

Para efectos de mitigar el impacto presentado por el llamado carrusel a nivel nacional se requiere diseñar y exponer en los sistemas de información de la compañía: CRM Legado, el Sistema de Soporte a la Operación, CRM eje cafetero, Siebel 8.1 y Siebel Móvil un WS (Web Service) que nos permita validar si nuestros usuarios han realizado retiros de sus servicios en un rango de tiempo inferior a 30 días y pretenden adquirirlos nuevamente dentro del mismo rango, dicha validación parte de una consulta inicial por medio del envió de información de entrada por parte de CRM Web y Móvil Interno.

- Dirección de consulta
- Producto a adquirir

Las validaciones también deben ser aplicadas en los intentos de ingreso desde los CRM's directos, claro está que bajo las premisas aplicadas al funcionamiento de cada uno de ellos.

El objetivo es que CRM Web y Móvil Interno reciban una respuesta por parte del WS de cada CRM y la interprete de manera que ventas pueda identificar mediante una alerta y mensaje emergente si es posible realizar el ingreso o si por el contrario éste deba ir a un proceso de manejo de excepción o prospección por motivo carrusel, el cual verifica internamente si se trata o no de una casuística de carrusel. La prospección en CRM Web y Móvil Interno debe quedar con motivo "Carrusel".

Adicionalmente se requiere que los canales que ingresan las ventas directamente a los CRM´s y a su vez posean perfiles especiales a solicitar, puedan contar con la posibilidad de extraer la información de un Log de trazabilidad para el posterior seguimiento y gestion desde el piloto designado por el área de procesos comerciales.

El repositorio debe ser expuesto por BI en una ruta especifica, con los campos relacionados a continuación y con la información ordenada ascendentemente por fecha:

- Nombre Cliente

- Cedula

- Dirección

- Teléfonos (Si aplica que pueda traer todos los registrados en la intención de venta)

- Servicios que no se vendieron (separados por coma)

- Fecha

- CRM - Departamento - Ciudad - Asesor - Canal de Venta - Nombre Canal de venta -Observaciones -Marcación para identificar que el cliente ya fue contactado -Resultado (Anulado, pendiente, instalado) Asimismo, para generar control y seguimiento de carrusel sobre los ingresos directos a los CRM's se requiere crear perfiles especiales para usuarios de CRM Legado y el Sistema de Soporte a la Operación que permitan de manera excepcionada realizar el ingreso una vez se apliquen las validaciones de segundo nivel. Las ventas ingresadas con estos usuarios no deben validar características aplicables a carrusel en los CRM's.
Justificación
Con este requerimiento se pretende atenuar las inconsistencias por motivo carrusel de venta, representando un ahorro para la compañía cercano a los $5.000.000.000 anuales.
Mercado objetivo
Negocio Hogares
Cobertura Geográfica
Nacional
Reglas de precio *(Es opcional, se describe si se requiere)*
Reglas de Negocio *(Es opcional, se describe si se requiere)*
Reglas de Facturación *(Es opcional, se describe si se requiere)*
RESPUESTA AL REQUERIMIENTO ***(Diligenciado por TIC y devuelve a la UEN)***
Este requerimiento fue asignado a:

La fecha de desarrollo será			La fecha de prueba será		
Día	**Mes**	**Año**	**Día**	**Mes**	**Año**
RAZON DE NO DESARROLLO (En caso de que aplique) *(Diligenciado por TIC y se debe devuelve a la UEN)*					

ANEXO 2: REQUERIMIENTO AUTOCONSUMOS

DETALLE DEL REQUERIMIENTO	
Título o Identificador	**Autoconsumos GPON**
Tipo de Requerimiento *(seleccione la opción)*	**Caja rápida** ☐ **Mantenimiento** ☐ **Mejora X** ☐

DESCRIPCIÓN DEL REQUERIMIENTO
(Diligenciado por la UEN, Se debe adjuntar Manual de Producto si existe)

Se requiere configurar autoconsumos para los planes Hogares en las diferentes tecnologías que tiene AAA a nivel nacional.

DESCRIPCIÓN DETALLADA DEL REQUERIMIENTO
(Diligenciado por la UEN)

Definición
(Describir en forma detallada lo que se pretende solucionar o realizar con el requerimiento)

Para que la fuerza de ventas, logre llegar a nuestros clientes con las ofertas que actualmente ofrecemos, requerimos que los planes Hogares en los tres servicios base (Televisión, Telefonía fija, Internet Banda Ancha) se configuren en sistemas de información con modalidad autoconsumos para cualquier tecnología.

Funcionamiento requerido
(Describir en forma detallada lo que se pretende solucionar o realizar con el requerimiento)

Permitir autoconsumos para los servicios de Telefonía, Televisión e Internet con acceso GPON para los planes vigentes de Hogares.

- Los servicios entregados para autoconsumo deben obedecer exclusivamente a servicios para uso interno de AAA, no para clientes particulares o que correspondan a un tipo de negociación especial como demostraciones, pruebas, etc.
- En ningún caso se debe ingresar como autoconsumo un servicio que se esté prestando a un cliente.
- Se registran a nombre de AAA, no de un tercero y/o filial.
- No se generan facturas ni IVA
- Debe estar asociado a un centro de actividad
- Los pedidos deben ser ingresados a nombre de AAA MMM Telecomunicaciones, bajo el Nit 00000000
- Los equipos asociados a los servicios deben estar adscritos a la cartera de cada empleado.
- Dentro de la facturación, deben estar asociados a cada centro de actividad (número asociado en el facturador).
- En principio deben estar en las instalaciones de AAA (propias o arrendadas).
- Los pedidos serán atendidos centralizadamente desde la Subdirección logística (próximamente se canalizarán a través de la mesa de ayuda).
- Las solicitudes deberán llegar a través de un Tiquete de soporte con asunto CR.
- Deberán quedar consignados en el ciclo xx de facturación, donde quedan exentos de impuestos, no generan cargos por interés por mora ni suspensión del servicio.
- Los servicios deberán estar marcados como indirecto y asociar una suscripción válida para AAA MMM Telecomunicaciones
- Aplica para los planes Hogares y Empresariales bajo tecnología GPON. Así sean planes Hogares, las órdenes de trabajo para la instalación, deben ser atendidos por grupos de empresas

- Permitir oferta de Internet Banda ancha especial en ventas o peticiones, tal como se encuentra actualmente
- Permitir configuración de segundo acceso.

Mercado objetivo

Clientes actuales y potenciales de AAA.

Cobertura Geográfica

Todas las ciudades con cobertura de servicios de AAA actuales y las que ingresen a futuro.

Reglas de precio
(Es opcional, se describe si se requiere)

- Aplican las mismas que existen hoy para el portafolio de productos y la que tenga el cliente.

Reglas de Negocio
(Es opcional, se describe si se requiere)

-

Reglas de Facturación
(Es opcional, se describe si se requiere)

- NA

RESPUESTA AL REQUERIMIENTO
(Diligenciado por TIC y devuelve a la UEN)

Este requerimiento fue asignado a:
(Diligenciar el nombre o nombres de los funcionarios asignados a la solución)

La fecha de desarrollo será			La fecha de prueba será		
Día	**Mes**	**Año**	**Día**	**Mes**	**Año**

RAZON DE NO DESARROLLO (En caso de que aplique)
(Diligenciado por TIC y se debe devuelve a la UEN)