UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

FACULTAD 3

COMPONENTE PARA LA ESTRATIFICACIÓN ESPACIO-TEMPORAL DE TERRITORIOS BASADA EN VARIABLES GEORREFERENCIADAS

Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas.

Autor (es):

Alejandro Alea González

Alberto Pérez Ojeda

Tutores:

Ing. Yadian Guillermo Pérez Betancourt

Ing. Roger Godofredo Rivero Morales

Ing. Liset González Polanco

"La Habana. Junio 2018"

"Año 60 de la Revolución"

DECLARACIÓN JURADA DE AUTORÍA

Declaramos ser los autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmamos la presente a los ____ días del mes de _____ del año _____.

_____
Alejandro Alea González
Autor

_____
Alberto Pérez Ojeda
Autor

_____
Ing. Yadian Guillermo Pérez Betancourt
Tutora

_____
Ing. Roger Godofredo Rivero Morales
Tutor

_____
Ing. Liset González Polanco
Tutor

AGRADECIMIENTOS

De Alejandro

Quiero agradecer a mi familia, en especial a mis padres por toda la confianza depositada en mí y por todo el apoyo que me han dado siempre en los momentos más difíciles de mi carrera.

A mi hermano Javier, que siempre ha sido mi ejemplo a seguir por su brillante inteligencia y dedicación con lo que quiere alcanzar.

A todos mis amigos por los momentos tan lindos que hemos compartido, en especial a los del edificio 58 que compartimos tiempos inolvidables y por ayudarme en múltiples situaciones.

A todos de corazón muchas gracias por acompañarme todos estos años.

A mis tutores en especial al profesor Roger.

De Alberto

Agradezco a toda mi familia, a cada de uno de ellos de ser como es.

A mis amigos por todo lo que han dicho o hecho pensando en mi bien.

DEDICATORIA

De Alejandro

A mi mamá Carmen.

A mi papá Adalberto.

A mi hermano Javier.

A todos mis abuelos.

De Alberto

A mi familia por toda la preocupación que han tenido sobre mis estudios (un pocoBastante exagerada), principalmente en esta etapa universitaria.

RESUMEN

El desarrollo de las tecnologías de la información y las comunicaciones en la actualidad, ha potenciado el uso de la información geográfica en disímiles ramas como la agricultura, la meteorología, el turismo y la medicina, implicando la utilización de los mapas como factor importante en la toma de decisiones. El uso de los Sistemas de Información Geográfica en la rama de la salud ha aumentado considerablemente, sin embargo, su utilización en el sector se limita en gran medida a presentar información en la cartografía, sin explotar en su totalidad la componente espacial y temporal de los datos. En la presente investigación se desarrolló una propuesta para la estratificación de territorios basada en indicadores de salud. Para ello se efectuó un estudio del panorama actual de los Sistemas de Información Geográfica y se realizó un análisis crítico de las herramientas existentes que soportan los procesos de estratificación de territorios. Se definió la arquitectura y los principales patrones de diseño utilizados. Se diseñó el algoritmo de agrupamiento ST-HDBSCAN basado en densidad que permite la estratificación a partir de tres o más dimensiones en los datos.

Se desarrolló un componente que permite integrar datos de naturaleza temática, espacial y temporal para el análisis y construcción de estratos; la solución tiene su base en las técnicas de agrupamiento de datos y contribuye a la identificación de riesgos de salud de los territorios cubanos y a la toma de decisiones en las entidades de salud.

Palabras claves: estratificación, agrupamiento espacio-temporal, sistemas de información geográfica, salud.

ABSTRACT

The development of information and communication technologies nowadays has promoted the use of geographical information in different branches such as agriculture, meteorology, tourism and medicine, implying the use of maps as an important factor in the decision making. The use of Geographic Information Systems in the field of health has increased considerably, however, their use in the sector is largely limited to presenting information in the cartography, without fully exploiting the spatial and temporal component of the data. In this research a proposal for the stratification of territories based on health indicators was developed. To this end, a study was made of the current landscape of Geographic Information Systems and a critical analysis of the existing tools that support the processes of stratification of territories was carried out. The architecture and the main design patterns used were defined. The ST-HDBSCAN density-based clustering algorithm was designed that allows stratification from three or more dimensions in the data.

A component was developed that allows the integration of data of thematic, spatial and temporal nature for the analysis and construction of strata; The solution is based on the clustering techniques and contributes to the identification of health risks in Cuban territories and decision-making in health entities.

Key words: *stratification, spatio-temporal clustering, geographic information systems, health*

TABLA DE CONTENIDO

TABLA DE CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

En la actualidad se ha evidenciado un aumento de forma exponencial de la cantidad de datos que se producen, debido a la aparición de nuevas tecnologías, dispositivos, medios de comunicación y aplicaciones. La geolocalización permite situar un objeto en un punto concreto del espacio (Beltrán y López 2012; Murazzo et al. 2016). De esta manera se crea información espacial, que según los datos espaciales o geodatos se refieren a localizaciones específicas en el espacio (Vanegas 2014). El acelerado desarrollo de información ha potenciado el uso de la información geográfica en disímiles ramas como la agricultura, la meteorología, el turismo y la salud pública, por lo que se han desarrollado herramientas con el objetivo de analizar estos datos.

Los Sistemas de Información Geográfica (SIG) son aplicaciones que permiten capturar almacenar, recuperar, transformar y visualizar información espacial del mundo real mediante herramientas (Vanegas 2014). Los datos relacionados con coordenadas geográficas (datos espaciales) y atributos descriptivos, su uso y demanda se encuentran en crecimiento en los últimos años (Vanegas 2014; Martínez, Torres y Moreno 2008). Los SIG permiten obtener una representación cartográfica del conjunto de datos en función de su componente espacial.

En el estudio de los geodatos es muy difícil pensar que exista independencia espacial entre los mismos, pues es muy probable que lo que ocurra en una unidad territorial repercuta casi directamente en otra. Por tanto, el valor que tome una variable en una unidad geográfica no solo está determinado por cuestiones internas, sino también por lo que ocurre en las unidades vecinas (Serrano 2000).

Al realizar un estudio sobre determinados fenómenos no solo es necesario analizar su comportamiento en el espacio sino también en el tiempo, ya que el tiempo es una dimensión para modelar la evolución de comportamientos geográficos. Los datos espacio-temporales representan el espacio y el tiempo. Las coordenadas espaciales y el tiempo están mutuamente relacionadas, un cambio en un atributo (espacial o no espacial) en el tiempo, implica la introducción de un dato histórico. Los fenómenos y actividades espacio-temporales dependen directamente de la transformación del espacio geográfico en un determinado período de tiempo (Vanegas 2014; W3C 2018). Las entidades geográficas presentan una ruta espacio-temporal, que inicia en el momento de la toma del geodato y termina en el momento en el que se destruye el geodato. Bajo la anterior descripción, los datos espacio-temporales se representan como una inserción de la dimensión tiempo en entidades geográficas concebidas, donde el espacio geográfico se organiza en capas temáticas que incluyen la información de captura en un tiempo determinado (Vanegas 2014; Du, Yu y Liu 2009).

Los datos geoespaciales se destacan por presentar autocorrelación espacial. Por tanto, los eventos mientras más cercanos están en el espacio son más semejantes. Sin embargo, la mayoría de los estudios espacio-temporales existentes, rara vez consideran autocorrelaciones espacio-temporales y heterogeneidades entre las entidades espacio-temporales y el acoplamiento en el espacio y el tiempo no ha sido bien resaltado (Deng et al. 2013). Por lo que se evidencia la importancia de conocer sobre la realidad de la variable en el espacio como objeto de análisis, sin ignorar la componente temporal. La componente temporal modela la naturaleza dinámica de los objetos espaciales, interpretando y aprovechando el conocimiento implícito en ambas componentes y evitando añadir incertidumbre en la calidad del resultado de un agrupamiento.

Se debe realizar una buena recuperación de información (IR, por sus siglas en inglés), para un correcto análisis y usos específicos de los datos. La IR consiste en encontrar material, por lo general documentos de una naturaleza no estructurada, generalmente texto, que satisfaga una necesidad de información dentro de grandes colecciones de datos almacenada en las computadoras (Chacón et al. 2016; Raghavan y Wong 1986; Xu y Croft 2000). La IR puede tener una definición muy amplia. Hechos tan cotidianos como buscar una persona en alguna red social, buscar sobre un personaje reconocido en algún buscador o examinar dentro del ordenador para encontrar un archivo, todo utiliza algún método o algoritmo para recuperar la información requerida (Chacón et al. 2016; Dominich 2000). La IR tiene varios métodos para realizar sus búsquedas, los más utilizados son: Modelo booleano, Modelo probabilístico, Modelo Vectorial, Big Data y Minería de datos (MD) (Chacón et al. 2016; Xu y Croft 2000).

La minería de datos es la exploración automática o semiautomática de los grandes conjuntos de datos con la intención de descubrir patrones (Pérez 2016). La minería de datos es uno de los pasos que componen del proceso de extracción del conocimiento (KDD, por sus siglas en inglés) (Russo et al. 2016). La minería de datos emerge de las áreas de base de datos (DB, por sus siglas en inglés), repositorio de datos (Data Warehouse) y de las grandes bases de datos (Big Data en inglés), como un proceso de extracción de información fundamentado en la matemática y la estadística (Rojas y Gomez 2015). Debido a la generación de petabytes de datos (puede ser clásica, espacial, temporal o de tipo híbrido) diariamente desde diferentes fuentes, se requieren tareas de minería de datos para extraer conocimientos de lo oculto, de modo que estos abundantes datos pueden utilizarse de manera significativa (Agrawal et al. 2016). La minería de datos, como el proceso de extracción de la información dentro de los volúmenes de datos, permite demostrar la utilidad de las cosas en las diferentes áreas

del conocimiento y del saber, antes de la puesta de su funcionamiento (Rojas y Gomez 2015).

La mejora tecnológica en la última década ha ayudado a recopilar gran cantidad de datos geoespaciales. El creciente volumen de datos necesita ser transformado en conocimiento esencial mediante el desarrollo de nuevas tecnologías capaces de analizar y modelar los datos de forma rápida y precisa. No se pueden implementar análisis convencionales como métodos estadísticos simples para explorar las perspectivas de evolución y la relación oculta entre y dentro de los datos espaciales digitales (Peña-Caro et al. 2016). Cuando en la base de datos se incorpora el dato espacial, dada la complejidad de estos tipos de datos, los objetos que se almacenan (puntos, líneas, polígonos), las estructuras de datos, se dificulta la utilización de la minería de datos tradicional (Russo et al. 2016). La Minería de datos espacio temporal (MDE) provee los mecanismos y herramientas para abordar los problemas anteriores que traían los nuevos datos (Russo et al. 2016). La MDE se puede definir como el proceso automático o semiautomático de seleccionar, explorar, modificar, visualizar y valorar elevados volúmenes de datos espaciales con el objeto de descubrir conocimiento. La aplicación de la MDE busca resolver diversos problemas mediante el descubrimiento de conocimiento, aplicando diversas técnicas donde los objetos espaciales cuentan además con características no espaciales y sirven como entrada a algoritmos de minería (Russo et al. 2016).

El agrupamiento o clustering es una técnica que se encuentra comprendida dentro de la MD, se basa en técnicas iterativas para agrupar los casos de un conjunto de datos dentro de grupos que contienen características similares. Estas técnicas son útiles para la exploración de datos, la identificación de anomalías en los datos y la creación de predicciones, para futuros comportamientos (Rojas y Gomez 2015; Parimala, Lopez y Senthilkumar 2011). Un enfoque común en la agrupación espacial, es utilizar la distancia geográfica como medida de similitud, derivado del concepto llamado primera ley de la geografía (Leong y Sung 2015). El agrupamiento consiste en una técnica de aprendizaje automático sin supervisión (Cervantes Suarez 2017). El propósito de realizar grupos es identificar agrupamientos “naturales” en un conjunto muy grande de datos y generar una representación concisa del comportamiento del sistema (Arango González, Jaramillo Morales y Jaramillo Escobar 2016). El entrenamiento del modelo se hace a partir de las relaciones existentes entre los datos y la de los grupos que identifica el algoritmo. Las técnicas de análisis exploratorio de datos espaciales y la determinación de los efectos espaciales de dependencia o autocorrelación, proponen el uso de técnicas de mapeo

de la distribución del agrupamiento espacial (Gómez y Sánchez Soria 2016; Yrigoyen 2003).

El método de agrupar datos asociados por puntos o características comunes pasando de lo general a lo particular, es la estratificación (Díaz Rosales, Cecilia Simón y Nuñez Cruz 2017). La estratificación es la división en clases discretas y homogéneas de variables o conjuntos de variables que se expresan en gradientes continuos. La clasificación del territorio en estratos aporta un marco espacial válido para el análisis en áreas grandes y heterogéneas (BUSQUÉ y MAESTRO 2014; Metzger et al. 2012). Los SIG, potencian la estratificación y constituyen una herramienta que a través de su estudio espacial y temporal, permiten encontrar la expresión de la variabilidad de los diferentes parámetros considerados en el proceso de zonificación (Tiria y Marcela 2014).

Todos los fenómenos geográficos evolucionan a lo largo del tiempo, tanto espacial como temporalmente, de modo que existe una necesidad de realizar tareas relacionadas con la agrupación en datos espacio-temporales (Agrawal et al. 2016). El agrupamiento espacio-temporal ha sido un tema delicado en el campo de minería de datos y descubrimiento de conocimiento, la misma puede emplearse para descubrir e interpretar las tendencias de desarrollo del fenómeno geográfico en el mundo real. El clustering (agrupamiento) espacio-temporal ha sido un importante campo de investigación de la minería de datos espacio-temporales y descubrimiento de conocimiento; ha atraído gran atención de los investigadores en ciencias de la computación y la ciencia de la geo-información (Deng et al. 2013; Shekhar, Vatsavai y Celik 2009; Kisilevich et al. 2009). El agrupamiento espacio-temporal es un proceso de agrupación de objetos basados en su similitud espacial y temporal (Kisilevich et al. 2009). El agrupamiento espacio-temporal pretende detectar entidades espacio-temporales similares a partir de una base de datos espacio-temporales y, además, descubrir los grupos espacio-temporales. Estas agrupaciones pueden identificar las tendencias evolutivas o reglas relacionadas con el fenómeno geográfico (Deng et al. 2013). Los algoritmos de agrupamientos diseñados para datos espacio-temporales pueden utilizarse en muchas aplicaciones. Estas aplicaciones se pueden observar en sistemas de información geográfica, imágenes médicas, y la predicción meteorológica, entre muchas más (Birant y Kut 2007).

Las técnicas de agrupamiento tradicionales son ineficientes en la minería espacial-temporal o espacial de datos debido a que no incorporan la heterogeneidad de los dominios espaciales y temporales. Por lo tanto, se necesitan nuevas técnicas para hacer frente a estos retos y aportar soluciones eficaces para analizar los datos espacio-temporales (Wang 2014). El poco uso de la componente temporal por parte de las

técnicas de agrupamiento tradicionales, demostrado en la presente investigación, impide determinar el comportamiento de fenómenos a través de los años. Un ejemplo en concreto de esto se pone en práctica cuando en un año determinado se observa que una enfermedad afecta una región determinada, pero esto no implica que siempre va a suceder, en cambio, si durante varios años consecutivos se observa este comportamiento, se pueden obtener conclusiones significativas y tomar decisiones claves con respecto a las observaciones realizadas.

En esta investigación, se desarrolla un marco unificado para el análisis de agrupaciones de datos espacio-temporales y se propone un nuevo algoritmo de agrupación espacio-temporal basado en densidad.

Identificándose como **problema a resolver**, el insuficiente tratamiento a la componente temporal en la estratificación espacio-temporal de territorios, limita la detección de fenómenos locales y globales en estudios salubristas.

Como **objeto de estudio** se tiene el proceso de estratificación de territorios en los SIG, y el **campo de acción** en los algoritmos de agrupamiento espacio-temporal.

Para darle solución al problema anteriormente descrito se plantea el siguiente **objetivo:** desarrollar un componente para la estratificación espacio-temporal de territorios basada en variables georreferenciadas que permita detectar fenómenos locales y globales en estudios salubristas.

Objetivos específicos:

1. Construir el marco teórico referencial relacionado con la estratificación de territorios.
2. Diseñar un algoritmo de agrupamiento espacio-temporal sobre polígonos.
3. Implementar un complemento para la estratificación espacio-temporal de territorios basada en variables georreferenciadas sobre los SIG QGIS.
4. Verificar la solución informática propuesta aplicando diferentes pruebas y métricas.

En el desarrollo del presente trabajo de investigación fueron utilizados los ***métodos científicos*** siguientes:

- **Histórico-Lógico:** permitió realizar un estudio de los principales algoritmos que realizan procesos de agrupamiento en la estratificación de territorios.

- **Análisis y síntesis:** se utiliza para identificar y analizar las diversas funcionalidades de los SIG que pueden ser aplicadas al proceso de estratificación territorial y su posterior síntesis, conforme a las necesidades de Cuba en el sector de la salud.
- **Modelación:** se emplea para mostrar los diversos diagramas que se construyen como resultado del proceso de ingeniería de software.

Estructura del trabajo

El presente trabajo está estructurado de la siguiente forma: introducción, tres capítulos, conclusiones, recomendaciones, referencias bibliográficas, glosario de términos y anexos. A continuación, se muestra una breve descripción de cada uno de los capítulos.

Capítulo 1: Fundamentos teóricos y metodológicos de la minería de datos espacio-temporal y la estratificación de territorios.

En este capítulo se presentan elementos teóricos relacionados con el proceso de estratificación para lograr un mejor entendimiento del trabajo a desarrollar. Se hace una valoración sobre los SIG y sobre las principales herramientas que soportan procesos de estratificación. Además, se realiza un estudio de la metodología, herramientas, tecnologías y lenguajes a utilizar en el desarrollo de la solución.

CAPÍTULO 2: Algoritmo de agrupamiento para datos espacio-temporales.

En este capítulo se realiza una descripción general de la solución propuesta, se especifican los requisitos funcionales y no funcionales que se tendrán en cuenta para la implementación de la misma, y se detallan aspectos relacionados con su diseño y arquitectura. Se especifican los patrones del diseño aplicados y los artefactos derivados de la metodología de desarrollo de software seleccionada.

Capítulo 3: Verificación de la solución.

Este capítulo describe la etapa de implementación, se elaboran y documentan las pruebas realizadas a la solución propuesta para demostrar el correcto funcionamiento de la misma, y por último se analizan los resultados obtenidos tras la aplicación de la herramienta en un caso de estudio.

CAPÍTULO 1: Fundamentos teóricos y metodológicos de la minería de datos espacio-temporal y la estratificación de territorios

1.1 Introducción

En este capítulo se abordan un conjunto de elementos teóricos que conforman el marco referencial relacionado al objeto de estudio. Se hace referencia sobre la estratificación y se realiza un estudio del panorama actual de los SIG. Se realiza un estudio crítico sobre la obtención de conocimiento a través de algoritmos de agrupamiento en la minería de datos espacio temporal. Por último, se analizan las herramientas informáticas, tecnologías y metodologías a utilizar en el proceso de desarrollo del componente.

1.2 Análisis bibliométrico documental

En este epígrafe se realiza un análisis bibliométrico con el objetivo de mostrar la novedad de la revisión bibliográfica realizada, basándose en las fechas de las publicaciones consultadas. Las bases de datos utilizadas son: *IEEE, Google Scholar y Scielo*. Además, se relacionan los tipos de fuentes bibliográficas más citadas. Las fuentes bibliográficas son: artículos de revistas, libros, tesis (específicamente de maestrías o doctorados), artículos en congresos y sitios web. El análisis realizado se muestra en la Tabla 1.

Tabla 1: Análisis bibliométrico documental. Fuente: Elaboración propia.

Tipo de Fuente Bibliográfica	Cantidad	Cantidad de publicaciones en los últimos 5 años (2013-2018)
Libro	12	1
Artículo en conferencia	4	2
Artículo en revista científica	62	35
Tesis	7	6
Informe	1	
Página Web	2	2

Total	**88**	**46**

Figura 1: Bibliografía consultada. Fuente: Elaboración propia.

Marco teórico

Durante los últimos años los sistemas de información constituyen uno de los principales ámbitos de estudio en el área de organización de empresas. El entorno donde las compañías desarrollan sus actividades se vuelve cada vez más complejo. La creciente globalización, el proceso de internacionalización de la empresa, el incremento de la competencia en los mercados de bienes y servicios, la rapidez en el desarrollo de las tecnologías de información, el aumento de la incertidumbre en el entorno y la reducción de los ciclos de vida o comportamientos de enfermedades originan que la información se convierta en un elemento clave para la gestión, así como para la toma de decisiones (de Oca, Rivera y León 2018).

Todo sistema de información utiliza como materia prima los datos, los cuales almacena, procesa y transforma para obtener como resultado final información, la cual será suministrada a los diferentes usuarios del sistema, existiendo además un proceso de retroalimentación, en la cual se ha de valorar si la información obtenida se adecua a lo esperado (Trasobares 2003).

Primera ley de la geografía o principio de autocorrelación espacial, fue formulado por el geógrafo (Tobler 1970) de esta forma :

“Todas las cosas están relacionadas entre sí, pero las cosas más próximas en el espacio tienen una relación mayor que las distantes”.

Estratificar es analizar problemas, fallas, quejas o datos, clasificándolos o agrupándolos de acuerdo con los factores que se cree pueden influir en la magnitud de los mismos, para así localizar las mejores pistas para resolver los problemas de un proceso o para mejorarlo (López 2015).

La estratificación territorial es un proceso que permite dimensionar espacialmente los eventos a través de un proceso de agregación y desagregación de los territorios a evaluar, a partir de variables seleccionadas para dichos territorios que permitan agregaciones (por homologías de las características) o desagregaciones (por heterogeneidades de estas) (Batista Moliner et al. 2001). La utilización de los SIG en el análisis de la distribución espacial de enfermedades ha aumentado considerablemente, sustentado en las herramientas de análisis existentes que posibilitan resolver problemas asociados a la distribución espacial (Fotheringham y Rogerson 2013). Sin embargo, estas no son extensibles, su utilización se limita a llevar información a la cartografía, y la componente espacial de los datos no es explotada en su totalidad. Si bien el espacio es un elemento importante en estos estudios, no siempre se le da la importancia requerida, motivado por: el acceso limitado a los SIG por los costos que ellos implican, poco conocimiento de las herramientas o el tiempo de formación en el área de los SIG es elevado (González Polanco y Pérez Betancourt 2013).

Este proceso permite separar espacialmente los elementos representativos de los territorios (Batista Moliner et al. 2001). Su principal utilidad es identificar regiones de un país determinado en las cuales las condiciones de vida desiguales estén relacionadas con diferentes problemas de salud. En Cuba, su principal utilidad es identificar áreas con mayores necesidades de salud, con la finalidad de ofrecer a cada territorio de manera justa los recursos que realmente necesita y efectuar acciones específicas ante cada situación. Los SIG en salud pública con la ayuda de la estratificación son utilizados en el análisis de la situación de salud, la vigilancia de eventos, el estudio epidemiológico, la planeación y la evaluación de estrategias por zonas de salud, así como en la gestión y toma de decisiones (González Polanco y Pérez Betancourt 2013).

Sistemas de Información Geográfica como soporte para el proceso de estratificación.

Los SIG poseen gran importancia tanto en la esfera social como económica, atendiendo además que la solución que se propone en esta investigación va encaminada a este tipo de sistemas, se hace imprescindible abordar los elementos fundamentales de los mismos.

Se puede definir un SIG como una integración de software, hardware y datos geográficos, diseñado para capturar, analizar, almacenar, manipular y desplegar información geográficamente referenciada. Puede definirse también como un modelo de una parte de la realidad referido a un sistema de coordenadas terrestres, construido principalmente para satisfacer la necesidad de información y ubicación geográfica del mundo. Los SIG son capaces de ubicar un objeto determinado en el espacio; encontrar donde está un cuerpo con respecto a otro; brindar información sobre su perímetro y área; encontrar el camino mínimo de un punto a otro, así como la generación de modelos a partir de fenómenos o actuaciones simuladas (Bravo 2000).

Los SIG pueden definirse como: sistemas informáticos, con la capacidad de visualizar y analizar información georreferenciada de forma versátil e intuitiva, agilizando la toma de decisiones.

Georreferenciación es la asignación de algún tipo de coordenadas, ligadas a la Tierra, a los objetos de nuestro interés, naturales o artificiales, tales como ríos, montañas, bosques, rutas, edificios, parcelas, entre otros (para la Innovación Agraria 2017).

Análisis espacial de modo formal, se puede decir que es el estudio cuantitativo de aquellos fenómenos que se manifiestan en el espacio (Anselin 1989). Según (Olaya 2014) es una “colección de procesos con los que explotar los datos espaciales”. Ello indica una importancia clave de la posición, la superficie, la distancia y la interacción a través del propio espacio. Para que estos conceptos cobren sentido, se necesita que toda la información esté referenciada espacialmente.

Temporal son aquellos cambios que se producen con relación al paso del tiempo.

Un **SIG** ha de permitir la realización de las siguientes operaciones:

- Lectura.
- Edición.
- Almacenamiento.

- Gestión de datos espaciales.

El análisis de datos puede incluir desde consultas sencillas a la elaboración de complejos modelos y puede llevarse a cabo tanto sobre la componente espacial de los datos (la localización de cada valor o elemento) como sobre la componente temática (el valor o el elemento en sí). Generación de resultados tales como mapas, informes, gráficos, etc.

En función de cuál de estos aspectos se valore como más importante, encontramos distintas definiciones formales del concepto de un SIG. Una definición clásica es la de (Tomlin 1990), para quien un SIG es un elemento que permite "*analizar, presentar e interpretar hechos relativos a la superficie terrestre*". El mismo autor (Tomlin 1990) argumenta, no obstante, que "*esta es una definición muy amplia, y habitualmente se emplea otra más concreta. En palabras habituales, un SIG es un conjunto de software y hardware diseñado específicamente para la adquisición, mantenimiento y uso de datos cartográficos*".

En una línea similar, (Star y Estes 1990) define un SIG como un "*sistema de información diseñado para trabajar con datos referenciados mediante coordenadas espaciales o geográficas. En otras palabras, un SIG es tanto un sistema de base de datos con capacidades específicas para datos georreferenciados, como un conjunto de operaciones para trabajar con esos datos. En cierto modo, un SIG es un mapa de orden superior"*.

Ambas definiciones recogen el concepto fundamental de los SIG en el momento en que fueron escritas, pero la realidad hoy en día hace necesario recoger otras ideas, y la definición actual de un SIG debe fundamentarse sobre todo en el concepto de sistema como elemento integrador que engloba a un conjunto de componentes interrelacionados.

La MD, también conocida como Descubrimiento de Conocimiento en Bases de datos (sus siglas en inglés son "KDD – Knowledge Discovery in Databases"), es el campo que permite descubrir información nueva y potencialmente útil de grandes cantidades de datos. Se ha empleado en numerosos campos, incluyendo desde los ya conocidos casos de cesta de la compra hasta la bioinformática o investigaciones contra el terrorismo (Galindo y García 2010).

1.3 Métodos de descubrimiento de conocimiento

Los métodos de descubrimiento son una de las clasificaciones de minería de datos planteadas por (Terán, Alcivar y Puris 2016). Según (Valcárcel Asencios 2004) junto con

estos métodos se engloban también todas las técnicas predictivas y han de encontrar patrones potencialmente interesantes de forma automática. El resultado obtenido con la

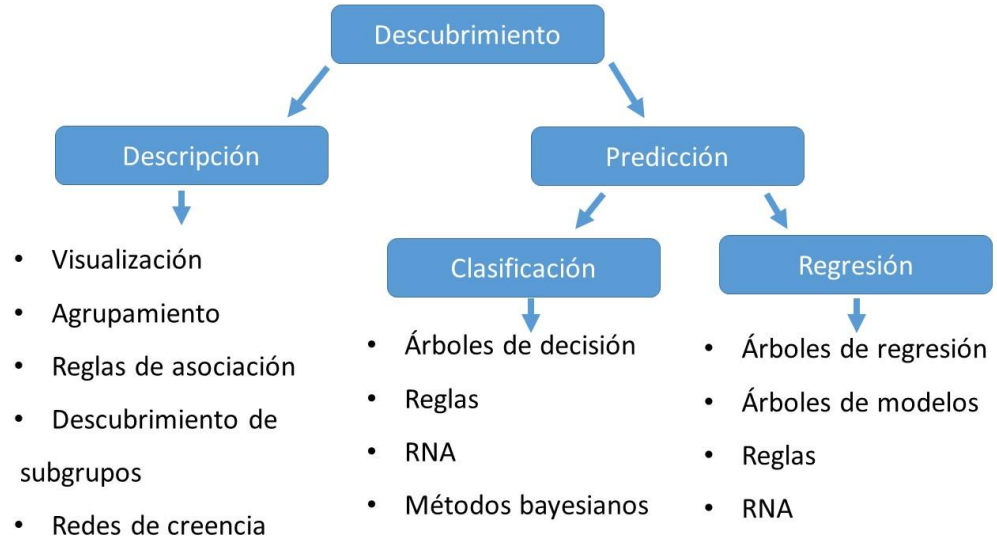

Figura 2: Fragmento de taxonomía de minería de datos. Fuente: (Terán, Alcivar y Puris 2016)

aplicación de algoritmos de MD pertenecientes al grupo de técnicas de descubrimiento, puede ser de carácter descriptivo o predictivo. Las predicciones sirven para prever el comportamiento futuro de algún tipo de entidad mientras que una descripción puede ayudar a su comprensión.

1.4 Descubrimiento de conocimiento en datos Espacio Temporales.

La Minería de datos espacio-temporales (STDM, por sus siglas en inglés) es la extracción de datos desconocidos y conocimiento implícito, estructuras, relaciones o patrones de estos grandes conjuntos de datos. Los agrupamientos espacio-temporales, han sido un importante campo de investigación de la minería de datos espacio-temporales y descubrimiento de conocimiento. La agrupación espacio-temporal pretende detectar entidades espacio-temporales similares a partir de una base de datos espacio-temporales y, además, descubrir los clústeres espacio-temporales. Estas agrupaciones pueden identificar las tendencias evolutivas o reglas relacionadas con el fenómeno geográfico (Deng et al. 2013).

1.5 Métodos para el descubrimiento de conocimientos en datos espacio-temporales.

Los métodos para el descubrimiento de conocimiento en datos espacio-temporales son básicamente los mismos que los empleados en la minería de datos espaciales, pero es evidente que, debido a la adición de una nueva componente, el manejo de los datos no puede ser el mismo, pero siguen en un principio las mismas metodologías. Los cambios

tienen y son introducidos ya que se quieren explotar de la mejor manera los datos y estos ya no son solo puntos ubicados en el espacio, si no que tienen una descripción de su comportamiento en el tiempo, por lo que debe cambiar su análisis estadístico. El presente trabajo muestra ejemplos de algunas de las variaciones en ciertas metodologías existentes.

1.5.1 Técnicas de minería de datos.

El gran volumen de datos existente en el sector de la salud, dificulta la toma de decisiones por parte de los especialistas, debido a que no se aplican técnicas que aprovechen al máximo la información disponible, ocasionando la dificultad de reconocer patrones de comportamiento y extraer conocimiento oculto de los datos almacenados. Además, la no predicción del comportamiento, basado en el conocimiento previo, puede acarrear un alto porcentaje de fracaso, más aún cuando se trata de un campo tan primordial como el de la salud. De aquí, surge la necesidad de aplicar técnicas de minería de datos, debido a que son capaces de extraer patrones, de predecir comportamientos, regularidades y, de sacar provecho a la información automatizada. Según (Tamayo y Marín 2017) las técnicas de minería de datos son algoritmos con diferente nivel de sofisticación que se aplican sobre un conjunto de datos para obtener resultados, y proceden de la inteligencia artificial y de la estadística. Una forma bastante buena y clara sobre la importancia de las técnicas de minería de datos, por parte de (Berry y Linoff 1997), dice: “Las técnicas de minería de datos nos mostrarán cómo aprovechar de forma vertiginosa y cómoda la mina de oro para soluciones de negocio escondidas en las bases de datos y sistemas de información”

Existen diversas técnicas de minería de datos que permiten la obtención de información implícita, entre las principales que se utilizan están:

Las **redes neuronales** son un paradigma de aprendizaje y procesamiento automático inspirado en el funcionamiento del sistema nervioso de los animales. Se trata de reproducir el esquema neuronal tanto en sus unidades de proceso (neuronas), como en sus conexiones y modo de establecerlas. Ejemplos de ellas son el perceptrón, el perceptrón multicapa y los mapas autoorganizados (Montaño Moreno 2017). Las redes son una herramienta importante en la modelación de variables en las cuales la existencia de un modelo estructural no es clara, pues no parten de supuestos a priori sobre los datos para el pronóstico, y todo lo que de ellas puede decirse es inherente a las observaciones (Misas, López-Enciso y Querubin 2015).

Regresión lineal es la más utilizada para formar relaciones entre datos en estadística, para predecir valores de variables continuas, debido a su fácil interpretación (Díaz Sepúlveda y Correa 2017). El propósito del análisis de la regresión lineal es examinar un modelo que pretende explicar el comportamiento de una variable (variable endógena, explicada o dependiente), denotada por Y, utilizando la información proporcionada por los valores tomados por un conjunto de variables (explicativas, exógenas o independientes), denotadas por X1, X2,…Xn (Sedano et al. 2015).

Los **árboles de decisión** son modelos de predicción utilizados en el ámbito de la Inteligencia Artificial y el análisis predictivo, son muy utilizados a la hora de analizar la toma de decisiones de individuos (Rokach y Maimon 2008). Son un método no paramétrico de aprendizaje supervisado, cuyo objetivo es crear un modelo para predecir los valores de una variable respuesta de interés, basado en reglas de decisión sencillas inferidas de los datos (Dianda, Quaglino y Pagura 2016).

Los **algoritmos de agrupamiento** son procedimientos de agrupación de una serie de vectores según criterios de cercanía, son estrategias encargadas de descubrir la estructura interna de los datos, pero cada uno de ellos supone condiciones en la base de datos que los hace eficientes para algunos tipos de datos e ineficientes para otros (González 2010). En general, la clasificación no supervisada o agrupamiento (clustering) consiste en dividir el conjunto de objetos en grupos de objetos similares llamados clusters, de modo tal que objetos pertenecientes a un mismo grupo son más similares que objetos de grupos diferentes (González 2010). Si se conoce el número de grupos o no se puede diferenciar entre ellos, estos se dividen en dos: agrupamiento jerárquico y agrupamiento no jerárquico (Barberá 2017).

Bajo el nombre genérico de algoritmos de agrupamiento, se incluyen todo un conjunto de procesos cuya finalidad general será dividir un conjunto de objetos en clases, para obtener un subconjunto representativo del conjunto de entrenamiento inicial que posteriormente pueda utilizarse en una regla de clasificación supervisada.

1.5.2 Clustering jerárquicos

Los clustering jerárquicos son aquellos que no se conoce a priori el número de grupos buscados. Estos métodos jerárquicos normalmente suelen ser ascendentes, es decir sucesivamente van fusionando grupos desde el elemento individual hacia arriba.

Los algoritmos de clustering jerárquicos se pueden dividir en dos grupos:

Aglomerativos: parten de tantos grupos como observaciones y van fusionando los grupos con características más similares formando grupos más grandes hasta llegar a formar un único grupo.

Divisivos: parten de un solo grupo y se van particionando en distintos grupos.

1.5.3 Clustering no jerárquicos

Estos, a diferencia de los clustering jerárquicos, no construyen un árbol (una jerarquía de conceptos).

De los algoritmos no jerárquicos el más conocido es el **k-means**. Este algoritmo lleva a cabo un proceso iterativo para encontrar k grupos que minimicen la suma de las distancias de sus centroides.

Debido a que se desconoce el número de grupos resultantes de este trabajo y según las clasificaciones vistas anteriormente en la bibliografía, se plantea desarrollar un algoritmo jerárquico.

1.6 Algoritmos de agrupamiento espacio temporales.

Los métodos de agrupación espacio-temporales pueden clasificarse en tres tipos: métodos analizando espacio-tiempo, métodos basados en densidad y métodos basados en la distancia (Deng et al. 2013).

Si se toma en cuenta toda la información que se obtiene de datos georreferenciados y además de eso aplicamos minería espacio-temporal (agrupamiento espacio-temporal) se pueden observar comportamientos de determinados fenómenos a través del tiempo y a su vez ayudar a la toma de decisiones.

Métodos basados en densidad

Los algoritmos basados en densidad tratan de identificar grupos en zonas altamente pobladas. Emplean diferentes técnicas para determinar los grupos: por grafos, basadas en histogramas, kernels (núcleos), aplicando la regla K-NN o tratando de descubrir subgrupos denso-conectados. Los algoritmos basados en densidad localizan zonas de alta densidad separadas por regiones de baja densidad. La idea general es que un grupo puede continuar creciendo, siempre y cuando la densidad (número de objetos en un espacio determinado) en el vecindario, sobrepase un límite determinado (Minpts) (Vallejos-Peña 2017). Estos algoritmos son capaces de descubrir grupos de formas y tamaños diferentes y algunos presentan estrategias para manejar el ruido (González 2010). Los grupos no-esféricos aparecen naturalmente en datos espaciales, es decir,

datos con una referencia a algún espacio concreto en dos o tres dimensiones que corresponde al mundo real. Los grupos en datos espaciales pueden tener formas arbitrarias debido a las restricciones impuestas por entidades geográficas tales como montañas y ríos. Incluso en datos de dimensiones más altas, asumir la existencia de una determinada cantidad de grupos de una determinada forma es muy fuerte y a menudo no se cumple lo supuesto (DEZEREGA 2016). DBSCAN (Ester et al. 1996) es uno de los primeros algoritmos de agrupamiento que emplea este enfoque.

Por estos puntos positivos descritos anteriormente se perfila el trabajo a desarrollar un algoritmo jerárquico basado en densidad.

K-Means es el algoritmo de agrupación de referencia para muchos simplemente porque es rápido, fácil de entender y está disponible en todas partes (hay una implementación en casi cualquier herramienta estadística o de aprendizaje automático que desee utilizar). Sin embargo, K-Means tiene algunos problemas. El primero es que no es un algoritmo de agrupamiento, es un algoritmo de partición. Es decir, K-means no 'encuentra clústeres'. Divide su conjunto de datos en tantos fragmentos (asumidos como globulares) como se solicite al intentar minimizar las distancias entre particiones. Eso lleva al segundo problema: necesita especificar exactamente cuántos clústeres espera. Si se sabe mucho sobre los datos, entonces es algo que podría esperar saber. Si, por otro lado, simplemente está explorando un nuevo conjunto de datos, entonces 'número de grupos' es un parámetro difícil para tener una buena intuición. La solución generalmente propuesta es ejecutar K-medias para muchos valores diferentes de 'número de clústeres' y puntuar cada agrupamiento con alguna medida de 'bondad de agrupamiento' (generalmente una variación en distancias intraclúster vs inter-clúster) e intentar encontrar un punto de inflexión [1]. Pero encontrar dicho codo generalmente no es tan fácil, y tampoco se correlaciona necesariamente con el número real de conglomerados que pueda desear. Finalmente, K-Means también depende de la inicialización; darle múltiples inicios aleatorios diferentes y puede obtener múltiples agrupamientos diferentes. Esto no genera mucha confianza en ninguna agrupación individual que pueda resultar (Khan 2015).

Affinity Propagation es un algoritmo de agrupamiento que utiliza un enfoque basado en grafos para permitir que los puntos 'voten' en su 'ejemplar' preferido. El resultado final es un conjunto de 'ejemplares' de grupos de los que derivan grupos básicamente haciendo lo que hace K-Means y asignando cada punto al clúster de su ejemplar más

[1] Es el punto de cambio de las curvas en la búsqueda de la cantidad óptima del número de clusters (Liu et al. 2010)

cercano. Affinity Propagation tiene algunas ventajas sobre K-Means. En primer lugar, la votación ejemplar basada en grafos significa que el usuario no necesita especificar la cantidad de clústeres. En segundo lugar, debido a la forma en que el algoritmo funciona como caja negra con la representación gráfica, permite las diferencias no métricas (es decir, podemos tener diferencias que no obedecen a la desigualdad del triángulo, o que no son simétricas). Este segundo punto es importante si alguna vez se trabaja con datos que no están naturalmente integrados en un espacio métrico de algún tipo; algunos algoritmos de agrupamiento admiten, por ejemplo, diferencias no simétricas. Finalmente, Affinity Propagation tiene, al menos, una mejor estabilidad sobre las ejecuciones, pero no sobre los rangos de parámetros.

Los puntos débiles de Affinity Propagation son similares a K-Means. Dado que divide los datos como K-Means, se espera ver el mismo tipo de problemas, particularmente con datos ruidosos. Si bien Affinity Propagation elimina la necesidad de especificar el número de clústeres, tiene parámetros de "preferencia" y "amortiguación". Elegir bien estos parámetros puede ser difícil. La implementación en sklearn prefiere por defecto la diferencia de mediana. Esto tiende a generar una gran cantidad de grupos. Un valor mejor es algo más pequeño (o negativo) pero depende de los datos. Finalmente, la Affinity Propagation es lenta; dado que admite disimilitudes no métricas, no puede tomar ninguno de los accesos directos disponibles para otros algoritmos, y las operaciones básicas son costosas a medida que crece el tamaño de los datos (El-Samak y Ashour 2015).

Mean shift es otra opción si no se desea tener que especificar la cantidad de clústeres. Está basado en centroides, como K-Means y Affinity Propagation, pero puede devolver clústeres en lugar de una partición. La idea subyacente del algoritmo Mean Shift es que existe alguna función de densidad de probabilidad a partir de la cual se extraen los datos, y trata de ubicar los centroides de los conglomerados en el máximo de esa función de densidad. Se aproxima a esto a través de técnicas de estimación de la densidad del kernel, y el parámetro clave es el ancho de banda del kernel utilizado. Esto es más fácil de adivinar que el número de clústeres, pero puede requerir cierta observación, por ejemplo, de las distribuciones de distancias por pares entre los puntos de datos para elegir con éxito. El otro problema (al menos con la implementación de sklearn) es que es un depsite bastante lento que potencialmente tiene una buena escala (Senoussaoui et al. 2014).

Spectral clustering se puede considerar mejor como un agrupamiento de grafo. Para datos espaciales uno puede pensar en inducir un grafo basado en las distancias entre

puntos (potencialmente un grafo k-NN, o incluso un grafo denso). Spectral clustering ha atraído la atención de personas en las últimas décadas debido a su sólida fundamentación teórica (Tian et al. 2014). A partir de allí, Spectral clustering observará los vectores propios del operador laplaciano[2] del grafo para intentar encontrar una buena componente (baja dimensional) del grafo en el espacio euclidiano. La complejidad de una implementación directa de descomposición de valores propios es cúbica a la cantidad de nodos en el grafo (Tian et al. 2014). Este es esencialmente un tipo de aprendizaje múltiple, al encontrar una transformación de nuestro espacio original a fin de representar mejor las múltiples distancias para algunos de los que se supone que los datos se encuentran. Una vez que se tiene el espacio transformado, se ejecuta un algoritmo de agrupamiento estándar; con sklearn, el valor predeterminado es K-Means. Eso significa que la clave para Spectral clustering es la transformación del espacio. Suponiendo que se pueda respetar mejor la variedad, se conseguirá una mejor agrupación; no debe preocuparse menos por los cúmulos globulares de K-Means, ya que son meramente globulares en el espacio transformado y no en el espacio original. Desafortunadamente conserva algunas debilidades de K-Means: aún particiona los datos en lugar de agruparlos; se tiene el parámetro "número de grupos" difícil de adivinar; tiene problemas de estabilidad heredados de K-Means. Peor aún, si se opera en el grafo denso de la matriz de distancia, posee un paso inicial muy costoso y sacrifica el rendimiento (Yang et al. 2015).

Agglomerative clustering es realmente un conjunto de algoritmos, todos basados en la misma idea. La idea fundamental es que comience con cada punto en su propio clúster y luego, para cada clúster, utilice algún criterio para elegir otro clúster para fusionarse. Se hace esto repetidamente hasta que se tenga solo un grupo y se obtenga una jerarquía, o árbol binario, de grupos que se bifurquen hasta la última capa que tiene una hoja para cada punto en el conjunto de datos. La versión más básica de este enlace, único, elige el clúster más cercano para fusionar, y por lo tanto el árbol se puede clasificar por distancia en cuanto a cuando los clústeres se fusionaron / dividieron. Las variaciones más complejas utilizan elementos como la distancia media entre los clústeres o la distancia entre los centroides del grupo, para determinar qué clúster fusionar. Una vez que se tenga una jerarquía de clúster, se puede elegir un nivel o un corte (según algunos criterios) y tomar los clústeres en ese nivel del árbol. Para sklearn

[2] El operador laplaciano es un operador diferencial elíptico de segundo orden, relacionado con problemas de minimización de magnitudes sobre un dominio determinado (Liu, Jia y Ge 2017).

usualmente se elige un corte basado en un parámetro de 'número de clústeres' (Murtagh y Legendre 2014).

La ventaja de este enfoque es que los conglomerados pueden crecer "siguiendo la variedad subyacente" en lugar de suponerse que son globulares. También puede inspeccionar el dendrograma de grupos y obtener más información sobre cómo se descomponen los grupos. Por otro lado, si se quiere un conjunto de conglomerados planos, se debe elegir un corte del dendrograma, y eso puede ser difícil de determinar. Puede tomar el enfoque de sklearn y especificar una cantidad de clústeres, pero como ya se ha planteado, ese no es un parámetro particularmente intuitivo cuando realiza análisis exploratorio de datos. Se puede ver el dendrograma e intentar elegir un corte natural, pero esto es similar a encontrar el "codo" entre los diversos valores de k para K-Means: en principio está bien, y los ejemplos de los libros de texto siempre lo hacen parecer fácil, pero en la práctica en datos desordenados del mundo real, la opción "obvia" a menudo dista de ser obvia. También sigue particionando en lugar de agrupar los datos, por lo que todavía tiene ese problema persistente de ruido que contamina los grupos. Afortunadamente, el rendimiento puede ser bastante bueno; la implementación de sklearn es bastante lenta, pero fastcluster (biblioteca de python) proporciona agrupamiento aglomerativo de alto rendimiento si eso es lo que necesita (Ackermann et al. 2014).

DBSCAN es un algoritmo basado en densidad: asume grupos para regiones densas. También es el primer algoritmo real de agrupamiento que se analiza: no requiere que cada punto se asigne a un clúster y, por lo tanto, no particiona los datos, sino que extrae los clústeres "densos" y deja el fondo escaso clasificado como 'ruido'. En la práctica, DBSCAN está relacionado con la agrupación aglomerativa. Como primer paso, DBSCAN transforma el espacio de acuerdo con la densidad de los datos: los puntos en regiones densas se dejan solos, mientras que los puntos en regiones dispersas se mueven más lejos. La aplicación de clústeres de vinculación única al espacio transformado da como resultado un dendrograma, que se corta de acuerdo con un parámetro de distancia (llamado épsilon o eps en muchas implementaciones) para obtener grupos. De manera importante, cualquier agrupación semifallada en ese nivel de corte se considera "ruido" y se deja sin agrupar. Esto proporciona varias ventajas: se obtiene el siguiente comportamiento múltiple de aglomeración de clústeres y obtiene clústeres reales en lugar de particiones. Mejor aún, dado que se puede enmarcar el algoritmo en términos de consultas de la región local, se puede usar varios trucos, como kdtrees, para obtener un rendimiento excepcionalmente bueno y escalar a tamaños de

conjuntos de datos que de otro modo serían inalcanzables con algoritmos distintos de K-Means. Sin embargo, hay algunas capturas. Obviamente, épsilon puede ser difícil de elegir; puede hacer algunos análisis de datos y obtener una buena estimación, pero el algoritmo puede ser bastante sensible a la elección del parámetro. La transformación basada en densidad depende de otro parámetro (min_samples en sklearn). Finalmente, la combinación de min_samples y eps equivale a una elección de densidad y la agrupación solo encuentra clústeres en esa densidad o por encima de ella; Si sus datos tienen grupos de densidad variable, entonces el DBSCAN perderá, dividirá o agrupará algunos de ellos, según los parámetros escogidos (Dudik et al. 2015).

HDBSCAN tiene como objetivo permitir grupos de densidad variable. El algoritmo comienza de manera muy similar a DBSCAN: transformamos el espacio de acuerdo con la densidad, exactamente como lo hace DBSCAN, y realizamos grupos de enlace único en el espacio transformado. Sin embargo, en lugar de tomar un valor épsilon como nivel de corte para el dendograma, se adopta un enfoque diferente: el dendrograma se condensa al ver divisiones que producen una pequeña cantidad de puntos que se separan cuando los puntos "caen de un grupo". Esto da como resultado un árbol más pequeño con menos grupos que 'pierden puntos'. Ese árbol se puede usar para seleccionar los clústeres más estables o persistentes. Este proceso permite que el árbol se corte a una altura variable, seleccionando los clústeres de densidad variable en función de la estabilidad del clúster. La ventaja inmediata de esto es que se puede tener clústeres de densidad variable; el segundo beneficio es que se ha eliminado el parámetro épsilon porque ya no se necesita para elegir un corte del dendrograma. En su lugar, se tiene un nuevo parámetro min_cluster_size que se usa para determinar si los puntos se 'caen de un clúster' o se dividen para formar dos nuevos grupos. Esto cambia un parámetro no intuitivo por uno que no es tan difícil de elegir para el análisis exploratorio de datos (el grupo de tamaño mínimo del que se está dispuesto a preocuparse) (McInnes, Healy y Astels 2017).

1.7 Comparación entre los algoritmos de agrupamientos

Tabla 2: Comparación entre algoritmos basados en densidad. Fuente: Elaboración propia.

	Precisión al agrupar	Parámetros intuitivos	Estabilidad	Rendimiento
K-Means	No	Regular	Por lo general	Simple y eficiente
Affinity Propagation	No	Regular	Es determinista	Muy lento

			sobre las ejecuciones	
Mean Shift	No	Más que los anteriores	Variable (depende de los parámetros)	Bueno
Spectral Clustering	No (mejor que los anteriores)	Regular	Ligeramente más que el K-means	Un poco lento
Agglomerative Clustering	Regular (asume que no hay ruido)	Regular	Si	En dependencia de la implementación
DBSCAN	Si	No	Si	Bueno
HDBSCAN	Si (y se elimina el problema de los grupos de densidad variable)	Regular (mejor que los anteriores)	Si	En dependencia de la implementación

Como se planteó anteriormente, estos algoritmos no presentan la capacidad de obtener correctamente los datos y de esta manera obtener toda la información ventajosa necesaria para el negocio. Esto se debe a su incapacidad de tratar con el comportamiento temporal de los eventos, dando paso a posteriores modificaciones que posibilitan el máximo aprovechamiento de los datos.

A partir del análisis y comparación realizado sobre los algoritmos de agrupamiento, se pudo determinar de que el algoritmo HDBSCAN mejora la precisión al realizar el agrupamiento, ya que minimiza la obtención de falsos positivos y falsos negativos. Permite un trabajo cómodo debido al poco conocimiento que se necesita a priori sobre los datos, por lo que los parámetros de entrada ya no son una barrera. Junto a las anteriores mejoras hay que adicionar su buena estabilidad y rendimiento, haciendo de este algoritmo una perfecta opción a usar.

1.8 Herramientas, lenguajes y tecnologías a utilizar

En todo proceso investigativo es necesario la utilización de sistemas de soporte que permitan organizar, facilitar, agilizar y automatizar las tareas generadas durante el transcurso de la investigación. Las herramientas, lenguajes y tecnologías empleadas que se describen a continuación son de vital importancia para una correcta realización de la misma.

Python es un lenguaje de programación interpretado, cuya sintaxis se favorece por su código legible. Se trata de un lenguaje de programación multiparadigma, ya que soporta

orientación a objetos, programación imperativa y, en menor medida, programación funcional. Es un lenguaje interpretado, usa tipado dinámico y es multiplataforma.

Es administrado por la Python Software Foundation. Posee una licencia de código abierto, denominada Python Software Foundation License, que es compatible con la Licencia pública general de GNU a partir de la versión 2.1.1.

Se seleccionó Python en su versión 2.7.5 porque su sintaxis es simple, clara y sencilla logrando de esta manera que los programas elaborados en este lenguaje parezcan pseudocódigo. Además, el tipado dinámico, el gestor de memoria, la gran cantidad de librerías disponibles y la potencia del lenguaje, entre otros, hacen que desarrollar una aplicación en Python sea sencillo y rápido.

Es importante tener en cuenta que al seleccionar QGIS como el software que soportará la integración de la solución, Python es un lenguaje de programación eficiente y conveniente para utilizar que, junto con el módulo PyQT entrega una solución óptima al desarrollo de plugins e interfaces gráficas de usuario.

PyQt es un binding de la biblioteca gráfica Qt para el lenguaje de programación Python. La biblioteca está desarrollada por la firma británica *Riverbank Computing* y está disponible para Windows, GNU/Linux y Mac OS X bajo diferentes licencias.

Qt Designer es una herramienta que permite acelerar el desarrollo de interfaces multilenguaje debido a que genera un archivo XML cuyo contenido es el formato de dicha interfaz, pudiéndolo convertir con los programas pertinentes a cada lenguaje. Esta herramienta provee características muy poderosas como la previa visualización de la interfaz, soporte para widgets y un editor de propiedades con gran variedad de opciones.

Un **entorno de desarrollo integrado** (IDE, por sus siglas en inglés) es una herramienta que permite a los desarrolladores de software escribir sus programas en uno o más lenguajes. Consiste básicamente en una plataforma en la que se integran, un editor de código, un compilador, un depurador y una interfaz gráfica de usuario (ACHI y VLADIMIR 2018).

Pycharm es un editor de código inteligente que proporciona soporte de primera clase para los lenguajes de programación: Python, JavaScript, CoffeeScript, TypeScript, HTML/CSS, Cython, lenguajes de plantilla, AngularJS y Node.js, y otros menos utilizados. Pycharm funciona en las plataformas Windows, Mac OS y Linux con una única clave de licencia.

La decisión de seleccionar como IDE, Pycharm en su versión 3.4, está dada a que ofrece auto-completación inteligente de código, comprobación de errores sobre la marcha, soluciones rápidas y fácil navegación en el proyecto. Además, Pycharm mantiene la calidad del código bajo control con chequeos, asistencia a pruebas, refactorizaciones inteligentes, y una serie de inspecciones, lo que ayuda a escribir un código limpio y fácil de mantener («PyCharm» 2018).

Gestor de base de datos

Los Gestores de Bases de Datos (GBD) permiten crear y mantener una base de datos, además actúan como interfaz entre los programas de aplicación y el sistema operativo. El objetivo principal es proporcionar un entorno eficiente a la hora de almacenar y recuperar la información de las base de datos. Estos softwares facilitan el proceso de definir, construir y manipular bases de datos para diversas aplicaciones (Terán Guerrero 2017).

PostgreSQL es un Sistema de gestión de bases de datos relacional, orientado a objetos y libre, publicado bajo la licencia PostgreSQL, similar a la BSD o la MIT. Como muchos otros proyectos de código abierto, el desarrollo de PostgreSQL no es manejado por una empresa o persona, sino que es dirigido por una comunidad de desarrolladores que trabajan de forma desinteresada, altruista y libre, o apoyados por organizaciones comerciales. Dicha comunidad es denominada PGDG (*PostgreSQL Global Development Group*)

Metodología de desarrollo

En el desarrollo de un software se debe contar con un proceso bien detallado, para esto se necesita aplicar una metodología que sea capaz de llevar a cabo el control total del producto. Las metodologías de desarrollo de software surgen ante la necesidad de utilizar una serie de procedimientos, técnicas, herramientas y soporte documental a la hora de desarrollar un producto de software (Letelier 2006).

Las metodologías se dividen en dos grupos, tradicionales (pesadas) y ágiles (ligeras). Las tradicionales, se centran en la definición detallada de los procesos y tareas a realizar, herramientas a utilizar, y requiere una extensa documentación, pretendiendo prever todo de antemano, además dependen de un equipo de desarrollo bastante grande. En las ágiles es más importante lograr que un producto de software se desarrolle con la calidad requerida, que realizar una buena documentación. En este tipo

de metodología el cliente está presente en todo momento y colabora con el proyecto, que posee un equipo de desarrollo pequeño (Letelier 2006).

Metodología XP

La **programación extrema** o *eXtreme Programming* (de ahora en adelante, XP) es una metodología de desarrollo de la ingeniería de software formulada por Kent Beck, autor del primer libro sobre la materia (Beck y Gamma 2000). Es el más destacado de los procesos ágiles de desarrollo de software. Al igual que éstos, la programación extrema se diferencia de las metodologías tradicionales principalmente en que pone más énfasis en la adaptabilidad que en la previsibilidad. Los defensores de la XP consideran que los cambios de requisitos sobre la marcha son un aspecto natural, inevitable e incluso deseable del desarrollo de proyectos. Creen que ser capaz de adaptarse a los cambios de requisitos en cualquier punto de la vida del proyecto es una aproximación mejor y más realista que intentar definir todos los requisitos al comienzo del proyecto e invertir esfuerzos después en controlar los cambios en los requisitos.

La metodología ágil XP surge con el propósito de transformar la esencia del desarrollo de los procesos que se ejecutan, al momento de llevar a cabo la planificación y ejecución de un proyecto de creación de software, esta metodología se enfoca en integrar en una mejora continua al usuario y al grupo de individuos que se encargaran de resolver la problemática percibida (Yolanda 2013).

Un proyecto XP tiene éxito, cuando el cliente selecciona el valor de negocio a implementar basado en la habilidad del equipo para medir la funcionalidad que puede entregar a través del tiempo. El ciclo de desarrollo consiste (a grandes rasgos) en los siguientes pasos:

1. El cliente define el valor de negocio a implementar.
2. El programador estima el esfuerzo necesario para su implementación.
3. El cliente selecciona qué construir, de acuerdo con sus prioridades y las restricciones de tiempo.
4. El programador construye ese valor de negocio.
5. Vuelve al paso 1.

A partir del estudio de XP, se concluye que responde a las necesidades principales de tiempo, entorno y cantidad de programadores, e incluye al cliente como parte fundamental del equipo de desarrollo. Además, se preocupa más en el avance exitoso del producto que en generar una documentación detallada del mismo, siendo capaz de

adaptarse a los cambios de requisitos en cualquier punto del ciclo de vida del proyecto. Estos elementos demuestran que es una metodología factible para guiar el proceso de desarrollo de la solución, por lo que se decide incluir en la propuesta.

1.9 Conclusiones parciales

Con el desarrollo del marco teórico referencial se obtuvo un mejor dimensionamiento del problema a partir del análisis de los principales conceptos asociados a la solución por lo que se concluye:

- El estudio de las técnicas de agrupamiento facilitó identificar que los algoritmos de agrupamientos basados en densidad son capaces de descubrir grupos de formas y tamaños diferentes y presentan estrategias para manejar el ruido.
- La revisión del panorama actual de los softwares SIG permitió definir QGIS por su visual y su procesamiento de información como la más adecuada para la integración de la solución.
- El estudio de varios algoritmos existentes que soportan procesos de estratificación facilitó obtener una mejor visión de la solución, permitiendo identificar el algoritmo espacial HDBSCAN como el que más se ajustaba a las necesidades requeridas.
- Los algoritmos de agrupamientos estudiados poseen como limitante el procesamiento de tres o más dimensiones que presenten los datos.

CAPÍTULO 2: Características de la solución

2.1 Introducción

En este capítulo se presenta una propuesta de solución para realizar el proceso de agrupamientos de datos y se describen cada una de las fases que la conforman. En la primera sección se plantea el método a seguir donde se explica el funcionamiento del algoritmo. En la segunda sección se desarrollan los pasos descritos anteriormente para darle solución al algoritmo.

2.2 Método

Un método es una actividad estructurada y ordenada para la obtención de nuevos conocimientos y su aplicación para la solución a problemas o interrogantes de carácter científico. Este conjunto de estrategias y herramientas se utilizan para llegar a un objetivo preciso. Por tanto, se consideran las siguientes fases:

1 Identificar fuentes de datos de información georreferenciada.
2 Empleo del algoritmo de agrupamiento espacio-temporal.
3 Visualización de los resultados.

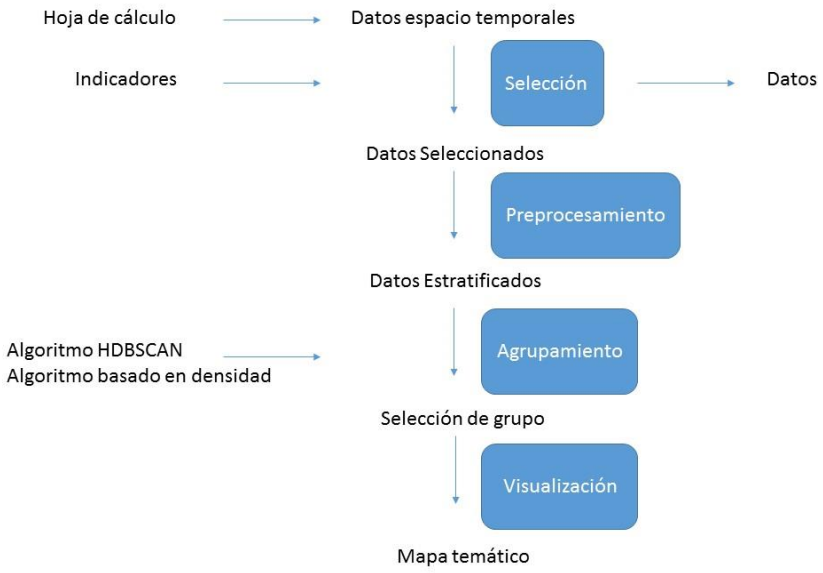

Figura 3: Método de estratificación espacio-temporal. Fuente: Elaboración propia.

Para el desarrollo del método propuesto y teniendo como base la revisión realizada con anterioridad se muestra en la Figura 3: Método de estratificación espacio-temporal. Fuente: Elaboración propia.. La misma describe en orden lógico los diferentes conceptos y algoritmos involucrados en cada una de las fases.

2.3 Instanciación del método propuesto

La fuente de dato a la cual se le realiza el análisis es una hoja de cálculo que contiene varias hojas, donde cada hoja representa el nivel de tiempo a utilizar. Las hojas contienen una tabla de valores numéricos donde las filas representan a los territorios que se agruparán y las columnas a los indicadores que describen dichos territorios. De la tabla se obtiene una matriz de valores (objeIndica_porAnnos) y estos son parametrizados.

1. Una vez obtenida la matriz objeIndica_porAnnos se aplica el algoritmo HDBSCAN a cada hoja de la fuente de datos, así se obtiene etiquetas con la clasificación de los territorios, donde cada posición representa un territorio y el valor en la posición el grupo al que pertenece, según la cantidad de nivel de tiempo existente (ejemplo: año por año o mes por mes).
2. Como segundo paso se construye una matriz con las etiquetas anteriores y se traspone de tal forma que la horizontal vulva a representar los territorios y la vertical el nivel de tiempo existente que contiene la pertenencia del territorio en cada momento de tiempo.
3. Sobre esta matriz se aplica nuevamente el HDBSCAN (re-clustering) obteniendo como resultado una etiqueta con los grupos de territorios que comparten similaridad en el tiempo, por lo que el algoritmo propuesto gana el nombre de espacio-temporal.

HDBSCAN se divide en los siguientes pasos:

1. Transformar el espacio de acuerdo a la densidad / dispersión.
2. Construir un árbol de expansión mínimo del grafo ponderado por distancia.
3. Construir una jerarquía de clúster de componentes conectados.
4. Condensar la jerarquía de clúster según el tamaño mínimo de clúster.
5. Extraer los clústeres estables del árbol condensado.

Pseudocódigo:

1. Calcula la distancia central w.r.t. **mpts** para todos los objetos de datos en **X**.
2. Calcula un **MST** de **Gmpts**, el Gráfico de Accesibilidad Mutua.
3. Extiende el MST para obtener MSText, agregando para cada vértice un "borde propio" con la distancia del núcleo del objeto correspondiente como peso.
4. Extrae la jerarquía HDBSCAN como un dendrograma de MSText:
    4.1 Para la raíz del árbol, asigna a todos los objetos la misma etiqueta (solo "clúster").

4.2 Elimina iterativamente todos los bordes de MSText en orden decreciente de pesos (en caso de ataduras, los bordes se deben quitar simultáneamente):

4.2.1 Antes de cada extracción, configure el valor de escala de dendrograma de la corriente nivel jerárquico como el peso del borde(s) a eliminar.

4.2.2 Después de cada extracción, asigna etiquetas a los componentes conectados que contiene(n) el vértice final(-ices) del borde(s) eliminado, para obtener el siguiente nivel jerárquico: asigne una nueva etiqueta de clúster a un componente si todavía tiene al menos un borde, de lo contrario, asígnele una etiqueta nula ("ruido").

MST (Mínimum Spanning Tree)
Gmpts (Mutual Reachability Graph)
X (data set)

2.3.1 Requisitos de software

Es una descripción completa del comportamiento del sistema que se va a desarrollar, la cual incluye un conjunto de casos de uso que describen todas las interacciones que tendrán los usuarios con el software. "*Un requisito es simplemente una declaración abstracta de alto nivel de un servicio que debe proporcionar el sistema o una restricción de éste*" (Sommerville 2010). La calidad con que se realiza la captura de los requisitos, afecta todo el proceso de desarrollo del software, repercutiendo en el resto de las fases de desarrollo del mismo. Además, contribuye a tomar mejores decisiones de diseño y de arquitectura.

2.3.2 Requisitos funcionales

Un requisito funcional (RF) define una función del sistema de software o sus componentes. Una función es descrita como un conjunto de entradas, comportamientos y salidas. Los requerimientos funcionales pueden ser: cálculos, detalles técnicos, manipulación de datos y otras funcionalidades específicas que se supone que un sistema debe cumplir. Estos son complementados por los requisitos no funcionales, que se enfocan en cambio, en el diseño o la implementación (Sommerville 2010).

A continuación, se muestran los RF identificados:

- **RF 1**: Importar indicadores estadísticos desde una hoja de cálculo**.**
- **RF 2**: Construir estratificación.
  - **RF 2.1:** Construir agrupamientos.
  - **RF 2.2:** Visualizar agrupamientos construidos en mapa temático
- **RF 3:** Exportar mapa temático de una estratificación como imagen.

2.3.3 Requisitos no funcionales

Los requisitos no funcionales (RNF) son propiedades o cualidades que el sistema debe tener. Estas propiedades o cualidades se refieren a las características que hacen al sistema estable, usable, rápido, confiable y escalable (Sommerville 2010).
A continuación, se muestran los RNF identificados:

Requisitos de Software

- **RNF 1:** Se debe tener instalada la herramienta QGIS en su versión 2.6 o superior.
- **RNF 2:** Se debe tener instalado el GBD PostgreSQL en su versión 9.0 o superior.
- **RNF 3:** Se debe tener instalado el módulo Postgis en su versión 2.1.5 o superior.

Requisitos de Hardware

- **RNF 4:** La estación de trabajo debe contar con al menos 1,0 GB de *Random Access Memory* (RAM, por sus siglas en inglés).

- **RNF 5:** La capacidad mínima de espacio en disco debe ser 2.0 GB.

Requisitos de Usabilidad

- **RNF 6:** Debe tener una interfaz gráfica atractiva y de fácil intuición, de tal forma que la aplicación podrá ser usada por cualquier usuario con conocimientos básicos sobre geografía e informática. Debe mostrar mensajes al usuario que le ayuden a llevar a cabo la tarea que realiza.

Requisitos de Interfaz

- **RNF 7:** Debe tener una interfaz amigable y con apariencia profesional.
- **RNF 8:** La interfaz debe tener un diseño sencillo y ser de fácil comprensión para el usuario.

Restricciones de diseño e implementación

- **RNF 9:** Se hace uso de la herramienta QGIS en su versión 2.6 e IDE Pycharm 3.4.
- **RNF 10:** El lenguaje de programación usado para la implementación es Python.

2.4 Fase de planificación

La metodología XP define como fase inicial la planificación. Durante esta etapa se lleva a cabo el proceso de identificación y confección de las historias de usuario, así como la familiarización del equipo de trabajo con las tecnologías y herramientas seleccionadas para el desarrollo del software. También el cliente especifica la prioridad en que se deben implementar las historias de usuario, además de una estimación del esfuerzo que

costará. El resultado de la fase es un plan de entregas donde se realiza una estimación de las versiones que tendrá el producto en su realización, de manera tal que guíe el desarrollo del mismo (Beck y Gamma 2000).

2.4.1 Historias de usuarios

Las historias de usuarios (HU) son la técnica utilizada en XP para especificar los requisitos del software. Se trata de tarjetas de papel en las cuales el cliente describe brevemente las características que el sistema debe poseer, sean requisitos funcionales o no funcionales. El tratamiento de las historias de usuario es muy dinámico y flexible, en cualquier momento historias de usuario pueden romperse, reemplazarse por otras más específicas o generales, añadirse nuevas o ser modificadas. Cada historia de usuario es lo suficientemente comprensible y delimitada para que los programadores puedan implementarla en unas semanas (Jeffries, Anderson y Hendrickson 2001).

A continuación, un ejemplo de tabla de historia de usuario, utilizando como RF “Importar indicadores estadísticos desde una hoja de cálculo” las restantes se encuentran en el anexo.

Tabla 3: Historia de usuario “Importar indicadores estadísticos desde una hoja de cálculo”. Fuente: Elaboración propia

<table>
<tr><td colspan="3">Historia de usuario “Importar indicadores estadísticos desde una hoja de cálculo”</td></tr>
<tr><td>Número: 1</td><td colspan="2">Nombre Historia de Usuario: Importar indicadores estadísticos desde una hoja de cálculo.</td></tr>
<tr><td colspan="3">Usuario: Experto</td></tr>
<tr><td>Prioridad en Negocio: Media</td><td colspan="2">Riesgo en Desarrollo: Media</td></tr>
<tr><td>Puntos Estimados: 7</td><td colspan="2">Iteración Asignada: 4</td></tr>
<tr><td colspan="3">Programador responsable: Alejandro Alea González y Alberto Pérez Ojeda</td></tr>
<tr><td colspan="3">Descripción:
La aplicación debe ser capaz de obtener los indicadores estadísticos de los territorios a evaluar desde una hoja de cálculo seleccionada por el usuario.</td></tr>
<tr><td colspan="3">Observaciones:</td></tr>
</table>

2.4.2 Estimación de esfuerzos por historias de usuario

En el presente epígrafe se realiza la estimación del esfuerzo por HU, se hace necesario tener en cuenta que estas deben ser programadas en un tiempo de una a tres semanas. Si la estimación es superior a tres semanas, se divide en dos o más HU. Si es menor de una semana, se combina con otra HU. Estas estimaciones permiten tener una medida de la velocidad del proyecto y ofrecen una guía a la cual ajustarse. Los resultados estimados se muestran en la Tabla 4.

Tabla 4: Estimación de esfuerzos por historias de usuario. Fuente: Elaboración propia.

Historia de usuario	Puntos de estimación(semanas)
HU 1: Importar indicadores estadísticos desde una hoja de cálculo	0.6
HU 2.1: Construir agrupamiento. HU 2.2: Visualizar estratos construidos en un mapa temático.	3
HU 3: Exportar mapa temático de una estratificación como imagen.	0.3

2.4.3 Plan de iteraciones

Una vez finalizadas las HU se debe crear un plan de iteraciones, indicando cuáles se desarrollarán en cada iteración del componente. En la Tabla 5 se muestra cómo quedó definido el plan de iteraciones de la solución propuesta.

Tabla 5: Plan de iteraciones. Fuente: Elaboración propia.

Iteraciones	Orden de las historias de usuario a implementar	Duración de las iteraciones (semanas)
Iteración 1	Construir agrupamiento	3
Iteración 2	Visualizar agrupamientos construidos en mapa temático.	3
Iteración 5	Exportar mapa temático de una estratificación como imagen.	0.6

Total	6.6

2.4.4 Plan de entrega

El plan de entregas establece cuáles HU serán agrupadas para conformar una entrega, y el orden de las mismas (Joskowicz 2008). En este plan se concentran las funcionalidades referentes a un mismo tema en módulos, esto permite un mayor entendimiento en la fase de implementación. El plan tiene como objetivo definir el número de liberaciones que se realizarán en el transcurso del proyecto y las iteraciones que se requieren para desarrollar cada una. De esta forma se puede trazar el plan de entrega en función de estos dos parámetros: el tiempo de desarrollo ideal y el grado de importancia para el cliente. En Tabla 6 se presenta el plan de entregas de la aplicación informática propuesta.

Tabla 6: Plan de duración de las entregas. Fuente: Elaboración propia.

	Final de la 1ra Iteración	**Final de la 2da Iteración**	**Final de la 3ra Iteración**	**Final de la 4ta Iteración**	**Final de la 5ta Iteración**
Módulos	4ta semana de abril	1ra semana de mayo	2da semana de mayo	3ra semana de mayo	4ra semana de mayo
Estratificación	**V 1.0**	**V1.1**	**V1.2**	**V1.3**	**Finalizado**

2.5 Fase de diseño

La metodología de desarrollo XP plantea prácticas especializadas que accionan directamente en la realización del diseño para lograr un sistema robusto y reutilizable. Se trata en todo momento de conservar su simplicidad, es decir, crear un diseño evolutivo que va mejorando incrementalmente y que permite hacer entregas pequeñas y frecuentes, de valor para el cliente, basado principalmente en el desarrollo de las tarjetas Clase-Responsabilidad-Colaboración (CRC).

2.5.1 Tarjetas Clase-Responsabilidad-Colaboración

Las tarjetas CRC son utilizadas para representar las responsabilidades de las clases y sus interacciones. Estas tarjetas permiten trabajar con una metodología basada en objetos, permitiendo que el equipo de desarrollo completo contribuya en la tarea del diseño. En cada tarjeta CRC el nombre de la clase se coloca a modo de título, las responsabilidades se colocan a la izquierda y las clases que se implican en cada responsabilidad a la derecha, en la misma línea que su requerimiento correspondiente. Una clase es cualquier persona, evento, concepto, pantalla o reporte. Las responsabilidades de una clase son las cosas que conoce y las que realizan, sus atributos y métodos. Los colaboradores de una clase son las demás clases con las que trabaja en conjunto para llevar a cabo sus responsabilidades (Casas y Reinaga 2008). En la Tabla 7 y Tabla 8 se muestran las tarjetas CRC correspondientes a las clases *Agrupamiento* y *Vecino*

Tabla 7: Tarjeta CRC para la clase Estratificación. Fuente: Elaboración propia.

Clase: Agrupamiento	
Responsabilidad	**Colaboración**
Crear instancias de la clase Indicador.	**Vecino, Estrato**

Tabla 8: Tarjeta CRC para la clase Estratificación. Fuente: Elaboración propia.

Clase: Vecino	
Responsabilidad	**Colaboración**
Calcular los vecinos	**ControladorVecino, Estratos**

2.6 Arquitectura de software

La arquitectura de software consiste en un conjunto de patrones y abstracciones coherentes que proporcionan un marco definido y claro para interactuar con el código fuente del software, es la definición y estructuración de una solución que cumple con los requisitos técnicos y operativos y a su vez optimiza atributos que implican una serie de decisiones, tales como la seguridad, el rendimiento y la capacidad de administración. Estas decisiones en última instancia, afectan la calidad de la aplicación, el mantenimiento, el rendimiento y el éxito global (Pressman 2010).

2.6.1 Estilo arquitectónico a utilizar

Un estilo es un concepto descriptivo que define una forma de articulación u organización arquitectónica. El conjunto de los estilos cataloga las formas básicas posibles de estructuras de software. Estos permiten expresar un esquema de organización estructural esencial para un sistema de software (Pressman 2010). En este trabajo se hace uso del estilo arquitectónico en capas, logrando que el sistema quede organizado y así tener un orden lógico en la programación del mismo.

Arquitectura en capas

La arquitectura en capas se define como una organización jerárquica donde cada capa proporciona servicios a la inmediatamente superior y se sirve de las prestaciones que le brinda la inmediatamente inferior. Con esto se logra abstraer las funcionalidades de una capa de manera tal que pueda ser totalmente remplazada sin afectar a las otras, solamente cambiar las referencias de las implicadas en el cambio (España y Fernando 2016; Pressman 2010). De esta forma es sencillo crear diferentes interfaces sobre un mismo sistema sin requerirse cambio alguno en la capa de datos o lógica.
En la Figura 4 se presenta una imagen de la arquitectura de la solución.

Figura 4: Evidencia de la arquitectura en capas. Fuente: Elaboración propia.

Capa de presentación: es la parte de la aplicación con que el usuario interactúa, por lo que deberá cumplir muchos requisitos. Estos requisitos abarcan factores generales como la facilidad de uso, rendimiento, diseño e interactividad. Es imprescindible que la aplicación tenga un buen diseño para apoyar una experiencia de usuario intuitiva, desde el principio, ya que la experiencia del usuario es influenciada por muchos aspectos diferentes de la arquitectura de la aplicación.

Capa de negocio: es donde residen las clases gestoras de la información, se reciben las peticiones del usuario y se envían las respuestas a la capa de presentación. Se nombra capa de negocio porque es aquí donde se establecen todas las reglas que deben cumplirse. Esta capa se comunica con la capa de presentación, para recibir las solicitudes del usuario y presentar los resultados obtenidos, y con la capa de acceso a

datos, para enviar datos que necesitan persistirse en la base de datos o recibirlos de la misma.

Capa de acceso a datos: está constituida por las clases gestoras del acceso a datos, encargadas de acceder a los mismos y realizan todo el almacenamiento de la información. Esta capa recibe solicitudes de almacenamiento o recuperación de información desde la capa de negocio.

2.7 Patrones de diseño

Los patrones de diseño constituyen la base para la búsqueda de soluciones a problemas comunes en el desarrollo de software y otros ámbitos referentes al diseño de interacción o interfaces. Un patrón de diseño resulta ser una solución a un problema de diseño. Para que una solución sea considerada un patrón debe poseer ciertas características. Una de ellas es que debe haber comprobado su efectividad resolviendo problemas similares en ocasiones anteriores. Otra es que debe ser reutilizable, lo que significa que es aplicable a diferentes problemas de diseño en distintas circunstancias.

2.7.1 Patrones Generales de Software para la Asignación de Responsabilidades

Los Patrones Generales de Software para la Asignación de Responsabilidades (GRASP, por sus siglas en inglés) son utilizados para describir los principios fundamentales del diseño y la asignación de responsabilidades (Craig 2007). Entre los que se utilizaron en la solución figuran los siguientes: Experto, Creador, Controlador, Bajo acoplamiento y Alta cohesión.

Experto: el patrón Experto define la forma de asignar de forma adecuada las responsabilidades en un modelo de clases. Indica que la responsabilidad de la creación de un objeto o la implementación de un método debe recaer en la clase que conoce toda la información necesaria para crearlo. Dicho patrón se evidencia en la aplicación informática, en la clase Territorio como esta posee toda la información necesaria para

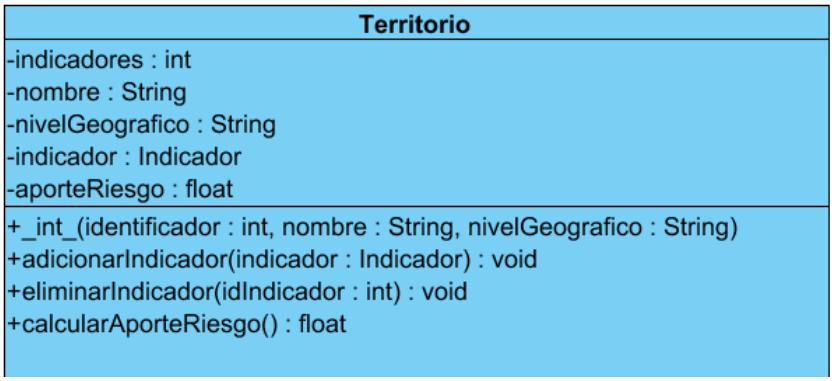

Figura 5: Evidencia del patrón Experto. Fuente: Elaboración propia.

calcular el aporte de riesgo de cada territorio se le es asignada dicha responsabilidad. En la Figura 5 se muestra una imagen de dicha clase.

Creador: Este patrón guía la asignación de responsabilidades relacionadas con la creación de objetos. La intención básica del mismo es encontrar un creador que necesite conectarse al objeto creado en alguna situación. En la aplicación informática se pone de manifiesto dicho patrón en la clase Estrato, a esta se le asigna la responsabilidad de crear instancias de la clase Territorio. En la Figura 6 se presenta la evidencia de dicho patrón.

Alta cohesión: la cohesión es una medida de cuán relacionadas y enfocadas están las responsabilidades de una clase. Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas que no realicen un trabajo enorme. Una baja cohesión hace muchas cosas no afines o realiza trabajo excesivo. En resumen, este patrón se observa cuando una clase tiene la responsabilidad de realizar una labor dentro del sistema, no desempeñada por el resto de los componentes del diseño. Este patrón se evidencia en la aplicación informática en conjunto con el patrón bajo acoplamiento, de forma tal que cada clase realice sus acciones y se evita que otra clase realice acciones correspondientes a la clase con la que está relacionada.

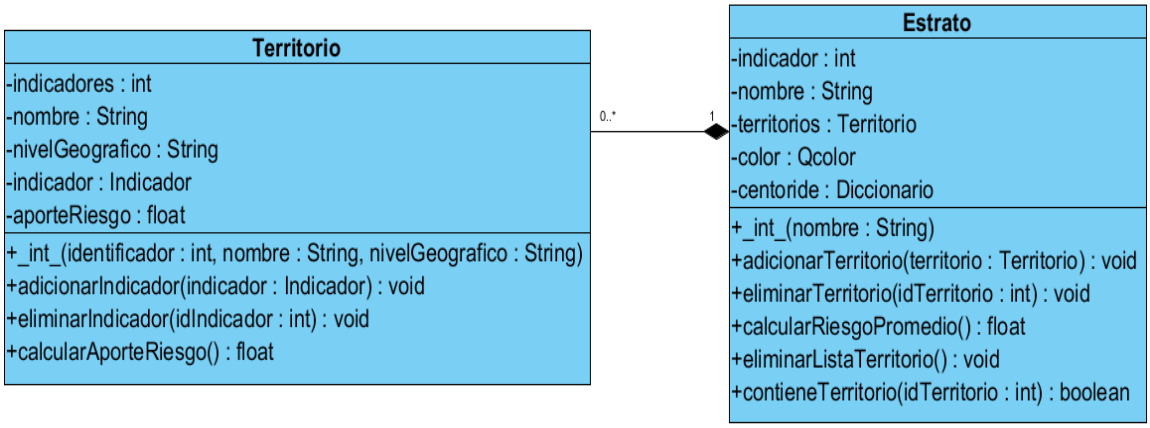

Figura 6: Evidencia del patrón Creador. Fuente: Elaboración propia.

Bajo acoplamiento: este patrón se garantiza en la aplicación basándose en la propia arquitectura del sistema, lo que permite que las dependencias entre las clases sea muy poca, ya que solamente las clases de una capa se pueden comunicar con las de la capa inmediatamente inferior.

Controlador: Permite asignar la responsabilidad de controlar el flujo de eventos del sistema, a clases específicas, facilitando la centralización de actividades. El controlador no realiza estas actividades, las delega en otras clases con las que mantiene un modelo de alta cohesión. Un error muy común es asignarle demasiada responsabilidad y alto nivel de acoplamiento con el resto de los componentes del sistema. Este patrón se evidencia en la aplicación informática, en la clase *ControladorEstratificador* se le asignó la responsabilidad de manejar los eventos del sistema generados por el usuario. En la Figura 7 se muestra una imagen de dicha clase.

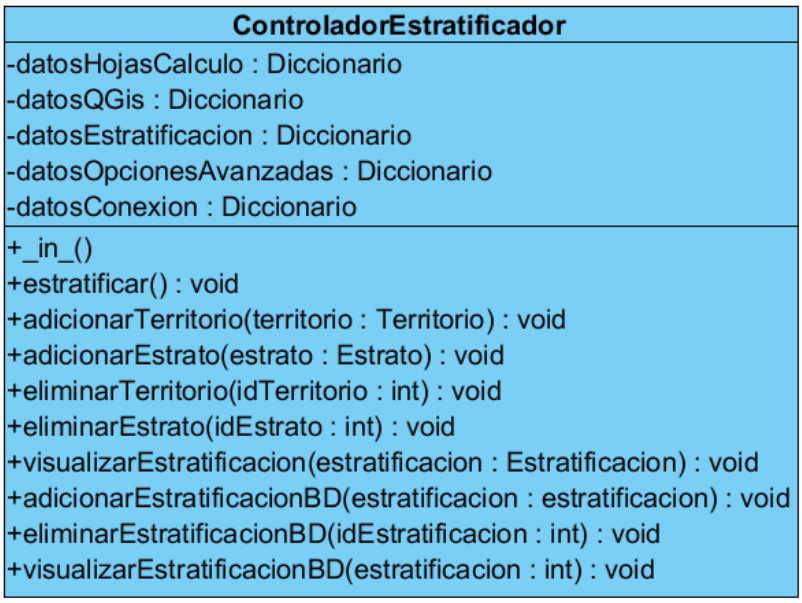

Figura 7: Evidencia del patrón Controlador. Fuente: Elaboración propia.

2.7.2 Patrones del Grupo de Cuatro

Los Patrones del Grupo de Cuatro (GoF, por sus siglas en inglés) resuelven problemas específicos de diseño de software (Rodriguez 2010). Estos patrones se agrupan en las siguientes categorías: creacionales, estructurales y de comportamiento.

Comportamiento: Contribuyen a definir la comunicación e interacción entre los objetos de un sistema (Rodriguez 2010).

Creacionales: Permiten abstraer el proceso de instanciación y ocultar los detalles de cómo los objetos son creados o inicializados (Rodriguez 2010).

Estructurales: Describen cómo las clases y objetos pueden ser combinados para formar grandes estructuras y proporcionar nuevas funcionalidades (Rodriguez 2010).

Método plantilla: es un patrón de comportamiento que define en una operación el esqueleto de un algoritmo, delegando en las subclases algunos de sus pasos, esto permite que las subclases redefinan ciertos pasos de un algoritmo sin cambiar estructura. Este patrón se evidencia en las clases *AlgoritmoST-HDBSCAN* esta heredan todas las funcionalidades de la clase *Algoritmo,* y redefinen el método y *run()* en función de sus características. En la Figura 8 se muestra cómo se evidencia el patrón plantilla en la aplicación informática propuesta.

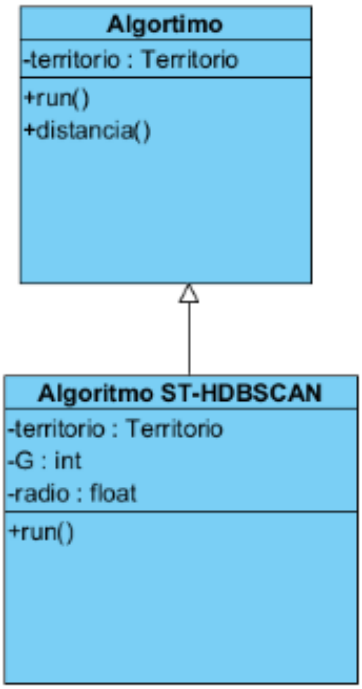

Figura 8: Evidencia del patrón Plantilla. Fuente: Elaboración propia.

2.8 Modelo de la vista lógica de la estructura del sistema

La vista lógica muestra un subconjunto significativo arquitectónicamente del modelo de diseño, es decir, un subconjunto de las clases, subsistemas y paquetes, y realizaciones. En la Figura 9 se muestra el diagrama de clases de la aplicación informática propuesta.

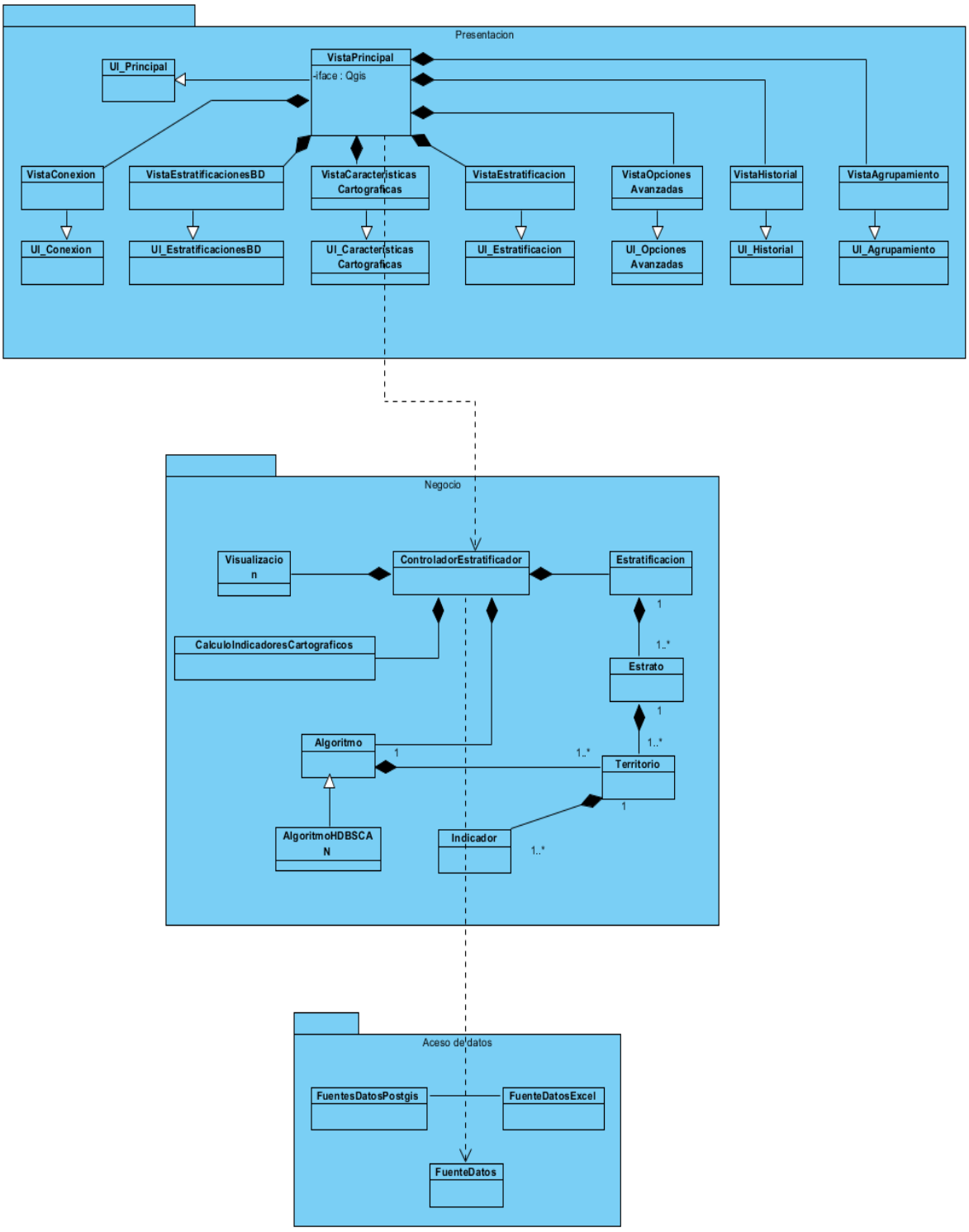

Figura 9: Modelo de la vista lógica de la estructura del sistema. Fuente: Elaboración propia.

2.9 Diagrama Entidad-Relación

Los Diagramas Entidad-Relación (DER) definen todos los datos que se introducen, se almacenan, se transforman y se producen dentro de una aplicación (Pressman 2010). Estos modelos representan a la realidad a través de un esquema gráfico empleando la terminología de Entidades, que son objetos que existen y son los elementos principales

que se identifican en el problema a resolver. En la Figura 10 se muestra el diagrama Entidad-Relación de la aplicación informática propuesta.

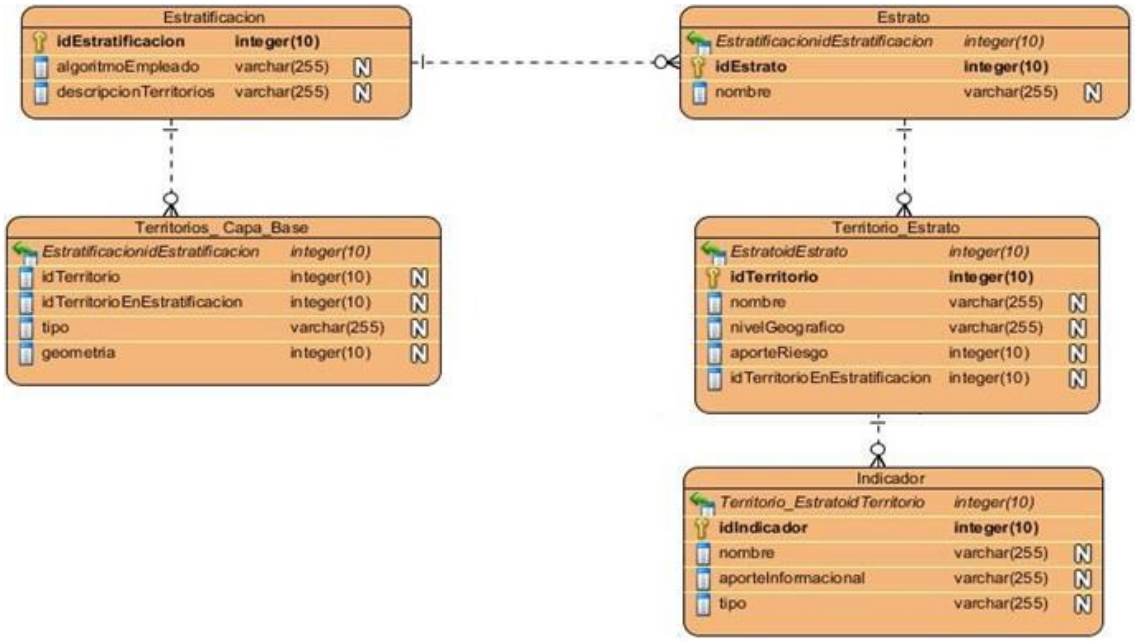

Figura 10: Diagrama Entidad-Relación. Fuente: Elaboración propia.

2.10 Fase de implementación

Luego de definir los elementos necesarios en la etapa de planificación y diseño se pasa a la de codificación o implementación de la aplicación, donde se da cumplimento al plan de iteraciones. En esta fase se realiza la implementación de las HU que fueron seleccionadas por cada iteración, además se crean las tareas de ingeniería para ayudar a organizar la implementación exitosa de las HU. Al finalizar esta fase el cliente estará listo para comenzar a realizar las pruebas.

2.11 Tareas de ingeniería

Cada HU está compuesta por una o varias tareas de ingeniería, éstas se realizan para especificar las acciones llevadas a cabo por los programadores. En la Tabla 9 se detallan para la iteración número uno, las tareas a desarrollar por cada HU y en la Tabla 10 se describe una tarea de ingeniería que responde a una HU arquitectónicamente significativa, el resto se encuentran especificadas en anexos.

Tabla 9: Distribución de tareas de ingeniería por HU (iteración 1). Fuente: Elaboración propia.

HU	Tareas de ingeniería por HU

Construir estratos	• Agrupar los territorios utilizando el algoritmo ST-HDBSCAN. • Obtener el aporte informacional de los indicadores. • Obtener el aporte de riesgo de salud de los territorios. • Normalizar los datos de los indicadores.

Tabla 10: Tarea de Ingeniería No.1. Fuente: Elaboración propia.

Tarea de ingeniería	
Número de Tarea: 1	Número Historia de Usuario: HU # 3.1
Nombre Tarea: Agrupar territorios utilizando el algoritmo ST-HDBSCAN	
Tipo Tarea: Desarrollo	Puntos Estimados: 1
Fecha Inicio: 26/04/2018	Fecha Fin: 30/04/2018
Programador :Alberto Pérez Ojeda y Alejandro Alea González	
Descripción: Esta tarea permite agrupar los territorios seleccionados por el usuario utilizando el algoritmo ST-HDBSCAN.	

2.11.1 Estándares de codificación

XP resalta que la comunicación de los programadores es a través del código, por lo que es necesario que sigan ciertos estándares de programación para lograr un entendimiento entre los programadores, de manera que cualquier persona del equipo de desarrollo pueda modificar el código. Además, se hace preciso que el código sea entendible para que posteriormente otros programadores puedan apoyarse en ese trabajo y desarrollen otras soluciones.

En el caso del componente que se desarrolla, el estándar que se utiliza es:

Máxima longitud de las líneas

- Todas las líneas se limitan a un máximo de 79 caracteres.

Importaciones

- Las importaciones se encuentran en líneas separadas.

Comentarios

- Se utilizan comentarios de una línea para hacer más entendible el código.

Comentarios de una línea: comentario pequeño que solo abarca una línea y describe el código que le sigue.

 # Esto es un comentario de una línea

Estilo de los nombres

- **Clases e Interfaces:** los nombres de las clases presentan la primera letra en mayúscula, en caso de ser un nombre compuesto, la inicial de cada palabra se representa en mayúscula. Se utilizan nombres simples y de alguna manera que describan el contenido, se usan palabras completas, a no ser que la abreviatura sea muy conocida.
- **Métodos y variables:** los nombres de los métodos se representan en minúscula, en caso de ser un nombre compuesto, la inicial de la primera palabra se simboliza en minúscula, y la de las otras palabras que lo componen en mayúscula. Los nombres de las variables son cortos, pero con significados lógicos, capaces de permitir a un observador identificar su función.

2.12 Conclusiones parciales

- La propuesta de solución definida facilita realizar el proceso de estratificación espacio-temporal de territorios utilizando SIG.
- El algoritmo ST-HDBSCAN, variación del HDBSCAN facilita la integración de la componente temporal y permite construir grupos a partir de tres dimensiones o más en los datos.
- La identificación de los requisitos permitió un mayor entendimiento de las necesidades del cliente. Las tres HU divididas por cinco iteraciones logró una mejor organización del trabajo y el establecimiento de fechas de culminación para cada iteración.

CAPÍTULO 3: Verificación de la solución

3.1 Introducción

En el siguiente capítulo se realizan las pruebas definidas por la metodología seleccionada, así como las pruebas de camino básico para verificar el código, y las pruebas de aceptación para comprobar si al final de cada iteración se consiguió la funcionalidad requerida. Se realiza un caso de estudio para verificar la validez de los resultados de la solución propuesta.

3.2 Fase de pruebas

Cuando se desarrolla una solución informática se deben realizar una gran cantidad de pruebas para verificar que el código esté correcto. Estas pruebas normalmente tienen que ser ejecutadas en varias ocasiones y se ven afectadas por los cambios que se introducen conforme se va construyendo la solución. XP divide las pruebas en dos grupos: pruebas de aceptación, o pruebas funcionales diseñadas por el cliente final, destinadas a evaluar si al final de una iteración se consiguió la funcionalidad requerida y pruebas unitarias, encargadas de verificar el código y diseñadas por los programadores.

3.2.1 Pruebas de aceptación

Las pruebas de aceptación XP son especificadas por el cliente, y se centran en las características y funcionalidades generales del sistema, que son visibles y revisables por parte del usuario. Estas pruebas derivan de las HU que se han implementado como parte de la liberación del software (Joskowicz 2008).

Los clientes son responsables de verificar que los resultados de estas pruebas sean correctos. Así mismo, en caso de que fallen varias pruebas, deben indicar el orden de prioridad de resolución. Una HU no se puede considerar terminada hasta tanto pase correctamente todas las pruebas de aceptación. Dado que la responsabilidad es grupal, es recomendable publicar los resultados de las pruebas de aceptación, de manera que todo el equipo esté al tanto de esta información (Joskowicz 2008).

Casos de prueba

En la Tabla 11 se muestra los casos de prueba de aceptación aplicados a las HU *Construir estratificación.*

Tabla 11: Caso de prueba de aceptación Construir agrupamientos. Fuente: Elaboración propia.

Caso de prueba de aceptación	
Código: HU2.1_P1	**Historia de Usuario: 2.1**

Nombre: Construir agrupamiento.
Descripción: Prueba para validar la funcionalidad construir estratos.
Condiciones de ejecución: • El usuario debe escoger la opción *seleccionar indicadores*. • El usuario debe seleccionar un algoritmo de agrupamiento. • El usuario debe seleccionar la opción *Aceptar*
Resultados esperados: En caso que se cumplan las condiciones de ejecución, el sistema obtiene el agrupamiento por los indicadores seleccionados. En caso contrario, el sistema muestra un mensaje informando el motivo por el cual no obtuvo las características cartográficas seleccionadas por el usuario.
Evaluación de la prueba: Prueba satisfactoria.

Análisis de los resultados

Para validar que el resultado obtenido por el sistema coincide con el resultado esperado por el cliente, se diseñaron un total de 9 casos de prueba de aceptación en conjunto cliente-desarrolladores. De este total, 6 arrojaron el resultado esperado mientras que 3 pruebas resultaron fallidas, las funcionalidades que respondían a estas pruebas fueron tratadas en la siguiente iteración y al volver a aplicar las pruebas de funcionalidad mostraron un resultado exitoso. Finalmente se obtuvieron un total 9 pruebas satisfactorias de 9 casos de prueba aplicados.

3.2.2 Pruebas de caja blanca

Las pruebas de caja blanca se centran en los detalles procedimentales del componente, por lo que su diseño está fuertemente ligado al código fuente. Se escogen distintos valores de entrada para examinar cada uno de los posibles flujos de ejecución del componente cerciorándose que se devuelvan los valores de salida adecuados (Pressman 2010).

Las pruebas de caja blanca intentan garantizar que:

- Se ejecuten al menos una vez todos los caminos independientes de cada módulo.
- Se utilicen las decisiones en su parte verdadera y en su parte falsa.
- Se ejecuten todos los bucles en sus límites.
- Se utilicen todas las estructuras de datos internas.

La técnica utilizada dentro de las pruebas de caja blanca fue camino básico.

Sentencias de código del método run*( )*.

```
def run (self):
    datos=self.objeIndica_porAnnos
    matriz_cluster2 = []

    clusterer = hdbscan.HDBSCAN(min_cluster_size=2)


    # primer agrupamiento año por año de los objetos y sus indicadores

    for n in len(datos):
        annoAct = np.array(datos[n])

        matriz_cluster2.append(clusterer.fit_predict(annoAct))

    matriz_cluster2 = zip(*matriz_cluster2)

    matriz_cluster2 = np.array(matriz_cluster2)


    # segudo agrupamiento para ver el comportamiento de los objetos
    con respecto al paso de   los años

    cluster_labels = clusterer.fit_predict(matriz_cluster2)


    return cluster_labels
```

Luego de haberse construido el grafo, se realiza el cálculo de la *complejidad ciclomática* mediante las tres fórmulas descritas a continuación, las cuales deben arrojar el mismo resultado para asegurar que el cálculo de la complejidad sea correcto.

1. La complejidad ciclomática coincide con el número de regiones del grafo de flujo.

2. La complejidad ciclomática, V(G), de un grafo de flujo G, se define como V(G) = Aristas - Nodos + 2.

3. La complejidad ciclomática, V(G), de un grafo de flujo G, también se define como V(G) = *Nodos de predicado* + 1.

A partir del grafo de flujo del método *run( )* que se presenta en la Figura 11, la complejidad ciclomática sería:

- Como el grafo tiene 2 regiones, V(G) = 2
- Como el grafo tiene 5 aristas y 5 nodos, V(G) = 5 - 5 + 2 = 2
- Como el grafo tiene 1 nodos de predicado, V(G) = 1 + 1 = 2

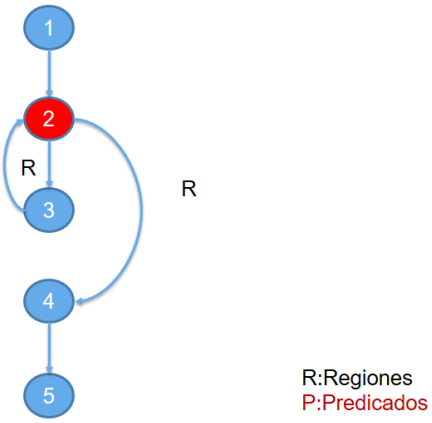

Figura 11: Grafo de flujo del método run( ). *Fuente: Elaboración propia.*

Dado a que el cálculo de las tres fórmulas anteriormente mencionadas arrojó el mismo resultado se puede plantear que la complejidad ciclomática del método es 2. Esto significa que existen 2 posibles caminos por donde el flujo puede circular. Este valor representa el número mínimo de casos de pruebas para el procedimiento tratado.

Caminos básicos identificados:

- Camino 2: 1-2-4
- Camino 1: 1-2-3-2-4

Para cada camino básico determinado se realiza un diseño de caso de prueba en la Tabla 12 y Tabla 13.

Tabla 12: Caso de prueba para el camino básico #1. Fuente: Elaboración propia.

Caso de Prueba para el camino básico #1 (1-2-4)	
Descripción	Prueba para comprobar los resultados de la función run()
Condición de ejecución	self.objeIndica_porAnnos=0
Entrada	self.objeIndica_porAnnos=[]
Resultado	cluster_labels
Resultado de la prueba	Prueba satisfactoria

Tabla 13: Caso de prueba para el camino básico #2. Fuente: Elaboración propia.

Caso de Prueba para el camino básico #1 (1-2-3-2-4)

Descripción	Prueba para comprobar los resultados de la función run()
Condición de ejecución	self.objeIndica_porAnnos>0
Entrada	self.objeIndica_porAnnos=[1,2]
Resultado	cluster_labels
Resultado de la prueba	Prueba satisfactoria

Análisis de los resultados

Para la validación del código generado en el desarrollo del componente se seleccionaron los métodos más relevantes, a los cuales se le realizaron las pruebas para poder evaluar si el funcionamiento de cada uno de estos métodos se comportó de la manera esperada. Se realizaron un total de 2 pruebas a las 3 funcionalidades, las cuales fueron satisfactorias 2 de las 3 funcionalidades en la primera iteración de pruebas. En la segunda iteración se erradicó la no conformidad. Comprobándose la estabilidad de la lógica aplicada en el código generado en el desarrollo del componente informático.

3.3 Caso de estudio

Con el fin de valorar los resultados de la solución propuesta se decide aplicar un caso de estudio, donde se utilizó como fuente de información el Anuario Estadístico de los años 2014-2017 y se seleccionaron los indicadores siguientes:

- Mortalidad infantil por cada 1000 nacidos vivos.
- Mortalidad infantil de los niños menores de cinco años por cada 1000 nacidos vivos.
- Mortalidad por enfermedades del corazón por cada 100 000 habitantes.
- Mortalidad por tumores malignos por cada 100 000 habitantes.
- Mortalidad por enfermedades cerebrovasculares por cada 100 000 habitantes.
- Mortalidad por influenza y neumonía por cada 100 000 habitantes.
- Mortalidad por accidentes por cada 100 000 habitantes.
- Mortalidad perinatal por cada 1000 nacidos vivos.

- Mortalidad por enfermedades infecciosas y parasitarias por cada 100 000 habitantes.
- Mortalidad materna por cada 100 000 nacidos vivos.
- Incidencia de tuberculosis por cada 100 000 habitantes
- Incidencia de hepatitis por cada 100 000 habitantes.
- Incidencia de diabetes por cada 100 000 habitantes.
- Incidencia de hipertensión por cada 100 000.
- Incidencia de asma por cada 100 000 habitantes.
- Incidencia de bajo peso al nacer.
- Consultas médicas por habitante.
- Ingresos por cada 100 habitantes.
- Camas de asistencia por cada 1000 habitantes.
- Consultas de puericultura por habitante.
- Consultas de pediatría por habitante.
- Densidad poblacional.
- Población mayor de 60 años.
- Población menor de 1 año.
- Población menor de 15 años.
- Natalidad por cada 1000 habitantes.

Aplicación del caso de estudio

Para realizar la clasificación de cada una de las provincias de Cuba utilizando el componente desarrollado se empleó el algoritmo de agrupamiento ST-HDBSCAN.

Resultados de la aplicación del caso de estudio

A continuación, se muestran los resultados de la estratificación de las provincias de Cuba a partir del proceso analítico-estadístico de las variables de salud escogidas. En la Figura 12 se aprecian los mismos de forma mapificada y en la Tabla 14 de manera más detallada.

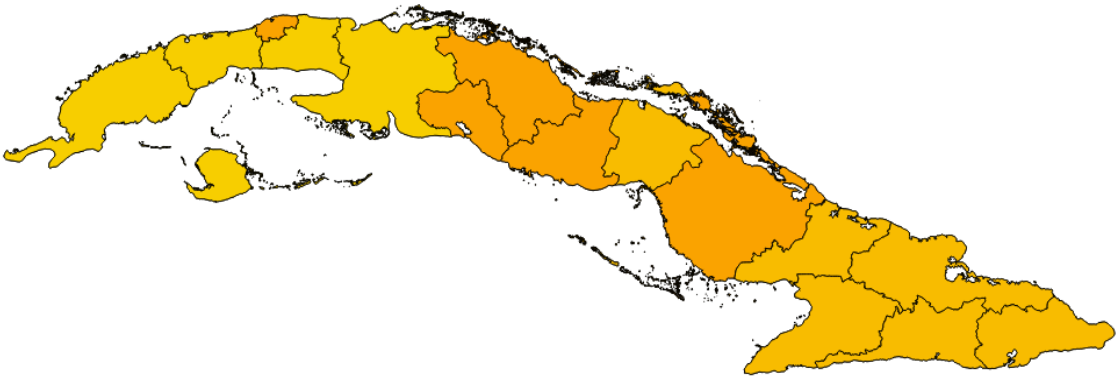

Figura 12: Mapa temático de la estratificación realizada utilizando la herramienta propuesta. Fuente: Elaboración propia.

Comparación de resultados

Para determinar la similitud existente entre los resultados obtenidos por el componente desarrollado y los del proceso realizado en (Companioni 2005), a partir del caso de estudio descrito anteriormente, se utilizó la medida para evaluar la exactitud de los estratos resultantes que se describe a continuación.

Suponiendo que el número final de estratos es k, la medida de exactitud r está dada por la ecuación siguiente (Hernández 2006).

$$r = \frac{\sum_{i=1}^{k} a_i}{n}$$

Donde n es el número de territorios de la estratificación, a_i es el número de territorios que aparecen clasificados correctamente en el estrato i y en su correspondiente clase, la cual es aquella que tenga el número máximo. En otras palabras, es el número de territorios con las etiquetas de la clase que dominan en el estrato i. Por lo tanto, el error está definido en la ecuación

$$e = 1 - r$$

Una medida de error en un rango ≤ 0.3, representa un valor aceptable de error (Hernández 2006).

A continuación se presenta una tabla comparativa entre los resultados del proceso de estratificación realizado utilizando el componente desarrollado y él efectuado en (Companioni 2005).

Tabla 14: Comparación de los resultados de los procesos de estratificación realizados (por estratos). Fuente: Elaboración propia.

Resultados del proceso realizado en (Companioni 2005)	Resultados obtenidos utilizando el componente desarrollado
Cienfuegos, Santiago de Cuba, Ciego de Ávila	Cienfuegos, Santiago de Cuba, Camagüey, Guantánamo, Sancti Spíritus, Granma
Pinar del Río, La Habana, Matanzas, Sancti Spíritus, Camagüey, Guantánamo, Granma	Pinar del Río, Matanzas, Holguín
Ciudad de la Habana , Holguín	Ciego de Ávila, Villa Clara, Las Tunas
, Las Tunas, Villa Clara,	Ciudad de la Habana

A partir de los resultados obtenidos en ambos procesos de estratificación, se aplica la medida de exactitud descrita inicialmente, obteniéndose un error de 0.2787.

Teniendo en cuenta lo anterior se evidencia que existe un valor del error aceptable entre los resultados obtenidos en ambos procesos. Esto permite verificar la veracidad de los estratos generados por el componente desarrollado a partir del caso de estudio propuesto. Los resultados expuestos demuestran la utilidad y validez del empleo de la estratificación territorial, permitiendo obtener una valiosa información estratificada de la situación salud-enfermedad del país, posibilitando implementar acciones más efectivas, y facilitando la distribución de los recursos con un enfoque equitativo.

3.4 Conclusiones parciales

- Las pruebas de aceptación facilitaron detectar, documentar y corregir las no conformidades existentes en el sistema implementado y las pruebas de caja blanca efectuadas determinan la complejidad ciclomática del algoritmo identificando la cantidad de caminos básicos que posee.
- El caso de estudio realizado posibilitó evaluar la aplicabilidad de la solución propuesta a través de los estratos obtenidos y la obtención de riesgo obtenido por estratos.

CONCLUSIONES GENERALES

Como resultados de la presente investigación se obtuvo una propuesta de solución para la estratificación de territorios utilizando SIG que contribuye a la (detección de fenómenos locales y globales en estudios salubristas). En función de los resultados obtenidos se arribó a las siguientes conclusiones.

- Los algoritmos de agrupamientos analizados como parte de la construcción del marco teórico presentan limitaciones para construir grupos sobre datos con más de dos dimensiones.
- El algoritmo ST-HDBSCAN, variación del HDBSCAN facilita la integración de la componente temporal en la estratificación de territorios y permite construir grupos a partir de tres dimensiones o más en los datos.
- La integración de la solución propuesta al sistema QGIS facilitó la realización del proceso de estratificación de territorios utilizando indicadores espaciales y temporales.
- Mediante las pruebas y métricas aplicadas para la verificación de la solución informática se puede afirmar que el complemento desarrollado permite la estratificación espacio-temporal de territorios basada en variables georreferenciadas y detectar fenómenos locales y globales en estudios salubristas.

RECOMENDACIONES

- Incluir al componente desarrollado, la funcionalidad de efectuar análisis estadísticos para realizar comparaciones entre los resultados de las estratificaciones obtenidas.

- Incluir al componente desarrollado, la funcionalidad de ver a través de una línea del tiempo, como se manifiesta el comportamiento de las enfermedades para un mejor análisis.

REFERENCIAS BIBLIOGRÁFICAS

ACHI, J. y VLADIMIR, S., 2018. *SISTEMA PARA LA GESTIÓN DE CLIENTES DE LA EMPRESA WHY5 MARKETING*. S.l.: Quito.

ACKERMANN, M.R., BLÖMER, J., KUNTZE, D. y SOHLER, C., 2014. Analysis of agglomerative clustering. *Algorithmica*, vol. 69, no. 1, pp. 184-215.

AGRAWAL, K.P., GARG, S., SHARMA, S. y PATEL, P., 2016. Development and validation of OPTICS based spatio-temporal clustering technique. *Information Sciences*, vol. 369, pp. 388-401.

ANSELIN, L., 1989. What is Special About Spatial Data? Alternative Perspectives on Spatial Data Analysis (89-4). ,

ARANGO GONZÁLEZ, M.A., JARAMILLO MORALES, J.D. y JARAMILLO ESCOBAR, L., 2016. Técnicas de clustering para detectar patrones espaciales de criminalidad en jóvenes y adultos en Medellín. Octubre del 2013 a noviembre del 2014. *Revista Criminalidad*, vol. 58, no. 1, pp. 25-45.

BARBERÁ, J.M.P., 2017. Análisis de los factores asociados a la elección de estudios universitarios utilizando técnicas de agrupamiento. ,

BATISTA MOLINER, R., COUTIN MARIE, G., FEAL CAÑIZARES, P., GONZÁLEZ CRUZ, R. y RODRÍGUEZ MILORD, D., 2001. Determinación de estratos para priorizar intervenciones y evaluación en Salud Pública. *Revista Cubana de Higiene y Epidemiología*, vol. 39, no. 1, pp. 32-41.

BECK, K. y GAMMA, E., 2000. *Extreme programming explained: embrace change*. S.l.: addison-wesley professional. ISBN 0-201-61641-6.

BELTRÁN, G. y LÓPEZ, G.B., 2012. *Geolocalización y redes sociales*. S.l.: Bubok. ISBN 84-686-1763-6.

BERRY, M.J. y LINOFF, G., 1997. *Data mining techniques: for marketing, sales, and customer support*. S.l.: John Wiley & Sons, Inc. ISBN 0-471-17980-9.

BIRANT, D. y KUT, A., 2007. ST-DBSCAN: An algorithm for clustering spatial–temporal data. *Data & Knowledge Engineering*, vol. 60, no. 1, pp. 208-221.

BRAVO, J.D., 2000. *Breve introducción a la cartografía ya los sistemas de información geográfica (SIG)*. S.l.: Ciemat.

BUSQUÉ, J. y MAESTRO, M.Y.J.S., 2014. Estratificación ambiental de Cantabria: metodología, resultados y aplicaciones de interés pascícola. *Busqué, J. et al*, pp. 2014-2020.

CASAS, S. y REINAGA, H., 2008. Identificación y modelado de aspectos tempranos dirigido por tarjetas de responsabilidades y colaboraciones. *XIV Congreso Argentino de Ciencias de la Computación*. S.l.: s.n.,

CERVANTES SUAREZ, C.A., 2017. *Identificacion de Patrones de Trayectorias Vehiculares Utilizando el Algoritmo MAPAS AUTO-ORGANIZADOS*. S.l.: Universidad de Guayaquil.

Facultad de Ciencias Matematicas y Fisicas. Carrera de Ingenieria en Sistemas Computacionales.

CHACÓN, J., VERA, M., ARIZA, P., BELTRAN, D., HUÉRFANO, Y., ROJAS, J., RODRIGUEZ, J., GRATEROL-RIVAS, M., ARIAS, V. y DEL MAR, A., 2016. Clasificador de perfiles de tratamientos de Diabetes Tipo 2 e Hipertensión Arterial utilizando métodos de recuperación vectorial de información: Caso IPS UNIPAMPLONA. *Revista Latinoamericana de Hipertensión*, vol. 11, no. 2.

COMPANIONI, Y.B., 2005. Construcción de tipologías: metodología de análisis para la estratificación según indicadores de salud. ,

CRAIG, L., 2007. UML y Patrones. *Editorial Prentice Hall,*

DE OCA, L.T.M., RIVERA, D.N. y LEÓN, A.M., 2018. Organización de los sistemas informativos para potenciar el control de gestión empresarial Information Systems Organization for Strengthening Business Management Control. *Revista Cubana de Contabilidad y Finanzas. COFIN HABANA*, no. 1, pp. 88-110.

DENG, M., LIU, Q., WANG, J. y SHI, Y., 2013. A general method of spatio-temporal clustering analysis. *Science China Information Sciences*, vol. 56, no. 10, pp. 1-14.

DEZEREGA, F.J.S., 2016. EVALUACIÓN DE ALGORITMOS DE AGRUPAMIENTO UTILIZANDO APACHE SPARK. ,

DIANDA, D.F., QUAGLINO, M.B. y PAGURA, J.A., 2016. Métodos predictivos de data Mining en el control de procesos industriales. ,

DÍAZ ROSALES, A., CECILIA SIMÓN, N. y NUÑEZ CRUZ, O., 2017. Implementación de la Tecnología de Gestión Total Eficiente de la Energía (TGTEE) en la Central Eléctrica Fuel-Oíl “Antonio Briones Montoto” de Pinar del Río. ,

DÍAZ SEPÚLVEDA, J.F. y CORREA, J.C., 2017. Comparison between CART regression trees and linear regression. ,

DOMINICH, S., 2000. A unified mathematical definition of classical information retrieval. *Journal of the Association for Information Science and Technology*, vol. 51, no. 7, pp. 614-624.

DU, Y., YU, C. y LIU, J., 2009. A study of GIS development based on KML and Google Earth. *INC, IMS and IDC, 2009. NCM’09. Fifth International Joint Conference on*. S.l.: IEEE, pp. 1581-1585. ISBN 1-4244-5209-0.

DUDIK, J.M., KUROSU, A., COYLE, J.L. y SEJDIĆ, E., 2015. A comparative analysis of DBSCAN, K-means, and quadratic variation algorithms for automatic identification of swallows from swallowing accelerometry signals. *Computers in biology and medicine*, vol. 59, pp. 10-18.

EL-SAMAK, A.F. y ASHOUR, W., 2015. Optimization of traveling salesman problem using affinity propagation clustering and genetic algorithm. *Journal of Artificial Intelligence and Soft Computing Research*, vol. 5, no. 4, pp. 239-245.

ESPAÑA, S. y FERNANDO, H., 2016. *Documentación y análisis de los principales frameworks de arquitectura de software en aplicaciones empresariales*. S.l.: Facultad de Informática.

ESTER, M., KRIEGEL, H.-P., SANDER, J. y XU, X., 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. *Kdd*. S.l.: s.n., pp. 226-231.

FOTHERINGHAM, S. y ROGERSON, P., 2013. *Spatial analysis and GIS*. S.l.: CRC Press. ISBN 0-203-22156-7.

GALINDO, A. y GARCÍA, H., 2010. Minería de Datos en la Educación. *Universidad Carlos III*, pp. 1-8.

GÓMEZ, P.S. y SÁNCHEZ SORIA, D., 2016. Concentración, dispersión y características sociodemográficas en la incorporación espacial de la migración peruana en la Ciudad de Córdoba, Argentina. *Población y Salud en Mesoamérica*, vol. 14, no. 1.

GONZÁLEZ, D.P., 2010. *Algoritmos de agrupamiento basados en densidad y variación de clusters*. S.l.: Universitat Jaume I, Departament de Llenguatges i Sistemes Informàtics.

GONZÁLEZ POLANCO, L. y PÉREZ BETANCOURT, G., 2013. La minería de datos espaciales y su aplicación en los estudios de salud y epidemiología. *Revista Cubana de Información en Ciencias de la Salud*, vol. 24, no. 4, pp. 482-489.

HERNÁNDEZ, E., 2006. Algoritmo de Clustering basado en entropía para descubrir grupos en atributos de tipo mixto. *México, DF*,

JEFFRIES, R., ANDERSON, A. y HENDRICKSON, C., 2001. *Extreme programming installed*. S.l.: Addison-Wesley Professional. ISBN 0-201-70842-6.

JOSKOWICZ, J., 2008. Reglas y prácticas en eXtreme Programming. *Universidad de Vigo*, vol. 22.

KHAN, M.E., 2015. K-means clustering.". ,

KISILEVICH, S., MANSMANN, F., NANNI, M. y RINZIVILLO, S., 2009. Spatio-temporal clustering. *Data mining and knowledge discovery handbook*. S.l.: Springer, pp. 855-874.

LEONG, K. y SUNG, A., 2015. A review of spatio-temporal pattern analysis approaches on crime analysis. *International E-journal of Criminal Sciences*, vol. 9, pp. 1-33.

LETELIER, P., 2006. Metodologías ágiles para el desarrollo de software: eXtreme Programming (XP). ,

LIU, X., JIA, M. y GE, W., 2017. The method of lower and upper solutions for mixed fractional four-point boundary value problem with p-Laplacian operator. *Applied Mathematics Letters*, vol. 65, pp. 56-62.

LIU, Y., LI, Z., XIONG, H., GAO, X. y WU, J., 2010. Understanding of internal clustering validation measures. *Data Mining (ICDM), 2010 IEEE 10th International Conference on*. S.l.: IEEE, pp. 911-916. ISBN 1-4244-9131-2.

LÓPEZ, M.E.B., 2015. EXAMEN COMPLEXIVO. ,

MARTÍNEZ, B., TORRES, M. y MORENO, M., 2008. Bases de Datos Espacio-Temporales 1. *Instituto Politécnico Nacional, México, DF*,

MCINNES, L., HEALY, J. y ASTELS, S., 2017. hdbscan: Hierarchical density based clustering. *The Journal of Open Source Software*, vol. 2, no. 11, pp. 205.

METZGER, M.J., SHKARUBA, A.D., JONGMAN, R.H.G. y BUNCE, R.G.H., 2012. Descriptions of the European environmental zones and strata. . S.l.: Alterra.

MISAS, M., LÓPEZ-ENCISO, E.A. y QUERUBIN, P., 2015. La inflación en Colombia: una aproximación desde las redes neuronales. *Artículos de revista*,

MONTAÑO MORENO, J.J., 2017. Redes neuronales artificiales aplicadas al análisis de datos. ,

MURAZZO, M.A., RODRIGUEZ, N.R., GUEVARA, M.J. y TINETTI, F.G., 2016. Identificación de algoritmos de cómputo Intensivo para big data y su implementación en clouds. *Workshop de Investigadores en Ciencias de la Computación (WICC 2016, Entre Ríos, Argentina)*. S.l.: s.n.,

MURTAGH, F. y LEGENDRE, P., 2014. Ward's hierarchical agglomerative clustering method: which algorithms implement Ward's criterion? *Journal of classification*, vol. 31, no. 3, pp. 274-295.

OLAYA, V., 2014. Sistemas de información geográfica. *Recuperado de: http://www. icog. es/TyT/files/Libro_SIG. pdf*,

PARA LA INNOVACIÓN AGRARIA, F., 2017. Resultados y lecciones en sistema de información geográfico para mejoramiento de la gestión apícola: proyecto de innovación en Región del Maule. ,

PARIMALA, M., LOPEZ, D. y SENTHILKUMAR, N.C., 2011. A survey on density based clustering algorithms for mining large spatial databases. *International Journal of Advanced Science and Technology*, vol. 31, no. 1, pp. 59-66.

PEÑA-CARO, W.J., DÍAZ, D., GAONA-GARCIA, P.A., MONTENEGRO-MARIN, C.E. y CASTRO, M., 2016. Aproximación al Modelo de Estimación Para El Uso De Agua Del Rio Bogotá, Basado En El Análisis De Vertimientos En Aguas Superficiales-Approximation to the Estimation Model for the Use of Water from the Rio Bogotá, Based on the Analysis of Vertimientos in Surface Waters. *Revista científica*, vol. 3, no. 26, pp. 92-108.

PÉREZ, G., 2016. Peligros del uso de los big data en la investigación en salud pública y en epidemiología. *Gaceta Sanitaria*, vol. 30, no. 1, pp. 66-68.

PRESSMAN, R.S., 2010. *Ingeniería del Software UIn enfoque práctico, Septima edición ed*. S.l.: McGraw-Hill.

PyCharm: Python IDE for Professional Developers by JetBrains. *JetBrains* [en línea], 2018. [Consulta: 18 junio 2018]. Disponible en: https://www.jetbrains.com/pycharm/.

RAGHAVAN, V.V. y WONG, S.M., 1986. A critical analysis of vector space model for information retrieval. *Journal of the American Society for information Science*, vol. 37, no. 5, pp. 279.

RODRIGUEZ, E., 2010. Patrones gof. [en línea], [Consulta: 19 junio 2018]. Disponible en: http://www.academia.edu/5903473/Patrones_gof.

ROJAS, F.M. y GOMEZ, C., 2015. Funcionalidades de la minería de datos. *Ingeniería y Región*, vol. 12, no. 2, pp. 31-40.

ROKACH, L. y MAIMON, O.Z., 2008. *Data mining with decision trees: theory and applications. Volume 69 of Series in machine perception and artificial intelligence*. S.l.: World Scientific Press.

RUSSO, C.C., CHARME, J., PIERGALLINI, M.R., GUASCH, M.M., TORRIGGINO, A. y SMAIL, A., 2016. Aplicación de minería de datos espacial en el área de salud en la zona de influencia de la UNNOBA. *XVIII Workshop de Investigadores en Ciencias de la Computación (WICC 2016, Entre Ríos, Argentina)*. S.l.: s.n., ISBN 950-698-377-1.

SEDANO, Á.F., COLLAZOS, A.Z., PALACIO, M.C. y GONZÁLEZ, C.A.M., 2015. Análisis del City Branding y la Imagen de Marca a través de los Medios de Comunicación Online y el Social Media: Caso Medellín (Colombia). *Revista ESPACIOS| Vol. 36 (Nº 18) Año 2015*,

SENOUSSAOUI, M., KENNY, P., STAFYLAKIS, T. y DUMOUCHEL, P., 2014. A study of the cosine distance-based mean shift for telephone speech diarization. *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, vol. 22, no. 1, pp. 217-227.

SERRANO, R.M., 2000. La Utilidad de la Econometría Espacial en el Ámbito de la Ciencia Regional* por Esther Vayá Valcarce. *DOCUMENTO DE TRABAJO*, vol. 2000, pp. 13.

SHEKHAR, S., VATSAVAI, R.R. y CELIK, M., 2009. Spatial and Spatio-temporal Data Mining: Recent Advances, Next Generation of Data Mining. *New York: CRC Press*, vol. 549, pp. 578.

SOMMERVILLE, I., 2010. *Software engineering*. S.l.: New York: Addison-Wesley.

STAR, J. y ESTES, J., 1990. Geographic Information Systems: An Introduction Prentice Hall. *Englewood Cliffs, New Jersey*,

TAMAYO, S. y MARÍN, D.P., 2017. Propuesta de Evaluación basada en Big Data para facilitar la integración de Agentes Conversacionales Pedagógicos en las aulas. *IE Comunicaciones: Revista Iberoamericana de Informática Educativa*, no. 26, pp. 13-23.

TERÁN GUERRERO, L.E., 2017. *Sistema web para el control de la alimentación, sanidad y comercialización de cuyes para la Fundación a Favor de la Vida.* S.l.: Quito: UCE.

TERÁN, H.E.E., ALCIVAR, M. y PURIS, A., 2016. Aplicaciones de minería de datos en marketing. *Revista Publicando*, vol. 3, no. 8, pp. 503-512.

TIAN, F., GAO, B., CUI, Q., CHEN, E. y LIU, T.-Y., 2014. Learning deep representations for graph clustering. *AAAI*. S.l.: s.n., pp. 1293-1299.

TIRIA, C. y MARCELA, P., 2014. *Zonificación climatológica según el modelo Caldas–Lang de la cuenca Rio rio negro mediante el uso del Sistema de Información Geográfica SIG*. S.l.: Universidad Militar Nueva Granada.

TOBLER, W.R., 1970. A computer movie simulating urban growth in the Detroit region. *Economic geography*, vol. 46, no. sup1, pp. 234-240.

TOMLIN, C.D., 1990. *Geographic information systems and cartographic modeling*. S.l.: Prentice Hall. ISBN 0-13-350927-3.

TRASOBARES, A.H., 2003. Los sistemas de información: evolución y desarrollo. *Proyecto social: Revista de relaciones laborales*, no. 10, pp. 149-165.

VALCÁRCEL ASENCIOS, V., 2004. Data Mining y el descubrimiento del conocimiento. *Industrial Data*, vol. 7, no. 2.

VALLEJOS-PEÑA, A., 2017. Propuesta de algoritmo que combina el agrupamiento en subespacios basado en densidad y el agrupamiento basado en restricciones para la detección de grupos que incluyan atributos de interés en conjuntos de datos de alta dimensionalidad. ,

VANEGAS, C.E.D., 2014. Asociación de datos espacio-temporales en bases de datos Oracle. *Ingenierías USBmed*, vol. 5, no. 2, pp. 100-108.

W3C, 2018. World Wide Web Consortium (W3C). *W3C* [en línea]. [Consulta: 25 marzo 2018]. Disponible en: https://www.w3.org/.

WANG, S., 2014. *Spatial and Spatio-Temporal Clustering*. S.l.: s.n.

XU, J. y CROFT, W.B., 2000. Improving the effectiveness of information retrieval with local context analysis. *ACM Transactions on Information Systems (TOIS)*, vol. 18, no. 1, pp. 79-112.

YANG, Y., MA, Z., YANG, Y., NIE, F. y SHEN, H.T., 2015. Multitask spectral clustering by exploring intertask correlation. *IEEE transactions on cybernetics*, vol. 45, no. 5, pp. 1083-1094.

YOLANDA, B.L., 2013. Metodología Ágil de Desarrollo de Software–XP. *línea]. Disponible en: http://www. runayupay. org/publicaciones/2244_555_COD_18_290814203015. pdf.[Accedido: 03-jun-2016]*,

YRIGOYEN, C.C., 2003. *Econometría espacial aplicada a la predicción-extrapolación de datos microterritoriales*. S.l.: Dirección General de Economía y Planificación. ISBN 84-451-2442-0.

Anexos

Tabla 15: Obtener características cartográficas a través de QGIS. Fuente: Elaboración propia.

Historia de usuario “Obtener características cartográficas a través de QGIS”	
Numero: 2	Nombre Historia de Usuario: Obtener características cartográficas a través de QGIS
Usuario: Experto	
Prioridad en Negocio: Media	Riesgo en Desarrollo: Media
Puntos Estimados: 7	Iteración Asignada: 4
Programador responsable: Alejandro Alea González y Alberto Pérez Ojeda	
Descripción: La aplicación debe ser capaz de capturar la información necesaria del SIG QGIS para construir una estratificación, así como la capa correspondiente de los territorios a evaluar; y si es preciso, las capas necesarias para el cálculo de los valores de los indicadores cartográficos: “cantidad de ríos con contaminación” y “cantidad de puntos contaminantes”.	
Observaciones:	

Tabla 16: Construir agrupamientos. Fuente: Elaboración propia.

Historia de usuario “Construir agrupamientos.	
Numero: 3.1	Nombre Historia de usuario: Construir agrupamientos.
Usuario: Experto	
Prioridad en Negocio: Media	Riesgo en Desarrollo: Media
Puntos Estimados: 7	Iteración Asignada: 4
Programador responsable: Alejandro Alea González y Alberto Pérez Ojeda	
Descripción: La aplicación debe ser capaz de construir agrupamientos, a partir de: • los indicadores estadísticos importados por el usuario desde una hoja de cálculo. • las características cartográficas seleccionadas por el usuario a través de QGIS.	

Observaciones:

Tabla 17: Visualizar agrupamientos construidos en mapa temático. Fuente: Elaboración propia.

Historia de usuario “Visualizar agrupamientos construidos en mapa temático”	
Numero: 3.2	Nombre :Historia de usuario: Visualizar agrupamientos construidos en mapa temático.
Usuario: Experto	
Prioridad en Negocio: Media	Riesgo en Desarrollo: Media
Puntos Estimados: 7	Iteración Asignada: 4
Programador responsable: Alejandro Alea González y Alberto Pérez Ojeda	
Descripción: La aplicación debe ser capaz de representar gráficamente en un mapa temático cada estratificación generada.	
Observaciones:	