UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

FACULTAD 3

Red neuronal artificial para el pronóstico de éxito en la mejora de procesos de software

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

Autores:
Roger Godofredo Rivero Morales
Marcell Alejandro Verdera Marcano

Tutores:
MSc. Ana Marys Garcia Rodríguez
Ing. Osvaldo Santos Acosta

La Habana, junio de 2016
"Año 58 de la Revolución"

DECLARACIÓN 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 _____.

_____	_____
Roger Godofredo Rivero Morales	Marcell Alejandro Verdera Marcano
Autor	Autor
_____	_____
MSc. Ana Marys García Rodríguez	Ing. Osvaldo Santos Acosta
Tutora	Tutor

DEDICATORIA

De Marcell

A Norelsa y Jose Antonio mis padres y mis mejores amigos.

A mis abuelos Carmen, Roberto y Fidelina.

A mi hermanita Laura.

A mi familia en general con todo mi cariño.

A mis amigos y mis compañeros.

A mis tutores y mis profesores.

A dios.

De Roger

A Delia y Lazaro, mis padres y mejores maestros.

A Rosmery, mi hermana y mejor amiga.

A mis abuelos, los mejores abuelos del mundo.

A Deivis, mi tía y segunda madre.

A toda mi familia y amigos.

A la Revolución Cubana.

AGRADECIMIENTOS

De Marcell

Gracias a dios por darme la fuerza para seguir adelante en cada momento de mi vida.

Gracias a mi familia en especial a mis padres, a mis abuelos y mis primos por hacer de mí el hombre que soy hoy, por su paciencia, su amor incondicional y su apoyo eterno en todos mis sueños y realizaciones.

Gracias a mis amigos por su apoyo y lealtad ante todo.

Gracias a mis compañeros en especial a los de mi aula por compartir conmigo buenos y malos momentos a lo largo de estos cinco años de carrera.

Gracias a mis tutores Osvaldo y Ana Marys por su apoyo y trabajo, por ser más que mis tutores mis amigos.

Gracias a mis profesores por los conocimientos y experiencias que me brindaron.

Gracias a todos los profesores que nos ayudaron en la realización de esta tesis en especial a la profesora Mailen y el profesor Julio.

Gracias a mi compañero de tesis por compartir conmigo esta experiencia inolvidable.

De Roger

Gracias a Delia y Lazaro, por el amor, la educación y consejos que me han dado.

Gracias a Rosmery por tu apoyo siempre que me ha hecho falta.

Gracias a mis abuelos, sin ustedes no estaría hoy aquí.

Gracias a Deivis por estar siempre orgullosa de mi.

Gracias a toda mi familia por su apoyo y cariño.

Gracias a mis tutores Ana Marys y Osvaldo por sus conocimientos y guiarme durante toda la tesis.

Gracias a los profesores que contribuyeron al desarrollo de la tesis, especialmente a la profe Mailen y el profesor Julio.

Gracias a los profesores que contribuyeron a mi formación durante estos cinco años.

Gracias a mis amigos y compañeros de aula, por su comprensión, amistad y consejos en especial a Erlis y Yamila.

Gracias a mi compañero de tesis, por su amistad y paciencia especialmente durante el desarrollo de la tesis.

Gracias a la FEU y todos los que implica por formarme integralmente.

A todos, muchas gracias.

RESUMEN

Las organizaciones de software han comenzado a centrarse en una competencia por posicionarse en un mercado mundial que exige cada vez productos de mejor calidad. Son muchas las investigaciones que plantean la importancia de la mejora de procesos de software y las limitantes que existen a su alrededor, como el alto costo monetario al insertarse en la misma. Lo que hace necesario realizar una evaluación a las organizaciones con vista a la mejora de procesos de software.

En la presente investigación, se realizó un estudio de metodologías para la minería de datos, de las cuales constituyen aplicables al desarrollo de la solución las fases de modelación y evaluación, y como técnica de clasificación una RNA evolutiva. Para ello se realizó un estudio con el objetivo de identificar los elementos para la construcción de una red neuronal evolutiva. La aplicación de esta técnica permitió la adaptación de una red a diferentes situaciones para analizar un conjunto de indicadores que inciden en el diseño y ejecución de las organizaciones, permitiendo conocer el estado en que se encuentran para enfrentar una mejora de procesos de software a partir de las experiencias de organizaciones similares. Como resultado de la validación se obtuvo que la red al clasificar posee una precisión superior al 90% demostrando la factibilidad de su uso para dotar a las organizaciones de información útil para saber si es el momento más idóneo de insertarse en la mejora de procesos de software.

Palabras claves: mejora de procesos de software, organizaciones, pronóstico, RNA evolutiva.

ABSTRACT

Software organizations started to focus on competing for a position in a global market that demands more and better quality products. There are many researches that state the importance of software process improvement. Also, there are some factors against it, like the high costs, which makes necessary to assess organizations to software process improvement.

In this research, a study was carried out to select the methodology for data mining. Only modeling and evaluation phases of the methodology were used and as classification technique an evolving artificial neural network was selected. This requires a study to identify the elements for the construction of an evolving neural network was performed. The application of this technique allowed the adaptation of a network to different situations to analyze a set of indicators that affect the design and management of organizations, allowing to know the state they are in to face a software process improvement based on the experiences of similar organizations. As a result of the validation was obtained by the network to classify has an accuracy above 90%, demonstrating the feasibility of their use to provide organizations with useful information to know if it's the right time to be inserted in software process improvement.

Keywords: *software process improvement, organizations, forecast, evolutionary artificial neural network.*

TABLA DE CONTENIDOS

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

INTRODUCCIÓN

El desarrollo de las Tecnologías de la Información y las Comunicaciones se ha expandido a cada uno de los sectores de la sociedad, agilizando sus procesos y viabilizando el trabajo de las personas. Los beneficios que proporciona la industria de software y la creciente demanda de informatización, han llevado a las organizaciones de software a centrarse en una competencia por posicionarse en un mercado mundial que exige cada vez productos de mejor calidad. Sin embargo, los esfuerzos aplicados para suplir la progresiva demanda de proyectos de software, no se encuentra a tono con la ejecución eficiente y eficaz de los procesos, quedando reflejado en estudios realizados por (STANDISH-GROUP 2015), que solo el 29 % de los proyectos resultaron exitosos, el 52 % presentaron problemas de retraso y el 19 % resultó fallido.

La importancia de la Mejora de Procesos de Software (MPS) con vista a la introducción de buenas prácticas en la ejecución de los procesos para elevar la madurez y capacidad de los mismos, queda reafirmada en diversas investigaciones (ASHRAFI 2003; BASILI *et al.* 2002; GARCIA 2016; PINO *et al.* 2008; TRUJILLO CASAÑOLA *et al.* 2013; ZAHRAN 1998). Muchas organizaciones, instituciones y comunidades científicas han optado por la aplicación de modelos, normas, guías y estándares en función de la MPS como: Modelo de Capacidad de Madurez Integrada (CMMI por sus siglas en inglés) (CMMI 2016), las normas ISO 9000 (ISO 2005a) e ISO 25000 (ISO 2005b), desarrolladas por la Organización Internacional para la Estandarización (ISO por sus siglas en inglés) (ISO 2016), la Guía de Fundamentos para la Dirección de Proyectos (PMBOK) (PMI 2013) propuesto por el Instituto de Gestión de Proyectos (PMI por sus siglas en inglés) (PMI 2016), entre otros. Sin embargo estos modelos, guías y estándares no establecen el cómo ejecutar la MPS, solo especifican qué hacer para establecerla en la organización.

Los aspectos sociales y las necesidades reales de las organizaciones alineadas a sus objetivos estratégicos, no son contemplados en los programas de mejora definidos. Esto influye de tal forma que al ejecutar los procesos definidos como parte del programa, existan inconsistencias y dificultades de entendimiento por parte del equipo de desarrollo que los ejecuta. Los resultados de la aplicación de iniciativas MPS en las organizaciones, muestran un gran número de fracasos, observándose un ascenso de hasta un 70%, reflejándose en estudios realizados (FORRADELLAS *et al.* 2005; TRUJILLO CASAÑOLA *et al.* 2013). Esto se debe en gran medida a que las iniciativas de mejora no contemplan el estado real de las organizaciones y las peculiaridades de cada entidad que representan un punto de partida diferente para el programa condicionando sus resultados (TRUJILLO *et al.* 2014).

En investigaciones (GARCIA 2016; NIAZI *et al.* 2010; NIAZI *et al.* 2006; TRUJILLO CASAÑOLA *et al.* 2013), se plantea que el uso de los factores críticos de éxito (FCE) en función de los contextos organizacionales favorece al éxito de las iniciativas en la MPS (DOUNOS and BOHORIS 2010; MONTONI and ROCHA 2010). Para valorar el estado de las organizaciones frente a la MPS, persisten insuficiencias asociadas a la reutilización del conocimiento adquirido para emitir evaluaciones que se aproximen cada vez más a la realidad experimentada. De acuerdo a las características de las organizaciones existe una variabilidad en la medidas que caracterizan a los FCE debido a que estas pueden cambiar en dependencia del entorno. Además, los pesos asignados a los FCE para el análisis de su influencia sobre la MPS no son reajustables y constituye una realidad que su relevancia varía de acuerdo al contexto de las organizaciones y del programa a aplicar. En investigaciones anteriores no se tuvieron en cuenta ninguno de estos dos aspectos.

A partir de lo anterior se define como **problema a resolver:** ¿cómo desarrollar una herramienta para la clasificación de las organizaciones con vista a la Mejora de Procesos de Software, a partir de las experiencias adquiridas en torno al comportamiento de los Factores Críticos de Éxito?

Se define como **objeto de estudio:** metodología de desarrollo de software en la toma de decisiones centrándose en el **campo de acción:** metodología de desarrollo de software en la toma de decisiones para la solución de problemas de clasificación.

Para darle solución al problema de define como **objetivo general:** desarrollar una herramienta para la clasificación de las organizaciones con vista a la Mejora de Procesos de Software, a partir de las experiencias adquiridas en torno al comportamiento de los Factores Críticos de Éxito.

Para dar complimiento al objetivo general se definen los siguientes **objetivos específicos:**

1. Definir el marco teórico de la investigación mediante el estudio y el análisis de los principales referentes teóricos para el desarrollo de la solución.
2. Diseñar una RNA para el pronóstico de éxito en la Mejora de Procesos de Software.
3. Implementar la solución propuesta.
4. Valorar la solución propuesta mediante la realización de un Cuasi-experimento y Validación cruzada.

Se definen como **tareas de la investigación:**

- Análisis de modelos de MPS.
- Realizar un estudio de metodologías de minería de datos para el desarrollo de la solución.

- Caracterización de las tecnologías y herramientas a utilizar para el desarrollo de la solución.
- Diseño del modelo de datos.
- Diseño de una RNA para el pronóstico de éxito en la Mejora de Procesos de Software.
- Realizar un estudio de modelos de RNA para dar solución a problemas de clasificación.
- Implementación de la solución propuesta.
- Validación de la implementación realizada.

Los métodos de trabajo científico utilizados son los siguientes:

Métodos teóricos:

- El método **analítico-sintético** para el análisis crítico de los trabajos anteriores con el objetivo de establecer un punto de referencia para la propuesta resultante.
- El método **inducción-deducción** para la identificación de la problemática, así como sus variantes de solución.

CAPÍTULO 1. FUNDAMENTACIÓN TEÓRICA

1.1 Introducción

En el presente capítulo se hace referencia a los principales conceptos relacionados con la toma de decisiones, así como técnicas de inteligencia artificial para la solución de problemas de clasificación. Se realizó un análisis de investigaciones afines, a partir de los cuales se identificaron los elementos que forman parte de la solución propuesta.

1.2 Análisis bibliométrico y documental

En la sección, se realiza un análisis bibliométrico, con el fin de mostrar la novedad de la revisión bibliográfica, basándose en las fechas de las publicaciones consultadas. Las bases de datos utilizadas fueron: IEEE, Google Scholar y Scielo. Las fuentes bibliográficas fueron: artículos de revistas, libros, tesis (específicamente de maestrías y doctorados), artículos y sitios web. El análisis realizado se muestra en la siguiente Tabla 1.

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

Tipo de Fuente Bibliográfica	Cantidad	Cantidad publicada en los últimos cinco años (2011-2016)
Artículos de revista	44	21
Libro	24	9
Tesis	9	9
Sitio web	8	8
Estándar	3	1
Conferencia	7	3
Total:	**95**	**51**

De la tabla anterior se obtiene el resultado mostrado en la Figura 1 donde se muestra que el 54% de la literatura consultada pertenece a los últimos cinco años, por lo que se evidencia la actualidad de la bibliografía consultada.

Figura 1: Bibliografía consultada. Fuente: Elaboración propia

1.3 Conceptos y definiciones asociados al dominio del problema

En el presente acápite se estudian y definen los principales conceptos que serán tratados a lo largo de la investigación.

Mejora de procesos de software

La mejora de procesos significa comprender los procesos existentes y modificarlos para incrementar la calidad del producto y reducir costos. La mejora de procesos se centra en el perfeccionamiento de los procesos para mejorar la calidad de los productos y en particular reducir el número de defectos de los entregables (SOMMERVILLE 2007).

Una vez alcanzado esto, el principal objetivo sería la reducción de los costos y el cronograma de trabajo. No significa simplemente adoptar métodos, herramientas o utilizar algún modelo, a pesar que las organizaciones que desarrollan productos similares tienen mucho en común, existen factores y particularidades que influyen en el proceso de manera diferente, por ello debe verse a la mejora como una actividad específica dentro de una organización, a continuación se muestran los tres estados principales (SOMMERVILLE 2007).

- Medición del proceso: Son medidos atributos del proyecto y el producto. El objetivo es mejorar las mediciones de acuerdo a los objetivos de la organización involucrada en la mejora de procesos.

- Análisis del proceso: Es evaluado el proceso y se identifican sus debilidades y riesgos.
- Cambio del proceso: Los cambios al proceso que deben ser identificados durante el análisis son introducidos.

La mejora de procesos de software es un patrón a seguir, mediante el cual, una empresa trata de modificar sus procesos para ser más eficaces y reducir costos. Mediante su seguimiento, que en la mayor parte de los casos implica reorganizaciones internas, se persigue implícitamente la mejora en la calidad de los productos desarrollados (SOMMERVILLE 2007).

Sobre la mejora de procesos de software (GARCIA 2013) plantea que existen un conjunto de Indicadores que influyen en estos de manera positiva o negativa (ver Tabla 2).

Tabla 2: Indicadores que influyen en la MPS. Fuente: (GARCIA 2013)

1. Influencia del personal				
Relaciones interpersonales	**Formación del personal**	**Experiencia del personal**	**Efectividad del programa de reconocimiento y remuneración**	**Motivación y compromiso del personal**
Colaboración - Competencia	Formación para la mejora de proceso	Experiencias en la producción	Reconocimientos y castigos	Motivación por el trabajo
Relaciones Individuo - Individuo	Capacidad de aprendizaje	Experiencias en roles	Satisfacción con la política de retribuciones	Satisfacción con el trabajo
Relaciones intergrupales	Capacidad de adaptación y autorrenovación		Satisfacción con la política de estimulaciones	Identificación con la organización
2. Influencia de la alta gerencia				
Orientación estratégica	**Administración estratégica**	**Atención al capital humano**	**Apoyo de la alta gerencia**	
Orientación a la mejora continua	Planeación estratégica	Selección de personal e inducción a la organización	Confianza en la dirección	
Orientación a la satisfacción del cliente	Establecimiento y dominio de los objetivos organizacionales	Programas de desarrollo y planes de superación	Competencia de los directivos	
Orientación a procesos	Establecimiento y delimitación de roles organizacionales	Evaluación del desempeño	Supervisión	

Gestión del cambio		Protección e higiene del trabajo	Estilo de dirección
3. Características de la organización			
Comunicación	**Funcionamiento**	**Disponibilidad de recursos**	
Participación	Perspectivas de la organización	Disponibilidad de las personas	
Información	Eficiencia	Disponibilidad de tiempo	
Comunicación	Eficacia	Disponibilidad de infraestructura	
	Estabilidad interna de la organización		
	Trabajo en equipo		

Toma de decisiones

Entre las funciones obligatorias de la gerencia se encuentra la toma de decisiones. En pocas ocasiones se considera el verdadero proceso secuencial y sistemático que implica tomar una decisión con el objetivo de obtener realmente la efectividad necesaria a partir de la decisión tomada. Es el proceso de identificación de un problema u oportunidad y la selección de una alternativa de acción entre varias existentes, es una actividad clave en todo tipo de organización (CHOO 1999; SAATY 2014).

El proceso de toma de decisiones demanda identificar información actualizada sobre qué alternativas se encuentran disponibles en el presente o cuáles se deben considerar, cuáles son las consecuencias de actuar según cada una de las diversas opciones y es indispensable la información sobre como pasar del presente al futuro: cuáles son los valores y las preferencias que se deben utilizar para seleccionar, entre las alternativas que, según los criterios establecidos, conducen del mejor modo a los resultados deseados (CHOO 1999; SAATY 2014). Dentro de la toma de decisiones uno de los tipos de problemas a resolver son los de clasificación.

Problemas de clasificación

El objetivo de un problema de clasificación es predecir la clase de pertenencia de nuevos patrones, a partir del modelo obtenido por el clasificador con una serie de patrones que son proporcionados como entradas. En (MITCHELL 1997), se define como aquellos en los que la tarea es clasificar ejemplos dentro de un conjunto discreto de posibles categorías (BALLESTEROS 2013; BENÍTEZ *et al.* 2014; PERDOMO 2015). Para dar solución a este tipo de problemas es recomendable la aplicación de técnicas de minería de datos.

Minería de datos

La minería de datos, es el proceso de detectar la información dentro de grandes conjuntos de datos, de manera automática o semiautomática, con el objetivo de deducir los patrones y tendencias que expliquen el comportamiento de los datos en un determinado contexto. Estos patrones no se pueden detectar mediante la exploración tradicional de los datos porque las relaciones son demasiado complejas o porque hay demasiado datos. Con este fin, se hace uso de prácticas estadísticas y, en algunos casos, de algoritmos de búsqueda próximos a la inteligencia artificial (BRAHA 2013; CIOS *et al.* 2012).

1.4 Estado del arte

En el presente epígrafe se realiza un estudio sobre el estado del arte del a temática en cuestión, abordando temas fundamentales para la investigación.

1.4.1 Minería de datos

Desde los años sesenta los estadísticos manejaban términos como data fishing[1] o data archaeology[2], con la idea de encontrar correlaciones sin una hipótesis previa en bases de datos. A principios de los años ochenta, Rakesh Agrawal, Gio Wie-derhold, Robert Blum y Gregory Piatetsky-Shapiro, entre otros comenzaron a consolidar los términos de minería de datos. A finales de los ochenta solo existían algunas pocas empresas que se dedicaban a brindar servicios referentes a este tema pero para el año 2002 el número aumentó a más de cien empresas brindando más de 300 servicios (CIOS *et al.* 2012).

Las técnicas de minería de datos son el resultado de un largo proceso de investigación y desarrollo de productos. Esta evolución comenzó cuando los datos fueron almacenados por primera vez en computadoras, y continuó con mejoras en el acceso a los datos, y más recientemente con tecnologías generadas para permitir a los usuarios navegar a través de los datos en tiempo real. La misma toma

[1] Conocido como "pesca de datos" es una práctica de minería de datos en la que grandes volúmenes de datos son analizados buscando las posibles relaciones entre los datos. A veces con fines no éticos, data fishing a menudo elude las técnicas de minería de datos tradicionales y puede conducir a conclusiones prematuras, es a veces descrito como "la búsqueda de más información de lo que realmente contiene un conjunto de datos."

[2] Se refiere a los métodos para recuperar información almacenada en formatos obsoletos. También puede referirse a la recuperación de información de formatos electrónicos dañados después de desastres naturales o provocados por el hombre.

este proceso de evolución más allá del acceso y navegación retrospectiva de los datos, hacia la entrega de información prospectiva y proactiva. Los algoritmos de minería de datos utilizan técnicas que han existido desde hace décadas, pero que sólo han sido implementadas recientemente como herramientas confiables y entendibles (BRAHA 2013; CIOS *et al.* 2012; GUPTA 2014).

Existen dos tipos de análisis dentro de la minería de datos:

- **Análisis predictivo**

Se caracteriza porque requiere un conjunto de entrenamiento, el cual está formado por un histórico de datos. Algunas de las aplicaciones comúnmente desarrolladas con análisis predictivo son: predecir riesgos, predecir activación de nuevos clientes, predicción de ventas, entre otras. Para el análisis predictivo se aplican las técnicas supervisadas, algunas de ellas son: Redes Neuronales, Árboles de Decisión, Máquinas de Soporte Vectorial, Métodos de Regresión, Método Bayesiano y Métodos basados en Ejemplos. Dentro de dicho análisis de encuentran las tareas de clasificación y regresión (OVIEDO CARRASCAL *et al.* 2015).

- **Análisis descriptivo**

En este tipo de análisis se pueden desarrollar tareas de agrupación y de asociación. El conjunto de datos requerido está conformado por los atributos que se desean analizar para encontrar similitudes o asociaciones entre los datos. Algunas de las aplicaciones más comunes del análisis descriptivo son: análisis del perfil de personas, detección de anomalías, detección de reglas que condicionen la venta de productos, entre otras. Para el análisis descriptivo se aplican las técnicas no supervisadas, algunas de ellas son: Método Particional, Método Jerárquico, Método Probabilístico, Redes Neuronales y Reglas de Asociación. Dentro de este análisis de encuentran las tareas de agrupamiento, reglas de asociación y análisis correlacional (OVIEDO CARRASCAL *et al.* 2015).

Para el desarrollo de proyectos de minería de datos existe una tendencia al uso de metodologías que proporcionan una serie de pasos a seguir con el fin de realizar una implementación adecuada (BRAHA 2013).

1.4.1.1 Metodologías de Minería de Datos

Según una encuesta publicada en (*KDnuggets* 2016), (ver Figura 2) las metodologías más usadas son CRIST-DM, SEMMA y KDD. En esta encuesta se tuvieron en cuenta 200 votos divididos de la siguiente

manera: 45.5% Estados Unidos y Canadá, 28.5% Europa, 14% Asia, 9.5% América Latina y 2.5% otros. La misma fue realizada en dos momentos diferentes, una primera vez en el año 2007 y posteriormente en el 2014.

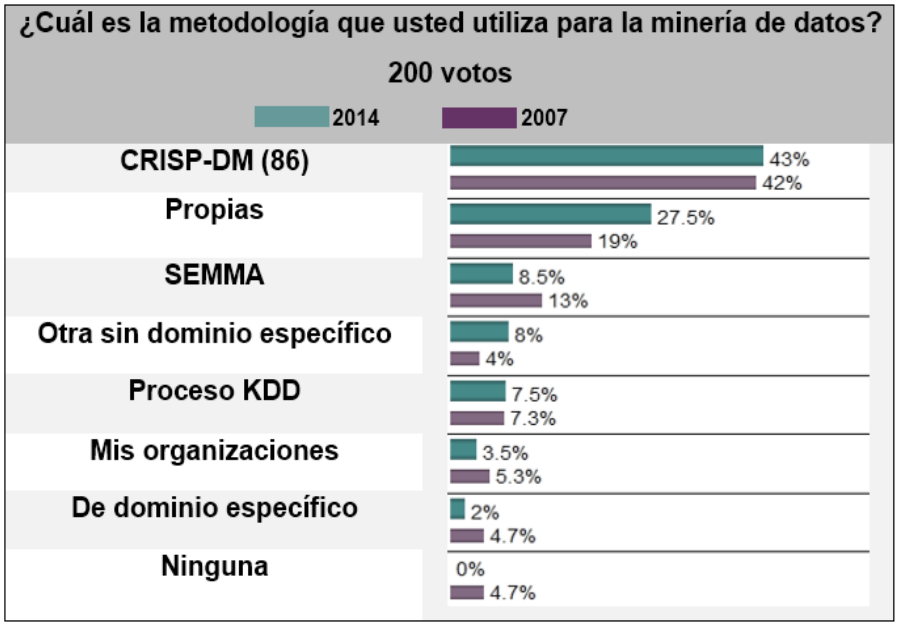

Figura 2: Resultados de la encuesta realizada por KDnuggets. Fuente:(KDnuggets 2016)

Proceso de estandarización industrial transversal para la minería de datos (CRISP-DM por sus siglas en inglés)

Fue concebida desde un enfoque práctico de acuerdo la experiencia de sus creadores: un consorcio de empresas europeas, incluyendo SPSS de IBM. Actualmente CRISP-DM es la guía de referencia más utilizada en el desarrollo de proyectos de minería de datos (CORRALES *et al.* 2014; MOINE 2013; TORRES 2013). Está constituida por seis fases: entendimiento del negocio, entendimiento de los datos, preparación de los datos, modelado, evaluación y despliegue (BARCLAY *et al.* 2015; DAVID and GOMEZ 2014; NIAKSU 2015; OVIEDO CARRASCAL *et al.* 2015).

1. **Entendimiento del negocio**

Se debe explorar las expectativas de la organización con respecto a la minería de datos e intentar implicar a la mayor cantidad de personas que sea posible en este proceso y documentar los resultados. En el paso final de esta fase se trata de producir un plan de proyecto utilizando la información que se

contiene en esta documentación. Conocer las razones comerciales para que los esfuerzos en minería de datos aseguren que todos los usuarios están de acuerdo antes de asignar recursos (IBM 2012).

2. Entendimiento de los datos

La fase de comprensión de datos de CRISP-DM implica estudiar más de cerca los datos disponibles de minería. Este paso es esencial para evitar problemas inesperados durante la siguiente fase (preparación de datos) que suele ser la fase más larga de un proyecto. La comprensión de datos implica acceder a los datos y explorarlos con la ayuda de tablas y gráficos. De esta forma se podrá determinar su calidad y describir los resultados de estos pasos en la documentación del proyecto (IBM 2012).

3. Preparación de los datos

La preparación de datos es uno de los aspectos más importantes y con frecuencia que más tiempo exigen en la minería de datos. De hecho, se estima que la preparación de datos suele llevar el 50-70% del tiempo y esfuerzo de un proyecto. Dedicar los esfuerzos adecuados a las primeras fases de comprensión comercial y comprensión de datos puede reducir al mínimo los gastos indirectos relacionados, pero aún se deberá dedicar una buena cantidad de esfuerzo para preparar y empaquetar los datos para la minería (IBM 2012).

Dependiendo de la organización y los objetivos, la preparación de datos suele implicar las tareas siguientes:

- Fusión de conjuntos y/o registros de datos.
- Selección de una muestra de un subconjunto de datos.
- Agregación de registros.
- Derivación de nuevos atributos.
- Clasificación de los datos para el modelado.
- Eliminación o sustitución de valores en blanco o ausentes.
- División en conjuntos de datos de prueba y entrenamiento.

4. Modelado

Los datos preparados se incorporan a las herramientas analíticas y se podrán ver los resultados del problema planteado en comprensión del negocio. Se debe seleccionar la técnica de modelado correcta. El modelado se suele ejecutar en múltiples iteraciones. Normalmente, los analistas de datos ejecutan

varios modelos utilizando los parámetros por defecto y ajustan los parámetros o vuelven la fase de preparación de datos para las manipulaciones necesarias por su modelo (IBM 2012).

5. **Evaluación**

Se habrá determinado, en la fase de modelado, que los modelos son técnicamente correctos y efectivos en función de los criterios de rendimiento de minería de datos qué se han definido previamente. Sin embargo, antes de continuar, se deben evaluar los resultados utilizando los criterios de rendimiento comercial establecidos en el inicio del proyecto. Es la clave para asegurar que la organización pueda utilizar los resultados que se han obtenido (IBM 2012).

6. **Despliegue**

La distribución es el proceso que consiste en utilizar sus nuevos conocimientos para implementar las mejoras en su organización. Puede significar una integración formal como la aplicación del modelo. La distribución puede significar que utilice los conocimientos adquiridos en minería de datos para aplicar modificaciones en su organización. Es posible que estos resultados no se integren formalmente en los sistemas de información, pero serán de gran utilidad para la planificación y toma de decisiones de marketing (IBM 2012).

Muestreo, Exploración, Modificación, Modelado y Evaluación (SEMMA por sus siglas en inglés)

Es la propuesta de SAS Analytics Solutions para desarrollar proyectos de minería de datos. La metodología establece cinco fases: muestreo, exploración, modificación, modelado y evaluación. Se caracteriza por incluir una fase de muestreo estadístico que no se considera en otras metodologías (DAVID and GOMEZ 2014; OVIEDO CARRASCAL *et al.* 2015; YARUSHKINA *et al.* 2015).

1. **Muestreo**

En esta etapa se toma una muestra del conjunto de datos disponible, que debe ser lo suficientemente grande para contener la información relevante, y lo suficientemente pequeña como para correr el proceso rápidamente. La etapa de muestreo es opcional, aconsejable cuando el tamaño del conjunto de datos es demasiado extenso (MOINE 2013).

2. **Exploración**

En esta fase, se hace un recorrido a través de los datos extraídos en la muestra para detectar, identificar y eliminar datos anómalos, ayudando a refinar los procesos de descubrimiento de información en fases siguientes del proceso. En este punto del proceso, la exploración se puede realizar a través de medios visuales, aunque muchas veces no es suficiente este método, es por eso, que además de la visualización se pueden manejar diferentes técnicas estadísticas como análisis de factores, análisis de correspondencias, entre otros (MOINE 2013).

3. **Modificación**

Consiste en explorar los datos en búsqueda de relaciones y tendencias desconocidas. Es una etapa especial para familiarizarse con los datos, y formular nuevas hipótesis a partir de su análisis (MOINE 2013).

4. **Modelado**

En esta fase, las herramientas de software se encargan de realizar una búsqueda completa de combinaciones de datos que juntos predecirán de una manera confiable los resultados buscados. Es en esta parte donde las técnicas y métodos de minería de datos entran a jugar un papel importante para la solución de los problemas que fueron identificados al iniciar el proyecto de minería de datos (MOINE 2013).

5. **Valoración**

Después de que la fase de modelación presente los resultados obtenidos de la aplicación de los métodos de minería de datos al conjunto de datos. Se debe realizar un análisis de los resultados para ver si estos fueron exitosos de acuerdo a las entradas que se tuvieron para analizar el problema. Una buena práctica para identificar si los resultados con el modelo creado son los esperados, es aplicar este modelo a una porción de datos diferente. Si el modelo funciona correctamente para esta muestra y para la muestra utilizada para el proceso de creación del modelo, se tiene una buena probabilidad de tener un modelo valido (MOINE 2013).

Descubrimiento de conocimientos en base de datos (KDD por sus siglas en inglés)

Se conoce como el descubrimiento de conocimiento en bases de datos y un proceso no trivial donde se identifican patrones válidos, novedosos, potencialmente útiles y en última instancia entendibles en

los datos. Algunos autores consideran a la minería de datos como una etapa en el de KDD. Sin embargo, según las encuestas de (*KDnuggets* 2016), se está utilizando KDD como metodología para hacer minería de datos. La metodología establece cinco fases: selección de datos, preprocesamiento, transformación, minería de datos e interpretación y evaluación (OVIEDO CARRASCAL *et al.* 2015; WIBOWO *et al.* 2014; YARUSHKINA *et al.* 2015).

1. **Selección de datos**

En esta primera etapa, se debería recolectar todo el conocimiento disponible y relevante sobre el dominio de aplicación e identificar los objetivos del proceso KDD desde el punto de vista del usuario (MOINE 2013).

2. **Preprocesamiento**

Esta etapa consiste en la elección de las fuentes de datos que se utilizarán, las cuales serán integradas y se seleccionarán las observaciones/atributos. Aunque no es estrictamente necesario, en este paso podría requerirse la construcción de un almacén de datos. Se deberían llevar a cabo tareas como limpieza de datos y tratamiento de datos faltantes (MOINE 2013).

3. **Transformación**

Se detectan características útiles de representación de los datos dependiendo del objetivo de la tarea de minería. Se incluye la utilización de métodos de transformación de los datos para reducir la cantidad de variables en discusión o para encontrar representaciones invariantes de los datos. En esta etapa es frecuente la transformación de los datos, calculando nuevos atributos o bien redefiniendo los existentes con otro formato (MOINE 2013).

4. **Minería de datos**

En esta fase, se deberá determinar la tarea de minería con la que se abordará el estudio (como agrupamiento, regresión, clasificación, o asociación) teniendo en cuenta los objetivos definidos. De acuerdo a la tarea de minería establecida en el punto anterior, en esta etapa se define el algoritmo (o algoritmos) que se aplicarán para la búsqueda de patrones sobre los datos. Incluye la determinación de qué modelos y parámetros son los más adecuados según la naturaleza del problema y de los datos disponibles. Se aplican los algoritmos y técnicas seleccionadas al conjunto de datos (MOINE 2013).

5. Interpretación y evaluación

En esta fase se realiza la interpretación de los patrones encontrados, visualizando y traduciendo los mismos en términos comprensibles por el usuario. Finalmente implementa el conocimiento descubierto, apoyando con el mismo la toma de decisiones o bien reportándolo a las partes interesadas. Incluye la verificación y resolución de potenciales conflictos con conocimiento descubierto previamente (MOINE 2013).

Como se ha evidenciado cada una de estas metodologías tienen cinco etapas en común (ver Tabla 3) que son: análisis y comprensión del negocio, selección y preparación de los datos, modelado, evaluación y por último implementación.

Tabla 3: Correspondencia entre las fases de cada metodología. Fuente: Elaboración propia

<table>
<tr><th>Fases</th><th>KDD</th><th>CRISP-DM</th><th>SEMMA</th></tr>
<tr><td>Análisis y comprensión del negocio</td><td>Selección de datos</td><td>Entendimiento del negocio</td><td>-</td></tr>
<tr><td rowspan="3">Selección y preparación de los datos</td><td rowspan="2">Preprocesamiento</td><td rowspan="2">Entendimiento de los datos</td><td>Muestreo</td></tr>
<tr><td>Exploración</td></tr>
<tr><td>Transformación</td><td>Preparación de los datos</td><td>Modificación</td></tr>
<tr><td>Modelado</td><td>Minería de datos</td><td>Modelado</td><td>Modelado</td></tr>
<tr><td>Evaluación</td><td rowspan="2">Interpretación y evaluación</td><td>Evaluación</td><td>Valoración</td></tr>
<tr><td>Implementación</td><td>Despliegue</td><td>-</td></tr>
</table>

A partir del análisis previo sobre las fases de las metodologías se identificó que en el contexto de la investigación , las fases de análisis y comprensión del negocio y de selección y preparación de los datos no se realiza porque en investigaciones como (GARCIA 2016; TRUJILLO *et al.* 2014) se realizó el análisis del negocio y la preparación de los datos a utilizar en las fases posteriores. En la fase de

modelado dentro de la actividad de selección de la técnica de minería de datos, se definió a priori para resolver el problema de clasificación el uso de una RNA. Dentro de la fase de evaluación solo se evalúa la precisión del modelo y no su interpretación en el dominio del problema. La fase de implementación no está dentro del alcance de la investigación. La presente investigación se centra en las etapas modelado y evaluación.

Modelado: consiste en aplicar las distintas técnicas de minería sobre el conjunto de datos para obtener los modelos buscados, según el tipo de análisis que se desee realizar (descriptivo o predictivo).

Evaluación: consiste en la evaluación de los modelos obtenidos en la fase anterior, determinando la precisión de los mismos y su interpretación en el dominio del problema.

1.4.2 Redes neuronales artificiales

Una red neuronal artificial (RNA) puede definirse como un sistema de procesamiento de información compuesto por un gran número de elementos de procesamiento (neuronas), profusamente conectados entre sí a través de canales de comunicación. Las conexiones establecen una estructura jerárquica y permiten la interacción con objetos del mundo real tratando de emular al sistema nervioso biológico. La computación neuronal permite desarrollar sistemas que resuelvan problemas complejos cuya formalización matemática es sumamente difícil. Lo anterior se logra gracias a los principios de funcionamiento de las redes neuronales: aprendizaje adaptativo, auto-organización, tolerancia a fallos, operación en tiempo real y fácil inserción en la tecnología existente. Una neurona artificial sigue un modelo genérico (ver Figura 3) que está compuesto por los siguientes elementos (ANDRADE TEPÁN 2013; BALLESTEROS 2013; LANZARINI *et al.* 2015; MARTIN DEL BRIO and SANZ MOLINA 2001; SÁNCHEZ *et al.* 2016):

- Conjunto de entradas: es el conjunto de valores que entran a la neurona, ya sea procedente de un sensor o de la salida de otra neurona.
- Pesos sinápticos: definen la intensidad de la interacción existente entre la neurona presináptica *j* y la postsináptica *i.*
- Regla de propagación: permite obtener, a partir de los pesos y las entradas, el valor del potencial postsináptico de la neurona.
- Función de activación (transferencia): proporciona el estado de activación actual a partir del potencial postsináptico.

- Función de salida: proporciona la salida global de la neurona en función de su estado de activación actual.

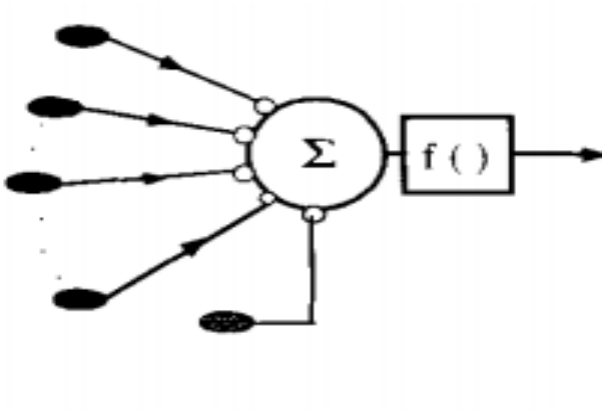

Figura 3: Modelo de una neurona artificial. Fuente: (MARTIN DEL BRIO and SANZ MOLINA 2001)

Algunos modelos también utilizan una entrada cuyo valor es 1, se conoce como umbral, neurona independiente o bias point. Esta entrada actúa exactamente igual a como actúa el peso de la conexión de una neurona cualquiera considerando que la activación siempre es 1 (ANDRADE TEPÁN 2013; FREEMAN and SKAPURA 1991; LANZARINI *et al.* 2015; ROJAS QUINCHO 2013; VILLADA *et al.* 2014).

Se define topología, como la estructura o patrón de conexionado de una red neuronal (ANDRADE TEPÁN 2013; MARTIN DEL BRIO and SANZ MOLINA 2001). En un sistema neuronal artificial las neuronas se conectan por medio de sinapsis, esta estructura de conexiones sinápticas determina el comportamiento de la red. Las conexiones sinápticas son direccionales, es decir, la información solamente puede propagarse en un único sentido (desde la neurona presináptica a la postsináptica). En general, las neuronas se suelen agrupar en unidades estructurales que se denominarán capas**.** Las neuronas de una capa pueden agruparse, a su vez, formando grupos neuronales (*clusters*). Dentro de un grupo, o de una capa si no existe este tipo de agrupación, las neuronas suelen ser del mismo tipo. Finalmente, el conjunto de una o más capas constituye la red neuronal (ANDRADE TEPÁN 2013; LANZARINI *et al.* 2015; MARTIN DEL BRIO and SANZ MOLINA 2001; ROJAS QUINCHO 2013; VILLADA *et al.* 2014).

Aprendizaje

La intensidad de una sinapsis no viene representada por una cantidad fija, sino que puede ser modulada en una escala temporal mucho más amplia que la del disparo de las neuronas (horas, días o meses). Esta plasticidad sináptica se supone que constituye, al menos en buena medida, el aprendizaje tal y

como postuló (HEBB 1949), encontrándose posteriormente evidencias experimentales de ello en estudios realizados por (KANDEL and HAWKINS 1992; LANZARINI *et al.* 2015; MARTIN DEL BRIO and SANZ MOLINA 2001; SÁNCHEZ *et al.* 2016).

Los modelos de RNA pueden ser clasificados según el aprendizaje y la arquitectura de la red (ver Figura 4). Según el aprendizaje se clasifican en supervisados, no supervisados, híbridos y reforzados. Por la arquitectura estos modelos pueden ser unidireccionales (la información fluye en un solo sentido) o realimentadas (la información fluye entre las capas en cualquier sentido) (ANDERSON 2007; ANDRADE TEPÁN 2013; LANZARINI *et al.* 2015; MARTIN DEL BRIO and SANZ MOLINA 2001).

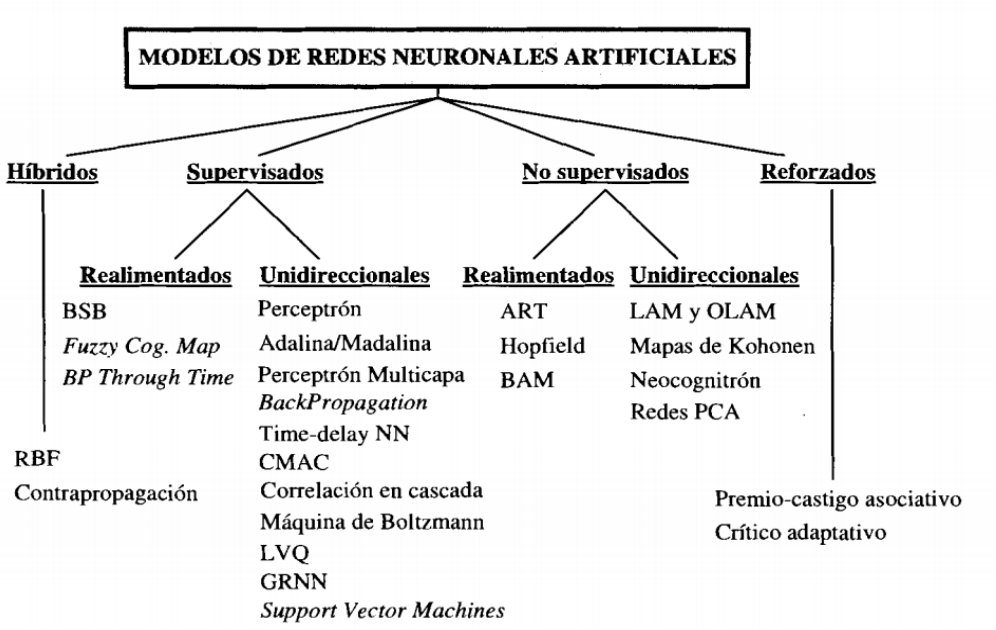

Figura 4: Modelos de RNA. Fuente: MARTIN DEL BRIO and SANZ MOLINA 2001)

Aprendizaje supervisado: en el aprendizaje supervisado se presenta a la red un conjunto de patrones, junto con la salida deseada u objetivo e iterativamente ésta ajusta sus pesos hasta que su salida tiende a ser la deseada, haciendo uso del error que se comete en cada paso. De esta forma la red es capaz de determinar la relación existente entre las entradas y las salidas (ANDRADE TEPÁN 2013; BALLESTEROS 2013; GARCÍA 2016; MARTIN DEL BRIO and SANZ MOLINA 2001; PERDOMO 2015).

Aprendizaje no supervisado: En este tipo de aprendizaje la red no hace uso de las salidas para llegar a una respuesta, mediante la utilización de la regla de aprendizaje reconoce regularidades en el conjunto de entradas, extraer rasgos o agrupar patrones. Debido a la poca información inicial para

trabajar, los algoritmos no supervisados son difíciles de construir y utilizar (ANDRADE TEPÁN 2013; BALLESTEROS 2013; MARTIN DEL BRIO and SANZ MOLINA 2001; ROJAS QUINCHO 2013).

Teniendo en cuenta la definición mencionada anteriormente de problemas de clasificación y los tipos de aprendizaje, cuando se desea resolver un problema de clasificación utilizando redes neuronales el tipo de aprendizaje a utilizar es el supervisado.

Modelos de redes neuronales

Se realizó un estudio de algunos modelos de RNA utilizados en los problemas de clasificación. Uno de los primeros modelos de RNA propuestos fue el de McCulloch-Pitts, llevado a cabo a partir de un modelo matemático y partiendo de la idea de que la red operaba a través de impulsos binarios utilizando el paso mediante un umbral. Posteriormente en (ROSENBLATT 1958) se le propone al modelo un algoritmo para el entrenamiento, basado en el ajuste de los pesos de las conexiones entre los niveles de entrada y salida, conocido como Perceptrón (CASTELLANOS PEÑUELA 2013; GRAUPE 2007; HAGAN *et al.* 2014; MARTIN DEL BRIO and SANZ MOLINA 2001; MINSKY and PAPERT 1969; ROJAS QUINCHO 2013; ROSENBLATT 1958).

A pesar de los grandes avances presentados por el modelo, en un estudio realizado por (MINSKY and PAPERT 1969) se demostró que el principal problema radica en que el Perceptrón de una sola capa al representar un discriminador lineal es incapaz de representar funciones que no sean linealmente separables (GRAUPE 2007; HAGAN et al. 2014).

Al insertarle capas ocultas a un Perceptrón simple se obtiene un Perceptrón multicapa (MLP por sus siglas en inglés), arquitectura que generalmente es entrenada utilizando el algoritmo de retro-propagación del error. La arquitectura más común de un MLP es la de una capa oculta con una salida lineal (ver Figura 5), pero la inclusión de neuronas no lineales en la capa de salida es una solución que se adopta generalmente en los problemas de clasificación (HAGAN et al. 2014; MARTIN DEL BRIO and SANZ MOLINA 2001).

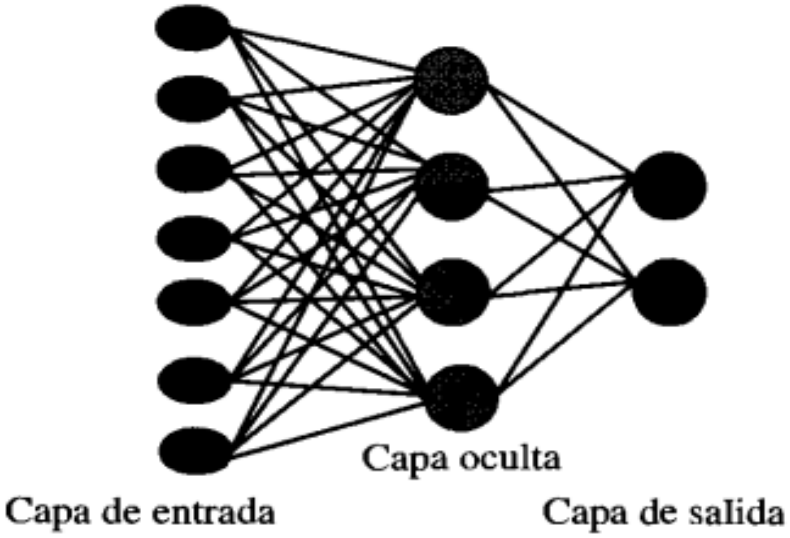

Figura 5: Arquitectura de un MLP. Fuente: (MARTIN DEL BRIO and SANZ MOLINA 2001)

El método de retropropagación del error (BP por sus siglas en inglés) fue formalizado por (RUMELHART et al. 1986), para que una red del tipo MLP aprendiera la asociación que existe entre un conjunto de patrones de entrada y sus salidas correspondientes. Esta red tiene la capacidad de generalización, que no es más que la facilidad de dar salidas satisfactorias a entradas que el sistema no ha visto nunca en su fase de entrenamiento.

Retropropagación del error

El algoritmo BP realiza una búsqueda del valor mínimo de la función de error en la colección de pesos de la red, haciendo usando del método Descenso de gradiente. El conjunto de pesos que minimizan la función de error se considera la solución al problema de aprendizaje, el cual constituye además la solución al problema de asignación de créditos, que consiste en que en caso de que exista un error en una neurona de la capa de salida, no se conoce cuál de las neuronas de las capas anteriores fue la que lo causó (ANDERSON 2007; ANDRADE TEPÁN 2013; HAGAN *et al.* 2014; MARTIN DEL BRIO and SANZ MOLINA 2001; RUMELHART *et al.* 1986; XABIER BASOGAIN OLABE 2004).

El Aprendizaje de Cuantificación Vectorial (LVQ por sus siglas en inglés) es un modelo híbrido, propuesto por (KOHONEN *et al.* 1996), el cual está orientado a la clasificación de patrones, se basa en premiar a aquellas neuronas que clasifican correctamente un determinado patrón, actualizando sus pesos con la regla convencional, y castigar a las que realizan una clasificación errónea, modificando

sus pesos en sentido contrario (ANDRADE TEPÁN 2013; HAGAN *et al.* 2014; HONG *et al.* 2014; MARTIN DEL BRIO and SANZ MOLINA 2001; NAOUM and AL-SULTANI 2012; OUYANG *et al.* 2014).

La red neuronal LVQ puede ser expresada como una red neuronal de tres capas (ver Figura 6). La primera capa es de entrada, tiene tantas neuronas como variables de entrada el patrón. La segunda, también conocida como capa competitiva y la tercera capa es de salida (NAOUM and AL-SULTANI 2012; OUYANG et al. 2014).

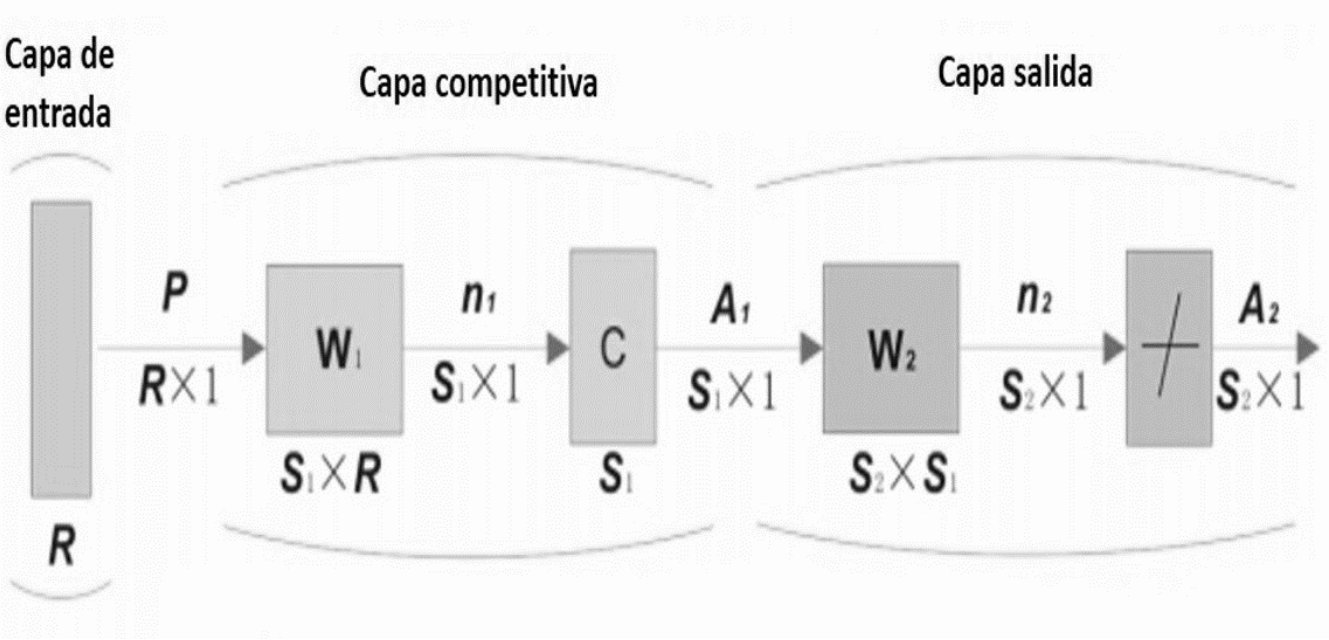

Figura 6: Arquitectura del LVQ. Fuente (HONG et al. 2014)

El rendimiento de las redes neuronales es sensible al número de neuronas. Muy pocas neuronas pueden resultar en mala aproximación, mientras que demasiadas neuronas pueden contribuir al sobreajuste de un problema. Obviamente, el número de neuronas es contradictorio entre la consecución de un mejor rendimiento de la red y la simplificación de la topología de red. Desde los años 1990, se han utilizado con éxito los Algoritmos Evolutivos (EAS por sus siglas en inglés) para optimizar el diseño y los parámetros de las RNA (DING *et al.* 2013).

1.4.3 Redes neuronales evolutivas

En la actualidad, una de las principales áreas de interés en la IA se relaciona con el diseño de sistemas auto-adaptativos, que son capaces de transformarse para resolver diferentes tipos de problemas o para adaptarse a los cambios del entorno (KONAR 2000). A pesar de las potencialidades que presentan las

RNA, en la resolución de una gran cantidad de problemas reales, debido a su capacidad de aprender por ejemplos y su capacidad de generalización, una arquitectura que da solución a un problema determinado no puede ser utilizada para resolver otro problema. En ese caso se hace necesario definir una nueva arquitectura desde el inicio: definir su estructura, decidir la cantidad de neuronas y capas ocultas y reajustar los pesos. No es un trabajo sencillo incluso para los profesionales más experimentados, debido al tamaño y la complejidad del espacio de búsqueda, incluso para los modelos de redes mejor entendidos (MANRIQUE 2001; SÁNCHEZ *et al.* 2016). En estas condiciones, se hace necesario la construcción de un sistema inteligente auto-adaptativo basado en redes neuronales para solucionar cualquier problema que se presenta como un conjunto de patrones de entrenamiento (BERMÚDEZ *et al.* 2013; LUNA 2013; RABUÑAL and DORADO 2006; SÁNCHEZ *et al.* 2016).

Actualmente, la decisión de seleccionar una arquitectura, se ha realizado mediante la engorrosa tarea de prueba y error, lo cual hace que el tiempo sea excesivamente largo. Existen múltiples algoritmos que intentan reducir el número de neuronas ocultas. Estos algoritmos del tipo constructivos/destructivos, dependen fuertemente de la topología inicial de la red y presentan ciertas facilidades para caer en mínimos locales, no garantizando una solución satisfactoria. Además como la estructura de la red está fuertemente ligada al algoritmo de aprendizaje, no se puede garantizar que el proceso de aprendizaje sea el más idóneo para un determinado algoritmo, y por otra parte si el proceso de aprendizaje no da buenos resultados, no es posible saber si ha fallado la red o el algoritmo de aprendizaje. La aplicación de los algoritmos genéticos al diseño de redes neuronales para resolver una determinada tarea está siendo dirigida desde tres campos de aplicación (BERMÚDEZ *et al.* 2013; CASTILLO, P. A. *et al.* 2003; CASTILLO, P.A. *et al.* 2005; LUNA 2013; MARÍN 1995):

- La evolución o búsqueda del conjunto óptimo de los pesos de conexión.
- La evolución o búsqueda de la topología óptima de la red.
- La evolución o búsqueda de las reglas de aprendizaje óptima.

Teniendo en cuenta el estudio realizado y dada la variabilidad de las medidas que describen los FCE las cuales pueden cambiar en dependencia de la organización y el tiempo, es necesario el uso de una RNA evolutiva que se adapte a las diferentes condiciones.

1.4.4 Algoritmos genéticos

Los Algoritmos Genéticos (AG) son métodos generalmente usados en problemas de búsqueda y optimización de parámetros, basados en la reproducción y en el principio de la supervivencia del más

apto. En (DEB and GOLDBERG 1989) se da una definición más formal, *"los algoritmos genéticos son algoritmos de búsqueda basados en la mecánica de la selección natural y de la genética natural. Combinan la supervivencia del más apto entre estructuras de secuencias con un intercambio de información estructurada, aunque aleatorizado, para construir así un algoritmo de búsqueda que tenga algo de las genialidades de las búsquedas humanas"*. Los algoritmos genéticos utilizan soluciones que por su evaluación han demostrado ser buenas para combinarlas y producir otras todavía mejores. Para ello hacen uso de la codificación, la selección, el cruce y la mutación siendo los operadores fundamentales a utilizar al aplicar un AG (BERMÚDEZ *et al.* 2013; GONZÁLEZ PEREA *et al.* 2016; POSE 2010).

Codificación

Si un problema es factible de ser representado por un conjunto de parámetros (conocidos como genes), éstos pueden ser unidos para formar una cadena de valores (cromosoma), a este proceso se le llama codificación (GESTAL *et al.* 2010). El cual contiene la información necesaria para construir un organismo conocido como fenotipo[3]. Esos mismos términos se aplican en algoritmos genéticos. El conjunto de los parámetros representados por un cromosoma particular recibe el nombre de genotipo[4]. El genotipo contiene la información necesaria para la construcción del organismo. Desde los primeros trabajos de (GOLDBERG and HOLLAND 1988) la codificación suele hacerse mediante valores binarios. Se asigna un número determinado de bits a cada parámetro, el número depende del ajuste que se desea realizar y no todos los parámetros deben estar codificados con la misma cantidad de bits. También existen otros tipos de codificación como las Basadas en el orden y las Reales (POSE 2010; RUSSELL and NORVIG 2003).

Selección

Es el proceso mediante el cual un individuo (cromosoma) es copiado proporcionalmente a su evaluación, formando un conjunto intermedio de individuos. El conjunto intermedio se convierte tentativamente en una nueva población a la cual se le aplican los otros operadores genéticos. Se ha de otorgar un mayor número de probabilidades a los individuos más aptos, sin descartar a los menos aptos con el objetivo de lograr en un futuro generaciones homogéneas. Una opción bastante común es

[3] Conjunto de caracteres visibles que un individuo presenta como resultado de la interacción entre su genotipo y el medio.

[4] Conjunto de los genes que existen en el núcleo celular de cada individuo.

seleccionar el primer individuo utilizando uno de los métodos de selección y el otro de manera aleatoria.(POSE 2010; RUSSELL and NORVIG 2003).

Cruce

El cruce se lleva a cabo sobre el conjunto intermedio generado por la selección. Primero se selecciona aleatoriamente una pareja de individuos para ser cruzados. Después, con el uso de la teoría de las probabilidades se determina si habrá cruce entre los dos individuos seleccionados o no. Normalmente el cruce se maneja dentro de la implementación del AG con un porcentaje que indica con qué frecuencia se efectuará. Lo anterior significa que no todas las parejas de cromosomas se cruzarán, sino que habrá algunas que pasarán intactas a la siguiente generación. (POSE 2010; RUSSELL and NORVIG 2003)

A partir del estudio realizado se pudo apreciar que existen métodos que hacen evolucionar las RNA, donde se combinan las metodologías de RNA y los algoritmos genéticos, los cuales se comportan como una eficaz herramienta para tratar problemas que presentan superficies con múltiples mínimos locales y grandes espacios de búsqueda.

En la presente investigación en la fase de modelado planteada se realiza el diseño y posteriormente la implementación de una RNA evolutiva. Se selecciona como modelo de RNA el MLP.

1.5 Conclusiones parciales

- La revisión bibliográfica evidenció la necesidad de realizar un pronóstico de éxito en vista de la MPS visualizándose la factibilidad del uso de una RNA.
- El análisis de las metodologías de minería de datos determinó la necesidad de ejecutar las fases de modelación y evaluación para el desarrollo de la solución.
- Como resultado del estudio de los modelos de redes neuronales fue identificado el MLP como el modelo a utilizar para la solución.
- De acuerdo a la variabilidad de las medidas de los FCE en la MPS se seleccionó el uso de una red evolutiva que permita adaptarse a las condiciones de las distintas organizaciones.

CAPÍTULO 2. CARACTERÍSTICAS DE LA SOLUCIÓN

2.1 Introducción

En el presente capítulo se realiza la descripción de la actividad **construcción del modelo** correspondiente a la fase de modelado mencionada en el capítulo anterior. En dicha actividad se realiza la selección del tipo de codificación, el diseño de la topología y la forma de evolución de los pesos de la RNA.

2.2 Modelado de la solución

El proceso de diseño de la RNA evolutiva consta de las etapas siguientes: en primer lugar es necesario elegir la topología de la red y por último, escoger el método de entrenamiento de la red.

2.2.1 Elección de la codificación y los pesos iniciales

Tanto en la evolución de los pesos como en la topología de la red, la selección de la codificación adecuada es fundamental, ya sea para la representación de las redes (codificación binaria o real), o como para la cantidad de información que cada individuo de la población codifica (codificación directa, indirecta o de Arquitecturas Básicas) (BALLESTEROS 2013; BARRIOS 2001; BERMÚDEZ *et al.* 2013; CASTILLO, P.A. *et al.* 2005; GRANITTO *et al.* 2013).

Cualquiera de estos métodos de codificación debe emplear un operador de cruce adecuado, como el de un punto, de dos puntos, uniforme, y los operadores generalizados, como el cruzamiento de Hamming (H-X) (BARRIOS 2003). Este último está basado en el concepto de distancia de Hamming, está especializado para trabajar con la codificación Arquitecturas Básicas, aprovechando sus beneficios para obtener un mejor rendimiento en la búsqueda de arquitecturas neuronales que otros operadores de cruce binarios.

Sin embargo hacer uso de la codificación de Arquitecturas Bases y el operador de cruce de Hamming combinando una parte de una RNA con otra parte de otra RNA, posiblemente hará que ambas redes pierdan funcionalidad (BARRIOS 2003). Otra variante de hacer evolucionar la topología de una RNA es mediante el uso de una codificación indirecta, los algoritmos constructivos/destructivos y el uso de algoritmos genéticos (GUTIÉRREZ *et al.* 2005) siendo la variante seleccionada.

La inicialización de los pesos de la red es clave para obtener una convergencia rápida en un MLP, ya que dependiendo del punto del espacio de búsqueda del que se parta, se obtienen mejores o peores resultados al realizar el entrenamiento.

Existen diversos métodos de inicialización de los pesos, el más simple de todos es realizar una inicialización aleatoria de los pesos, que resulta sencillo y generalmente produce múltiples soluciones. Otros métodos más elaborados requieren de un análisis estadístico de los patrones de entrenamiento lo que los hace poco eficientes. También se recomienda un rango pequeño de inicialización, que puede hacer que el proceso de aprendizaje sea más rápido. Por otra parte existen métodos que mediante la aplicación de algoritmos genéticos permiten evolucionar los pesos de conexión de la RNA afín de encontrar los valores de los pesos que minimizan el error cuadrático medio (BERMÚDEZ *et al.* 2013; CASTILLO, P.A. *et al.* 2005; DING *et al.* 2013) , siendo esta variante la seleccionada para la investigación.

2.2.2 Evolución de los pesos de conexión

El algoritmo de retropropagación del error y sus variantes se han aplicado en numerosas tareas obteniéndose resultados satisfactorios aunque presentan algunos problemas ya que se basan en descenso del gradiente. Una forma de resolver el problema es aplicar métodos que hagan evolucionar los pesos de conexión directamente. Así se puede utilizar un AG para buscar un conjunto de pesos, de forma global, sin utilizar información del gradiente. De esta forma se evita tener que ajustar los parámetros del algoritmo de aprendizaje (CASTILLO, P.A. *et al.* 2005; DING *et al.* 2013; GRANITTO *et al.* 2013).

La forma habitual para entrenar las redes neuronales utilizando algoritmos genéticos es construir, a partir de una red neuronal y un conjunto de patrones de entrenamiento, un AG que sea capaz de encontrar los valores de los pesos de las conexiones que minimizan el error cuadrático medio o asegurarse de que es al menos inferior a lo que se considera que es aceptable (BALLESTEROS 2013; BERMÚDEZ *et al.* 2013; GRANITTO *et al.* 2013; LUNA 2013; RABUÑAL and DORADO 2006).

A continuación se describe un proceso genérico para el entrenamiento de una red unidireccional utilizando algoritmos genéticos (BALLESTEROS 2013; GRANITTO *et al.* 2013; LUNA 2013; RABUÑAL and DORADO 2006):

- Generación de la población inicial de individuos (cromosomas), cada uno de los cuales codifica un conjunto de valores para los pesos de las conexiones y los bias de la red neuronal.

- Evaluación de la aptitud de cada individuo mediante el cálculo del error cuadrático medio, con el objetivo de ser utilizado en la selección.
- Aplicación del operador de cruzamiento para generar nuevos individuos hasta que se converja a un individuo que minimice el error.

Para la evolución de los pesos es necesario también realizar la selección de la forma de codificación. La forma más común es codificar el valor binario de cada peso de la red y construir el individuo mediante la concatenación de estos números binarios en una cadena de bits. Otra forma de codificación es la de Gray, la cual se utiliza muy a menudo para reducir la posibilidad de pequeños cambios en los bits que conducen a grandes variaciones en los valores que representan (BALLESTEROS 2013; RABUÑAL and DORADO 2006).

En la presente investigación es utilizada la **codificación real** de los pesos ya que es mucho más precisa que la codificación binaria y la de Gray, ya que permite explorar el dominio de la función objetivo con gran precisión (MANRIQUE GAMO 2006).

Como método de selección se escoge la **selección por torneo**, que constituye un procedimiento de selección muy extendido en el cual, la idea consiste en escoger al azar tantos individuos de la población como se haya prefijado el tamaño del torneo (con o sin reemplazamiento), se selecciona el mejor individuo de los que constituyen el grupo de torneo, y se repite el proceso hasta alcanzar el número deseado de individuos seleccionados. Habitualmente, el tamaño del torneo es de dos, y generalmente se utiliza una versión probabilística en la cual se permite la selección de individuos sin que necesariamente sean los mejores de los diferentes torneos que tienen lugar (MANRIQUE GAMO 2006).

Posteriormente para el cruzamiento se selecciona el **cruce morfológico** el cual se trata de una reinterpretación de la operación de gradiente morfológico, muy común en la segmentación de la imagen digitalizada para obtener una medida de la diversidad genética en línea (MANRIQUE GAMO 2006). Esta medida aumenta la velocidad de búsqueda local del AG en el proceso de formación, que es el principal inconveniente reportado por los investigadores que prefieren combinar algoritmos genéticos y métodos de descenso de gradiente, al tiempo que encuentra los óptimos locales menos probable (RABUÑAL and DORADO 2006).

Cruce morfológico

Opera con poblaciones de *λ* individuos formados por cadenas de números reales de longitud "*l"* de la siguiente forma. Partiendo de un número n impar de cadenas progenitoras (n ≤ λ), obtenidas sin repetición de la población actual, se obtiene un conjunto del intervalos, denominados intervalos de cruce *C0,..., Cl-1*, a partir de los cuales se generan las cadenas descendientes del operador, denotadas por *o = (o0,..., ol-1) y o' = (o0',..., ol-1')*. Para cada *i = 0,..., l-1, oi* es un valor aleatorio perteneciente al intervalo de cruce Ci, mientras que oi' se obtiene como:

oi' = (ai+ bi) –oi

Se denomina matriz progenitora, y se denota por *G*, a la matriz que representa todos los valores de los genes de las cadenas padre:

$$G = \begin{pmatrix} a_{10} & a_{11} & \dots & a_{1l-1} \\ a_{20} & a_{21} & \dots & a_{2l-1} \\ \dots & \dots & \dots & \dots \\ a_{n0} & a_{n1} & \dots & a_{nl-1} \end{pmatrix}$$

Figura 7: Matriz progenitora genérica. Fuente:(MANRIQUE 2001)

Para cada una de las *l* columnas de *G* se define el vector unidimensional *fi* como:

$$f_i = (a_{1,i}, a_{2,i}, \dots, a_{n,i}) \text{ con } i = 0, \dots, l-1$$

Figura 8: Definición genérica del vector unidimensional fi. Fuente:(MANRIQUE 2001)

Partiendo de estos *l* vectores columna: *f0,..., fl-1*, el cruce morfológico se lleva a cabo siguiendo tres pasos: cálculo de la medida de la diversidad genética, cálculo de los intervalos de cruce y obtención de la descendencia.

1. Cálculo de la medida de la diversidad genética

El primer paso del operador de cruce morfológico es el cálculo de un valor que proporcione una medida de la diversidad genética de la población. De esta forma, es posible poder decidir de forma dinámica, según se evoluciona en el proceso de convergencia hacia el óptimo global, si es más interesante aplicar técnicas de explotación sobre los esquemas intervalo actualmente existentes, o si, por el contrario, se está en peligro de convergencia subóptima. En este último caso, hay que tomar medidas más intensas de exploración, provocando la ruptura de patrones intervalo.

La obtención de la medida de la diversidad genética se realiza gen a gen a partir de los *n* individuos tomados como progenitores, es decir, se toma una medida *gi* para cada vector columna *fi* de la matriz progenitora. Puesto que se trabaja con cada uno de los genes que forman los individuos independientemente, es posible conocer cómo evoluciona cada uno de ellos por separado, lo cual proporciona una mayor información, y es posible actuar de forma distinta (explorando o explotando) según cada caso.

La transformación gradiente morfológico consiste en la resta entre la dilatación y la erosión, obteniendo la función *g*. Su expresión es la siguiente:

$$g(s) = (fi \oplus b) - (fi \ominus b)$$

El funcionamiento interno de esta operación aplicada sobre la componente situada en la posición media del vector *fi*, es decir, el cálculo de $g\ (E\ (n/2)+1)$ se realiza en tres pasos:

1. En primer lugar, se aplica una operación de dilatación del vector *fi* sobre el punto $E(n/2)+1$, con el elemento estructurante *b.* El resultado de esta operación es el valor máximo de las componentes del vector, puesto que el elemento estructurante recorre a *fi* desde la componente $E(n/2)+1-E(n/2) = 1$; hasta $E\ (n/2) +1 +E\ (n/2) = 2\ E(n/2) +1 = n$ *(ya que n es impar).*
2. Posteriormente se calcula, de la misma forma, la erosión del vector *fi* sobre el punto $E(n/2)+1$. Esta operación consiste en el cálculo del valor mínimo de las componentes del vector *fi*.
3. Finalmente, se realiza la resta entre los valores máximo y mínimo de las componentes de *fi* obtenidas en los dos pasos anteriores.

Definición: Sea *G* la matriz progenitora de dimensión $(n \times l)$ y sean $f0,..., fi,..., fl-1$ sus *l* vectores columna, $fi= (a1, i, a2, i,..., an, i)$ que contienen los *n* valores de los *n* progenitores para el gen *i.* Se define como medida de la diversidad genética del gen *i* en la población, el valor $gi \in [0,1]$ calculado como:

$$gi= g\,(E\,(n/2)+1) = (fi \oplus b)\,(E\,(n/2)+1) - (fi \ominus b)\,(E\,(n/2)+1)$$

Es decir, se toma como medida de la diversidad genética del gen número *i* el valor del gradiente morfológico aplicado sobre la componente situada en la posición media del vector columna *fi* de la matriz progenitora.

Una característica muy importante de esta medida definida en la ecuación, es su bajo coste computacional al constar únicamente del cálculo del valor máximo y mínimo de un vector y, posteriormente, la resta entre ambos. Al contrario que el resto de medidas existentes hasta el momento, no hay que aplicar ni una sola multiplicación o división, que es lo computacionalmente costoso cuando se trabaja con números reales.

2. Cálculo de los intervalos de cruce

Como se ha dicho anteriormente, el cruce morfológico toma un número impar de progenitores sin repetición de la población actual para generar dos nuevos descendientes $o= (o0,..., ol-1)$ y $o' = (o'0..., o'l-1)$. Cada uno de los pares de genes *oi* y *oi'* de la descendencia se obtienen como dos valores pertenecientes al intervalo de cruce *Ci.* El cálculo de cada uno de los *l* intervalos de cruce *Ci* depende de una función ϕ denominada función de exploración/explotación.

Sea *gimax* el gen máximo, definido como la resta entre la dilatación del vector *fi* en el punto medio y el valor de la función ϕ en el punto *gi* (valor de la diversidad genética del gen número *i*), $gimax= (fi \oplus b)\,(E\,(n/2)+1) - \phi\,(gi)$:

$$gimax= máx\,(fi) - \phi\,(gi)$$

Asimismo, se define el gen mínimo *i*, y se denota por *gimin*, como la suma entre la erosión del vector *fi* en el punto medio, y el valor de la función ϕ en el punto *gi*, $gimin= (fi \ominus b)\,(E\,(n/2)+1) + \phi\,(gi)$:

$$gimin= mín\,(fi) + \phi(gi)$$

El gen máximo y el gen mínimo son los valores que determinan finalmente los bordes de los intervalos de cruce, de tal forma que $Ci= [gimin, gimax]$, con $i = 0,..., l-1$. La función ϕ de exploración/explotación (FEE), es crucial en el funcionamiento del cruce morfológico, ya que determina cómo van a ser los intervalos de cruce.

Mediante la inclusión de la FEE en el cruce morfológico es posible controlar la construcción de los intervalos de cruce, para que éstos no sean siempre iguales a los intervalos definidos por los padres.

Puesto que depende de la variable *gi*, es decir, de la diversidad genética de la población actual, la FEE debe de proporcionar una regla que permita, localizar la búsqueda dentro del intervalo *[mín (fi), máx (fi)]* (explotación), o bien por el contrario, tratar de buscar nuevos puntos dentro del espacio de búsqueda (exploración).

Cuando los valores que toma algún determinado gen *i* en los individuos de la población son muy similares entre sí, converge hacia algún punto. El valor obtenido de *gi* será muy cercano a cero y por tanto, la FEE debe expandir el intervalo *[mín (fi), máx (fi)]*, para permitir la exploración de nuevos puntos dentro del espacio de búsqueda, con el fin de evitar la convergencia hacia un punto que no sea el óptimo. La FEE debe proporcionar valores negativos para valores pequeños de *gi*, con lo que *[mín (fi), máx (fi)] ⊆ [gimin, gimax].*

Existe también la posibilidad de que los valores de un determinado gen sean muy diferentes unos de otros en las cadenas que forman la población actual. O bien, el caso general de que los individuos de la población estén muy dispersos dentro de todo el espacio de búsqueda. Por ejemplo, en la población inicial o en las primeras etapas del proceso de convergencia del AG. En estos casos, el valor de *gi* será muy elevado (más cercano al valor 1) y, por tanto, la FEE debe estrechar el intervalo de referencia. Para ello, la FEE debe proporcionar valores positivos tal que *[gimin, gimax] ⊆ [mín (fi), máx (fi)].*

3. Cálculo de la descendencia

Una vez obtenidos los *l* intervalos de cruce *C0, C1,..., Ci,..., Cl-1, el* tercer paso del cruce morfológico es el cálculo de la descendencia *o = (o0,..., ol-1) y o'= (o'0,..., o'l-1)*, resultado final del operador. Para cada gen *i* (*i = 0,..., l-1)*, este paso se realiza en dos etapas:

1. *oi* es un valor aleatorio escogido del intervalo de cruce *Ci*.
2. *oi'* se obtiene a partir del valor de *oi* según la expresión:

$$oi' = (mín\ (fi) + máx\ (fi)) - oi,\ con\ i = 0, 1,..., l-1$$

Una vez calculada la descendencia mediante el cruce morfológico, se realiza el reemplazo de los nuevos individuos por los peores individuos de la población de partida, tomando como función objetivo el error cuadrático medio. Posteriormente se realiza nuevamente la selección por torneo obteniéndose una nueva matriz progenitora. Este procedimiento se realiza en varias ocasiones hasta alcanzar los valores de los pesos de conexión que minimicen el error cuadrático medio para la configuración de la red en cuestión.

En general los AG son ineficientes al realizar una búsqueda final local mientras que son muy eficientes en la búsqueda global. Por lo que el entrenamiento mediante AG puede ser mejorado si se le incorpora un método de búsqueda local, seleccionándose BP. El AG busca una región adecuada en el espacio de búsqueda y posteriormente el método de búsqueda local afina la solución encontrada obteniendo un resultado más cercano al óptimo en dicha región. Teniendo en cuenta que el BP debe utilizarse varias veces para encontrar los pesos iniciales adecuados, ya que depende de las condiciones iniciales, es muy adecuada realizar la combinación (CASTILLO, P.A. *et al.* 2005; DING *et al.* 2013).

2.2.3 Evolución de la topología de la red

Lo primero a tener en cuenta en el diseño de una red neuronal es que el espacio de búsqueda tiene un alto cardinal. La búsqueda de la arquitectura óptima es, en principio, un buen candidato para ser resuelto mediante algoritmos genéticos.

El uso de un AG para diseñar una red neuronal, como cualquier otro problema de optimización al abordarse mediante esta técnica, exige un método de codificación del espacio de búsqueda de las redes neuronales (GARRO *et al.* 2012; RABUÑAL and DORADO 2006). Del mismo modo, se necesita de la función de evaluación para ser optimizado. Se asume en adelante que el óptimo es la red neuronal que produce el valor mínimo de esta función, por lo que, teniendo en cuenta dos RNA, la red cuya aptitud es más baja es considerada la mejor solución.

El acondicionamiento físico que generalmente se da por una cadena dada, es el error cuadrático medio alcanzado en la formación de la salida de la red neuronal mediante la decodificación de esta cadena. El valor de aptitud se suele ponderar con otros términos que tengan en cuenta hechos como la complejidad de la arquitectura neuronal codificada. Es decir, las personas que representan arquitecturas con una gran cantidad de elementos de procesamiento y conexiones entre ellas se penalizan más que las de arquitecturas simples. Esto se debe a que la arquitectura sencilla tiene una velocidad de ejecución superior y una mayor capacidad de generalización. Una vez que todos los individuos han sido evaluados, los operadores de reproducción, cruce y de mutación genética se aplican a la salida de la nueva generación (población), repitiendo estos pasos hasta que la población converge hacia la arquitectura neuronal óptima para resolver un problema dado (GARRO *et al.* 2012; RABUÑAL and DORADO 2006).

En la Figura 9 se define como relacionar una matriz con una arquitectura de un a red unidireccional con una capa oculta. Sea *n* el número de neuronas de entrada; si $i \leq n$ entonces *(i, j)* representa una conexión entre la neurona de entrada *i* y la *j*-ésima neurona oculta; si $i > n$, *(i, j)* representa una conexión entre la *j*-ésima neurona oculta y el *(i - n)* -ésimo neurona de salida.

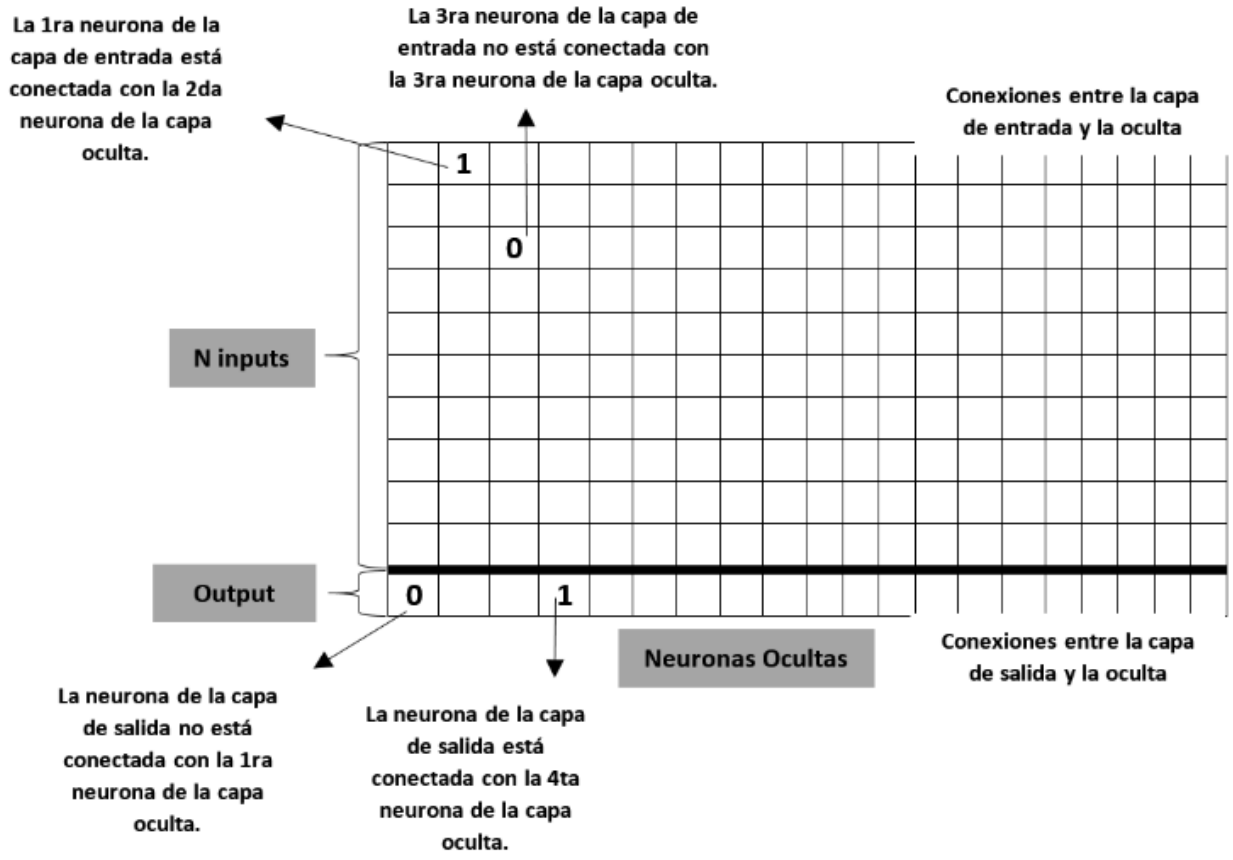

Figura 9: Representación en forma de matriz de una RNA. Fuente: Elaboración propia

Por lo tanto, el significado de cada 1 en M se interpreta como una conexión y un 0 como la ausencia de conexión. Antes de la obtención de la arquitectura definida de la RNA unidireccional, la matriz M debe ser transformado a una matriz más corta de la siguiente manera:

- Se eliminan las columnas con valores 0 en la matriz. Si los elementos de la columna de la orden *k* son 0, no hay conexiones de las entradas a las neuronas k-ésima ocultas y no existen conexiones de la k-ésima neurona oculta a las salidas. Por lo tanto, esa neurona oculta debe ser eliminada.
- Cuando hay una neurona en la capa oculta sin ninguna conexión con la capa de salida, el nodo es también eliminado de la red, ya que no tendrá ninguna influencia en las salidas de la red.

- Cuando hay una neurona de la capa de entrada sin ninguna conexión con la capa oculta, esta neurona será eliminada.

La dinámica completa del enfoque puede entonces ser descrita como sigue:

1. Los individuos de la población son generados al azar, se generan posiciones aleatorias de semillas[5].
2. Cada cromosoma se decodifica y se convierte en las ubicaciones de la red de acuerdo a la codificación. Cada dos genes se representa una semilla. El primer gen representa la coordenada ***x*** de la semilla y el otro, la coordenada ***y***. Además se establece la configuración inicial, (la matriz se encuentra llena por 0 por lo que se hace necesario configurarla para poder aplicar el crecimiento y la poda), definida a continuación:
    a. Es necesario replicar cada semilla de crecimiento secuencialmente, sobre su vecindad cuadrática. La replicación se realiza de tal manera que si una nueva semilla tiene que ser colocada en una posición previamente ocupada por otra semilla, la primera será reemplazada.
3. Los algoritmos de crecimiento y de poda se desarrollan siguiendo el procedimiento siguiente:
    a. Crecimiento: esta regla se aplica hasta que no hay más condiciones de la regla. La configuración obtenida es la configuración final del algoritmo de crecimiento:
        i. La idea es copiar una s_k semilla que crece en particular cuando una celda está inactivo (es decir $a_{i,j} = 0$) y hay al menos tres semillas que crecen idénticas en su vecindario. La regla se define en la Figura 10:

$$
\begin{aligned}
a_{i,j}^{(t+1)} = s_k \text{ IF } a_{i,j}^{(t)} = 0 \text{ AND} \\
a_{i-1,j-1}^{(t)} = a_{i-1,j}^{(t)} = a_{i-1,j+1}^{(t)} = s_k \text{ OR} \\
a_{i+1,j-1}^{(t)} = a_{i+1,j}^{(t)} = a_{i+1,j+1}^{(t)} = s_k \text{ OR} \\
a_{i-1,j-1}^{(t)} = a_{i,j-1}^{(t)} = a_{i+1,j-1}^{(t)} = s_k \text{ OR} \\
a_{i-1,j+1}^{(t)} = a_{i,j+1}^{(t)} = a_{i+1,j+1}^{(t)} = s_k \text{ OR} \\
a_{i-1,j-1}^{(t)} = a_{i-1,j}^{(t)} = a_{i,j-1}^{(t)} = s_k \text{ OR} \\
a_{i-1,j}^{(t)} = a_{i-1,j+1}^{(t)} = a_{i,j+1}^{(t)} = s_k \text{ OR} \\
a_{i,j-1}^{(t)} = a_{i+1,j-1}^{(t)} = a_{i+1,j}^{(t)} = s_k \text{ OR} \\
a_{i+1,j}^{(t)} = a_{i+1,j+1}^{(t)} = a_{i,j+1}^{(t)} = s_k \\
a_{i,j}^{(t+1)} = a_{i,j}^{(t)} \text{ otherwise.}
\end{aligned}
$$

Figura 10: Condiciones para el crecimiento. Fuente: Elaboración propia

[5] Representa una coordenada en la matriz.

b. Poda: las semillas de poda se colocan en la disminución de la red para obtener la configuración inicial del metodo decreciente. Si hay algunas semillas en esos lugares, se sustituyen. Se aplica esta regla hasta que se alcanza la configuración final:
    i. La regla de poda está diseñada para eliminar las semillas que crecen en la red. Se extrae una creciente semilla s_k cuando dos celdas vecinas contiguas contienen semillas que crecen idénticas y otra celda vecina contiene una semilla decreciente. Si dos semillas decrecientes están presentes en la vecindad, la regla no se activa. La regla se define en la Figura 11:

$$a_{i,j}^{(t+1)} = d_r \text{ IF } (a_{i,j}^{(t)} = a_{i-1,j-1}^{(t)} = a_{i,j-1}^{(t)} = s_k \text{ AND } a_{i-1,j}^{(t)} = d_r) \text{ OR}$$
$$(a_{i,j}^{(t)} = a_{i-1,j-1}^{(t)} = a_{i-1,j}^{(t)} = s_k \text{ AND } a_{i,j-1}^{(t)} = d_r)$$
$$(a_{i,j}^{(t)} = a_{i-1,j+1}^{(t)} = a_{i,j+1}^{(t)} = s_k \text{ AND } a_{i-1,j}^{(t)} = d_r)$$
$$(a_{i,j}^{(t)} = a_{i-1,j}^{(t)} = a_{i-1,j+1}^{(t)} = s_k \text{ AND } a_{i,j+1}^{(t)} = d_r)$$
$$(a_{i,j}^{(t)} = a_{i,j-1}^{(t)} = a_{i+1,j-1}^{(t)} = s_k \text{ AND } a_{i+1,j}^{(t)} = d_r)$$
$$(a_{i,j}^{(t)} = a_{i,j+1}^{(t)} = a_{i+1,j+1}^{(t)} = s_k \text{ AND } a_{i+1,j}^{(t)} = d_r)$$

$$a_{i,j}^{(t+1)} = a_{i,j}^{(t)} \text{ otherwise.}$$

Figura 11: Condiciones para la poda. Fuente: Elaboración propia

c. Una matriz binaria *M* se obtiene finalmente, en donde se sustituyen las semillas en crecimiento por un 1 y las semillas de poda por un 0. La matriz será configurada para obtener una arquitectura de una RNA lineal, como se describe al inicio.

4. Se entrena la red mediante la evolución de los pesos de conexión.
5. Se calcula un valor de medición de la eficiencia de la arquitectura. Esta función de evaluación debe definirse teniendo en cuenta: el error cuadrático medio entre la salida deseada y la obtenida por la red. El valor se utiliza como la función de aptitud del cromosoma.
6. Los pasos del 2 al 5 se repiten para todos los individuos de la población. Haciendo uso de los operadores habituales de cruce de AG se generan nuevas poblaciones de individuos.
7. Los pasos del 2 al 6, se lleva a cabo a través de diferentes generaciones, hasta que la función de aptitud se optimiza.

2.3 Implementación de la solución

Una vez seleccionado el tipo de codificación a usar, la forma en que se va realizar la evolución de los pesos y la topología que va a tener la red neuronal se pasa a la definición de la arquitectura que va a tener la solución.

2.3.1 Descripción de la arquitectura

La propuesta de solución sigue una arquitectura de flujo de datos. Es aplicada cuando los datos de entrada se transforman en datos de salida mediante una serie de componentes aplicando el cálculo o la manipulación. La estructura utilizada es secuencial por lotes. Los componentes son programas independientes; el supuesto es que cada paso se ejecuta hasta completarse antes que se inicie el paso siguiente (PRESSMAN 2002).

La solución fue dividida en paquetes con el fin de agrupar las funcionalidades y proveer un orden con el estilo arquitectónico aplicado. Los paquetes son:

configuración: paquete encargado de realizar la configuración inicial de la red, para ello crea una matriz, la cual representa la topología, véase epígrafe **2.2.3**. Además se encarga de generar las semillas de crecimiento y poda, útiles para el proceso de creación de la red. El proceso se realiza siguiendo las indicaciones planteadas en el epígrafe antes mencionado.

construcción: paquete encargado de la creación de la topología. Una vez obtenida la configuración inicial se procede a realizar el crecimiento de la red, cumpliendo con las restricciones planteadas en el epígrafe **2.2.3**. Luego se realiza la configuración necesaria para la poda y seguidamente se procede a la realización de la poda cumpliéndose las restricciones mencionadas.

codificación: paquete encargado de dar inicio al proceso de entrenamiento con AG. Una vez realizada la configuración final cumpliéndose con todas las restricciones, se realiza la decodificación de la RNA. Posteriormente se generan un conjunto de pesos los cuales son codificados para ser utilizados en el componente **cruceMorfológico**.

cruceMorfológico: paquete encargado de realizar la selección y el cruzamiento de los pesos codificados. El proceso se realiza como se describe en la sección **2.2.2**, obteniéndose los más cercanos al óptimo, considerándose los mejores pesos iniciales para ser utilizados por el ***backpropagation***.

backpropagation: paquete encargado de realizar el entrenamiento de la red neuronal. Una vez obtenidos los pesos iniciales se realiza el entrenamiento de la red haciendo uso de los datos almacenados en la base de datos.

Para un mejor entendimiento de la arquitectura propuesta, en la Figura 12 se describe cómo interactúan cada uno de los componentes y paquetes según lo definido en el método propuesto; así como las principales características y funcionalidades presentes en los diferentes paquetes de la solución desarrollada.

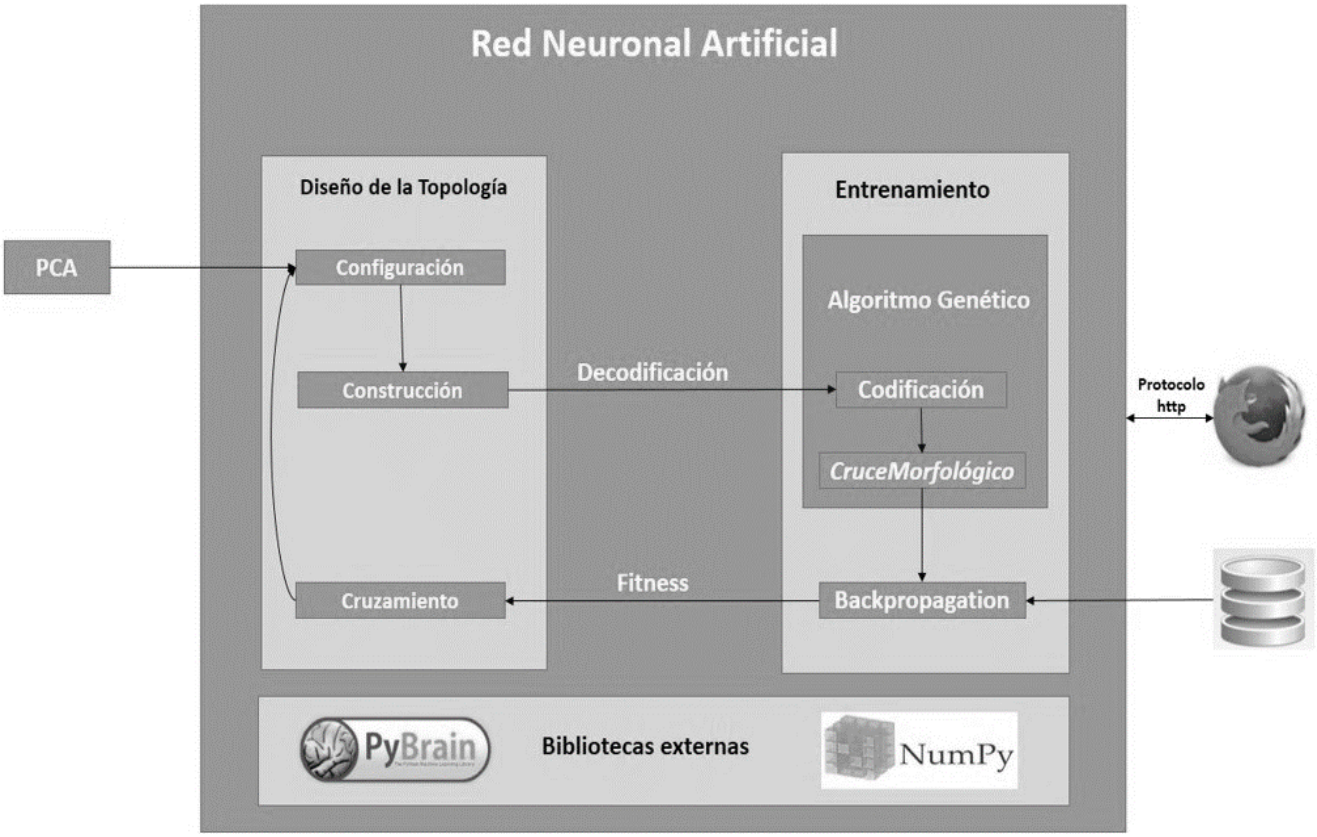

Figura 12: Arquitectura de la RNA. Fuente: Elaboración propia

A partir de un Análisis de Componentes Principales (PCA por sus siglas en inglés), con el cual se reduce la cantidad de medidas de entrada a la RNA, se procede a realizar el diseño y entrenamiento de la RNA. Primeramente se realiza el diseño de la topología de la red: en el paquete ***configuración*** se crea la configuración inicial de la red para posteriormente en el paquete ***construcción*** se crea la topología de la red. Una vez terminado este proceso se realiza la decodificación de la red y la configuración final.

Luego se inicia el proceso de entrenamiento de la red: en el paquete ***codificación*** se codifican los pesos de la red los cuales serán utilizados en el paquete ***cruceMorfológico*** en el cual se realiza la búsqueda de los pesos óptimos para la topología de red obtenida. Estos valores serán utilizados en el paquete ***backpropagation*** como los pesos iniciales y al obtener los datos de la base de datos se realizaría el entrenamiento de la RNA obteniéndose un **fitness**[6], valor que permite determinar qué tan buena es la red. Luego se realiza el proceso del paquete ***cruzamiento*** para obtener una nueva topología de la RNA. El proceso completo se repite hasta obtener la arquitectura de RNA de menor fitness.

2.3.2 Diseño de la base de datos

En el modelo de datos se definen los conceptos que son utilizados en el sistema y que describen la estructura de la base de datos diseñada. En este modelo se representan los datos, sus atributos y tipos, sus relaciones.

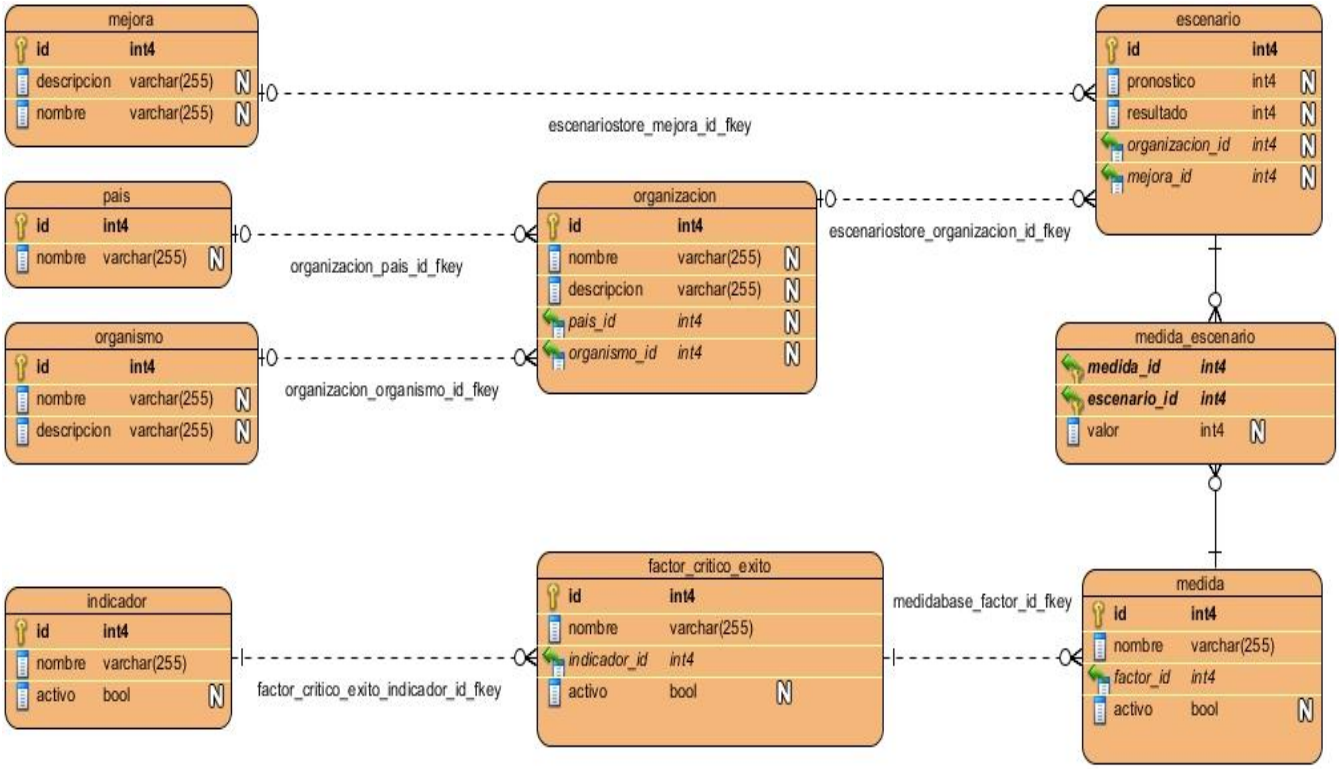

Figura 13: Diagrama Entidad-Relación. Fuente: Elaboración propia

[6] Para la investigación se considera como el valor del error cuadrático medio de la red una vez que ha sido entrenada.

Descripción de las tablas

Se identificaron 9 tablas, a continuación se detallan 2 de ellas, encontrándose el resto en los anexos del 1 al 7.

Tabla 4: Descripción de la tabla escenario. Fuente: Elaboración propia

Nombre: escenario		
Descripción: almacena todos los escenarios de la base de datos		
Atributo	**Tipo**	**Descripción**
id	integer	Identificador
pronostico	integer	Resultado pronosticado previo a la inserción en la mejora de procesos de software
resultado	integer	Resultado obtenido en la inserción en la mejora de procesos de software
organizacion_id	integer	Llave foránea que hace referencia al identificador de la *tabla organizacion*
mejora_id	integer	Llave foránea que hace referencia al identificador de la *tabla mejora*

Tabla 5: Descripción de la tabla factor_critico_exito. Fuente: Elaboración propia

Nombre: factor_critico_exito		
Descripción: almacena todos los factores críticos de la base de datos		
Atributo	**Tipo**	**Descripción**
id	integer	Identificador
nombre	string	Nombre literal de factor crítico de éxito
indicador_id	integer	Llave foránea que hace referencia al identificador de la *tabla indicador*
activo	boolean	Indica si el factor crítico de éxito está activo o inactivo

A continuación se describen las herramientas y tecnologías a usar en la construcción de la solución.

2.3.2.1 Lenguaje de modelado

El Lenguaje Unificado de Modelado (UML por sus siglas en inglés) es un lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar artefactos de un sistema de software. Se usa para entender, diseñar, configurar, mantener y controlar la información sobre los sistemas a construir (FOWLER and SCOTT 1999; STEVENS and POOLEY 2002).

UML capta la información sobre la estructura estática y el comportamiento dinámico de un sistema. El mismo pretende unificar la experiencia pasada sobre técnicas de modelado e incorporar las mejores prácticas actuales en un acercamiento estándar. UML es una notación con la cual se construyen sistemas por medio de conceptos orientados a objetos. Prescribe un conjunto de notaciones y diagramas estándares, y describe la semántica esencial de lo que estos diagramas y símbolos significan (BOOCH 2005).

2.3.2.2 Visual Paradigm 8.0

Durante la etapa de modelación y para diseñar el sistema se utilizó Visual Paradigm, una herramienta de modelado multiplataforma que no se inclina por ninguna metodología específica. Además ofrece un entorno de creación de diagramas para UML, con soporte para los 13 diagramas de la última versión (UML 2.0). Por otro lado el diseño es centrado en casos de uso y enfocado al negocio, con soporte para los Diagramas de Procesos de Negocios y Diagramas de Flujos de Datos, lo cual genera un software de mayor calidad (PARADIGM 2013).

Usa un lenguaje estándar común para todo el equipo de desarrollo que facilita la comunicación y está capacitado para la ingeniería directa e inversa. Presenta dos tipos de diagramas de modelado de bases de datos: entidad-relación (ERD) y mapeo objeto-relacional (ORM). Los diagramas ERD modelan la base de datos a nivel físico y los ORM muestran la relación entre las clases y las entidades.

2.3.3.3 Programación del lado del servidor

A continuación se describe el lenguaje de programación empleado así como los marcos de trabajo utilizados.

Python 2.7

Python es un lenguaje de programación interpretado cuya filosofía insiste en una sintaxis que favorezca un código legible. Se trata de un lenguaje de programación multiparadigma, ya que soporta orientación a objetos, programación imperativa y, en menor medida, programación funcional. Python usa tipado dinámico y conteo de referencias para la administración de memoria (PYTHON 2016).

Una característica importante de Python es la resolución dinámica de nombres; es decir, lo que enlaza un método y un nombre de variable durante la ejecución del programa (también llamado enlace dinámico de métodos). Otro objetivo del diseño del lenguaje es la facilidad de extensión. Python puede incluirse en aplicaciones que necesitan una interfaz programable (PYTHON 2016).

Django 1.8

Django es un marco de trabajo de desarrollo web de código abierto, escrito en Python, que respeta el patrón de diseño conocido como Modelo-Vista-Plantilla. Django proporciona una aplicación incorporada para administrar los contenidos, que puede incluirse como parte de cualquier página hecha con Django y que puede administrar varias páginas hechas con Django a partir de una misma instalación; la aplicación administrativa permite la creación, actualización y eliminación de objetos de contenido, llevando un registro de todas las acciones realizadas sobre cada uno, y proporciona una interfaz para administrar los usuarios y los grupos de usuarios. La meta fundamental de Django es facilitar la creación de sitios web complejos. Django pone énfasis en la reutilización, la conectividad y extensibilidad de componentes, el desarrollo rápido y el principio “No te repitas”. Python es usado en todas las partes del marco de trabajo, incluso en configuraciones, archivos, y en los modelos de datos (JAISWAL and KUMAR 2015).

Django Rest Framework 2.4.3

Django Rest Framework es una aplicación Django que permite construir proyectos software bajo la arquitectura Transferencia de Estado Representacional (por sus siglas en inglés REST) que es un estilo de arquitectura software para sistemas hipermedia distribuidos como la World Wide Web, incluye gran cantidad de código para reutilizar (Views, Resources, etc.) y una interfaz administrativa desde la cual es posible realizar pruebas sobre las operaciones HTTP como lo son: POST y GET. Hace uso intensivo de las Generic Views, las cuales desde Django 1.3 se basan en clases (class) y no en funciones (def), con el objetivo de aprovechar las ventajas de la programación orientada a objetos (CHRISTIE 2015).

Se utilizó para la creación de un servicio que hace uso de la RNA evolutiva para pronosticar el éxito con vistas a la MPS.

2.3.2.4 Base de datos

Un sistema gestor de bases de datos es un sistema de software que permite la definición de bases de datos, la elección de las estructuras de datos necesarios para el almacenamiento y la búsqueda de los datos, puede ser de forma interactiva o a través de un lenguaje de programación (BERTINO and MARTINO 1995), a continuación se describe el utilizado en la implementación de la solución.

PostgreSQL 9.4

PostgreSQL es un sistema gestor de bases de datos: relacional y de código abierto. Se ejecuta en los sistemas operativos más importantes, incluyendo Linux, Unix, y Windows. Soporta distintos tipos de datos como INTEGER, NUMERIC, BOOLEAN, CHAR, VARCHAR, DATE, INTERVAL y TIMESTAMP. También soporta el almacenamiento de objetos binario grandes, incluyendo imágenes, sonidos y videos. Tiene interfaces nativas de programación para C/C++, Java, .Net, Perl, Python, Ruby, Tcl, ODBC, entre otros y una amplia documentación (POSTGRESQL 2016).

PostgreSQL, cuenta con características avanzadas como Control de Concurrencia Multiversión, espacios de tablas, replicación asincrónica, transacciones anidadas, salvas en línea, un planificador de consultas sofisticadas y se provee de tolerancia a fallos. Es altamente escalable tanto en la enorme cantidad de datos que puede manejar y en el número de usuarios simultáneos que puede acomodar (POSTGRESQL 2016).

2.3.2.5 Bibliotecas de Python

A continuación se describen las librerías de Python utilizadas para el trabajo con vectores y matrices y para el trabajo con la red neuronal.

NumPy

Es una extensión de Python, que agrega mayor soporte para vectores y matrices, constituyendo una biblioteca de funciones matemáticas de alto nivel para operar con los mismos (OLIPHANT 2014). La biblioteca fue utilizada para el trabajo con matrices y números aleatorios.

PyBrain 0.3

Es una biblioteca modular de aprendizaje automático para Python. Su objetivo es ofrecer algoritmos flexibles y fáciles de usar, manteniendo el alcance para las tareas de aprendizaje y una variedad de entornos predefinidos para probar y comparar los algoritmos (RUANGCHAI 2014). Permitió la generación de las redes neuronales, así como su entrenamiento mediante el uso del algoritmo BP.

2.4 Conclusiones parciales

- La arquitectura seleccionada para el diseño define el desarrollo de la RNA evolutiva. Agrupa los dos elementos fundamentales en paquetes y evidencia el flujo de información entre ellos.
- El empleo de los AG para la evolución de una RNA permitió la adaptación de la misma a diferentes condiciones respondiendo a las características específicas de cada organización.
- El uso del algoritmo BP refinó la solución obtenida por el AG al realizar la evolución de los pesos de la red.
- Las herramientas y tecnologías utilizadas son fundamentales en el modelado de la solución e implementación de la RNA evolutiva, aprovechándose las principales características y beneficios de las mismas.

CAPÍTULO 3. VALIDACIÓN DE LA PROPUESTA

3.1 Introducción

En este capítulo se realizará la validación de la red neuronal aplicando un Cuasi-experimento y la Validación cruzada. En las siguientes secciones se describen las pruebas para posteriormente realizar un análisis de los resultados obtenidos, estando en consecuencia con la fase de evaluación mencionada anteriormente.

3.2 Aplicación del Cuasi-experimento

Teniendo en cuenta, que los programas de MPS tardan años y que la manifestación del impacto de los FCE no se evidencia en toda su magnitud desde los inicios del programa de la MPS, se consideró emplear un Cuasi-experimento. La decisión se tomó a raíz de que el Cuasi-experimento constituye adecuado para analizar efectos a mediano o largo plazo, cuando se tienen bases para suponer que la influencia de la variable independiente sobre la dependiente, como es el caso, tarda en manifestarse.

Para el Cuasi-experimento, se tuvieron en cuenta como entrada para el pronóstico, los datos obtenidos en el diagnóstico aplicado a 14 centros de desarrollo de la UCI en el año 2012. Los datos que se manejan son resultados de un diagnóstico y no un resultado real, por tanto no se manejan en términos de “éxito” o “fracaso”, sino “muy adecuado”, “adecuado”, “poco adecuado” y “no adecuado”. Sin embargo, se consideró oportuno establecer una comparación entre los resultados que se obtienen en el diagnóstico y los resultados del pronóstico.

Para efectuar la valoración del funcionamiento se siguieron los siguientes pasos:

1. Seleccionar aleatoriamente el conjunto de estados resultantes del diagnóstico:
   Se seleccionaron los centros: CENIA, CDAE, CEGEL, CEIGE, CIDI, TLM, CESIM, CEDIN, CESOL, CISED y DATEC.
2. Seleccionar aleatoriamente los estados a realizar el pronóstico:
   Se seleccionaron los centros: FORTES, GEYSED e ISEC.
3. Someter los estados para realizar el pronóstico:
   Una vez sometidos los 3 estados al pronóstico el resultado fue:
   - FORTES: arrojó un pronóstico de fracaso.
   - GEYSED: arrojó un pronóstico de fracaso.
   - ISEC: arrojó un pronóstico de fracaso.

4. Comparar resultados obtenidos en el pronóstico Vs. resultados reflejados en el diagnóstico.

En el diagnóstico realizado a la UCI el resultado obtenido para los tres centros fue “Poco Adecuado”, en cuyo caso se sugiere que no se inicie la MPS pues existe una tendencia al fracaso.

Al utilizar la red para el pronóstico se obtuvo como resultado *“fracaso”*, al comprobarse en el diagnóstico que no se sugiere el inicio en la MPS por ser *“Poco Adecuado”*, se propone no iniciarse en la MPS, por lo que se puede apreciar que en los 3 estados coincide el pronóstico.

3.3 Validación cruzada

En las técnicas de minería de datos se aplica el método de Validación cruzada, que consiste en dividir el conjunto de datos en *n* partes construyendo un modelo con *n-1* de ellas y reservando la restante para la validación de los resultados obtenidos, con lo que finalmente para cada repetición del algoritmo se ajustarán n modelos validados sobre observaciones ‘nuevas’. Este proceso se repite *m* veces consiguiendo por lo tanto *m x n* modelos ajustados con conjuntos de entrenamiento distinto y validado sobre observaciones no utilizadas en su construcción. Se ha elegido *n = 5* y *m= 5* en este estudio.

En un primer momento los datos de entrenamientos están compuestos por la partición 1, 2, 3, 4 y la partición 5 fue seleccionada para realizar la validación.

En la Figura 14, las dos primeras celdas de la diagonal muestran el número y porcentaje de clasificaciones correctas por la red entrenada. En la primera 393 estados se clasifican correctamente como 0 o “fracaso”. Esto corresponde al 99.49% del total de 395 estados clasificados como fracaso. Del mismo modo, 336 estados se clasifican correctamente como 1 o “exitosos”. Esto corresponde al 82.15% de todos los estados clasificados como exitosos.

De los estados exitosos, 2, se clasifican incorrectamente como fracasos y esto corresponde a un 0.51% del total de 395 estados clasificados como fracasos. Del mismo modo, 73 de los estados de fracaso se clasifican incorrectamente como exitosos y esto corresponde al 17.85% de todos los datos clasificados como exitosos.

Además de 395 predicciones de fracaso, el 99.5% son correctos y el 0.5% están incorrectos, y de 409 predicciones de éxitos, el 82.2% son correctos y el 17.8% están incorrectos. Por otra parte de 466 casos de fracaso, el 94.47% se predijo correctamente como fracasos y el 5.53% se predice como éxitos.

De los 338 casos exitosos, el 99.41% son clasificados correctamente como exitosos y el 0.59% son clasificados como fracasos.

En general, el 90.67% de las predicciones son correctas y el 9.33% son las clasificaciones erróneas.

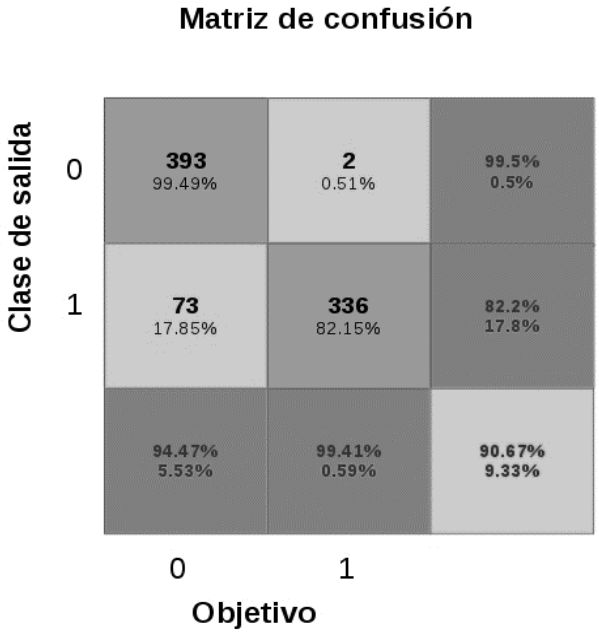

Figura 14: Primera matriz de confusión. Fuente: Elaboración propia

En un segundo momento los datos de entrenamientos están compuestos por la partición 1, 2, 3, 5 y la partición 4 fue seleccionada para realizar la validación.

En la Figura 15, las dos primeras celdas de la diagonal muestran el número y porcentaje de clasificaciones correctas por la red entrenada. En la primera 373 estados se clasifican correctamente como 0 o “fracaso”. Esto corresponde al 94.43% del total de 395 estados clasificados como fracaso. Del mismo modo, 385 estados se clasifican correctamente como 1 o “exitosos”. Esto corresponde al 94.13% de todos los estados clasificados como exitosos.

De los estados exitosos, 22, se clasifican incorrectamente como fracasos y esto corresponde a un 5.57% del total de 395 estados clasificados como fracasos. Del mismo modo, 24 de los estados de fracaso se clasifican incorrectamente como exitosos y esto corresponde al 5.87% de todos los datos clasificados como exitosos.

Además de 395 predicciones de fracaso, el 94.4% son correctos y el 5.6% están incorrectos, y de 409 predicciones de éxitos, el 94.1% son correctos y el 5.9% están incorrectos. Por otra parte de 397 casos

de fracaso, el 93.95% se predijo correctamente como fracasos y el 6.05% se predice como éxitos. De los 407 casos exitosos, el 94.59% son clasificados correctamente como exitosos y el 5.41% son clasificados como fracasos.

En general, el 94.3% de las predicciones son correctas y el 5.7% son las clasificaciones erróneas.

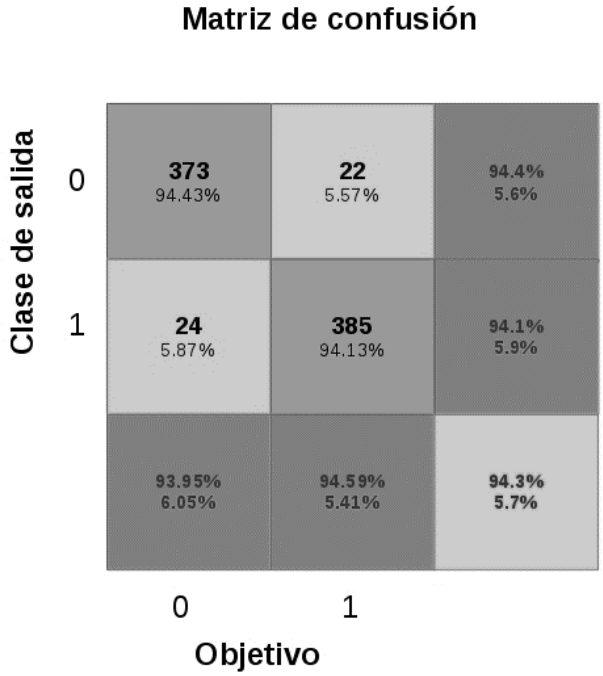

Figura 15: Segunda matriz de confusión. Fuente: Elaboración propia

En un tercer momento los datos de entrenamientos están compuestos por la partición 1, 2, 4, 5 y la partición 3 fue seleccionada para realizar la validación.

En la Figura 16, las dos primeras celdas de la diagonal muestran el número y porcentaje de clasificaciones correctas por la red entrenada. En la primera 604 estados se clasifican correctamente como 0 o “fracaso”. Esto corresponde al 99.5% del total de 607 estados clasificados como fracaso. Del mismo modo, 161 estados se clasifican correctamente como 1 o “exitosos”. Esto corresponde al 81.73% de todos los estados clasificados como exitosos.

De los estados exitosos, 3, se clasifican incorrectamente como fracasos y esto corresponde a un 0.5% del total de 607 estados clasificados como fracasos. Del mismo modo, 36 de los estados de fracaso se clasifican incorrectamente como exitosos y esto corresponde al 18.27% de todos los datos clasificados como exitosos.

Además de 607 predicciones de fracaso, el 99.5% son correctos y el 0.5% están incorrectos, y de 197 predicciones de éxitos, el 81.7% son correctos y el 18.3% están incorrectos. Por otra parte de 640 casos de fracaso, el 94.38% se predijo correctamente como fracasos y el 5.62% se predice como éxitos. De los 164 casos exitosos, el 98.17% son clasificados correctamente como exitosos y el 1.83% son clasificados como fracasos.

En general, el 95.15% de las predicciones son correctas y el 4.95% son las clasificaciones erróneas.

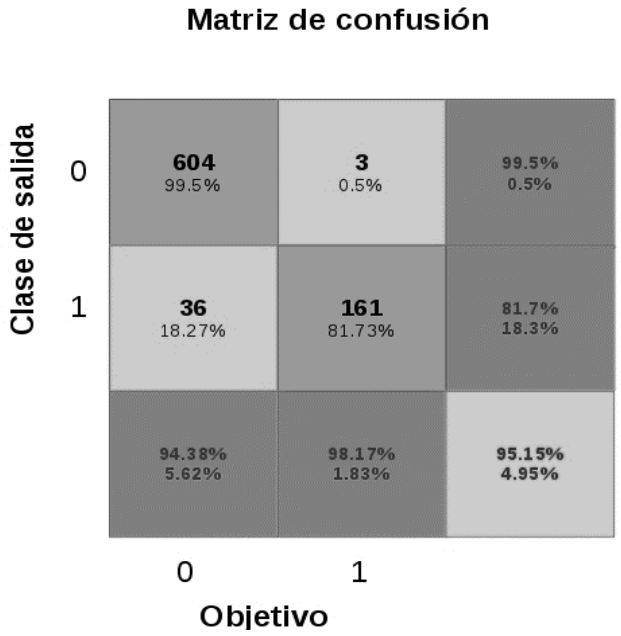

Figura 16: Tercera matriz de confusión. Fuente: Elaboración propia

En un cuarto momento los datos de entrenamientos están compuestos por la partición 1, 3, 4, 5 y la partición 2 fue seleccionada para realizar la validación.

En la Figura 17, las dos primeras celdas de la diagonal muestran el número y porcentaje de clasificaciones correctas por la red entrenada. En la primera 385 estados se clasifican correctamente como 0 o “fracaso”. Esto corresponde al 97.47% del total de 395 estados clasificados como fracaso. Del mismo modo, 377 estados se clasifican correctamente como 1 o “exitosos”. Esto corresponde al 92.18% de todos los estados clasificados como exitosos.

De los estados exitosos, 10, se clasifican incorrectamente como fracasos y esto corresponde a un 2.53% del total de 395 estados clasificados como fracasos. Del mismo modo, 32 de los estados de fracaso se clasifican incorrectamente como exitosos y esto corresponde al 7.82% de todos los datos clasificados como exitosos.

Además de 395 predicciones de fracaso, el 97.5% son correctos y el 2.5% están incorrectos, y de 409 predicciones de éxitos, el 92.2% son correctos y el 7.8% están incorrectos. Por otra parte de 417 casos de fracaso, el 92.32% se predijo correctamente como fracasos y el 7.68% se predice como éxitos. De los 387 casos exitosos, el 97.42% son clasificados correctamente como exitosos y el 2.58% son clasificados como fracasos.

En general, el 94.77% de las predicciones son correctas y el 5.23% son las clasificaciones erróneas.

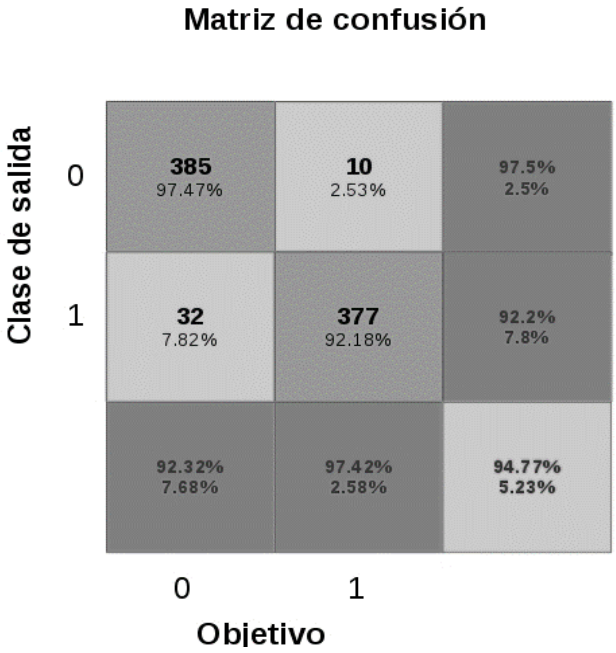

Figura 17: Cuarta matriz de confusión. Fuente: Elaboración propia

Por último los datos de entrenamientos están compuestos por la partición 2, 3, 4, 5 y la partición 1 fue seleccionada para realizar la validación.

En la Figura 18, las dos primeras celdas de la diagonal muestran el número y porcentaje de clasificaciones correctas por la red entrenada. En la primera 379 estados se clasifican correctamente como 0 o “fracaso”. Esto corresponde al 95.95% del total de 395 estados clasificados como fracaso. Del mismo modo, 384 estados se clasifican correctamente como 1 o “exitosos”. Esto corresponde al 93.89% de todos los estados clasificados como exitosos.

De los estados exitosos, 16, se clasifican incorrectamente como fracasos y esto corresponde a un 4.05% del total de 395 estados clasificados como fracasos. Del mismo modo, 25 de los estados de fracaso se clasifican incorrectamente como exitosos y esto corresponde al 6.11% de todos los datos clasificados como exitosos.

Además de 379 predicciones de fracaso, el 95.9% son correctos y el 4.1% están incorrectos, y de 409 predicciones de éxitos, el 93.9% son correctos y el 6.1% están incorrectos. Por otra parte de 404 casos de fracaso, el 93.81% se predijo correctamente como fracasos y el 6.19% se predice como éxitos. De los 400 casos exitosos, el 96% son clasificados correctamente como exitosos y el 4% son clasificados como fracasos.

En general, el 94.9% de las predicciones son correctas y el 5.1% son las clasificaciones erróneas.

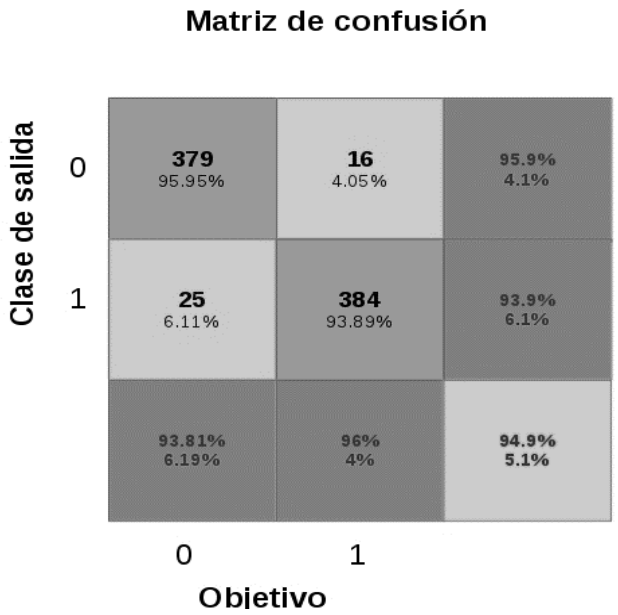

Figura 18: Quinta matriz de confusión. Fuente: Elaboración propia

3.4 Conclusiones parciales

- Los resultados del Cuasi-experimento realizado demostraron que la RNA evolutiva clasifica correctamente.
- La Validación cruzada permitió evaluar la RNA evolutiva obteniéndose como resultado que las predicciones correctas están por encima del 90%.

CONCLUSIONES GENERALES

- La revisión bibliográfica demostró la necesidad de realizar un pronóstico de éxito con vista a la MPS evidenciándose la factibilidad del uso de una RNA evolutiva.
- El diseño de una RNA evolutiva para la clasificación de las organizaciones con vista a la MPS, favoreció su adaptación a las diferentes condiciones respondiendo a las características específicas de cada organización.
- La implementación de un AG para el diseño y entrenamiento de la red permitió la creación de un modelo evolutivo de RNA, y el uso del algoritmo BP refinó la solución obtenida por el AG al realizar la evolución de los pesos de la red.
- Los resultados obtenidos por la Validación cruzada evidenció que la RNA clasifica con una precisión de más del 90% y el Cuasi-experimento permitió corroborar que la RNA realiza la clasificación correctamente.

RECOMENDACIONES

- Utilizar algún algoritmo para la creación de reglas que permitan al usuario conocer con mayor exactitud los motivos por los que alcanzó o no el éxito.
- Realizar experimentos con otros modelos de redes neuronales con el objetivo de verificar si se obtienen mejores resultados.

REFERENCIAS BIBLIOGRÁFICAS

ANDERSON, J. *Redes Neurales*. Primera Edición. Mexico, Alfaomega, 2007. 616 p. 978-970-15-1265-4

ANDRADE TEPÁN, E. C. *Estudio de los principales tipos de redes neuronales y las herramientas para su aplicación*, Universidad Politécnica Salesiana Sede Cuenca 2013. p.

ASHRAFI, N. The impact of software process improvement on quality: in theory and practice *Information and Management*, 2003, 40(6): 677 - 690.

BALLESTEROS, A. J. T. *Nuevos Modelos de Redes Neuronales Artificiales Evolutivas para la Clasificación. Aplicación a Unidades Producto y Unidades Sigmoide.*: Lenguajes y Sistemas Informáticos, Sevilla, 2013. p.

BARCLAY, C.; A. DENNIS*, et al.* Application of the CRiSP-DM Model in Predicting High School Students' examination (CSeC/CXC) Performance *Knowledge Discovery Process and Methods to Enhance Organizational Performance*, 2015: 279.

BARRIOS, D., CARRASCAL, A., MANRIQUE, D., & RÍOS, J. An algebraic model for generating and adapting neural networks by means of optimization methods. *Annals of Mathematics and Artificial Intelligence*, 2001.

--- Cooperative binary-real coded genetic algorithms for generating and adapting artificial neural networks. *Neural Computing and Applications, 12(2)*, 2003.

BASILI, V. R.; F. E. MCGARRY*, et al. Lessons learned from 25 years of process improvement: the rise and fall of the NASA software engineering laboratory*. *Proceedings of the 24th International Conference on Software Engineering*. Orlando, Florida, ACM, 2002. 69-79.

BENÍTEZ, R.; G. ESCUDERO*, et al. Inteligencia artificial avanzada*. Editorial UOC, 2014. p. 8490643210

BERMÚDEZ, M.; A. ALONSO*, et al.* Modelo de redes neuronales optimizadas con algoritmos genéticos. Una aplicación para proyecciones sobre pacientes con cáncer hospitalizados en la CCSS, 2013.

BERTINO, E. A. and L. A. MARTINO Sistemas de bases de datos orientadas a objetos *Ediciones Díaz de Santos*, 1995.

BOOCH, G. *The Unified Modeling Language User Guide*. Pearson Education, 2005. p.

BRAHA, D. *Data mining for design and manufacturing: methods and applications*. Springer Science & Business Media, 2013. p. 1475749112

CASTELLANOS PEÑUELA, A. *Algoritmos para Minería de Datos con Redes de Neuronas*. Facultad de Informática. Madrid. España, Universidad Politécnica de Madrid, 2013. 170. p.

CASTILLO, P. A.; M. G. ARENAS*, et al.* Artificial Neural Networks Design using Evolutionary Algorithms. en: *Advances in Soft Computing*. BENÍTEZ, J.;CORDÓN, O.*et al*, Springer London, 2003. 43-52.p.

CASTILLO, P. A.; J. G. CASTELLANO*, et al.* Diseño de Redes Neuronales Artificiales mediante Algoritmos Evolutivos *Universidad de Granada*, 2005.

CIOS, K. J.; W. PEDRYCZ*, et al. Data mining methods for knowledge discovery*. Springer Science & Business Media, 2012. p. 1461555892

CMMI, I. *CMMI Institute. Clearmodel*, Carnegie Mellon, 2016. [2016]. Disponible en: http://cmmiinstitute.com/

CORRALES, D.; A. LEDESMA*, et al.* A new dataset for coffee rust detection in Colombian crops base on classifiers *Revista S&T*, 2014: 9-23.

CHOO, C. W. *La organización inteligente: el empleo de la información para dar significado, crear conocimiento y tomar decisiones*. México D.F, Oxford University Press, 1999. p.

CHRISTIE, T. *Django REST Framework*, 2015.

DAVID, L. and C. E. P. GOMEZ DATA MINING TO FIND PROFILES OF STUDENTS *European Scientific Journal, ESJ*, 2014, 10(30).

DEB, K. and D. E. GOLDBERG. *An investigation of niche and species formation in genetic function optimization*. Proceedings of the 3rd international conference on genetic algorithms, Morgan Kaufmann Publishers Inc., 1989. 42-50 p. 1558600663

DING, S.; H. LI*, et al.* Evolutionary artificial neural networks: A review *Springer Science*, 2013.

DOUNOS, P. and G. BOHORIS. *Factors for the Design of CMMI-based Software Process Improvement Initiatives*. *Informatics (PCI), 2010 14th Panhellenic Conference on* Tripoli IEEE Xplorer Digital Library, 2010. 43 - 47.

FORRADELLAS, P.; G. PANTALEO*, et al. El modelo CMM/CMMI - Cómo garantizar el éxito del proceso de mejoras en las organizaciones, superando los conflictos y tensiones generados por su implementación*. Universidad CAECE, Av. de Mayo 866, Capítulo Argentino de la IEEE COMPUTER SOCIETY e it-Mentor, 2005. 21.

FOWLER, M. and K. SCOTT. *UML gota a gota*. Pearson Education, 1999. p.

FREEMAN, J. A. and D. M. SKAPURA. *Neural Networks. Algorithms, Applications, and Programming Techniques*. California, Addison Weasley, 1991. 401 p. 0-201-51376-5

GARCIA, R. A. M. *PROCESO PARA PRONOSTICAR EL ÉXITO EN LA MEJORA DE PROCESOS DE SOFTWARE*, Universidad de las Ciencias Informáticas, 2013. p.

---. *Pronóstico de éxito en la Mejora de Procesos de Software*. *Informática 2016*. Habana, 2016.

GARCÍA, V. G. V. *Estimación y clasificación de daños en materiales utilizando modelos AR y redes neuronales para la evaluación no destructiva con ultrasonidos*, 2016. [2016]. Disponible en: http://ceres.ugr.es/~alumnos/esclas/#Cap3

GARRO, B. A.; H. SOSSA*, et al.* Diseño Automático de Redes Neuronales Artificiales mediante el uso del Algoritmo de Evolucion Diferencial., 2012.

GESTAL, M.; D. RIVERO*, et al. Introducción a los algoritmos genéticos y la programación genética*. Universidade da Coruña, Servicio de Publicacións, 2010. p.

GOLDBERG, D. E. and J. H. HOLLAND Genetic algorithms and machine learning *Machine learning*, 1988, 3(2): 95-99.

GONZÁLEZ PEREA, R.; E. CAMACHO POYATO*, et al. Optimización de la predicción de demanda de agua mediante algoritmos neuro-genéticos para un conjunto de datos reducido*. XXXIV Congreso Nacional de Riegos, Sevilla 2016, Escuela Técnica Superior de Ingeniería Agronómica, 2016. p.

GRANITTO, P.; P. VERDES*, et al. ENTRENAMIENTO DE REDES NEURONALES: ANÁLISIS DE MÉTODOS GENERALIZADOS DE MINIMIZACIÓN POR DECENSO SEGÚN EL GRADIENTE*. ANALES AFA, 2013. p. 1850-1168

GRAUPE, D. *Principles of Artificial Neural Network*. Segunda Edición. Singapore, World Scientific, 2007. p. 978-981-270-624-9

GUPTA, G. *Introduction to data mining with case studies*. PHI Learning Pvt. Ltd., 2014. p. 8120350022

GUTIÉRREZ, G.; A. SANCHIS*, et al.* Non-direct encoding method based on Cellular Automata to Design Neural Network Architectures *Computing and Informatics*, 2005, 24(3): 23.

HAGAN, M.; H. DEMUTH*, et al. Neural Network Design*. Oklahoma, Estados Unidos, 2014.

HEBB, D. The Organization of Behaviour *Wiley*, 1949.

HONG, H.; Z. HENGYING*, et al.* APPLICATION OF PCA-LVQ NEURAL NETWORK MODEL IN LITHOLOGY IDENTIFICATION BY LOGGING DATA IN PANZHUANG AREA *Pakistan Journal of Statistics*, 2014, 30 (6).

IBM. *Manual CRISP-DM de IBM SPSSModeler*. Estados Unidos, IBM Corp, 2012.

ISO. *ISO 9000: 2005. Quality management systems.*, ISO, 2005a.

---. *ISO. International Organization for Standardization*, ISO, 2016. [2016]. Disponible en: http://www.iso.org/iso/home.htm

---. *ISO/IEC 25000:2005. Software Engineering -- Software product Quality Requirements and Evaluation (SQuaRE)*, ISO, 2005b.

JAISWAL, S. and R. KUMAR. *Learning Django Web Development*. Packt Publishing Ltd, 2015. p. 1783984414

KANDEL, E. R. and R. D. HAWKINS Bases biolbgicas del aprendizaje y de la individualidad *Investigacion y Ciencia*, 1992: 58.

KDnuggets: Data Mining Community Top Resource. 2016. [Disponible en: http://www.kdnuggets.com/

KOHONEN, T.; J. HYNNINEN*, et al. LVQ PAK: The Learning Vector Quantization Program Package*. Helsinki, Finlandia, Universidad de Tecnología de Helsinki, 1996. 28.

KONAR, A. Artificial intelligence and soft computing. *Boca Raton, FL: CRC Press*, 2000.

LANZARINI, L. C.; W. HASPERUÉ*, et al. Redes neuronales artificiales*. XVII Workshop de Investigadores en Ciencias de la Computación (Salta, 2015), 2015. p.

LUNA, E. D. J. R. ENTRENAMIENTO EVOLUTIVO DE REDES NEURONALES DINÁMICAS PARA LA DETECCIÓN DE APNEA-BRADICARDIA EN NEONATOS PREMATUROS, 2013.

MANRIQUE, D. Neural networks design and new optimization methods by genetic algorithms *Ph.D. thesis, Universidad Politécnica de Madrid, Spain.*, 2001.

MANRIQUE GAMO, D. *Computación Evolutiva:Algoritmos Genéticos*. Madrid, Universidad de Madrid, 2006.

MARÍN, F. J. S., FRAMCISCO Diseño de Redes Neuronales Artificiales mediante Algoritmos Genéticos, 1995.

MARTIN DEL BRIO, B. and A. SANZ MOLINA. *Redes Neuronales y Sistemas Borrosos*. Segunda Edición. Editorial RA-MA, 2001. 416 p. 9788478974665

MINSKY, M. and S. PAPERT. *Perceptrons: An Introduction to Computational Geometry*. Cambridge, Reino Unido, MIT Press, 1969. p. 0-262-63022-2

MITCHELL, T. M. *Machine Learning*. New York, USA, 1997. p. 0-07-042807-7

MOINE, J. M. *Metodologías para el descubrimiento de conocimiento en bases de datos: un estudio comparativo*, Facultad de Informática, 2013. p.

MONTONI, M. A. and A. R. ROCHA. *Applying Grounded Theory to Understand Software Process Improvement Implementation*. *Proceedings of the 2010 Seventh International Conference on the Quality of Information and Communications Technology*, IEEE Computer Society, 2010. 25-34.

NAOUM, R. S. and Z. N. AL-SULTANI Learning Vector Quantization (LVQ) and k-Nearest Neighbor for Intrusion Classification *World of Computer Science and Information Technology Journal*, 2012, 2.

NIAKSU, O. CRISP Data Mining Methodology Extension for Medical Domain *Baltic Journal of Modern Computing*, 2015, 3(2): 92.

NIAZI, M.; M. A. BABAR*, et al.* Software Process Improvement barriers: A cross-cultural comparison *Information and Software Technology*, 2010, 52(11): 1204-1216.

NIAZI, M.; D. WILSON*, et al.* Critical success factors for software process improvement implementation: an empirical study *Software Process: Improvement and Practice*, 2006, 11(2): 193-211.

OLIPHANT, T. *NumPy: Open Source Scientific Tools for Python. Version 1.7*, 2014.

OUYANG, A.; K. LI1*, et al.* Improved LDA and LVQ for Face Recognition *Applied Mathematics & Information Sciences*, 2014.

OVIEDO CARRASCAL, E. A.; A. I. OVIEDO CARRASCAL*, et al.* MINERÍA DE DATOS: APORTES Y TENDENCIAS EN ELSERVICIO DE SALUD DE CIUDADES INTELIGENTES *Revista Politécnica*, 2015, 11: 111-120.

PARADIGM, V. Visual paradigm for uml *Visual Paradigm for UML-UML tool for software application development*, 2013.

PERDOMO, J. G. Un nuevo enfoque para la resolución de problemas: redes neuronales *Revista EAN*, 2015, (24): 35-40.

PINO, F. J.; F. GARCIA*, et al.* Software process improvement in small and medium software enterprises: a systematic review *Software Quality Journal*, 2008, 16(2): 237-261.

PMI. *A Guide to the Project Management Body of Knowledge*, 5th. Project Management Institute, 2013. [2013]. Disponible en: http://www.pmi.org/PMBOK-Guide-and-Standards.aspx

---. *PMI. Project Management Institute.*, 5. Project Management Institute, Inc., 2016. [2016]. Disponible en: http://www.pmi.org/PMBOK-Guide-and-Standards.aspx

POSE, M. G. Introducción a Algoritmo Genéticos *Coruña: Depto.Tecnología de la Infromación y las Comunicaciones*, 2010.

POSTGRESQL. *PostgreSQL*, 2016. [2016]. Disponible en: http://www.PostgreSQL.org/about

PRESSMAN, R. S. *Ingeniería del software: un enfoque práctico*. 5ta. Mikel Angoar, 2002. p.

PYTHON. *Python*, 2016. [2016]. Disponible en: https://www.python.org/about/

RABUÑAL, J. and J. DORADO. *Artificial neural-networks in real-life applications*. EEUU, Idea Group Publishing, 2006.

ROJAS QUINCHO, J. P. *REDES NEURONALES ARTIFICIALES PARA LA PREDICCIÓN DE LA CONCENTRACIÓN DE OZONO TROPOSFÉRICO EN EL DISTRITO DE ATE-LIMA.*, 2013. p.

ROSENBLATT, F. The Perceptron: A Probabilistic Model for Information Storage & Organization in the Brain *Psychological Review*, 1958, 65(6): 23.

RUANGCHAI, W. *Identification of protein complexes using machine learning (PyBrain and Scikit-Learn) based on DNA sequence data*. TEXAS A&M UNIVERSITY-COMMERCE, 2014. p. 1321440227

RUMELHART, D. E.; H. G.E.*, et al.* Learning representations by backpropagating errors, 1986.

RUSSELL, S. and P. NORVIG Artificial Intelligence: A Modern Approach., 2003.

SAATY, T. L. *Toma de decisiones para líderes*. RWS Publications, 2014. p. 1888603291

SÁNCHEZ, S. E. T.; M. O. RODRÍGUEZ*, et al.* Implementación de Algoritmos de Inteligencia Artificial para el Entrenamiento de Redes Neuronales de Segunda Generación *JÓVENES EN LA CIENCIA*, 2016, 2(1): 6-10.

SOMMERVILLE, I. *Software Engineering*. 8va. Addison- Wesley, 2007. p.

STANDISH-GROUP. *Chaos Manifiesto 2015*. *The Standish Group*, The Standish Group International, Inc, 2015.

STEVENS, P. and R. POOLEY. *Utilización de UML en Ingeniería del Software con Objetos y Componentes*. 2. Addison-Wesley Publishing Company, 2002. p.

TORRES, D. *Diseño y aplicación de una metodología para análisis de noticias policiales utilizando minería de textos*. Chile, Universidad de Chile, 2013. p.

TRUJILLO CASAÑOLA, Y.; A. FEBLES ESTRADA*, et al. Modelo para valorar las organizaciones previo a la mejora de procesos de software*. *VI Taller de Calidad en las Tecnologías de la Información y las Comunicaciones*. La Habana, Informática 2013, 2013. 10.

TRUJILLO , Y.; A. FEBLES *, et al.* Diagnóstico al iniciar la mejora de proceso de software *Ingeniería Industrial*, 2014, 35(2): 172-183.

TRUJILLO, Y.; A. FEBLES*, et al.* Diagnóstico al iniciar la mejora de proceso de software *Ingeniería Industrial*, 2014, 35(2): 172-183.

VILLADA, F.; D. R. CADAVID*, et al.* Pronóstico del precio de la energía eléctrica usando redes neuronales artificiales *Revista facultad de ingeniería*, 2014, (44): 111-118.

WIBOWO, A.; M. I. DESA*, et al.* Data Cleaning in Knowledge Discovery Database (KDD)-Data Mining, 2014.

XABIER BASOGAIN OLABE Redes Neuronales Artificiales y sus Aplicaciones, 2004.

YARUSHKINA, N.; T. AFANASIEVA*, et al.* Fuzzy Trends Data Mining in Knowledge Discovery Process. en: *Creativity in Intelligent Technologies and Data Science*. Springer, 2015. 115-123.p.

ZAHRAN, S. *Software Process Improvement: Practical Guidelines for Business Success*. 1. ADDISON WESLEY Publishing Company Incorporated, 1998. 447 p. 978-0201177824

ANEXOS

Anexo 1

Tabla 6: Descripción de la tabla mejora. Fuente: Elaboración propia

Nombre: mejora		
Descripción: almacena todas las mejoras de la base de datos		
Atributo	**Tipo**	**Descripción**
id	integer	Identificador
descripcion	String	Descripción de la mejora en cuestión
nombre	String	Nombre literal de la mejora

Anexo 2

Tabla 7: Descripción de la tabla pais. Fuente: Elaboración propia

Nombre: pais		
Descripción: almacena todos los paises de la base de datos		
Atributo	**Tipo**	**Descripción**
id	integer	Identificador
nombre	String	Nombre literal del país

Anexo 3

Tabla 8: Descripción de la tabla organismo. Fuente: Elaboración propia

Nombre: organismo		
Descripción: almacena todos los organismos de la base de datos		
Atributo	**Tipo**	**Descripción**
id	integer	Identificador
nombre	String	Nombre literal del organismo
descripcion	String	Descripción del organismo en cuestión

Anexo 4

Tabla 9: Descripción de la tabla indicador. Fuente: Elaboración propia

Nombre: indicador		
Descripción: almacena todos los indicadores de la base de datos		
Atributo	**Tipo**	**Descripción**
id	integer	Identificador
nombre	String	Nombre literal del indicador
activo	Boolean	Toma el valor *true* si está activo el indicador y *false* en el caso contrario

Anexo 5

Tabla 10: Descripción de la tabla organizacion. Fuente: Elaboración propia

Nombre: organizacion		
Descripción: almacena todas las organizaciones de la base de datos		
Atributo	**Tipo**	**Descripción**
id	integer	Identificador
nombre	String	Nombre literal de la organización
descripcion	String	Descripción de la organización en cuestión
pais_id	integer	Llave foránea que hace referencia al identificador de la *tabla pais*
organismo_id	integer	Llave foránea que hace referencia al identificador de la *tabla organismo*

Anexo 6

Tabla 11: Descripción de la tabla medida. Fuente: Elaboración propia

Nombre: medida		
Descripción: almacena todas las medidas de la base de datos		
Atributo	**Tipo**	**Descripción**
id	integer	Identificador
nombre	String	Nombre literal de la medida
factor_id	integer	Llave foránea que hace referencia al identificador de la *tabla factor*
activo	Boolean	Toma el valor *true* si está activa la medida y *false* en el caso contrario

Anexo 7

Tabla 12: Descripción de la tabla medida_escenario. Fuente: Elaboración propia

Nombre: medida_escenario		
Descripción: almacena el valor de la medida asociada a un escenario		
Atributo	**Tipo**	**Descripción**
medida_id	integer	Llave foránea que hace referencia al identificador de la tabla medida
escenario_id	integer	Llave foránea que hace referencia al identificador de la tabla escenario
valor	integer	Valor correspondiente a una medida