DEFINICION DE UN MODELO DE EVALUACIÓN DE RIESGOS EN SEGURIDAD DE LA INFORMACIÓN BAJO LOS LINEAMIENTOS DE LA NORMA ISO 27001, UTILIZANDO TÉCNICAS DE REDES NEURONALES

JULIÁN ANDRÉS ARIAS LEÓN
JUAN GUILLERMO RUÍZ CORREA

UNIVERSIDAD TECNÓLOGICA DE PEREIRA
FACULTAD DE INGENIERÍAS: ELÉCTRICA, ELECTRÓNICA, FISICA Y CIENCIAS DE LA COMPUTACIÓN
MAESTRÍA EN INGENIERÍA DE SISTEMAS Y COMPUTACIÓN
PEREIRA
MARZO 2019

DEFINICION DE UN MODELO DE EVALUACIÓN DE RIESGOS EN SEGURIDAD DE LA INFORMACIÓN BAJO LOS LINEAMIENTOS DE LA NORMA ISO 27001, UTILIZANDO TÉCNICAS DE REDES NEURONALES

JULIÁN ANDRÉS ARIAS LEÓN
JUAN GUILLERMO RUÍZ CORREA

TRABAJO DE INVESTIGACIÓN PRESENTADO COMO REQUISITO PARA OPTAR AL TÍTULO DE: MAGISTER EN INGENIERÍA DE SISTEMAS Y COMPUTACIÓN

DIRECTOR: ING. JORGE IVÁN RÍOS PATIÑO

UNIVERSIDAD TECNÓLOGICA DE PEREIRA
FACULTAD DE INGENIERÍAS: ELÉCTRICA, ELECTRÓNICA, FISICA Y CIENCIAS DE LA COMPUTACIÓN
MESTRÍA EN INGENIERÍA DE SISTEMAS Y COMPUTACIÓN
PEREIRA
MARZO 2019

NOTA DE ACEPTACIÓN

_____

_____

_____

_____

_____

_____

_____

FIRMA DIRECTOR

_____

FIRMA JURADO

_____

FIRMA JURADO

Pereira, Risaralda, marzo 2019

Agradecimientos

Son varias las personas que han contribuido al proceso y conclusión de este trabajo de grado. En primer lugar, a Dios por darnos la oportunidad de continuar creciendo profesional y académicamente. Queremos agradecer al Ing. Jorge Iván Ríos Patiño, director de este trabajo y quien, con su experiencia y paciencia, nos brindó aportes clave para darle estructura al desarrollo y conclusión de este. Al Ing. Julio Cesar Chavarro Porras, quien nos asesoró hábilmente a enfocar la idea que teníamos inicialmente para convertirla luego en el anteproyecto. A nuestros familiares por estar pendientes de este proceso y alentarnos a culminarlo y a nuestros amigos quienes aportaron ideas y brindaron apoyo constante para cumplir esta meta.

Tabla de contenido

Índice de figuras

Índice de tablas

Índice de anexos

Resumen

Proteger la información de una organización es un problema donde no basta con adquirir equipos y tecnología, es un proceso de mejora continua donde no solo intervienen activos de información sino también aquellos expertos encargados de realizar la evaluación del riesgo en el ecosistema de información empresarial.

Para definir un sistema de gestión del riesgo en seguridad de la información, es preciso definir reglas para la gestión y evaluación de estos, de tal manera que sean utilizadas por toda la organización en la misma forma. Este suele ser un problema común a muchas organizaciones, pues depende exclusivamente de los expertos en el área. Teniendo en cuenta lo anterior, se hace necesario establecer desde el comienzo el tipo de evaluación, la escala de medición que se ha de utilizar y los niveles aceptables de riesgo, ya que no todos estos, tienen la misma importancia o la misma probabilidad de ocurrencia. Pero también se debe tener claro que algunos de los riesgos que se identifican muestran niveles inaceptables que hay que saber gestionar cuando se materialicen.

En el proceso de evaluación de los riesgos en seguridad de la información suelen existir juicios ambiguos por parte de los expertos encargados de asignar valores a los niveles de probabilidad de ocurrencia e impacto esperado en los riesgos y posiblemente estas divergencias solo aporten información imprecisa al analizar los resultados, de este modo, es que las técnicas de inteligencia artificial como la lógica difusa y las redes neuronales pueden ser muy útiles en este ámbito.

Es así como en el presente trabajo se implementaron modelos basados en lógica difusa y redes neuronales para establecer un análisis comparativo ante los mismos escenarios de prueba en ambos modelos, donde dicho análisis permite convertirse en una herramienta para apoyar el plan de continuidad del negocio y apoyo al proceso de toma de decisiones ante la materialización de amenazas inminentes.

Abstract

Protecting the information of an organization is a business problem in which the solution is much more than acquiring equipment and technology from third parties, it is a continuous improvement process in which not only information assets are involved but also those experts in charge of carrying out the evaluation of risk in the business information ecosystem.

To define a risk management system in information security, it is necessary to define rules for the management and evaluation of these, in such a way that they are used by the entire organization in the same way. This is usually a problem common to many organizations, since it depends exclusively on the experts in the area. Taking into account the above, it is necessary to establish from the beginning the type of evaluation, the scale of measurement to be used and the acceptable levels of risk, since not all risks have the same importance or the same probability of occurrence. But it must also be clear that some of the risks that are identified show unacceptable levels that must be managed when they materialize.

In the process of assessing information security risks, there are usually ambiguous judgments by the experts in charge of assigning values to the levels of probability of occurrence and expected impact on the risks and possibly these divergences only provide inaccurate information when analyzing the results, in this way, is that artificial intelligence techniques such as fuzzy logic and neural networks can be very useful in this field.

This is how in the present work models based on fuzzy logic and neural networks were implemented to establish a comparative analysis before the same test scenarios in both models, where such analysis allows to become a tool to support the business continuity plan and support to the decision-making process before the materialization of imminent threats.

1 Generalidades

1.1 Introducción

La medición de los riesgos en general permite emprender acciones ante la posibilidad de materialización de estos, siendo importante su evaluación al momento de priorizar recursos y esfuerzos orientados a mitigar, controlar, transferir o asumirlos.

La norma ISO 31000[1] es la norma técnica que brinda los principios y las directrices genéricas sobre la gestión del riesgo. En uno de sus apartados, se enuncia la evaluación del riesgo, el cual define como: "Proceso de comparación de los resultados del análisis del riesgo con los criterios del riesgo, para determinar si este, su magnitud o ambos son aceptables o tolerables". (ICONTEC, 2011a) .

Para realizar la comparación de resultados mencionada en la definición anterior, primero debe existir una medición del riesgo como tal. Esta medición, puede variar de experto a experto, ya que cada uno puede valorar de diferentes maneras un riesgo, lo que puede conllevar a que estos no se traten de una manera adecuada o incluso que no se tomen correcta y oportunamente las decisiones ante la materialización de una amenaza. Dicha valoración puede ser semi automatizada usando técnicas en el campo de la inteligencia artificial, generando la materia prima que usará el experto en el ciclo de vida del riesgo[2].

En el presente trabajo se encontrará con la implementación de dos modelos de evaluación de riesgos en seguridad de la información, donde cada uno usa técnicas de inteligencia artificial tales como: la lógica difusa y las redes neuronales artificiales respectivamente, y en cuyo análisis de resultados se obtuvo evidencia que al usarlos, se convierten en una herramienta de ayuda a los expertos que emiten los juicios sobre la evaluación de dichos riesgos y facilitan la toma de decisiones ante la materialización de amenazas inminentes.

[1] La Norma ISO 31000 es una norma internacional que ofrece las directrices y principios para gestionar el riesgo de las organizaciones

[2] Proceso por el cual pasa un riesgo desde que se identifica, se analiza, se evalúa y se trata.

1.2 Formulación del problema

La evaluación de riesgos en seguridad de la información admite valores que no son precisos, esto es, admite valores que se establecen de acuerdo con una escala cuantitativa por rangos, por ejemplo, de 1 a 5, o una escala cualitativa, por ejemplo, con valores del tipo: bajo, medio y alto. En este tipo de medición es muy importante que exista un criterio homogéneo de valoración que permita comparar entre activos de información. Dicha medición, no sigue un modelo simple o un modelo matemático de precisión, si no que se basa en el conocimiento de un experto, el cual utiliza conceptos imprecisos o ambiguos para realizarla. Esto puede hacer que no haya una medición confiable y que a partir de dicha medición se tomen decisiones que no mitiguen, controlen o eviten a satisfacción el impacto generado por la materialización de una amenaza.

Es por esto, que el presente proyecto de grado planteó la definición de un modelo de medición de riesgos basado en redes neuronales y el análisis del modelo basado en lógica difusa planteado en (Angarita Vivas & Tabares Isaza, 2015) , con el fin de realizar un análisis comparativo de los resultados generados por ambas soluciones, las cuales permitirán a los expertos contar con una herramienta en la que se puedan plasmar sus percepciones sobre los niveles de riesgo y escalas de valoración de una forma más precisa y así obtener unos resultados que faciliten el tratamiento y la toma de decisiones frente a los mismos

1.3 Justificación

La gestión y medición del riesgo en seguridad de la información en cualquier proyecto, ha de ser una preocupación constante a nivel de toda la organización.
La implementación de la propuesta no pretende reemplazar la valoración hecha por los expertos, por el contrario, lo que busca es dotar a dichos expertos de una herramienta que construya sobre la experiencia de quienes conocen el tema a evaluar y se aproveche dicha experiencia para poder realizar una valoración más precisa, que reduzca notablemente la subjetividad y proporcione resultados más eficientes que se traduzcan en mayores beneficios y optimización de los recursos para la organización.

El valor agregado que brinda la presente propuesta a los expertos al utilizar los aplicativos generados de las implementaciones de los modelos con lógica difusa y redes neuronales, es que podrán tener a disposición un mecanismo que les permita analizar los datos obtenidos y valorar de una manera más coherente los niveles de riegos a los que se vean expuestos ante determinados escenarios posibles y posteriormente tomar decisiones acordes a dichos resultados. Además, con el modelo basado en redes neuronales artificiales, se puede garantizar que después del entrenamiento de la red, se asegura que ante cualquier escenario propuesto y cuyos valores de entrada ingresen al sistema, existirá una valoración dada a la variable de salida, producto del aprendizaje obtenido por la red y no dependerá exclusivamente de la base de conocimiento dado inicialmente.

En términos generales, el uso de estas técnicas en la evaluación de riesgos en seguridad de la información permite robustecer el modelo, incluir otros parámetros que los expertos vean de interés o eliminar restricciones del sistema.

1.4 Objetivo

1.4.1 Objetivo General

Evaluar los riesgos de seguridad de la información bajo los lineamientos de la norma ISO 27001, utilizando técnicas de redes neuronales

1.4.2 Objetivos Específicos

- Diseñar un modelo de medición de riesgos de seguridad de la información basado en redes neuronales artificiales
- Implementar el modelo de medición de riesgos de seguridad de la información basado en redes neuronales artificiales.
- Implementar el modelo de medición de riesgos de seguridad de la información basado en lógica difusa planteado en (Angarita Vivas & Tabares Isaza, 2015).
- Realizar un análisis comparativo de los resultados generados por los modelos implementados.

1.5 Alcance del proyecto

El presente trabajo tiene como alcance la implementación de los modelos de evaluación del riesgo en seguridad de la información, cuyos resultados son objeto del análisis comparativo. En la implementación del modelo con lógica difusa, tomando la definición hecha por los autores (Angarita Vivas & Tabares Isaza, 2015), se usó como función de membresía o pertenencia una función triangular, la cual es la más usada para definir conjuntos difusos y es muy adecuada para definir situaciones en las que se tiene un valor óptimo central. Para el modelo basado en redes neuronales artificiales, se usó el algoritmo base de entrenamiento de redes llamado: el algoritmo de retropropagación, mediante el cual se van adaptando todos los parámetros de la red.

Otros tipos de funciones de membresía para el modelo basado en lógica difusa y otros tipos de funciones de activación o de algoritmos de entrenamiento para el modelo de redes neuronales, pueden ser objeto de un trabajo futuro.

1.6 Metodología de la investigación

1.6.1 Enfoque Metodológico

El tipo de investigación que se aplicó en el presente trabajo fue cuantitativa y analítica, ya que se basó en datos numéricos sobre el trabajo de investigación realizado por Angarita y Tabares (Angarita Vivas & Tabares Isaza, 2015), los cuales se tomaron como referencia para realizar el proceso de análisis y comparación de resultados brindados por el modelo basado en lógica difusa y el modelo basado en redes neuronales respectivamente, una vez implementados y aplicados los mismos escenarios de prueba en cada uno de ellos.

1.6.2 Etapas del proyecto

Para el desarrollo y alcance de objetivos del proyecto, se planearon una serie de pasos que reflejan la secuencia del trabajo de investigación de la siguiente manera:

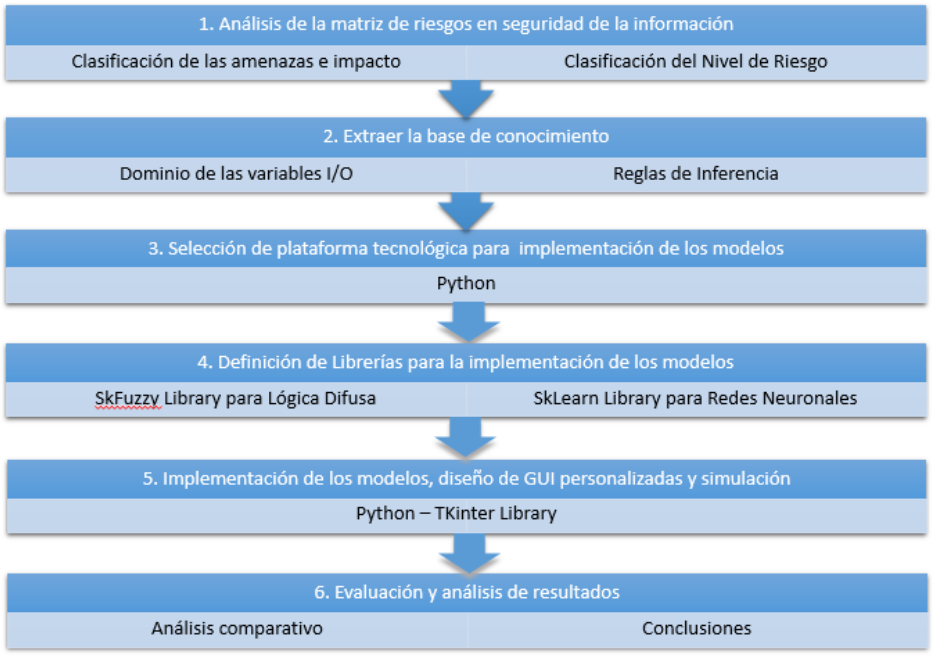

Figura 1. Etapas del proyecto

2 Marco de Referencia

2.1 Antecedentes

El riesgo de origen tecnológico puede incidir sobre las metas y objetivos organizacionales y ser causa de otro tipo de riesgos al estar directamente relacionado con el uso de tecnología. Por ello el daño, interrupción, alteración o falla derivada del uso de TI puede implicar pérdidas significativas en las organizaciones, pérdidas financieras, multas o acciones legales, afectación de la imagen de una organización y causar inconvenientes a nivel operativo y estratégico.

Dentro del contexto de los modelos de evaluación de riesgos en seguridad de la información, se realizó un trabajo de grado de la Maestría en Ingeniería de Sistemas y Computación de la Universidad Tecnológica de Pereira llamada: "DEFINICIÓN DE UN MODELO DE MEDICIÓN DE RIESGOS DE LA SEGURIDAD DE LA INFORMACIÓN APLICANDO LÓGICA DIFUSA Y SISTEMAS BASADOS EN EL CONOCIMIENTO. CASO DE ESTUDIO: EMPRESA DE SERVICIOS PUBLICOS" (Angarita Vivas & Tabares Isaza, 2015), en la cual los autores establecen un modelo basado en la teoría de la lógica difusa para proveer a los expertos de la organización de una herramienta que sirviera de apoyo en la toma de decisiones teniendo en cuenta sus criterios subjetivos.

El modelo parte de la tipificación de amenazas que se pueden presentar, se procede a valorar la probabilidad y el impacto en caso de que se materialicen, donde dichas valoraciones son realizadas por los expertos de la organización. Con estos datos, se construyó la materia prima del proyecto conocida como: la matriz de riesgos y con base en esta, los expertos construyen la base de reglas de inferencia del sistema.

El modelo simula ciertos escenarios de entrada, y con estos, se activan ciertas reglas de la base de conocimiento teniendo en cuenta su implicación difusa. A cada regla activada, se le aplican algunas operaciones que provienen de la combinación de los conjuntos difusos de las diferentes variables de entrada y posterior a todo este proceso se arroja un resultado difuso (variable de salida) que indica la valoración del riesgo.

2.2 Marco Conceptual

En este apartado se definirán los conceptos más importantes que tienen relación con los temas de la presente propuesta de trabajo de grado, como lo son: el análisis y la evaluación de riesgos en seguridad de la información, las redes neuronales artificiales y la lógica difusa.

2.2.1 Conceptos del análisis y evaluación de riesgos

Tomando los conceptos citados por los autores en el portal de libre difusión en español de la serie de normas ISO 27000 y de los sistemas de gestión de seguridad de la información (Neira & Spohr, 2012) y por la norma NTC ISO 31000 (ICONTEC, 2011b) , definen:

- Activos de información: Se refiere a cualquier información o elemento relacionado con el tratamiento de esta (sistemas, soportes, edificios, personas...) que tenga valor para la organización. También se definen como los elementos o componentes del sistema de información que deben ser protegidos con el fin de evitar que como resultado de la materialización de una amenaza sufran algún tipo de daño.
- Amenaza: Causa potencial de un incidente no deseado, que puede provocar daños a un sistema o a la organización.
- Análisis de riesgo: Proceso para comprender la naturaleza del riesgo y determinar el nivel de riesgo. Es el proceso dirigido a determinar la probabilidad de que las amenazas se materialicen sobre los bienes informáticos e implica la identificación de los bienes a proteger, las amenazas que actúan sobre ellos, su probabilidad de ocurrencia y el impacto que puedan causar.
- Evaluación del riesgo: Proceso de comparación de los resultados del análisis del riesgo con los criterios del riesgo para determinar si el riesgo, su magnitud, o ambos son aceptables o tolerables.
- Evento: Ocurrencia o cambio y un conjunto particular de circunstancias
- Impacto: El coste para la empresa de un incidente -de la escala que sea-, que puede o no ser medido en términos estrictamente financieros, por ejemplo: pérdida de reputación, implicaciones legales, etc. Es el daño producido por la materialización de una amenaza.

- Matriz de Riesgo: Herramienta para clasificar y visualizar el riesgo mediante la definición de rangos para la consecuencia y la probabilidad.
- Nivel de Riesgo: Magnitud de un riesgo o de una combinación de riesgos, expresada en términos de la combinación de las consecuencias y su posibilidad.
- Probabilidad: Medida de la oportunidad de la ocurrencia, expresada como un número entre 0 y 1, en donde 0 es la imposibilidad y 1 es la certeza absoluta.
- Riesgo: Posibilidad de que una amenaza concreta pueda explotar una vulnerabilidad para causar una pérdida o daño en un activo de información. Suele considerarse como una combinación de la probabilidad de un evento y sus consecuencias.
- Riesgo residual: Es el riesgo remanente después de aplicados controles de seguridad para minimizarlo.
- Seguridad: Es usado en el sentido de minimizar los riesgos a que están sometidos los activos de información hasta llevarlos a niveles adecuados.
- Sistema de información: Es el conjunto de activos de información de los cuales dispone una entidad u organización para su correcto funcionamiento y la consecución de los objetivos.
- Vulnerabilidad: En un sistema de información es un punto o aspecto susceptible de ser atacado o de dañar su seguridad; representan las debilidades o aspectos falibles o atacables en el sistema de información y califican el nivel de riesgo de este.

2.2.2 Conceptos de lógica difusa

- Variables Lingüísticas: Variables cuyos posibles valores son palabras y pueden ser representadas mediante conjuntos difusos.
- Reglas Difusas: Conjunto de proposiciones SI-ENTONCES que modelan el problema que se quiere resolver. Permiten expresar la condición de una determinada variable y la relación entre variables de diferentes conjuntos.
- Fuzzyficador: Convierte las entradas del sistema, que son valores numéricos nítidos en conjuntos borrosos aplicando una función de Fuzzyficación.
- Base de conocimiento (Reglas Difusas): Almacena las reglas SI-ENTONCES obtenidas de los expertos.

- Motor de inferencia: Simula el razonamiento humano haciendo inferencia sobre las entradas recibidas y las reglas SI-ENTONCES almacenadas.
- Desfuzzyficador: Convierte el conjunto borroso obtenido por el motor de inferencia en un valor numérico nítido que puede ser reutilizado.

2.2.3 Conceptos de Redes Neuronales Artificiales

Tomando de (Antonio J Serrano, Emilio Soria, 2010) los conceptos fundamentales de las redes neuronales:

- Función de activación: La función de activación define la salida de un nodo dado un conjunto de entradas. Es la representación básica a cómo funciona el perceptrón en una red neuronal artificial. Esta función se puede decir que es binaria, es decir que el resultado final debe ser de activación o no y para producir esto, se utiliza la función escalón.
- Perceptrón: Es una neurona básica que inspiró la red de inteligencia artificial perceptrón, básicamente permite separar elementos en un plano en grupos categorizados como deseados y no deseados. Estas neuronas pueden asociarse para poder solucionar problemas más complejos.
- Aprendizaje Adaptativo: Capacidad de aprender a realizar tareas basadas en un entrenamiento o en una experiencia inicial.
- Aprendizaje supervisado: El aprendizaje supervisado se caracteriza porque el proceso de aprendizaje se realiza mediante un entrenamiento controlado por un agente externo (supervisor, maestro) que determina la respuesta que debería generar la red a partir de una entrada determinada. El supervisor controla la salida de la red y en caso de que ésta no coincida con la deseada, se procederá a modificar los pesos de las conexiones, con el fin de conseguir que la salida obtenida se aproxime a la deseada.
- Aprendizaje no supervisado: Las redes con aprendizaje no supervisado (también conocido como autosupervisado) no requieren influencia externa para ajustar los pesos de las conexiones entre sus neuronas. La red no recibe ninguna información por parte del entorno que le indique si la salida generada en respuesta a una determinada entrada es o no correcta.
- Algoritmo backpropagation: Se considera una etapa de funcionamiento donde se presenta, ante la red entrenada, un patrón de entrada y éste se transmite a través de las sucesivas capas de

neuronas hasta obtener una salida y posteriormente una etapa de entrenamiento o aprendizaje donde se modifican los pesos de la red de manera que coincida la salida deseada por el usuario con la salida obtenida por la red.

3 Marco Teórico

Dentro del presente trabajo de grado se abordaron varios temas y conceptos que fueron el pilar fundamental para el desarrollo de esta, los temas de estudio en los que se basa el trabajo son: La gestión de riesgos y su proceso de evaluación, por otra parte, algunas ramas de la inteligencia artificial como las redes neuronales y la lógica difusa.

El aumento de la complejidad de las empresas y la necesidad de adaptarse a un entorno cada vez más amplio y variado, junto con otros factores, han hecho que las organizaciones estén más expuestas a los riesgos y que estos, en muchos casos, les afecten en mayor medida. A partir de esto, existe una norma que permite realizar una eficaz gestión de los riesgos, la norma ISO 31000 la cual fue adoptada sin cambios en la Norma Técnica Colombiana ISO 31000.

3.1 La norma ISO 31000

La norma ISO 31000 es una norma internacional que ofrece las directrices y principios para gestionar el riesgo de las organizaciones. Esta norma fue publicada en noviembre del 2009 por la Organización Internacional de Normalización (ISO)[3] la cual tiene por objetivo que organizaciones de todos los tipos y tamaños puedan gestionar los riesgos en la empresa de forma efectiva, por lo que recomienda que las organizaciones desarrollen, implanten y mejoren continuamente un marco de trabajo cuyo objetivo es integrar el proceso de gestión de riesgos en cada una de sus actividades.

La ISO 31000 permite a las organizaciones:

- Fomentar una gestión proactiva libre de riesgo.
- Mejorar la identificación de oportunidades y amenazas.
- Cumplir con las exigencias legales y reglamentarias, además de las normas internacionales.
- Aumentar la seguridad y confianza y mejorar la prevención de pérdidas y manejo de incidentes.
- Mejorar el aprendizaje organizacional.
- Mejorar la eficiencia y eficacia operacional.

[3] Sigla de la expresión inglesa International Organization for Standardization, 'Organización Internacional de Estandarización', sistema de normalización internacional para productos de áreas diversas.

Buscar que el riesgo se gestione de manera eficaz en cualquier organización, es una práctica que se ha desarrollado a lo largo del tiempo en todas las compañías, con procesos coherentes que permiten un manejo integral de los Sistemas de Gestión. La ISO 31000, es una norma orientada a cualquier organización, independientemente del tamaño o sector. (ICONTEC, 2011b)

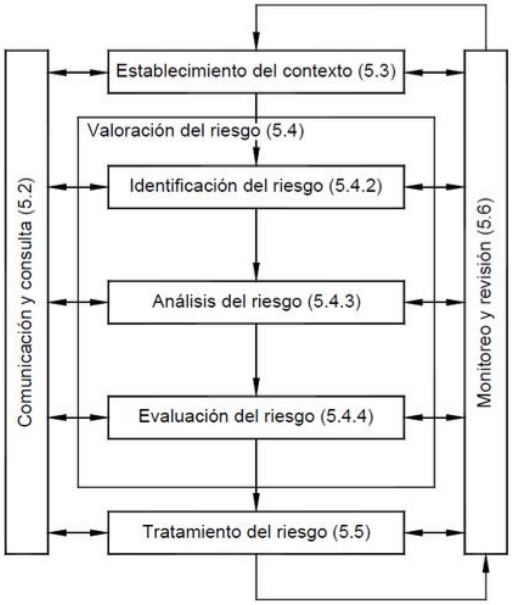

Figura 2. El proceso de gestión de riesgos, tomada de (ICONTEC, 2011b)

Al diseñar los objetivos de la gestión del riesgo de la organización, es importante y necesario considerar todos los principios que la rigen, aunque cada uno de ellos puede variar según el marco de referencia considerado en la organización y su aplicación en la misma, por lo que es necesario conocer la implicación que tiene cada uno de ellos y aplicarlos de forma continua. La implementación eficaz de estos principios determinará tanto la eficacia como la eficiencia de la gestión del riesgo en la organización.

3.1.1 Principios de la gestión de riesgos

De acuerdo con lo que establece la Norma Técnica Colombiana ISO31000, para que la gestión de riesgos sea efectiva, se debería cumplir en toda la organización con los siguientes 11 principios: (ICONTEC, 2011b)

a) La gestión de riesgos crea y protege los valores. Contribuye al cumplimiento de los objetivos y a la mejora del desempeño por ejemplo en: seguridad y salud humana, seguridad patrimonial, cumplimiento legal y reglamentario, aceptación del público (imagen, percepción, y aceptación), protección ambiental, calidad del producto, gestión de proyectos, eficiencia de la operación, autoridad y prestigio.
b) La gestión de riesgos es una parte integral de todos los procesos de la organización. La gestión de riesgos es parte de las responsabilidades de gestión y un componente integral de todos los procesos de la organización, incluyendo la planificación estratégica y, todos los proyectos y cambios en los procesos de gestión.
c) La gestión de riesgos es parte de la toma de decisiones.
d) La gestión de riesgos de manera explícita considera la incertidumbre.
e) La gestión de riesgos es sistemática, estructurada y programada.
f) La gestión de riesgos está basada en la mejor información disponible.
g) La gestión de riesgos está hecha a la medida de la organización.
h) La gestión de riesgos considera los factores humanos y culturales.
i) La gestión de riesgos es transparente e incluyente.
j) La gestión de riesgos es dinámica, reiterativa y sensible a los cambios.
k) La gestión de riesgos facilita la mejora continua de la organización.

3.1.2 Evaluación de riesgos

Como lo enuncia en su libro (Casares San José-Martí & Lizarzaburu Bolaños, 2016). En el apartado sobre evaluación del riesgo:

Los riesgos son analizados, teniendo en cuenta la probabilidad de ocurrencia y el impacto, lo que permitirá determinar su tratamiento. Se evalúan los riesgos inherentes y residuales.

La evaluación de riesgos es el proceso de análisis y priorización de los riesgos relevantes para el logro de los objetivos de la entidad y para determinar una respuesta apropiada.

Mediante la identificación y análisis de los riesgos relevantes y de la capacidad institucional de gestionarlos, se evalúa la vulnerabilidad de la organización preparándose la respuesta necesaria para enfrentarlos y resguardándose la consecución de los objetivos y resultados previstos por la empresa.

Criterios de evaluación de riesgos:

- Impacto (I): Nivel de exposición financiera de la empresa ante un riesgo o cuantía de la pérdida financiera que se generaría si ocurriera un evento de riesgo.
- Probabilidad (P): Grado de posibilidad de que ocurra el evento de riesgo en un periodo de tiempo determinado. Puede ser estimado en función a cuántas veces históricamente ha ocurrido el evento de riesgo en la organización y la posibilidad que vuelva a ocurrir en el futuro.

Como complemento a esta norma se ha desarrollado otro estándar: la ISO 31010 "Gestión del riesgo. Técnicas de evaluación de riesgos". Esta norma provee de una serie de técnicas para la identificación y evaluación de riesgos, tanto positivos como negativos.

3.1.3 La norma ISO 31010

La Norma ISO 31010 especifica las técnicas y herramientas de evaluación de riesgo en un proceso de Gestión de riesgo que pueden ser usadas dependiendo de la necesidad de la organización, las técnicas descritas en la norma pueden ser clasificadas de diferentes maneras con el fin de facilitar la comprensión de sus aplicaciones, elementos de entrada, procesos, resultados, relativas fortalezas y limitaciones. La norma lista 31 técnicas para la evaluación de riesgo en su Anexo B, incluso proporciona una guía para la selección y aplicación de cada una de estas técnicas en las etapas de identificación, análisis y evaluación del riesgo según la propuesta de la norma ISO 31000. (Casares San José-Martí & Lizarzaburu Bolaños, 2016)

Tabla 1. Técnicas usadas en el proceso de evaluación del riesgo.

Proceso de Evaluación del Riesgo						
Grupo	Técnica	Identificación del Riesgo	Análisis del Riesgo			Evaluación del Riesgo
			Consecuencia	Probabilidad	Nivel de Riesgo	
Métodos de Búsqueda	Listas de verificación	FA	NA	NA	NA	NA
	Análisis preliminar de riesgos	FA	NA	NA	NA	NA
Métodos de Apoyo	Lluvia de ideas	FA	NA	NA	NA	NA
	Entrevistas estructuradas	FA	NA	NA	NA	NA
	Técnica Delphi	FA	NA	NA	NA	NA
	Técnica What if	FA	FA	FA	FA	FA
	Análisis de Fiabilidad Humana	FA	FA	FA	FA	A
Análisis de escenarios	Análisis Causa-Raíz	NA	FA	FA	FA	FA
	Análisis de Escenario	FA	FA	A	A	A
	Valoración de riesgo medio ambiental	FA	FA	FA	FA	FA
	Análisis del impacto en el negocio	A	FA	A	A	A
	Análisis de árbol de fallos	A	NA	FA	A	A
	Análisis de árbol de sucesos	A	FA	A	A	NA
	Análisis de causa y consecuencia	A	FA	FA	A	A
	Análisis de Causa y efecto	FA	FA	NA	NA	NA
Métodos Estadísticos	Análisis de Markov	A	FA	NA	NA	NA
	Simulación de Monte Carlo	NA	NA	NA	NA	FA
	Esatdística y Redes Bayesianas	A	FA	NA	NA	FA
	Matriz de consecuencia/probabilidad	FA	FA	FA	FA	A
	Análisis Coste/Beneficio	A	FA	A	A	A
Análisis de Funciones	Arbol de decisión	NA	FA	FA	A	A
	Curvas FN	A	FA	FA	A	FA
	Indice de riesgos	A	FA	FA	A	FA
	Análisis modal de daños potenciales	FA	FA	FA	FA	FA
	Análisis de errores de diseño	FA	FA	FA	FA	FA
	Estudio de riesgos operacionales	A	NA	NA	NA	NA
	Estudio de peligros y operabilidad	FA	FA	A	A	A
	Análisis de puntos de control crítico	FA	FA	NA	NA	FA
Evaluaciones de Control	Análisis de niveles de protección	A	FA	A	A	NA
	Análisis del nudo de pajarita	NA	A	FA	FA	A
Evaluaciones de Control	Análisis de Decisión Multicriterio	A	FA	A	FA	A

FA: Fuertemente aplicable A: Aplicable NA: No Aplicable

El propósito de la evaluación de riesgos es proporcionar información y análisis basados en evidencia para tomar decisiones sobre cómo tratar riesgos particulares y cómo seleccionar entre varias opciones para prevenirlos o mitigarlos.

Los principales beneficios de una evaluación de riesgos incluyen:

- Proporcionar información objetiva a los tomadores de decisiones en la organización.
- Una comprensión del riesgo y su impacto potencial sobre los objetivos.
- Identificar, analizar y evaluar los riesgos y determinar la necesidad de su tratamiento.
- La cuantificación o clasificación de riesgos.
- Identificación de los contribuyentes importantes a los riesgos y enlaces débiles en sistemas y organizaciones.
- Comparación de riesgos en sistemas alternativos, tecnologías o enfoques.
- Identificación y comunicación de riesgos e incertidumbres.
- Ayudan a establecer prioridades para la salud y la seguridad.
- Racionalizar una base para el mantenimiento preventivo y la inspección.
- Investigación y prevención posterior al incidente.
- Seleccionar diferentes formas de tratamiento de riesgo.
- Cumplir con los requisitos reglamentarios
- Proporcionar información que ayudará a evaluar la tolerabilidad del riesgo en comparación con los criterios predefinidos.

En síntesis, la norma ISO 31000 es un estándar para la gestión de riesgos, que al igual que la ISO 27001 que se explicará a continuación para los sistemas de gestión de seguridad de la información, puede ser implementado en organizaciones de todo tipo y tamaños, sin importar el objeto de negocio, los procesos y sus niveles, debido a que cualquiera puede enfrentar factores internos y externos, que generan incertidumbre sobre si las organizaciones lograrán o no sus objetivos. El efecto que esta incertidumbre tiene en los objetivos de una organización es el “riesgo”.

ISO 31000 no suministra ninguna recomendación específica sobre el tratamiento de la seguridad de la información y la gestión de tales riesgos. Por el contrario, ISO 27001 si lo hace y de una forma sobresaliente.

La norma ISO 27001 proporciona el Know-how necesario para identificar activos, amenazas y vulnerabilidades, pero también para evaluar las consecuencias de un determinado riesgo y calcular su probabilidad de ocurrencia.

Es de aclarar que no se necesita ISO 31000 para implementar ISO 27001. Lo anterior no significa que ISO 31000 no resulte de utilidad, sobre todo para identificar contextos externos e internos y para obtener un marco apropiado para la gestión de todo tipo de riesgos a nivel de toda la organización. Por el contrario, si tenemos en cuenta que ISO 31000 ayuda a abordar la gestión de riesgos de forma estratégica y global, podemos considerar este estándar como un excelente marco de gestión de todo tipo de riesgos, que puede trabajar en asocio y no como dependiente de cualquier otra norma, incluida por supuesto ISO 27001.

3.2 La norma ISO 27001

En la búsqueda del mejor estándar para gestionar la seguridad de la información en una compañía, generalmente los resultados suelen están alrededor de la serie de normas ISO 27000[4] ya que reúne todos los lineamientos en materia de gestión de seguridad de la información. Una de las normas más importantes, que además es certificable, es la ISO 27001, la cual es una norma internacional emitida por la Organización Internacional de Normalización (ISO) y describe cómo gestionar la seguridad de la información en una empresa. La revisión más reciente de esta norma fue publicada en 2013 y ahora su nombre completo es ISO/IEC 27001:2013. La primera revisión se publicó en 2005 y fue desarrollada en base a la norma británica BS 7799-2.

La ISO/IEC 27001 no establece requisitos absolutos para la gestión de riesgos de seguridad de la información, pero sí brinda lineamientos para que de una manera eficaz se puedan minimizar dichos riesgos, al asegurar que son identificados, evaluados y gestionados, considerando el impacto para la organización. (ICONTEC, 2015)

Su objetivo fundamental es la gestión de la confidencialidad, la integridad, y la disponibilidad de cualquier bien que tenga valor para la organización, basándose en el ciclo Deming PDCA (siglas en inglés) o PHVA (siglas en español) o el ciclo de mejora continua. Una vez hecha la evaluación

[4] Familia de números reservado por la ISO para las normas relacionadas con sistemas de gestión de seguridad de la información. En el 2005 incluyó en ella la primera de la serie (ISO 27001)

de los riesgos y aplicados todos los controles, siempre queda lo que se conoce como un riesgo residual que debe ser revisado por lo menos una vez al año para tomar las medidas preventivas.
Además del ciclo Deming, el Sistema de Gestión en Seguridad de la Información basado en la norma ISO/IEC 27001 cuenta con indicadores y métricas con los cuales es posible medir la eficiencia de los distintos controles empleados. De esta manera aporta diariamente datos reales relacionados con la seguridad de los sistemas de información. Con dichos datos se puede contar con mecanismos de alerta y, consecuentemente, hacer las correcciones y mejoras del caso.

De una forma general, la norma ISO 27001 obliga a:

- Realizar un análisis de los riesgos relacionados con la seguridad de la información
- Definir los objetivos específicos de seguridad
- Implementar medidas para mitigar y gestionar los riesgos (la norma recomienda un conjunto de controles de seguridad)
- Asignar la responsabilidad de la gestión del riesgo
- Seguimiento de las medidas tomadas para mitigar los riesgos mediante auditorías y revisiones
- Mantener un enfoque de mejora continua.

3.3 La Lógica Difusa

La lógica difusa es una línea de la inteligencia artificial que proporciona una manera simple y elegante de obtener una conclusión a partir de información de entrada vaga, ambigua, imprecisa, con ruido o incompleta. En general la lógica difusa imita como una persona toma decisiones basada en información con las características mencionadas. Una de las ventajas de la lógica difusa es la posibilidad de implementar sistemas basados en ella tanto en hardware como en software o en combinación de ambos. La lógica difusa es una técnica de la inteligencia computacional que permite trabajar con información la cual tiene un alto grado de imprecisión, en esto se diferencia de la lógica convencional que trabaja con información bien definida y precisa. Es una lógica multivaluada que permite valores intermedios para poder definir evaluaciones entre si/no, verdadero/falso, negro/blanco, caliente/frío, etc. (Jang, J.-S. R., Sun, 1997)

3.3.1 Inferencia y Reglas difusas

Como en la lógica clásica, la lógica difusa se ocupa del razonamiento formal con proposiciones, pero a diferencia de esta, los valores de las proposiciones pueden tener valores intermedios entre falso y verdadero (0-1). El concepto de variable lingüística fue un escalón al concepto de reglas difusas SI-ENTONCES, estas son una base para la lógica difusa que a menudo es utilizada en aplicaciones prácticas (Zadeh, 1975; Rutkowska, 2002; y Zadeh, 1996)

Con las reglas difusas se especifican la relación entre las variables de entrada y salida del sistema y dichas relaciones difusas determinan el grado de presencia o ausencia de asociación o interacción entre los elementos de 2 o más conjuntos.

Según (Lofti A Zadeh, 1988), una regla difusa tipo Mamdami se asume como:

"Si x_1 es A_1 y x_2 es A_2 y.....y x_k es A_k Entonces Y es B"

Donde A_1, A_2,..., A_k, B son valores lingüísticos definidos mediante conjuntos difusos para las variables lingüísticas en el universo del discurso x_1, x_2,.., x_k y Y respectivamente. Como en la lógica clásica a menudo "x es A" es llamada antecedente o premisa y "y es B" es llamada la conclusión o consecuente. Este tipo de reglas puede ser utilizado para modelar y analizar un sistema.

La regla anterior, define una relación difusa en el espacio k+1 dimensional caracterizada por una función de pertenencia:

$$\mu A_k\text{->B } (x_1 , x_2 , \ldots, x_k \text{ ,Y}) \hat{I} [0; 1]$$

La base de las reglas difusas en general se obtiene del conocimiento de expertos mediante entrevistas, cuestionarios o técnicas de panel, sin embargo, en muchas ocasiones no se tiene acceso a dichos expertos, pero se cuenta con una base de datos de las variables de entrada-salida. En situaciones como ésta, es posible generar reglas difusas que definan una adecuada correspondencia entre las variables de entrada y salida. La interpretación de una regla SI-Entonces involucra dos pasos, el primero es evaluar el antecedente mediante la aplicación de cualquier operador difuso y el segundo paso es la implicación o la aplicación del resultado del antecedente al consecuente. Esto se hace evaluando la función de pertenecía μA->B (x_1 , x_2 ,…, x_k ,Y). Es decir, se trata de evaluar

la activación de una regla (activación del consecuente) en función del grado de cumplimiento del antecedente. Para realizar dicha tarea, se hace uso de operadores de composición de conjuntos difusos y de la aplicación de un sistema de inferencia.

Sea cual sea la forma de las variables y reglas, el proceso de inferencia se realiza en tres etapas:

1. Calcular la compatibilidad entre los valores actuales de las entradas y los antecedentes de las reglas.
2. Encontrar los resultados de las inferencias de cada regla (en qué grado se verifica cada regla).
3. Encontrar el resultado de la inferencia completa como un promedio de los resultados de las inferencias de cada regla.

3.3.2 Control Difuso

En general, el proceso de inferencia descrito anteriormente responde a un esquema de modelado que permite manipular reglas de inferencia sobre conjuntos difusos, y que puede ser resumido de la siguiente forma (Sakti, 2014) :

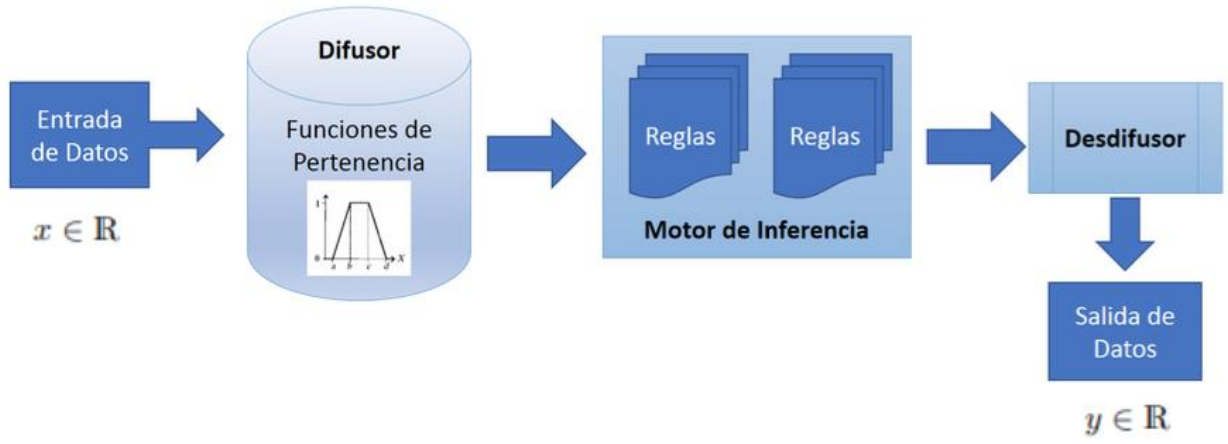

Figura 3. Estructura de un modelo difuso.

- Difusor: Convierte valores reales en valores difusos y se asignan los grados de pertenencia a cada una de las variables de entrada con relación a cada uno de los conjuntos difusos

previamente definidos utilizando las funciones de pertenencia asociadas a los conjuntos difusos.

- Motor de inferencia: Relaciona conjuntos difusos de entrada y de salida y que representa a las reglas que definen el sistema.
- Desdifusor: Se utilizan métodos matemáticos simples como el centroide, promedio ponderado o membresía del medio del máximo, para obtener un valor de salida numérico que representa el resultado.

3.4 Las Redes Neuronales Artificiales

Las redes neuronales emulan ciertas características propias de los humanos, como son la capacidad de memorizar y de asociación de hechos y eventos. Muchos de los problemas que no se pueden resolver mediante algoritmos tradicionales se podrían aproximar mediante el entrenamiento, cuya finalidad es adquirir experiencia. Las personas pueden resolver estos problemas gracias a la acumulación de experiencia, esto da pie a pensar que una forma de aproximarse a la solución es mediante la construcción de un sistema que se asemeje a esta característica humana; dicha aproximación se basa en modelos artificiales que se asemejen a un cerebro humano simplificado, tomando como núcleo la neurona.

Una neurona por sí sola no posee mayor poder o experiencia acumulada, pero cuando se agrupan y forman redes más complejas su poder y capacidad de aprender de estímulos las convierten en una herramienta muy poderosa para resolver problemas computacionales.

Una neurona biológica básicamente funciona de la siguiente manera: la neurona se ve estimulada por medio de sus entradas y cuando se alcanza un cierto umbral determinado, la neurona se activa, pasando una señal hacia su axón. Mediante la repetición de estos estímulos se refuerzan o debilitan las conexiones que previamente se han ido creando, adquiriendo así una especie de experiencia, pudiendo reaccionar de una forma determinada a estímulos controlados o nuevos.

3.4.1 Reseña histórica

- En 1936 Alan Turing[5] fue el primero en estudiar el cerebro como una forma de ver el mundo de la computación. Sin embargo, los primeros teóricos que concibieron los fundamentos de la computación neuronal fueron Warren McCulloch, un neurofisiólogo, y Walter Pitts, un matemático, quienes, en 1943, lanzaron una teoría acerca de la forma de trabajar de las neuronas modelando una red neuronal simple mediante circuitos eléctricos.
- En 1949 Donald Hebb[6] fue el primero en explicar los procesos del aprendizaje (que es el elemento básico de la inteligencia humana) desde un punto de vista psicológico, desarrollando una regla de como el aprendizaje ocurría. Aun hoy, este es el fundamento de la mayoría de las funciones de aprendizaje que pueden hallarse en una red neuronal. Su idea fue que el aprendizaje ocurría cuando ciertos cambios en una neurona eran activados. También intentó encontrar semejanzas entre el aprendizaje y la actividad nerviosa. Los trabajos de Hebb formaron las bases de la Teoría de las Redes Neuronales.
- En 1950 Karl Lashley[7] en sus series de ensayos, encontró que la información no era almacenada en forma centralizada en el cerebro, sino que era distribuida encima de él.
- En 1957 Frank Rosenblatt[8] comenzó el desarrollo del Perceptrón, considerada la red neuronal más antigua conocida y usada hoy en día para identificar patrones. Este modelo era capaz de generalizar, es decir, después de haber aprendido una serie de patrones podía reconocer otros similares, aunque no se le hubiesen presentado en el entrenamiento. Sin embargo, tenía una serie de limitaciones, por ejemplo, su incapacidad para resolver el problema de la función OR-exclusiva y, en general, era incapaz de clasificar clases no separables linealmente.
- En 1959 Frank Rosenblatt escribió "Principios de Neurodinámica" (Rosenblatt, 1959), en este libro confirmó que, bajo ciertas condiciones, el aprendizaje del Perceptrón convergía hacia un estado finito (Teorema de Convergencia del Perceptrón).

---

[5] Matemático, lógico, científico de la computación, criptógrafo y filósofo, nacido en Londres (1912-1954)

[6] Psicólogo, neurocientífico y profesor Canadiense (1904 - 1985)

[7] Psicólogo y profesor universitario, investigador del área de neuropsicología, nacido en Estados Unidos (1890-1958)

[8] Licenciado en letras, Psicólogo estadounidense notable en el campo de inteligencia artificial (1928-1971)

- En 1960 Bernard Widroff/Marcian Hoff desarrollaron el modelo Adaline (ADAptative LINear Elements). Esta fue la primera red neuronal aplicada a un problema real (filtros adaptativos para eliminar ecos en las líneas telefónicas) que se ha utilizado comercialmente durante varias décadas.
- En 1961 Karl Steinbeck construye "Die Lernmatrix", una red neuronal para simples realizaciones técnicas (memoria asociativa).
- En 1969 Marvin Minsky y Seymour Papert casi producen la "muerte abrupta" de las Redes Neuronales, ya que probaron matemáticamente que el Perceptrons no era capaz de resolver problemas relativamente fáciles, tales como el aprendizaje de una función no-lineal. Esto demostró que el Perceptrón era muy débil, dado que las funciones no-lineales son extensamente empleadas en computación y en los problemas del mundo real.
- En 1974 Paul Werbos desarrolló la idea básica del algoritmo de aprendizaje de propagación hacia atrás (backpropagation); cuyo significado quedó definitivamente aclarado en 1985.
- En 1977 Stephen Grossberg desarrolla la "Teoría de Resonancia Adaptada" (TRA), la cual es una arquitectura de red que se diferencia de todas las demás previamente inventadas. La misma simula otras habilidades del cerebro: memoria a largo y corto plazo.
- En 1982, Teuvo Kohonen[9] presentó un modelo de red neuronal con capacidad para formar mapas de características de manera similar a como ocurre en el cerebro, denominado mapas auto-organizados o SOM (Self-Organizing Maps), basado en ciertas evidencias descubiertas a nivel cerebral este tipo de red posee un aprendizaje no supervisado competitivo. En este modelo hay neuronas que se organizan en muchas zonas, de forma que la información captada del entorno a través de los órganos sensoriales se representa internamente en forma de mapas bidimensionales.
- En 1985 John Hopfield provoca el renacimiento de las redes neuronales con su libro: "Computación neuronal de decisiones en problemas de optimización."
- En 1986 David Rumelhart y G. Hinton redescubrieron el algoritmo de aprendizaje de propagación hacia atrás (backpropagation).

[9] Científico, informático y docente Finlandés, quien ha hecho grandes contribuciones en inteligencia artificial, principalmente en el área de las redes neuronales

A partir de 1986, el panorama fue alentador con respecto a las investigaciones y el desarrollo de las redes neuronales. En la actualidad, son numerosos los trabajos que se realizan y publican cada año.

3.4.2 Elementos básicos

Las redes neuronales están constituidas por un conjunto de neuronas interconectadas y arregladas en capas. El esquema tradicional está conformado por 3 capas: una capa de entrada, una capa oculta y una capa de salida, pero existen implementaciones que emplean más de una capa oculta.

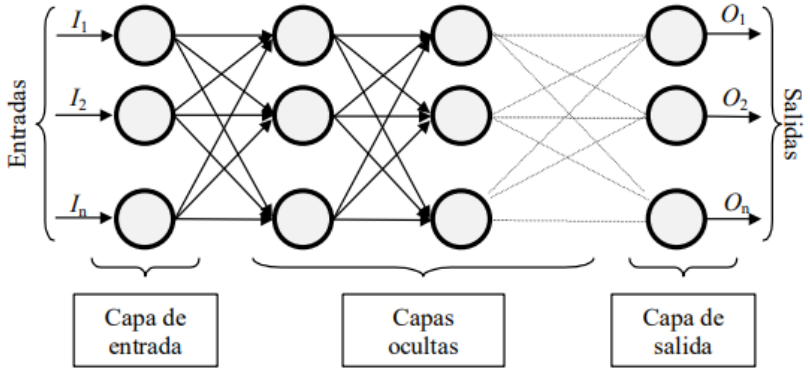

Figura 4. Estructura de una red neuronal.

Una neurona puede estar activa o inactiva, para determinar este estado se emplea una función de activación que está definida por una función matemática que normalmente se representa en forma lineal, sigmoidal o de tangente hiperbólica.

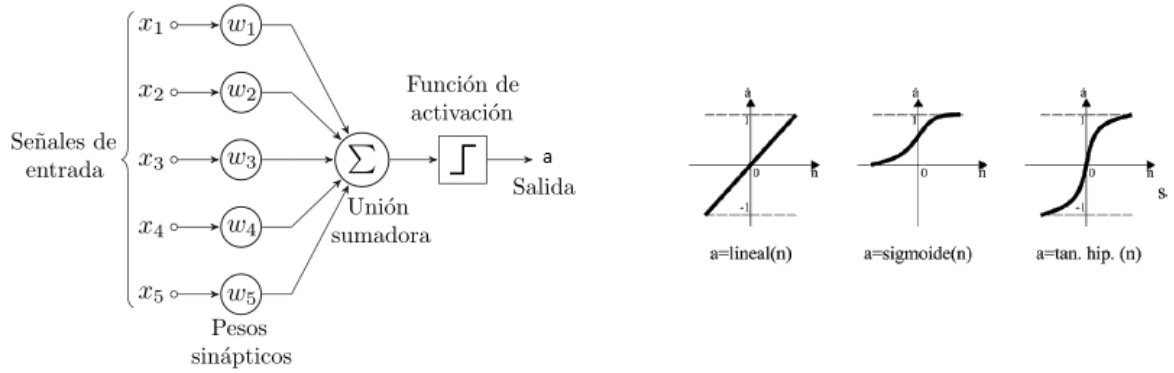

Figura 5. Funciones de activación.

Estas funciones de activación están definidas por las siguientes expresiones matemáticas y son las encargadas de definir el momento en el que cada neurona entra en acción:

Función lineal

$$f(x) = \begin{cases} -1 & para\ x \leq -1/a \\ a * x & para -1/a < x < 1/a \\ 1 & para\ x \geq 1/a \end{cases} \quad (1)$$

Función sigmoidea

$$f(x) = \frac{1}{1 + e^{-gx}} \quad (2)$$

Función tangente hiperbólica

$$f(x) = \frac{e^{gx} - e^{-gx}}{e^{gx} + e^{-gx}} \quad (3)$$

3.4.3 Mecanismos de aprendizaje

- **Aprendizaje supervisado**: En este tipo de aprendizaje se cuenta con un agente externo el cual realiza un entrenamiento controlado brindando las entregas y obteniendo una respuesta. La salida obtenida es comparada contra la deseada, en caso de no corresponder el agente externo procede a modificar los pesos de las conexiones, con el fin de conseguir que la salida se aproxime a la deseada.
- **Aprendizaje por corrección de error**: Aquí los pesos se ajustan según la diferencia entre la salida obtenida y la salida deseada teniendo un cambio muy grande si la diferencia es grande, permitiendo llegar de una forma rápida al ajuste de los pesos.
- **Aprendizaje por refuerzo**: En este método se cuenta con un indicador el cual determina si la salida obtenida se ajusta a la deseada modificando los pesos de las conexiones mediante un mecanismo de probabilidad.
- **Aprendizaje estocástico**: En el aprendizaje estocástico se realizan cambios aleatorios en los pesos, evaluando la salida siguiendo distribuciones de probabilidad.
- **Aprendizaje no supervisado**: Este tipo de aprendizaje no cuenta con un agente externo que supervise las salidas y ajuste los pesos de las conexiones entre neuronas. Al no poseer

información sobre si las salidas son correctas o no, este tipo de aprendizaje puede representar la similitud entre un conjunto de datos de entrada nuevos y los dados presentados en el pasado, también puede representar un sistema de codificación en el cual la salida es una representación de la entrada con un número mayor o menor de bits o realizar un mapeo de características similares.

- **Aprendizaje Hebbiano**: Este tipo de aprendizaje ajusta las conexiones de las neuronas según la correlación existente, si dicha correlación es positiva se genera un reforzamiento, de lo contrario la conexión se hace más débil.
- **Aprendizaje competitivo y comparativo**: Cada neurona compite y coopera con el fin de llevar a cabo una tarea dada. El objetivo es que las neuronas de salida compitan por activarse de tal forma que solo una quede activa. Por tanto, las neuronas compiten por activarse, quedando finalmente una, o una por grupo, como neurona vencedora.

4 Estado del arte

En esta sección se presenta la revisión de los proyectos, tesis o artículos que están relacionados con el desarrollo de este trabajo de grado, y que abordan temáticas en común tales como: la evaluación de riesgos, medición de riesgos en una organización o modelos/metodologías para evaluación o medición de riesgos y se explica cuál es el aporte que cada uno brinda al presente trabajo.

4.1 Definición de un modelo de medición de riesgos de la seguridad de la información aplicando lógica difusa y sistemas basados en el conocimiento. Caso de estudio: empresa de servicios públicos. (Angarita Vivas & Tabares Isaza, 2015)

Trabajo de grado de Maestría en Ingeniería de Sistemas y Computación, desarrollada por los ingenieros Alexis Angarita y Cesar Tabares, la cual describe el planteamiento de un modelo y la posterior implementación de un prototipo que busca proporcionar una herramienta que ayude a los expertos a expresar sus juicios probabilísticos o sus valores sobre los activos de información en forma más exacta, evitando así la ambigüedad y la declaración de conceptos subjetivos permitiendo el tratamiento computacional de los mismos. Según los autores su trabajo se justificó de la siguiente manera:

"El proceso de análisis y gestión de riesgos en el ámbito de la seguridad de la información, se ha vuelto una práctica cada vez más común en las organizaciones. Este proceso permite realizar la planificación de las acciones respectivas que se deben ejecutar a corto, mediano y largo plazo y tomar decisiones en tiempo real frente a eventos o incidentes de seguridad, pero en éste existe un considerable potencial que se desaprovecha, ya que para enfrentar estos riesgos no es suficiente con identificarlos y evaluarlos de una forma cualitativa, también se requiere de un sistema eficaz y efectivo que permita medirlos y controlarlos de una manera más acertada y que permitan minimizarlos o mitigarlos en gran medida."

4.2 Modelo de evaluación de riesgos en activos de TIC'S para pequeñas y medianas empresas del sector automotriz. (Moncayo Racines, 2014)

"Actualmente, las pequeñas y medianas empresas no conocen de forma clara los riesgos a la que está sometida su organización, parte de ellas aplican planes de contingencia a nivel empresarial, pero no cuantifican sus activos y el riesgo por cada uno de ellos.

Por tal motivo, a partir de este estudio se propone la creación de un modelo de evaluación de riesgos, basado en las metodologías Magerit, Octave y normas NIIF (Normas Internacionales de Información Financiera), el mismo que aportará a las empresas a obtener información sobre los riesgos, amenazas, y protecciones que deben considerar para evitar y tomar medidas de prevención oportunas y adecuadas.

El nuevo modelo, va a completar un ciclo de identificación de activos proponiendo su clasificación a servicios, hardware, software, soportes de información y personas, de los cuales se pueda determinar los puntos más débiles o vulnerables, de cada activo TIC's.

La metodología utilizada para la aplicación del modelo fue de tipo cuantitativo y cualitativo, ya que se aplicó una encuesta a los participantes de las dos empresas de estudio, y por otra parte se accedió a documentación que permitió conocer la situación actual referente a infraestructura, equipamiento y activos TIC's.

El modelo es de fácil aplicación y ayudará a las organizaciones a dirigir y gestionar las evaluaciones de riesgos por sí mismos, tomando las mejores decisiones para controlar y mitigar sus riesgos".

4.3 Las redes neuronales y la evaluación del riesgo de crédito. (Pérez Ramírez & Fernández Castaño, 2007)

"A pesar del escepticismo del mundo académico sobre los avances de la inteligencia artificial, las redes neuronales han abierto un campo de exploración bursátil que aún tiene mucho por investigar. Atendiendo a las ventajas del uso de las redes neuronales artificiales (ANN, por sus siglas en inglés) y a su capacidad para estimar modelos no lineales, en este artículo se muestra la aplicación de las redes neuronales a la cuantificación del riesgo de crédito. Además, se hace el desarrollo

teórico de los fundamentos básicos de las redes neuronales. Para presentar las metodologías de medición de riesgo de crédito basados en redes neuronales, y aplicarlas a la base de datos de una cartera comercial, fue necesario elaborar un análisis exploratorio de cada una de las variables e investigar la correlación entre ellas. El objetivo del análisis es encontrar algunas relaciones para grupos determinados de la población, de acuerdo con sus características particulares. Por tanto, se cruzan variables de cada cliente, del crédito y del comportamiento contra la variable default (fallidos y no fallidos). Variable que establece un procedimiento de clasificación, y permite determinar las ponderaciones necesarias y, además, establece la probabilidad de fallido.

Las técnicas para medir el riesgo de crédito son hoy en día muy variadas, y abarcan procedimientos que van desde simples cálculos, hasta sofisticadas metodologías con simulaciones dinámicas del futuro más próximo. Estos procedimientos se han desarrollado tratando de representar cómo varía la capacidad de pago y qué efectos tienen estas variaciones sobre las finanzas de las instituciones"

4.4 Modelos para medir el riesgo de crédito de la banca. (Saavedra García & Saavedra García, 2010)

Este artículo es el resultado de la fase inicial del proyecto de investigación "Los derivados de crédito para la mitigación del riesgo bancario en México", que inició el 10 de enero de 2008 y concluyó el 26 de febrero de 2009, realizado por los autores, con financiamiento de la Universidad La Salle, Dirección de Posgrado e Investigación, México. El artículo se recibió el 27-04-2009 y se aprobó el 21-05-2010

"Este trabajo describe los principales modelos de determinación de riesgos de crédito de la banca, a fin comparar y dar a conocer su utilidad en la administración del riesgo de crédito bancario y, de esta manera, brindar un marco de referencia para estudiar este tema en la teoría y práctica financiera. El estudio, de tipo descriptivo, define el riesgo de crédito y analiza los principales modelos tradicionales (sistemas expertos y sistemas de calificación), modernos (el de Kecholfer, McQuown y Vasicek [KMV]) y el Capital y Riesgo de Crédito en Países Emergentes (CyRCE), creado por el Banco de México. Según los hallazgos, los modelos tradicionales se basan en un esquema para el análisis de ciertos componentes básicos, con el fin de evaluarlos de manera integral. Entre tanto, los modelos modernos intentan registrar la alta volatilidad a la que están sujetos los valores y emplean técnicas más sofisticadas para su determinación. Estos resultados

indican que los modelos han evolucionado en correspondencia con la complejidad del entorno que rodea al sistema bancario

El sistema de medición de riesgo de crédito tiene por objeto identificar los determinantes del riesgo de crédito de las carteras de cada institución, con el propósito de prevenir pérdidas potenciales en las que podría incurrir. Por ello en este tipo de análisis es importante considerar los criterios de calificación de las carteras crediticias de la institución, la estructura y composición de los portafolios crediticios, el impacto de las variables macroeconómicas y sectoriales en los portafolios y las características históricas de las carteras de crédito de cada institución. Existen múltiples modelos de valuación del riesgo (Para mayor detalle, véanse Chorafas (2000, p. 52) y Ong (1999, p. 63)). A diferencia del riesgo de mercado, el desarrollo de metodologías para medir el riesgo de crédito ha sido menos cuantioso, ya que estas dependen de las características propias de cada institución de crédito, pero seguimos la clasificación de Galicia (2003), resumida en la tabla 3:

Tabla 2. Tomada de (Saavedra García & Saavedra García, 2010)

Modelos de valuación de riesgo de crédito

Modelos tradicionales	Modelos modernos
• Sistemas expertos (Galicia, 2003) • Sistemas de calificación*	• Modelo KMV • Modelo de valuación de Merton** • Modelo Credimetrics de J. P. Morgan (1997b) • Modelo Credit Risk + (Morgan, 1997a) • Modelo de retorno sobre capital ajustado al riesgo (Falkenstein, 1997) • Modelo CyRCE***

Los modelos para estimar la probabilidad de incumplimiento surgieron de manera formal durante la década de los setenta; sin embargo, desde los años treinta ya se habían iniciado estudios basados en el análisis tradicional de razones financieras.

Es necesario considerar que para entender el riesgo de crédito se deben visualizar los conceptos de pérdida esperada y pérdida no esperada (Elizondo y López, 1999). El deterioro que presenta un

crédito en el momento del análisis de riesgo se traduce en una pérdida esperada que producirá una minusvalía para el banco y por lo cual se deberá crear una reserva preventiva"

4.5 Sistema difuso para la evaluación de un modelo de riesgo de mercado en un portafolio de deuda pública en Colombia. (Cardona Ochoa, 2015)

"El riesgo de mercado se define como la posibilidad de incurrir en pérdidas económicas debido a las fluctuaciones de los precios de mercado de los activos (Superintendencia Financiera de Colombia, 1995).En la actualidad existen diversas metodologías para identificar, medir, controlar y monitorear ese riesgo; sin embargo, dichas técnicas no siempre logran captar los movimientos adversos de los factores asociados con el mismo, ya sea por la incertidumbre inherente a los datos, por la imposibilidad de cuantificarlos o por la complejidad computacional de los modelos. El presente trabajo busca incorporar los criterios de expertos a un modelo de valor en riesgo, VaR, por simulación histórica, para un portafolio de deuda pública colombiana, mediante la aplicación de sistemas de inteligencia artificial o de inferencia basados en lógica difusa, utilizando software de modelación matemática MATLAB®. Para el trabajo se consultaron varios autores en las áreas de inteligencia artificial, riesgos y auditoría, con el fin de incorporar variables no tenidas en cuenta por un modelo VaR por simulación histórica, pero que tienen un efecto directo sobre el comportamiento del portafolio y, por lo tanto, sobre los niveles de riesgo de mercado asumidos, escogiendo de manera complementaria diferentes niveles de exposición al riesgo".

5 Descripción e Implementación del modelo

5.1 Descripción del modelo

Las Redes de Neuronas Artificiales son sistemas de aprendizaje basados en ejemplos. La capacidad de una red para resolver un problema estará asociada de forma fundamental al tipo de ejemplos de los cuales se dispone en el proceso de aprendizaje. Desde el punto de vista de los ejemplos, el conjunto de aprendizaje debe poseer las siguientes características:

- Ser significativo: debe haber un número suficiente de ejemplos. Si el conjunto de aprendizaje es reducido, la red no será capaz de adaptar sus pesos de forma eficaz.
- Ser representativo: los componentes del conjunto de aprendizaje deberán ser diversos. Si un conjunto de aprendizaje tiene muchos más ejemplos de un tipo que el resto, la red se especializará en dicho subconjunto de datos y no será de aplicación general. Es importante que todas las regiones significativas del espacio de estados estén suficientemente representadas en el conjunto de aprendizaje.

La tarea de clasificar información puede usar un modelo estándar y básico que consiste en la selección de los datos, la creación, el entrenamiento de la red y su posterior evaluación. En el presente trabajo de grado, el modelo consta de las siguientes etapas:

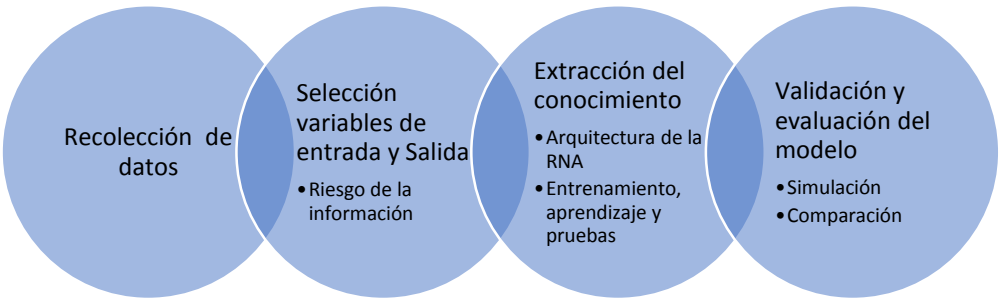

Figura 6. Etapas del modelo basado en RNA

5.2 Recolección de datos

La búsqueda y consolidación de la información con respecto a la salida que se pretende obtener es clave para el desarrollo del modelo. En esta etapa se obtiene la materia prima para encontrar las respuestas a los escenarios reales o de simulación específicos que se requieran plantear sobre el modelo.

En el caso específico del presente trabajo de grado, se usaron los datos recolectados en (Angarita Vivas & Tabares Isaza, 2015), ya que estos fueron sometidos a un proceso de selección previo según los expertos del área quienes crearon la base de conocimiento en la empresa Acueducto y Alcantarillado De Pereira S.A E.S.P como se menciona en su trabajo de grado.

En esta etapa es importante tener en cuenta la definición de los conjuntos de datos que se utilizarán para el proceso de entrenamiento y proceso de pruebas respectivamente.

5.3 Selección de variables de entrada y salida

La selección de variables de entrada reduce el tamaño de los datos que se emplearán en la base de conocimiento, para lo cual se eligen las variables más influyentes en el problema. En el caso del presente trabajo de grado se utilizaran las variables tanto de entrada como de salida descritas en (Angarita Vivas & Tabares Isaza, 2015), en donde las variables de entrada fueron tipificadas como amenazas con probabilidad de materializarse para un activo de información, y la variable de salida, como el nivel de riesgo al que están expuestos dichos activos.

A continuación, se ilustran las variables de entrada y salida en las siguientes tablas:

Tabla 3. Variables de entrada del modelo

Variables de Entrada: Amenazas e Impacto

Tipo	**Acrónimo**
Amenaza de Daño Físico	AMENAZADF
Amenaza de Compromiso de la información	AMENAZACI
Amenaza de Fallas Técnicas	AMENAZAFT
Amenaza de Acciones No autorizadas	AMENAZAANA
Impacto generado	IMPACTO

Tabla 4. Variable de salida del modelo

Variables de salida
Variable
Nivel de Riesgo

Cada una de estas variables está definida en un rango de valores que previamente fue determinado por los expertos en el área de seguridad de la información. En el anexo 3, se encuentran descritos los rangos de valores y los valores lingüísticos para cada una de las variables tanto de entrada como se salida que usa el modelo.

5.4 Extracción de la base de conocimiento

En esta etapa se describen los procesos de selección de la arquitectura de la red neuronal artificial (tipo de red, número de capas, neuronas por cada capa) y posteriormente se describen los procesos de aprendizaje y pruebas de la red.

5.4.1 Arquitectura de la RNA

Para efectos de encontrar una red que entregara más opciones y mejores resultados, se decidió utilizar una arquitectura de red neuronal multicapa. Las neuronas de una red de este tipo están conectadas entre sí, es decir, las neuronas se agrupan en distintas capas: una capa de entrada, otra de salida y en una o varias capas ocultas. La salida de cada neurona se propaga por igual por las conexiones hasta las neuronas de destino de la siguiente capa. Cada conexión tiene un peso asociado que pondera el valor numérico de la señal que viaja por esta. Todas las neuronas que contiene una capa se conectan con todas las neuronas de la siguiente capa, de esta manera, cuando una neurona obtiene un resultado, lo envía a todas las neuronas de la capa siguiente.
El modelo de la red neuronal artificial del presente trabajo se representa de la siguiente manera:

- A cada neurona i de la capa entrada se le entrega un valor que se encuentra en un vector X, así a la neurona i le ingresa el valor X(i). En el caso de las capas ocultas, su valor de entrada está dado por la salida de la neurona i-esima de la capa anterior.

- Cada neurona i de la capa j tendrá un peso sináptico W_{ij} que representa la intensidad de interacción entre la neurona presináptica j y la neurona postsináptica i. El valor del Potencial Sináptico $h_i(t)$ que se pasa a la neurona i de la siguiente capa, está dado por la regla de propagación (por ejemplo: una suma ponderada o un producto escalar de los vectores de entrada y de peso).

$$h_i(t) = \sum_{i=1}^{n} X_i * W_{ij} \tag{4}$$

- Cuando el valor del potencial sináptico para la neurona i-esima sobrepasa un valor numérico θ, denominado umbral, entonces la neurona i se activa y el número resultante de la regla de propagación se ingresa en una función denominada Función de Transferencia, descrita como:

$$h_i(t) > \theta \; Entonces \; f(h_i(t)) \tag{5}$$

En resumen, para la neurona estándar, la regla de propagación es la suma ponderada y la función de salida es la identidad. Se añade un grado de libertad a la neurona denominado "umbral" que consiste en restar al conjunto de pesos un parámetro θ_i y al aplicar la función de transferencia se tendrá el valor de la salida $Y_i(t)$, que será el valor de entrada de la neurona i de la siguiente capa, matemáticamente se define como:

$$Y_i(t) = f_i(\sum_{i=1}^{n} X_i * W_{ij} - \theta_i) \tag{6}$$

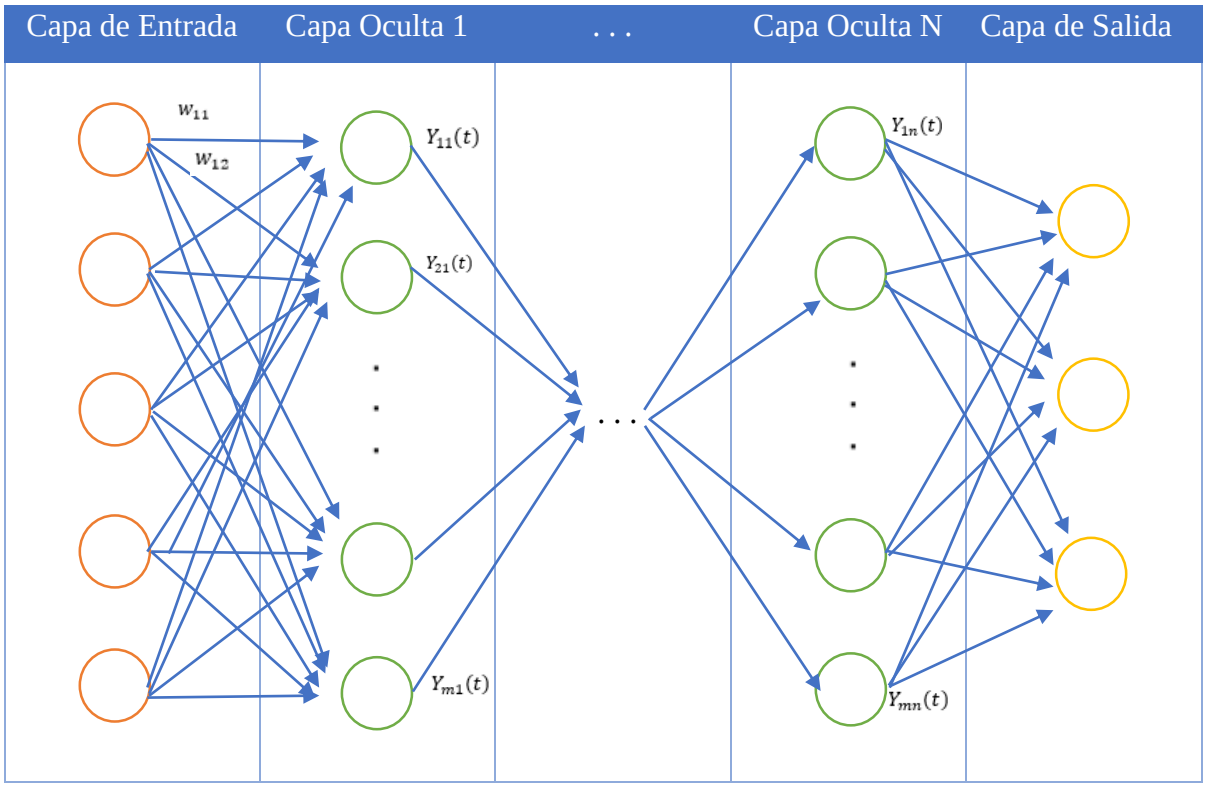

Figura 7. Arquitectura de la red neuronal multi capa

Como se observa en el gráfico de la arquitectura, no se determina de manera fija en número de capas ocultas de la red porque este será uno factores que variará en el modelo con el objetivo de observar el rendimiento ante diferentes valores de dicho parámetro.

En el apartado sobre las pruebas de la red neuronal se podrán observar los diferentes valores de precisión que se obtienen en la red neuronal multicapa al variar aleatoriamente parámetros como el número de capas ocultas y neuronas en cada capa de estas.

5.4.2 Aprendizaje de la red neuronal

Durante el aprendizaje, en la mayor parte de los modelos se produce una variación de los pesos sinápticos que miden la intensidad de interacción entre las neuronas. La red neuronal ajusta una función matemática que trata de minimizar los errores, mediante un proceso de cálculo numérico iterativo.

Para encontrar los pesos sinápticos de una red neuronal se utilizan distintos tipos de aprendizaje o entrenamiento. Un aspecto importante respecto al aprendizaje es conocer cómo se modifican los valores de los pesos; cuáles son los criterios para cambiar el valor asignado a las conexiones cuando se pretende que la red aprenda una nueva información. Estos criterios determinan el tipo de aprendizaje, siendo dos los tipos principales de aprendizaje: Aprendizaje Supervisado y Aprendizaje No Supervisado.

El aprendizaje supervisado es una técnica donde se muestran los patrones a la red y la salida deseada para esos patrones y se usa una fórmula matemática de minimización del error que ajuste los pesos para dar la salida más cercana posible a la salida deseada, dicho en otras palabras, hay un grupo de datos que servirán de entrenamiento para la red.

El aprendizaje no supervisado es una técnica en la que no se dispone de una salida esperada que asociar al grupo de datos de ejemplo, sino que únicamente a partir de las propiedades de los datos se intenta dar una agrupación o caracterización (clasificación, clustering) de los datos de trabajo según la similitud entre sus propiedades.

En el caso de presente trabajo, según la arquitectura seleccionada para la red neuronal, ésta se encuentra clasificada dentro del tipo de aprendizaje supervisado. El algoritmo usado dentro de la arquitectura es el algoritmo de retropropagación de errores o BackPropagation, el cual sirve como regla de aprendizaje para esta arquitectura y con el cual, debido a su gran potencial, se ha demostrado que es eficaz en la solución de problemas de clasificación.

5.4.3 Entrenamiento de la red neuronal

Para que la red neuronal ejecute una tarea es preciso entrenarla. Durante esta fase se puede producir la incorporación de nuevas neuronas o la pérdida de algunas de ellas. Si la red no aprende correctamente se pueden cambiar diversos parámetros y volver a entrenar hasta obtener una precisión o un mínimo de error aceptable.

Las redes neuronales supervisadas son técnicas para extraer datos a partir de las relaciones de entrada-salida y para almacenar tales relaciones en ecuaciones matemáticas que pueden utilizarse en actividades de pronóstico o en la toma de decisiones. Este tipo de red requiere que el usuario especifique la salida deseada y, por tanto, la red aprende a detectar la relación entre las entradas y las salidas suministradas, mediante un proceso iterativo y adaptativo; una vez que la red se ha entrenado, se puede utilizar con datos que nunca haya visto o puede ser incluida en un programa para el apoyo a las decisiones.

Para el presente trabajo, una vez cargados los datos, se procede a dividirlos en dos grupos, un subconjunto de entrenamiento correspondiente al 80% del total de estos y el restante 20% para pruebas, como se puede consultar en el Anexo 4 – Implementación del modelo.

5.4.4 Pruebas de la red neuronal

Para lograr obtener una red neuronal con el mejor nivel de precisión posible en la capa de salida, se hicieron una serie de pruebas modificando ciertos parámetros de la red como lo son: el tipo de solver, el número máximo de iteraciones para el entrenamiento, el número de neuronas en las capas ocultas e incluso el cambio de la función de activación.

La evaluación del desempeño del modelo está basada en el conteo de los elementos correcta e incorrectamente clasificados en el conjunto de datos de prueba de acuerdo con las salidas de clasificación del modelo. Es posible comparar el desempeño de un modelo usando la matriz de confusión o matriz de error. Una matriz de confusión se utiliza en clasificación para indicar la calidad de la solución e indica cuantos elementos fueron clasificados correctamente. Esta matriz es una matriz cuadrada de tamaño NxN donde N es el número de valores a clasificar y se contrasta contra las salidas generadas por la red neuronal en la fase de pruebas. Cuando la red no comete

errores, la matriz de confusión es diagonal, esto quiere decir, todos los valores se encuentran en la diagonal principal, teniendo un 100% de exactitud en el proceso de clasificación.
En caso de que la matriz de confusión contenga elementos distintos de cero fuera de la diagonal principal, esto indica que la red confundió un objeto de un tipo con otro tipo de objeto.

Tabla 5. Ejemplo definición matriz de confusión de 2 clases

	Clasificador	
	A	B
A	VP	FN
B	FP	VN

Cada columna de la matriz representará el número de predicciones realizadas por el modelo para cada clase y cada fila los valores reales por cada clase. Con lo cual, los conteos quedan divididos en 4 clases, TP, FN, FP y TN, que significan lo siguiente:

- VP – Verdaderos Positivos: Son el número verdaderos positivos, es decir, de predicciones correctas para la clase A.
- FN – Falsos Negativos: Son el número de falsos negativos, es decir, la predicción es negativa cuando realmente el valor tendría que ser positivo. A estos casos también se les denomina errores de tipo II.
- FP – Falsos Positivos: Son el número de falsos positivos, es decir, la predicción es positiva cuando realmente el valor tendría que ser negativo. A estos casos también se les denomina errores de tipo I.
- VN – Verdaderos Negativos: Son el número de verdaderos negativos, es decir, de predicciones correctas para la clase B.

Se desarrolló un script que toma la función de creación de la red neuronal y varía aleatoriamente los parámetros hasta encontrar el mejor porcentaje de precisión en dichas pruebas. Se emplea una semilla para la generación de los valores pseudoaleatorios correspondientes a la cantidad de neuronas en las capas ocultas, así como en la generación de los grupos de entrenamiento y pruebas de la red neuronal donde se emplea la función train_test_split. Esto se hace con el fin de tener la posibilidad de repetir el entrenamiento con valores aleatorios determinados por la semilla y lograr

un refinamiento de la red y sus resultados. Con la configuración de parámetros que mejor precisión arrojó en la capa de salida, es con la que se realizaron las simulaciones de los escenarios de prueba y posterior análisis comparativo.

```
salida = ['',0,0,'']
valores = []
np.random.seed(2019)
for j in np.arange(1,11,1):
    print
    print "Prueba para ",j," capas ocultas"
    print
    random.seed(20190327)
    co = ocultas_aleatorias(j)
    for slv in ['lbfgs','sgd','adam']:
        for act in ['logistic','tanh','relu']:
            for contador  in np.arange(1,11,1):
                YTrain, YTest, XTrain, XTest = train_test_split(entradas,salidas,train_size=0.80)
                red = MLPClassifier(solver=slv,alpha=0.00000001,max_iter=10000000,
                                    hidden_layer_sizes=co,random_state=1,activation=act)
                red.fit(YTrain, XTrain)
                y_pred = red.predict(entradas)
                score = accuracy_score(salidas, y_pred)
                valores.append(score)
                print "SOLVER: ",slv," activacion: ",act," Ocultas: ",j," Neuronas ",co," Accuracy: ",score
                if salida[1] < score:
                    salida = [slv,score,j,act]
            print "STD: ",statistics.stdev(valores)
            valores = []
print
print "Mejor resultado de la red neuronal"
print
print "Solver: ", salida[0]
print "Accuracy: ", salida[1]
print "Neuronas ocultas: ", salida[2]
print "Función de activación: ", salida[3]
print "Matriz de Confusion"
print
red = MLPClassifier(solver=salida[0],alpha=0.00000001,max_iter=10000000,
                    hidden_layer_sizes=(salida[2]),random_state=1,activation=salida[3])
YTrain, YTest, XTrain, XTest = train_test_split(entradas,salidas,train_size=0.80)
red.fit(YTrain, XTrain)
y_pred = red.predict(entradas)
cMat = confusion_matrix(salidas, y_pred)
print cMat
```

Figura 8. Algoritmo de prueba de la red neuronal.

Se realizan 10 pruebas con valores seleccionados de forma aleatoria, los resultados de estas pruebas son analizados por medio de la desviación estándar (7) para determinar la dispersión de estos y hayas los valores óptimos para realizar la comparación con el modelo de lógica difusa.

$$\sigma = \sqrt{S^2} = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \bar{x})^2}{n-1}} \tag{7}$$

A continuación, se muestran los resultados obtenidos con la red neuronal empleando diferentes configuraciones con el fin de determinar los parámetros a emplear para realizar la comparación contra el modelo de lógica difusa. Se desarrolló un algoritmo el cual realiza una serie de pruebas variando el número de capas ocultas, el algoritmo de activación y el solver a emplear, se ejecutan 10 pruebas aleatorias con cada uno de ellos obteniendo los resultados mostrados en la tabla 6.

Tabla 6. Resultados pruebas variables Redes Neuronales

Capas Ocultas	Solver	Activation	Media exactitud	Desviación estándar
1	**lbfgs**	**logistic**	**0,966181818**	**0,003172425**
1	**lbfgs**	**tanh**	**0,966818182**	**0,002615557**
1	**lbfgs**	**relu**	**0,963909091**	**0,005653769**
1	sgd	logistic	0,344181818	0,009717781
1	sgd	tanh	0,404818182	0,015617383
1	sgd	relu	0,330363636	0,009895691
1	adam	logistic	0,369636364	0,013641077
1	adam	tanh	0,451818182	0,032095473
1	adam	relu	0,472636364	0,032113921
2	**lbfgs**	**logistic**	**0,966909091**	**0,001828255**
2	**lbfgs**	**tanh**	**0,965909091**	**0,004247832**
2	**lbfgs**	**relu**	**0,966818182**	**0,002650433**
2	sgd	logistic	0,238454545	0,111311614
2	sgd	tanh	0,480636364	0,021290116
2	sgd	relu	0,400909091	0,018586408
2	adam	logistic	0,366000000	0,015189044
2	adam	tanh	0,496272727	0,044990459
2	adam	relu	0,595000000	0,034300701
3	**lbfgs**	**logistic**	**0,967090909**	**0,003631310**
3	**lbfgs**	**tanh**	**0,969545455**	**0,002277772**
3	**lbfgs**	**relu**	**0,969272727**	**0,002339425**
3	sgd	logistic	0,096818182	0,010621201
3	sgd	tanh	0,502000000	0,015331057
3	sgd	relu	0,397181818	0,048183629
3	adam	logistic	0,326818182	0,010393984
3	adam	tanh	0,503727273	0,052783756
3	adam	relu	0,537363636	0,037149216
4	lbfgs	logistic	0,089272727	0,005451177
4	**lbfgs**	**tanh**	**0,969545455**	**0,003127237**
4	**lbfgs**	**relu**	**0,968272727**	**0,001385349**
4	sgd	logistic	0,090090909	0,002587318
4	sgd	tanh	0,452090909	0,022453094
4	sgd	relu	0,447636364	0,033509726
4	adam	logistic	0,158454545	0,108779902
4	adam	tanh	0,515727273	0,034465751
4	adam	relu	0,598636364	0,035755650

5	lbfgs	logistic	0,093000000	0,007856028
5	**lbfgs**	**tanh**	**0,967818182**	**0,003297326**
5	**lbfgs**	**relu**	**0,968000000**	**0,003476274**
5	sgd	logistic	0,090818182	0,000287480
5	sgd	tanh	0,500363636	0,009381223
5	sgd	relu	0,457727273	0,024993112
5	adam	logistic	0,202727273	0,118606850
5	adam	tanh	0,461727273	0,048555725
5	adam	relu	0,517272727	0,062993871
6	lbfgs	logistic	0,089818182	0,003449757
6	**lbfgs**	**tanh**	**0,967727273**	**0,002785572**
6	**lbfgs**	**relu**	**0,966000000**	**0,003183982**
6	sgd	logistic	0,090909091	0
6	sgd	tanh	0,460363636	0,032207431
6	sgd	relu	0,389090909	0,042307198
6	adam	logistic	0,090909091	0
6	adam	tanh	0,474818182	0,032761066
6	adam	relu	0,479000000	0,057483898

De los datos de la tabla 6 se seleccionan aquellos con un valor de medía superior a 90% y de estos seleccionamos aquel cuya desviación estándar sea la menor para realizar la comparación.

Tabla 7. Resultados pruebas variables Redes Neuronales mayores a 90% exactitud

Capas Ocultas	**Solver**	**Activation**	**Media exactitud**	**Desviación estándar**
1	lbfgs	logistic	0,966181818	0,003172425
1	lbfgs	tanh	0,966818182	0,002615557
1	lbfgs	relu	0,963909091	0,005653769
2	lbfgs	logistic	0,966909091	0,001828255
2	lbfgs	tanh	0,965909091	0,004247832
2	lbfgs	relu	0,966818182	0,002650433
3	lbfgs	logistic	0,967090909	0,003631310
3	**lbfgs**	**tanh**	**0,969545455**	**0,002277772**
3	lbfgs	relu	0,969272727	0,002339425
4	lbfgs	tanh	0,969545455	0,003127237
4	lbfgs	relu	0,968272727	0,001385349
5	lbfgs	tanh	0,967818182	0,003297326
5	lbfgs	relu	0,968000000	0,003476274
6	lbfgs	tanh	0,967727273	0,002785572
6	lbfgs	relu	0,966000000	0,003183982

De estos valores se selecciona el correspondiente a 3 capas de neuronas ocultas en la capa oculta con un grado de exactitud del 96,9%. En la tabla 8 se muestra la cantidad de neuronas empleadas por capa en la prueba para cada una de las capas con la cuales se obtuvieron los resultados presentados.

Tabla 8. Cantidad de neuronas capa oculta

Neuronas capa oculta	Cantidad de neuronas
1	292
2	197, 179
3	194, 97, 84
4	88, 75, 160, 187
5	134, 247, 211, 163, 157
6	224, 136, 208, 106, 120, 143

En el siguiente gráfico se muestra la distribución de los datos, los cuales están enmarcados en una distribución normal con media 0,969545455 y desviación estándar 0,002277772.

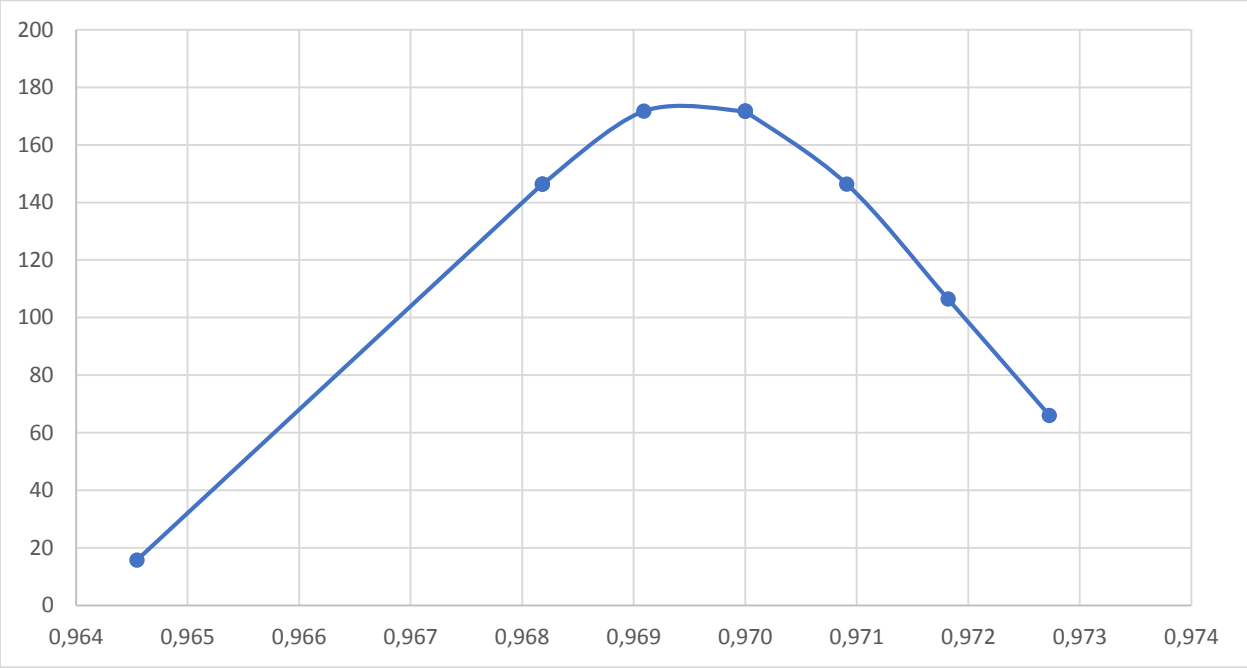

Figura 9. Distribución normal de los datos para 3 capas ocultas

$$\begin{bmatrix} 96 & 1 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 96 & 1 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 100 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 97 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 97 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 95 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 99 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 96 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 98 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 100 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 4 & 0 & 0 & 94 \end{bmatrix}$$

Figura 10. Matriz de confusión para 3 capas ocultas

Como se puede apreciar en la matriz de confusión, los elementos que se encuentran en la diagonal principal son los aciertos de clasificación de la red neuronal, mientras que los elementos encontrados por fuera de la diagonal son casos donde la salida de la red no era la salida esperada.

5.4.5 Rendimiento de la red neuronal

Después de realizar las pruebas sobre la red neuronal encontrando el nivel de precisión y la matriz de confusión (quien es la que muestra para cada dato de prueba cuál sería su salida contrastándola con la salida esperada), se decide trabajar con la configuración que arrojó un mayor porcentaje de precisión, siendo en este caso la obtenida con:

Tabla 9. Parámetros con mejor rendimiento

Parámetro	**Valor**
Solver	lbfgs
Función de activación	tanh
Número de capas ocultas	3
Número de neuronas por capa	194, 97, 84
Precisión	96,95%

Para la interfaz gráfica se emplea la misma desarrollada en el modelo de Python de lógica difusa, pero invocando a la red neuronal.

En síntesis, para tener una visión más genérica del modelo desarrollado y utilizado en el presente trabajo, a continuación, se ilustra gráficamente:

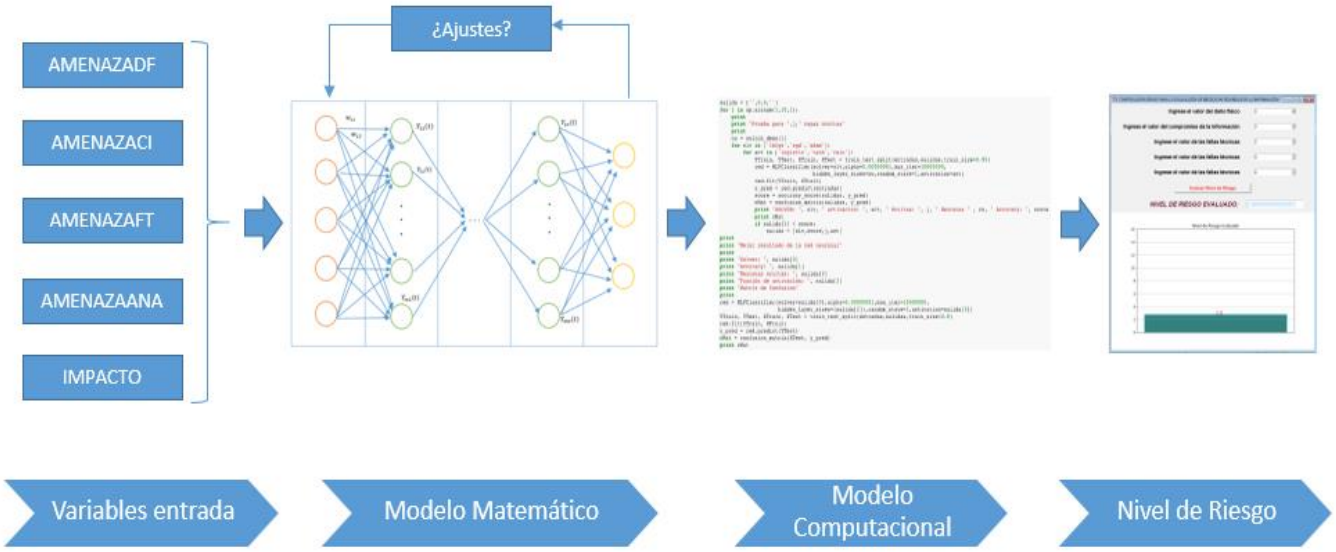

Figura 11. Estructura general del modelo desarrollado

6 Análisis de resultados

Para el proceso de simulación y posterior análisis de resultados, se tomarán datos reales emitidos por expertos en el tema, los cuales se introducirán en los campos correspondientes de las interfaces desarrolladas para el prototipo de evaluación de riesgos de seguridad de la información tanto con el modelo de lógica difusa como para el modelo de redes neuronales.

Todos los datos se supondrán acorde a la realidad expresada mediante el concepto del experto, arrojando así un resultado numérico que indicará la calificación del nivel de riesgo en el que se encuentra el activo evaluado. (Angarita Vivas & Tabares Isaza, 2015)

Este ejercicio busca realizar una comparación de los mismos escenarios de prueba en las diferentes implementaciones que se hicieron para los modelos de lógica difusa y redes neuronales, con el objetivo de analizar las salidas, determinar si un modelo puede ser más preciso que el otro y también definir si el uso de las diferentes plataformas, librerías o lenguajes usados tienen algún nivel de influencia en los resultados obtenidos.

6.1 Escenarios de prueba

La simulación en cada escenario se hará tomando como base los mismos escenarios de prueba ejecutados en el apartado de simulación de (Angarita Vivas & Tabares Isaza, 2015)

6.1.1 Escenario de Prueba 1

Este escenario se toma el activo de Información “SERVIDORES” tomando como fuente la calificación de los expertos acorde a los datos que se deben ingresar al prototipo:

Tabla 10. Parámetros entrada escenario de prueba 1

Valoración de Amenazas e impacto para el activo de Información SERVIDORES

AMENAZA	**VALOR**
Daño Físico	3
Compromiso de la Información	3
Fallas Técnicas	3
Acciones no autorizadas	4
Impacto	4

Se procede a ingresar los valores en el prototipo desarrollado en Python para los modelos basados en lógica difusa y la red neuronal:

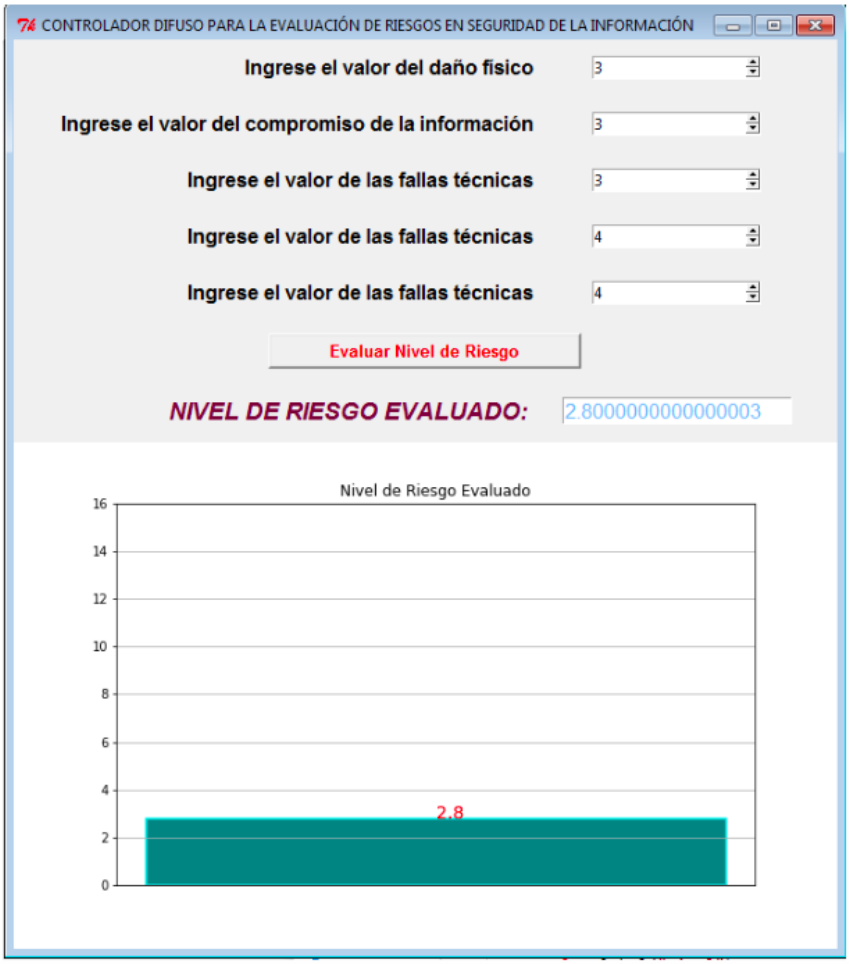

Figura 12. Simulación escenario 1 prototipo en Python modelo lógica difusa.

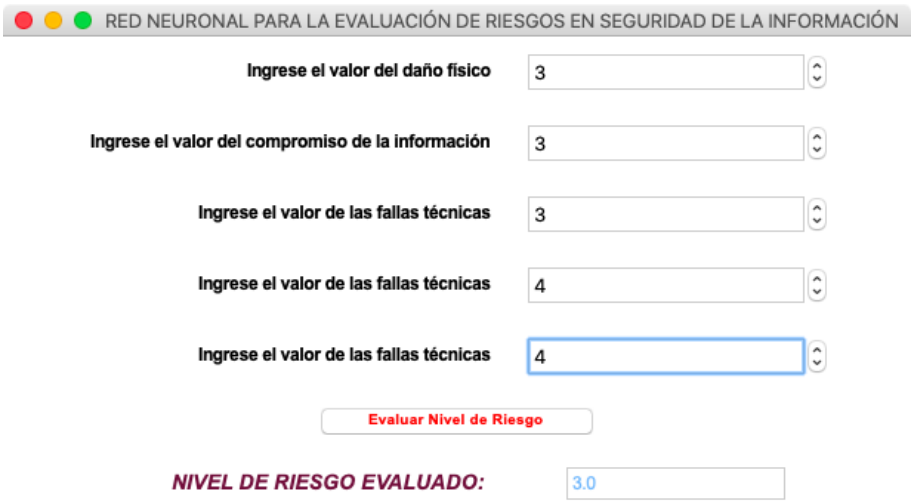

Figura 13. Simulación escenario 1 prototipo en Python modelo red neuronal.

6.1.2 Escenario de Prueba 2

Este escenario se toma el activo de Información "SERVIDORES" tomando como fuente la calificación de los expertos acorde a los datos que se deben ingresar al prototipo:

Tabla 11. Parámetros entrada escenario de prueba 2

Valoración de Amenazas e impacto para el activo de Información SERVIDORES

AMENAZA	**VALOR**
Daño Físico	12
Compromiso de la Información	12
Fallas Técnicas	3
Acciones no autorizadas	1
Impacto	10

Se procede a ingresar los valores en el prototipo desarrollado en Python para los modelos basados en lógica difusa y la red neuronal:

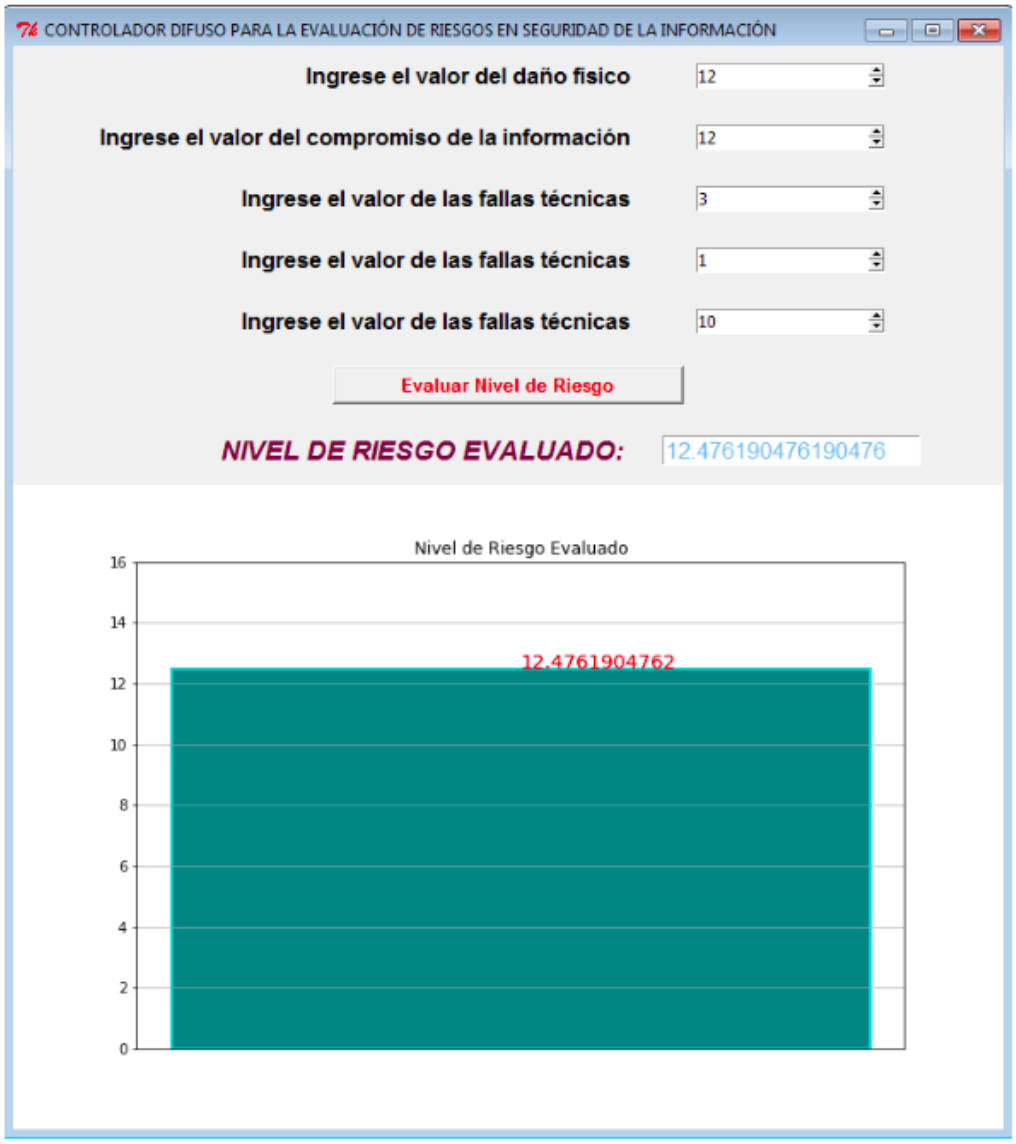

Figura 14. Simulación escenario 2 prototipo en Python modelo lógica difusa.

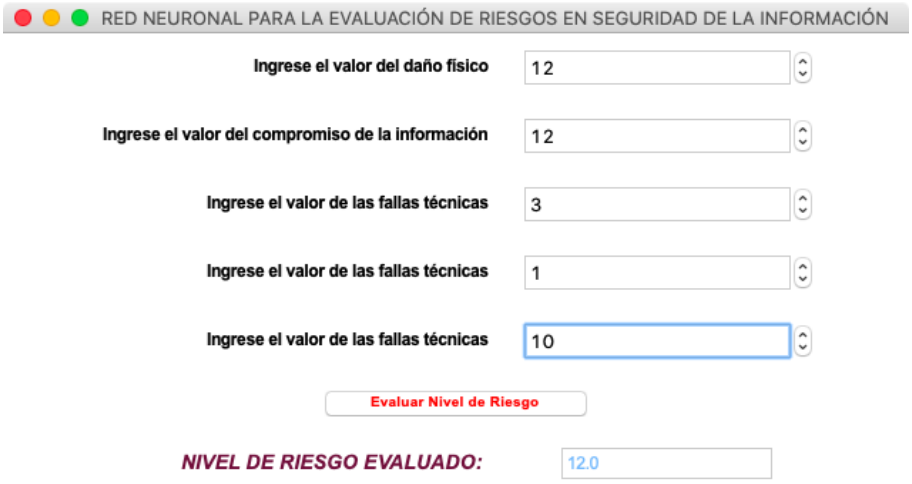

Figura 15. Simulación escenario 2 prototipo en Python modelo red neuronal.

6.2 Interpretación de resultados

Los resultados obtenidos de la comparación del modelo basado en lógica difusa y el modelo basado en redes neuronales según los datos de los niveles de riesgo obtenidos son muy similares, las diferencias son decimas que para el caso de estudio no son significativos pues están en el mismo rango de valores de clasificación dados por los expertos.

7 Conclusiones

Para las empresas los activos de información son de vital importancia y requieren una adecuada gestión del riesgo pasando desde su identificación hasta la auditoría y ajuste de los controles definidos en cada uno. La inteligencia artificial juega un papel importante en la valoración del riesgo, y en este caso, las redes neuronales permiten determinar el nivel de riesgo al que está expuesto un activo, ayudando en la toma de decisiones para darle un adecuado tratamiento.

Se logró diseñar un modelo de red neuronal multicapa para medir el nivel de riesgo mediante los datos proporcionados por un experto, brindándole al usuario la posibilidad de una mayor interpretación a la subjetividad envuelta en el análisis de los expertos, favoreciendo el proceso de creación del Plan de continuidad del negocio.

El modelo basado en redes neuronales implementado en el desarrollo de este proyecto se puso a prueba modificando algunos parámetros con el fin de maximizar la precisión en el valor de salida de la red, encontrando valores de precisión por encima del 95% que permitieron hacer una clasificación eficiente ante diversos escenarios de prueba.

Ante diversos escenarios tanto las redes neuronales como la lógica difusa permiten clasificar el nivel de riesgo de un activo de información estando en los mismos rangos de respuesta, lo que indica que ambas soluciones se pueden utilizar en el contexto de la evaluación de riesgos en seguridad de la información obteniendo una respuesta similar a la dada por un experto en el tema.

El uso de técnicas aplicadas de la inteligencia artificial en la evaluación de riesgos en seguridad de la información permite robustecer el modelo, convirtiéndose en un sistema de alertas tempranas donde los expertos pueden realizar simulaciones de diferentes posibles escenarios y poder preparar planes de contingencia o de tratamiento de los riesgos cuando realmente se materialice alguna amenaza para garantizar así la continuidad del negocio.

Las redes neuronales brindan una respuesta a cada entrada dada así no haya sido presentada previamente ya que cuentan con un mecanismo automático de aprendizaje, por otro lado, la lógica

difusa debe tener en su base de conocimientos la suficiente cantidad de reglas para dar una respuesta ante cualquier escenario propuesto, implicando así un mayor nivel de detalle en la definición de estas.

8 Recomendaciones y Trabajo futuro

Se plantea como trabajo futuro probar la implementación del modelo basado en redes neuronales, usando otros algoritmos mejorados de aprendizaje supervisados tales como: método elástico de retropropagación, regla delta de generalización, heurísticas de aproximación, métodos de orden de convergencia mayor o el algoritmo de Levenberg-Marquardt que se usa por defecto en el ToolBox de redes neuronales que usa MatLab.

También es viable implementar soluciones incluso con otras técnicas de aprendizaje como las no supervisadas o la probabilísticas si se quiere. Dentro de clasificación de redes no supervisadas sería muy interesante ver como se dan las salidas de la red usando cualquiera de sus modelos de aprendizaje, tales como: aprendizaje Hebbiano, aprendizaje cooperativo y competitivo usando por ejemplo un enfoque de mapas auto organizados (SOM), en el cual, estos últimos son capaces de manipular grandes volúmenes de datos para realizar procesos de clasificación consumiendo menos recursos computacionales que un algoritmo de aprendizaje supervisado.

Por último, se recomendaría a futuro investigar en la solución que ofrecen modelos híbridos como por ejemplo los modelos Neuro-Difusos (Neuro-Fuzzy) que están enmarcados dentro de la computación flexible (Soft-Computing), la cual engloba un conjunto de técnicas que tienen en común la robustez en el manejo de la información imprecisa e incierta que existe en los problemas relacionados con el mundo real, entre ellos, la toma de decisiones y los procesos de clasificación.

9 Bibliografía

Angarita Vivas, A. A., & Tabares Isaza, C. A. (2015). *DEFINICIÓN DE UN MODELO DE MEDICIÓN DE RIESGOS DE LA SEGURIDAD DE LA INFORMACIÓN APLICANDO LÓGICA DIFUSA Y SISTEMAS BASADOS EN EL CONOCIMIENTO. CASO DE ESTUDIO: EMPRESA DE SERVICIOS PUBLICOS*. Universidad Tecnológica de Pereira.

Antonio J Serrano, Emilio Soria, J. D. M. (2010). *Redes Neuronales Artificiales*. Recuperado a partir de http://ocw.uv.es/ingenieria-y-arquitectura/1-2/libro_ocw_libro_de_redes.pdf

Cardona Ochoa, L. G. (2015). *SISTEMA DIFUSO PARA LA EVALUACIÓN DE UN MODELO DE RIESGO DE MERCADO EN UN PORTAFOLIO DE DEUDA PÚBLICA EN COLOMBIA*. Universidad Eafit. Recuperado a partir de https://repository.eafit.edu.co/bitstream/handle/10784/7431/LuisGuillermo_CardonaOchoa_2015.pdf?sequence=2%26isAllowed=y

Casares San José-Martí, I., & Lizarzaburu Bolaños, E. R. (2016). *INTRODUCCIÓN A LA GESTIÓN INTEGRAL DE RIESGOS EMPRESARIALES ENFOQUE: ISO 31000* (Primera edición). Lima: Platinum. Recuperado a partir de https://fundacioninade.org/sites/inade.org/files/web_libro_3_la_gestion_integral_de_riesgos_empresariales.pdf

ICONTEC. (2011a). *COMPENDIO DE NORMAS DE GESTION DEL RIESGO*. (ICONTEC, Ed.). Bogotá, Colombia: ICONTEC Internacional.

ICONTEC. (2011b). *NORMA TÉCNICA COLOMBIANA NTC-ISO 31000 GESTIÓN DEL RIESGO. PRINCIPIOS Y DIRECTRICES*. (ICONTEC, Ed.). Bogotá: ICONTEC Internacional.

ICONTEC. (2015). *Compendio Seguridad de la Información*. (ICONTEC, Ed.) (Segunda). Bogotá: ICONTEC Internacional.

Jang, J.-S. R., Sun, C.-T. y M. (1997). *Neuro-fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence*. (N. Prentice & Hall, Eds.).

Lofti A Zadeh. (1988). Fuzzy Logic, 83. Recuperado a partir de https://ieeexplore.ieee.org/document/53/authors#authors

Moncayo Racines, D. E. (2014). *MODELO DE EVALUACIÓN DE RIESGOS EN ACTIVOS DE TIC'S PARA PEQUEÑAS Y MEDIANAS EMPRESAS DEL SECTOR AUTOMOTRIZ*.

Escuela Politécnica Nacional. Recuperado a partir de http://bibdigital.epn.edu.ec/handle/15000/8499

Neira, A. L., & Spohr, J. R. (2012). ISO 27000.es. Recuperado a partir de http://www.iso27000.es

Pérez Ramírez, F. O., & Fernández Castaño, H. (2007). LAS REDES NEURONALES Y LA EVALUACIÓN DEL RIESGO DE CRÉDITO*. *Revista Ingenierías Universidad de Medellín*, 77–91. Recuperado a partir de http://www.scielo.org.co/pdf/rium/v6n10/v6n10a07.pdf

Rosenblatt, F. (1959). *PRINCIPLES OF NEURODYNAMICS*. Cornell University, Ithaca, N.Y. Recuperado a partir de http://www.dtic.mil/dtic/tr/fulltext/u2/256582.pdf

Saavedra García, M. L., & Saavedra García, M. J. (2010). Modelos para Medir el riesgo de crédito de la banca. *Cuad. Adm. Bogotá*, *23*(40), 295–319. Recuperado a partir de http://www.scielo.org.co/pdf/cadm/v23n40/v23n40a13.pdf

Sakti, I. (2014). Methodology of fuzzy logic with mamdani fuzzy models applied to the microcontroller. Information Technology, Computer and Electrical Engineering (Icitacee), 1st International Conference on IEEE (pp. 93-98).

Team, T. scikit-image. (2016). The scikit-fuzzy Documentation.

Anexos

Anexo 1. Datos de entrenamiento de la red neuronal

Tabla 12. Datos de entrenamiento de la Red Neuronal

DF	CI	FT	ANA	IMP	OUT	DF	CI	FT	ANA	IMP	OUT	DF	CI	FT	ANA	IMP	OUT
12	4	10	5	3	3	5	10	7	8	2	7	9	8	7	8	11	10
4	4	2	1	7	3	2	7	7	2	8	7	9	11	6	12	12	10
3	6	4	12	2	3	6	6	4	10	6	7	8	9	7	7	12	10
1	12	5	4	3	3	2	12	5	6	1	7	3	11	7	11	11	10
7	3	3	2	2	3	5	6	8	6	5	7	1	6	11	11	12	10
2	3	4	12	2	3	3	1	5	10	10	7	11	11	3	4	11	10
4	12	6	4	1	3	1	6	10	6	9	7	9	10	7	8	12	10
1	11	1	6	4	3	7	4	3	6	10	7	7	12	7	9	11	10
8	1	1	2	3	3	4	7	3	9	6	7	11	12	7	11	11	10
2	4	3	4	1	3	5	10	5	12	1	7	1	4	12	11	11	10
4	4	6	8	3	3	6	8	5	3	9	7	5	10	4	11	12	10
4	1	5	2	2	3	6	12	6	8	4	7	11	4	11	5	11	10
1	12	8	4	6	3	7	4	2	11	2	7	12	11	4	10	2	10
8	3	11	3	2	3	9	4	9	6	2	7	4	12	9	6	10	10
2	4	1	7	4	3	2	1	6	12	2	7	11	6	12	6	11	10
3	4	2	6	3	3	4	6	11	7	5	7	11	12	1	11	12	10
4	4	3	1	12	3	4	1	6	12	6	7	11	7	11	12	5	10
7	2	1	6	3	3	1	9	2	2	9	7	12	10	5	8	11	10
4	3	3	3	4	3	5	10	5	11	2	7	2	12	11	6	11	10
4	4	6	2	9	3	2	6	1	12	6	7	5	12	11	4	11	10
1	4	2	6	4	3	10	2	6	4	9	7	7	11	11	4	10	10
3	4	3	1	4	3	2	10	3	8	6	7	6	11	9	7	12	10
4	4	9	6	2	3	8	3	6	2	11	7	5	11	10	7	12	10
4	4	5	2	3	3	5	7	6	11	3	7	10	11	4	8	10	10
5	5	7	1	4	3	5	7	11	6	4	7	11	1	9	3	12	10
2	3	8	2	4	3	3	6	4	4	12	7	4	12	7	8	10	10
1	6	4	3	1	3	10	12	3	2	4	7	6	12	10	3	12	10
3	4	7	4	1	3	4	5	6	7	9	7	9	11	3	11	12	10
3	11	4	4	1	3	11	2	2	11	3	7	11	12	6	5	11	10
2	1	2	9	3	3	12	1	8	10	2	7	11	11	8	10	12	10
4	6	4	5	2	3	5	2	3	10	10	7	7	11	10	7	12	10
3	2	8	1	7	3	1	5	4	12	6	7	8	11	8	10	11	10
3	6	5	3	2	3	2	8	6	9	3	7	11	11	10	10	11	10
7	2	10	3	3	3	8	2	6	6	6	7	1	11	9	8	10	10
4	1	11	2	3	3	10	4	6	6	2	7	9	11	8	6	10	10
3	7	6	1	1	3	6	9	10	1	9	7	12	11	5	8	11	10
4	4	6	1	12	3	1	9	10	2	6	7	7	4	8	12	11	10
2	2	4	5	3	3	2	2	7	9	4	7	11	10	5	6	12	10
4	3	8	3	4	3	2	2	7	7	2	7	7	11	8	6	11	10
4	10	7	2	2	3	10	3	3	6	11	7	9	10	11	7	12	10
2	7	5	4	2	3	4	6	2	9	6	7	4	10	1	11	11	10
6	6	4	4	1	3	6	8	2	9	6	7	11	11	5	3	12	10
4	4	11	4	3	3	10	1	6	1	10	7	7	7	11	10	10	10
5	12	12	4	4	3	6	2	3	6	8	7	8	4	10	11	12	10
6	3	2	3	3	3	10	6	8	5	6	7	7	11	11	9	11	10
6	1	5	3	3	3	1	2	6	6	3	7	4	10	11	7	10	10
4	1	1	2	2	3	5	1	8	12	2	7	11	11	5	10	11	10
2	6	1	4	2	3	5	6	10	3	8	7	11	9	11	12	3	10

2	9	1	3	5	3	6	6	11	7	2	7	8	9	11	7	11	10
2	4	9	3	1	3	7	10	11	2	3	7	1	11	11	8	12	10
8	10	1	1	1	3	6	6	3	10	10	7	8	10	1	11	12	10
DF	**CI**	**FT**	**ANA**	**IMP**	**OUT**	**DF**	**CI**	**FT**	**ANA**	**IMP**	**OUT**	**DF**	**CI**	**FT**	**ANA**	**IMP**	**OUT**
4	4	9	10	3	3	6	8	3	2	10	7	6	11	8	7	10	10
10	1	1	3	3	3	6	7	2	5	8	7	4	5	6	11	12	10
9	1	2	6	3	3	1	12	10	3	4	7	11	1	9	6	11	10
3	8	1	3	8	3	6	5	5	3	7	7	5	11	10	9	12	10
4	8	8	3	4	3	3	6	9	2	6	7	10	10	3	11	11	10
3	5	3	1	7	3	3	6	11	3	6	7	10	11	7	11	12	10
4	3	5	4	10	3	4	1	10	12	2	7	12	10	11	10	11	10
10	5	2	4	2	3	6	9	8	3	6	7	7	12	5	10	10	10
1	5	2	8	2	3	9	5	8	6	2	7	6	10	4	11	11	10
4	9	2	4	4	3	6	7	8	2	8	7	3	10	12	7	11	10
12	1	4	6	1	3	1	6	6	6	12	7	9	11	9	6	12	10
8	4	6	3	1	3	9	10	5	11	5	7	4	10	10	11	11	10
1	2	1	4	10	3	2	6	2	1	10	7	8	11	3	8	11	10
4	1	1	7	2	3	2	7	6	4	6	7	10	12	11	10	11	10
3	7	11	2	4	3	6	6	12	11	6	7	10	10	12	3	11	10
11	3	1	3	4	3	2	6	1	9	8	7	2	11	11	11	12	10
8	1	8	2	2	3	1	11	7	6	6	7	1	10	9	11	12	10
1	9	6	2	2	3	10	3	3	12	2	7	8	11	8	9	10	10
2	1	7	4	3	3	4	8	2	6	7	7	7	12	9	7	10	10
3	7	7	4	1	3	2	2	10	2	9	7	4	3	8	11	12	10
3	2	8	4	1	3	3	5	2	10	6	7	4	8	1	5	11	10
6	3	2	1	4	3	5	2	12	9	4	7	9	8	12	8	11	10
11	2	5	3	3	3	5	10	9	1	8	7	10	10	7	8	11	10
2	2	6	1	3	3	8	12	6	6	6	7	12	1	9	2	11	10
1	7	2	6	1	3	2	11	11	7	2	7	5	12	7	9	10	10
4	10	9	2	1	3	9	1	2	9	12	7	11	4	11	4	12	10
12	3	2	3	2	3	4	6	2	10	7	7	9	10	11	6	10	10
2	1	4	1	8	3	6	9	3	6	5	7	10	12	11	1	12	10
10	1	4	1	4	3	2	12	10	4	6	7	10	5	8	12	12	10
6	11	2	1	3	3	2	5	9	3	9	7	9	8	12	10	10	10
3	4	6	2	4	3	5	2	11	10	4	7	11	1	10	3	11	10
3	3	6	4	3	3	7	8	10	2	4	7	8	11	10	7	12	10
1	6	4	2	3	3	7	6	10	6	3	7	11	10	5	2	12	10
1	5	1	12	4	3	6	11	10	10	5	7	8	9	10	9	11	10
1	9	2	5	4	3	8	7	9	8	2	7	5	11	4	10	11	10
4	2	3	8	3	3	6	6	9	10	2	7	7	11	5	6	12	10
4	6	2	1	6	3	2	6	3	1	11	7	5	5	8	11	11	10
1	2	2	6	4	3	1	2	2	9	7	7	6	11	3	11	12	10
1	9	3	1	2	3	1	9	2	8	6	7	7	7	8	7	11	10
11	4	2	3	3	3	1	9	9	1	6	7	5	4	7	11	11	10
2	6	5	4	4	3	1	11	7	6	5	7	10	4	9	11	12	10
3	9	1	3	2	3	10	10	2	1	3	7	7	12	11	5	10	10
2	8	4	10	2	3	4	7	8	3	6	7	8	10	11	1	11	10
10	2	2	5	3	3	6	3	10	9	1	7	5	11	5	9	12	10
12	2	2	8	2	3	3	6	6	10	6	7	9	11	1	12	11	10
4	7	4	10	2	3	2	10	6	11	3	7	12	3	12	6	11	10
1	3	4	12	4	3	10	9	1	6	9	7	3	12	12	11	11	10
6	2	4	5	2	3	6	10	5	1	8	7	7	11	10	9	10	10
2	7	1	3	6	3	8	8	7	1	6	7	11	12	3	7	12	10
8	12	10	4	7	4	6	3	4	8	11	8	7	11	12	11	9	12
4	2	4	4	1	4	7	8	5	3	12	8	6	5	12	11	10	12
12	7	11	2	1	4	4	4	6	11	7	8	12	12	8	12	9	12
8	1	1	8	3	4	5	8	8	10	11	8	6	9	1	12	12	12
8	4	1	4	12	4	9	7	2	9	11	8	10	12	8	5	10	12
4	1	10	3	4	4	2	8	11	12	1	8	12	9	11	9	12	12
1	5	1	7	2	4	6	4	11	7	1	8	5	12	1	3	12	12
5	12	2	1	2	4	3	7	12	5	10	8	8	1	8	12	10	12
5	4	8	12	1	4	11	8	6	7	3	8	9	1	11	12	12	12

12	7	9	1	4	4	8	4	1	11	4	8	9	9	12	1	12	12
11	1	2	7	4	4	3	8	9	8	12	8	11	12	12	12	9	12
5	9	1	1	2	4	5	12	6	12	1	8	5	9	12	10	10	12
1	5	9	1	3	4	8	1	11	5	7	8	10	12	4	1	12	12
10	3	1	1	1	4	11	10	2	10	6	8	4	12	5	1	12	12

DF	**CI**	**FT**	**ANA**	**IMP**	**OUT**	**DF**	**CI**	**FT**	**ANA**	**IMP**	**OUT**	**DF**	**CI**	**FT**	**ANA**	**IMP**	**OUT**
7	1	12	1	4	4	6	11	12	7	8	8	12	12	7	9	1	12
8	4	8	4	1	4	9	11	11	11	5	8	12	11	1	7	12	12
3	4	12	4	7	4	5	1	11	11	9	8	8	9	12	11	11	12
7	1	5	1	2	4	7	6	5	9	7	8	12	7	10	1	12	12
3	9	1	1	6	4	9	6	10	8	3	8	10	12	7	5	12	12
5	11	1	1	2	4	5	12	12	1	5	8	8	3	11	12	9	12
9	3	3	8	4	4	12	11	7	6	1	8	4	12	1	5	12	12
2	1	8	4	10	4	7	6	9	9	9	8	11	12	12	8	9	12
8	1	3	1	4	4	1	4	9	11	9	8	8	9	12	1	11	12
12	7	4	7	2	4	3	1	9	10	12	8	12	6	2	9	12	12
2	1	8	4	7	4	4	9	7	9	3	8	8	1	11	12	9	12
3	11	1	1	3	4	10	1	10	1	8	8	3	9	2	12	12	12
2	11	3	1	4	4	11	10	4	11	5	8	9	12	11	5	9	12
7	2	8	1	4	4	8	5	11	7	2	8	4	12	9	10	9	12
10	12	4	9	4	4	10	8	11	8	8	8	12	3	12	7	12	12
8	12	2	9	4	4	6	9	7	10	9	8	12	9	2	5	12	12
8	2	3	1	2	4	11	2	5	7	6	8	12	12	5	12	3	12
7	2	4	8	3	4	11	9	4	8	5	8	12	9	7	10	12	12
8	2	4	7	4	4	6	7	11	4	11	8	4	10	12	9	9	12
9	1	5	1	3	4	8	11	3	9	4	8	12	7	3	9	12	12
1	4	9	4	1	4	12	11	1	8	7	8	12	6	9	6	12	12
4	12	7	4	8	4	10	7	5	6	5	8	3	1	9	12	12	12
10	3	3	8	4	4	5	9	11	12	5	8	12	12	1	4	12	12
11	4	2	7	3	4	5	5	10	8	12	8	12	9	5	12	12	12
3	6	1	12	1	4	11	5	4	6	10	8	12	12	11	9	9	12
1	4	12	3	1	4	5	1	4	11	8	8	8	6	12	9	9	12
1	2	1	1	5	4	7	6	12	1	3	8	12	12	9	3	9	12
4	12	1	7	1	4	7	4	12	3	7	8	8	12	12	7	9	12
8	4	3	1	12	4	12	5	4	11	11	8	11	4	12	1	12	12
4	9	4	1	4	4	10	10	5	11	7	8	4	11	12	12	9	12
11	1	4	8	2	4	10	9	9	11	3	8	7	12	3	1	12	12
5	4	1	1	12	4	12	4	6	5	8	8	5	12	11	8	9	12
11	12	4	10	4	4	5	8	3	12	9	8	12	10	9	5	12	12
4	1	8	6	3	4	10	11	9	3	9	8	8	9	12	1	12	12
4	7	5	2	1	4	11	4	11	1	6	8	10	1	8	12	12	12
3	1	5	12	4	4	12	8	6	5	10	8	12	12	11	2	9	12
3	7	12	3	1	4	8	3	10	10	6	8	9	12	10	4	9	12
12	4	9	3	2	4	5	8	10	3	11	8	1	9	6	12	12	12
6	1	1	7	3	4	9	4	3	10	5	8	12	5	9	6	12	12
11	3	8	3	1	4	12	3	8	2	9	8	8	7	12	10	9	12
1	4	1	5	3	4	12	9	4	6	6	8	2	12	7	4	12	12
7	3	8	1	3	4	4	5	5	8	12	8	1	12	3	9	12	12
12	7	4	8	3	4	11	11	8	3	12	8	5	5	12	9	10	12
4	5	9	1	1	4	1	8	7	9	10	8	12	9	7	3	12	12
9	4	5	1	2	4	6	3	10	7	9	8	12	8	12	9	3	12
5	3	3	4	1	4	10	5	8	5	11	8	3	12	8	5	12	12
7	1	11	3	1	4	12	6	9	5	6	8	11	6	3	12	12	12
12	5	9	4	1	4	8	10	11	2	7	8	11	7	12	3	12	12
11	4	8	2	4	4	2	5	12	5	11	8	10	12	11	8	9	12
1	4	11	12	3	4	5	4	9	7	7	8	5	9	12	10	9	12
8	6	1	4	3	4	1	7	2	9	10	8	10	12	4	9	9	12
4	5	5	4	1	4	2	11	5	10	8	8	2	9	7	12	12	12
4	8	3	4	1	4	11	10	7	2	7	8	6	12	1	3	12	12
7	6	3	1	1	4	12	7	5	4	5	8	9	9	12	3	12	12
4	12	11	1	4	4	11	10	6	1	4	8	12	9	2	9	12	12
11	1	2	8	1	4	5	1	7	5	8	8	4	12	10	8	9	12

8	5	8	1	4	4	11	6	7	5	5	8	11	12	11	4	9	12
11	3	9	4	4	4	8	5	7	9	8	8	12	8	12	11	1	12
4	11	1	2	1	4	9	11	12	9	5	8	11	12	4	1	12	12
1	5	1	4	4	4	7	9	4	3	8	8	4	9	12	12	9	12
12	3	8	1	4	4	10	2	11	1	11	8	12	6	9	1	12	12
9	1	10	4	1	4	12	1	1	10	11	8	12	12	4	11	9	12
7	1	2	1	3	4	3	3	7	9	8	8	12	12	8	10	9	12

DF	**CI**	**FT**	**ANA**	**IMP**	**OUT**	**DF**	**CI**	**FT**	**ANA**	**IMP**	**OUT**	**DF**	**CI**	**FT**	**ANA**	**IMP**	**OUT**
8	2	11	1	1	4	8	9	3	11	12	8	11	12	9	4	11	12
4	8	1	12	1	4	7	1	10	1	12	8	11	6	12	1	12	12
1	4	4	11	1	4	11	4	8	10	6	8	12	10	12	12	1	12
4	9	1	1	1	4	11	11	8	3	11	8	12	9	10	9	12	12
3	12	1	1	6	4	7	6	12	1	11	8	11	8	12	9	12	12
4	1	1	5	3	4	10	9	2	3	4	8	11	12	9	5	10	12
3	1	8	4	10	4	8	7	12	10	3	8	12	9	11	5	12	12
1	5	10	1	4	4	7	2	8	7	11	8	7	10	12	8	9	12
1	4	10	1	4	4	1	8	6	5	12	8	8	5	12	9	9	12
1	1	6	4	9	4	11	8	10	6	10	8	12	7	3	11	12	12
4	4	11	4	8	4	4	9	4	11	8	8	10	12	9	6	9	12
5	4	4	4	1	4	11	10	12	7	5	8	5	9	12	8	12	12
4	2	4	5	1	4	1	1	4	1	10	8	4	5	11	12	12	12
10	4	7	1	3	4	12	9	1	11	5	8	8	2	11	12	9	12
9	8	9	4	4	4	12	10	8	2	4	8	11	12	12	7	9	12
7	2	4	1	4	4	8	7	6	2	8	8	8	9	12	10	9	12
1	4	5	1	2	4	11	4	8	4	9	8	12	12	7	12	9	12
7	1	3	1	2	4	8	7	5	8	10	8	12	10	7	1	12	12
2	1	8	4	11	4	2	11	8	5	8	8	1	6	7	12	12	12
11	3	12	3	1	4	5	11	9	10	8	8	11	1	8	12	12	12
4	1	1	1	6	4	2	6	1	9	12	8	8	1	8	12	11	12
8	8	3	3	4	4	10	2	12	4	9	8	1	9	9	12	12	12
8	9	4	1	1	4	11	1	10	7	5	8	12	10	7	3	12	12
1	6	9	3	6	6	3	10	10	3	12	9	12	11	10	2	12	13
4	5	5	4	7	6	2	11	5	3	11	9	7	12	9	10	12	13
10	5	6	3	2	6	10	12	2	9	12	9	7	12	9	11	12	13
2	5	7	1	11	6	2	12	12	4	10	9	10	12	1	12	12	13
5	5	2	5	6	6	6	3	10	10	11	9	9	12	8	12	12	13
3	10	4	5	3	6	7	11	6	7	11	9	9	8	10	8	12	13
3	4	3	5	10	6	6	6	10	7	11	9	12	11	11	12	12	13
3	11	1	6	3	6	9	5	6	10	10	9	10	12	9	10	12	13
12	3	1	12	2	6	6	11	5	7	11	9	7	11	12	9	10	13
4	4	8	5	11	6	6	12	2	6	12	9	11	12	12	5	10	13
3	4	6	9	2	6	8	6	9	10	11	9	1	12	6	4	12	13
12	11	4	1	2	6	10	10	6	3	10	9	12	2	8	6	12	13
6	7	5	2	5	6	12	10	8	9	10	9	9	12	4	12	12	13
9	6	5	10	1	6	2	9	11	4	12	9	4	10	11	8	11	13
7	4	3	5	12	6	8	9	6	7	10	9	6	12	8	6	11	13
6	7	4	5	1	6	11	7	10	12	5	9	6	12	12	9	11	13
5	12	6	3	4	6	1	9	12	3	10	9	1	12	9	10	12	13
2	1	11	2	7	6	9	2	8	10	10	9	12	4	12	6	12	13
3	7	6	2	5	6	6	11	4	7	11	9	10	12	8	9	12	13
1	3	4	5	9	6	9	11	7	12	10	9	8	4	10	12	10	13
4	5	7	2	5	6	10	12	4	10	1	9	6	3	9	12	12	13
12	6	10	4	5	6	11	12	6	11	1	9	10	12	12	4	11	13
1	2	1	10	8	6	8	11	6	2	12	9	8	12	2	4	12	13
7	9	5	2	5	6	10	4	6	10	10	9	6	12	5	12	12	13
3	4	3	10	5	6	10	8	9	6	11	9	2	12	12	12	11	13
3	6	6	3	5	6	12	10	10	11	5	9	4	12	10	11	12	13
5	5	6	7	4	6	10	12	2	2	10	9	3	11	12	11	12	13
11	5	1	6	2	6	9	9	6	10	12	9	12	10	1	10	12	13
1	5	1	10	6	6	12	5	10	4	10	9	10	12	11	4	12	13
6	9	8	1	3	6	1	12	3	6	10	9	12	4	12	4	12	13
12	6	4	9	1	6	1	11	10	3	12	9	11	12	7	9	12	13

5	3	4	3	11	6	10	9	7	9	10	9	12	6	10	4	12	13
11	6	3	5	1	6	7	10	8	10	10	9	5	12	4	7	12	13
5	1	2	5	7	6	3	7	2	5	11	9	8	12	9	8	12	13
3	12	2	7	3	6	12	6	12	11	2	9	2	12	11	1	12	13
5	12	9	11	5	6	12	10	10	10	2	9	11	12	10	10	12	13
3	1	3	9	12	6	10	2	8	12	10	9	11	12	2	1	12	13
1	8	12	5	4	6	7	6	9	8	10	9	6	12	9	8	10	13
7	2	2	9	5	6	7	2	10	10	11	9	9	12	6	5	12	13
4	3	2	5	11	6	1	6	12	3	11	9	3	12	9	9	12	13
DF	**CI**	**FT**	**ANA**	**IMP**	**OUT**	**DF**	**CI**	**FT**	**ANA**	**IMP**	**OUT**	**DF**	**CI**	**FT**	**ANA**	**IMP**	**OUT**
2	3	9	1	11	6	10	6	12	11	6	9	12	3	10	6	12	13
3	5	1	6	5	6	6	11	6	3	12	9	8	7	12	6	10	13
7	6	2	4	5	6	8	6	1	7	10	9	8	11	12	2	11	13
5	12	10	2	5	6	7	12	6	5	11	9	12	12	7	8	12	13
4	2	6	9	5	6	1	12	7	7	10	9	8	12	7	8	12	13
5	7	5	3	7	6	9	12	10	2	10	9	10	11	12	9	12	13
5	7	2	4	6	6	6	7	2	6	12	9	10	6	12	5	12	13
6	5	11	8	5	6	8	9	10	9	10	9	6	12	5	6	12	13
3	9	3	11	2	6	4	6	7	11	11	9	9	12	12	2	11	13
2	1	8	5	1	6	6	8	9	7	10	9	8	12	10	6	11	13
1	10	9	5	4	6	7	10	1	1	12	9	3	10	2	12	12	13
1	5	9	1	11	6	7	12	2	9	10	9	12	12	5	7	11	13
4	3	1	7	7	6	2	12	12	5	10	9	5	12	4	2	12	13
2	3	5	9	1	6	9	11	2	4	10	9	2	6	9	12	12	13
7	10	4	5	1	6	7	11	2	9	10	9	4	12	6	11	12	13
9	6	7	12	1	6	2	9	12	3	10	9	3	10	12	9	12	13
4	8	10	6	1	6	4	4	8	10	10	9	5	11	4	12	12	13
5	1	12	5	1	6	7	12	7	10	10	9	11	12	8	12	12	13
3	1	5	9	8	6	7	6	9	11	12	9	4	11	9	11	12	13
6	1	11	1	10	6	10	6	2	7	11	9	12	2	10	2	12	13
1	9	10	7	3	6	6	7	8	6	12	9	4	12	6	12	12	13
5	3	3	1	12	6	6	10	12	2	12	9	9	12	4	5	12	13
3	9	5	11	5	6	11	10	11	11	2	9	4	12	12	4	12	13
11	12	4	1	1	6	4	9	1	6	12	9	5	12	2	1	12	13
8	3	8	5	4	6	10	10	10	11	6	9	7	12	12	2	12	13
4	12	5	11	4	6	3	6	9	10	11	9	7	12	6	4	12	13
1	9	5	5	2	6	10	8	11	12	2	9	1	12	12	5	12	13
10	3	5	10	1	6	11	12	5	11	2	9	8	12	10	7	11	13
5	7	3	6	5	6	7	8	9	10	12	9	12	12	12	12	12	13
3	10	1	6	5	6	6	11	5	7	10	9	8	12	9	6	10	13
2	3	7	5	11	6	9	9	6	6	11	9	7	12	4	9	12	13
4	7	5	8	5	6	3	10	2	9	10	9	12	3	10	4	12	13
3	7	10	12	1	6	7	5	7	10	11	9	8	11	11	8	12	13
3	1	10	5	12	6	10	6	10	6	11	9	4	12	11	11	10	13
5	5	12	11	2	6	8	11	9	12	10	9	7	7	12	10	12	13
1	1	10	5	2	6	7	3	1	6	10	9	4	2	8	12	12	13
6	5	4	2	11	6	10	2	9	10	10	9	7	12	10	8	11	13
2	8	5	2	7	6	11	12	2	11	2	9	9	10	12	10	12	13
2	5	4	2	9	6	4	10	3	7	11	9	4	10	12	12	12	13
8	2	4	2	11	6	3	6	10	10	11	9	3	6	6	12	12	13
2	10	9	1	1	6	3	4	7	10	12	9	8	12	6	8	11	13
5	12	7	5	4	6	4	12	12	3	10	9	4	12	8	8	10	13
2	8	4	4	9	6	10	9	12	12	2	9	12	12	12	10	12	13
3	9	5	8	3	6	11	12	6	2	10	9	11	12	5	8	10	13
5	3	7	3	5	6	6	7	7	6	10	9	4	10	12	10	12	13
7	2	2	10	1	6	11	10	11	11	6	9	8	12	7	11	12	13
5	11	5	4	1	6	10	3	10	5	10	9	4	12	10	2	12	13
2	1	2	5	10	6	2	12	4	11	10	9	8	12	2	12	11	13
4	7	4	1	12	6	10	6	6	9	11	9	10	10	12	4	12	13
3	5	5	5	4	6	4	12	6	7	10	9	2	6	10	12	12	13
3	9	1	11	2	6	2	10	6	7	10	9	12	11	12	7	12	13
6	9	7	1	3	6	1	12	2	5	11	9	12	8	1	12	12	13

3	9	12	3	5	6	3	10	2	8	11	9	2	11	12	8	12	13
2	1	5	9	3	6	7	11	10	3	12	9	12	12	10	4	10	13
11	4	3	9	1	6	10	2	10	5	12	9	4	11	10	10	12	13
5	6	2	8	4	6	11	9	11	11	6	9	12	12	6	3	12	13
5	1	6	1	9	6	9	10	1	1	11	9	1	12	7	12	12	13
8	1	5	5	11	6	8	10	6	5	12	9	3	11	12	10	12	13
5	7	5	1	10	6	1	10	4	5	12	9	4	11	12	8	10	13
5	10	1	5	4	6	9	6	10	10	10	9	6	10	12	10	10	13

- DF: Daño físico
- CI: Compromiso de la información
- FT: Fallas técnicas
- ANA: Acceso no autorizado
- IMP: Impacto
- OUT: Salida de la red

Anexo 2. Parámetros de entrenamiento de la red neuronal

Tomado de:
https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html
Clasificador de perceptrón multicapa (sklearn.neural_network.MLPClassifier)

class sklearn.neural_network.MLPClassifier(hidden_layer_sizes=(100,), activation='relu', solver='adam', alpha=0.0001, batch_size='auto', learning_rate='constant', learning_rate_init=0.001, power_t=0.5, max_iter=200, shuffle=True, random_state=None, tol=0.0001, verbose=False, warm_start=False, momentum=0.9, nesterovs_momentum=True, early_stopping=False, validation_fraction=0.1, beta_1=0.9, beta_2=0.999, epsilon=1e-08, n_iter_no_change=10)

Este modelo optimiza la función de pérdida de registros usando LBFGS[10] o descenso de gradiente estocástico.

Parámetros:
hidden_layer_sizes: vector, longitud = n_layers - 2, por defecto 100. El i-ésimo elemento representa el número de neuronas en la i-ésima capa oculta.
activation: {'identity', 'logistic', 'tanh', 'relu'}, valor por defecto 'relu'. Función de activación para la capa oculta. Estos valores pueden ser:

- 'identity', activación no operativa, útil para implementar el cuello de botella lineal, devuelve f(x)=x.
- 'logistic', función logística sigmoidea, devuelve f(x) = 1/(1+exp(-x)).
- 'tanh', función de saturación hiperbólica, devuelve f(x) = tanh(x).
- 'relu', función rectificada de la unidad lineal, devuelve f(x) = max(0, x).

solver: {'lbfgs', 'sgd', 'adam'}, por defecto 'adam'. El solucionador para la optimización del peso. Estos valores pueden ser:

[10] Algoritmo Broyden–Fletcher–Goldfarb–Shanno para optimización numérica

- ‘lbfgs’ es un optimizador en la familia de métodos cuasi-Newton.
- ‘sgd’ se refiere al descenso de gradiente estocástico.
- ‘adam’ se refiere a un optimizador estocástico basado en gradiente propuesto por Kingma, Diederik y Jimmy Ba.

Nota: El solucionador por defecto 'adam' funciona bastante bien en conjuntos de datos relativamente grandes (con miles de muestras de capacitación o más) en términos de tiempo de entrenamiento y puntaje de validación. Para conjuntos de datos pequeños, sin embargo, 'lbfgs' puede converger más rápido y funcionar mejor.

- alpha: real, opcional, por defecto 0.0001. Parámetro L2 de penalización (término de regularización).
- batch_size: entero, opcional, por defecto ‘auto’. Tamaño de minibatches para optimizadores estocásticos. Si el solucionador es 'lbfgs', el clasificador no usará minibatch. Cuando se establece en "auto", batch_size=min(200, n_samples).
- learning_rate: {‘constant’, ‘invscaling’, ‘adaptive’}, por defecto ‘constant’. Programa de tasa de aprendizaje para actualizaciones de peso. Solo es usada con el solver ‘sgd’. Estos valores pueden ser:
- ‘constant’ es una tasa de aprendizaje constante dada por ‘learning_rate_init’.
- ‘invscaling’ disminuye gradualmente la tasa de learning_rate_ en cada paso de tiempo 't' usando un exponente de escala inversa de 'power_t'. effective_learning_rate = learning_rate_init / pow (t, power_t).
- ‘adaptive’ mantiene constante la tasa de aprendizaje a 'learning_rate_init' siempre que la pérdida de entrenamiento siga disminuyendo. Cada vez que dos épocas consecutivas no disminuyen la pérdida de entrenamiento por al menos tol, o no aumentan la puntuación de validación al menos tol si está activada la 'early_stopping', la tasa de aprendizaje actual se divide por 5.
- learning_rate_init: real, opcional, por defecto 0.001. La tasa de aprendizaje inicial utilizada. Controla el tamaño del paso al actualizar los pesos. Solo se usa con el solver 'sgd' o 'adam'.
- power_t: real, opcional, por defecto 0.5. El exponente de velocidad de aprendizaje de escala inversa. Se usa para actualizar la tasa de aprendizaje efectiva cuando el valor de aprendizaje se establece en 'invscaling'. Solo se usa con el solver 'sgd'.

- max_iter: entero, opcional, por defecto 200. Número máximo de iteraciones. El solver itera hasta la convergencia (determinada por 'tol') o este número de iteraciones. Para solvers estocásticos ('sgd', 'adam'), se tiene en cuenta que esto determina el número de épocas (cuántas veces se usará cada punto de datos), no el número de pasos de gradiente.
- shuffle: booleano, opcional, por defecto True. Determina si se mezclan las muestras en cada iteración. Solo se usa con solver 'sgd' o 'adam'.
- random_state: entero, instancia de RandomState o None, opcional, por defecto None. Si se define un valor entero, este será la semilla utilizada por el generador de números aleatorios; Si se define una instancia de RandomState, este será el generador de números aleatorios, de lo contrario el generador de números aleatorios es la instancia de RandomState utilizada por np.random.
- tol: real, opcional, por defecto 0.0001. Tolerancia para la optimización. Cuando la pérdida o el puntaje no mejora al menos tol durante dos iteraciones consecutivas, a menos que learning_rate se configure como 'adaptativo', se considera que se ha alcanzado la convergencia y se detiene el entrenamiento.
- verbose: booleano, opcional, por defecto False. Determina si se deben imprimir los mensajes de progreso a la salida estándar.
- warm_start: booleano, opcional, por defecto False. Cuando se establece en True, se reutiliza la solución de la llamada anterior para que se ajuste como inicialización, de lo contrario, simplemente se borra la solución anterior.
- momentum: real, por defecto 0.9. Momento para la actualización de descenso de gradiente. Debe estar entre 0 y 1. Solo se usa con solver 'sgd'.
- nesterovs_momentum: booleano, por defecto True. Determina si se usa el impulso de Nesterov. Solo se usa con solver 'sgd' y momentum > 0.
- early_stopping: booleano, por defecto False. Determina si se efectúa una detención anticipada para finalizar el entrenamiento cuando la puntuación de validación no está mejorando. Si se establece en True, reservará el 10% de los datos de entrenamiento como validación y finalizará el entrenamiento cuando el puntaje de validación no mejore al menos tol durante dos épocas consecutivas. Solo se usa con solver 'sgd' o 'adam'.

- validation_fraction: real, opcional, por defecto 0.1. La proporción de datos de entrenamiento para separar como conjunto de validación para la detención temprana. Debe estar entre 0 y 1. Solo se usa si early_stopping es True
- beta_1: real, opcional, por defecto 0.9. La tasa de desintegración exponencial para las estimaciones del vector de primer momento en adam, debe estar en [0, 1). Solo se usa con solver 'adam'.
- beta_2: real, opcional, por defecto 0.999. La tasa de disminución exponencial para las estimaciones del vector de segundo momento en adam, debe estar en [0, 1). Solo se con solver 'adam'.
- epsilon: real, opcional, por defecto 1e-8. Valor para la estabilidad numérica en adam. Solo se usa con solver 'adam'.
- n_iter_no_change: entero, opcional, por defecto 10. Número máximo de épocas para no completar la mejora tol. Solo se usa con solver ’sgd’ o ‘adam’.

Atributos:

- classes_: arreglo o lista de arreglos de shape(n_classes,). Etiquetas de clase para cada salida.
- loss_: real. La pérdida actual calculada con la función de pérdida.
- coefs_: lista, longitud n_layers – 1. El i-ésimo elemento en la lista representa la matriz de ponderación correspondiente a la capa i.
- intercepts_: lista, longitude n_layers – 1. El i-ésimo elemento en la lista representa el vector de polarización correspondiente a la capa i + 1.
- n_iter_: entero. La cantidad de iteraciones que el solver ejecutó.
- n_layers_: entero. Número de capas.
- n_outputs_: entero. Número de salidas.
- out_activation_: cadena. Nombre de la función de activación de salida.

Anexo 3. Implementación del modelo basado en lógica Difusa

Para implementar en modelo definido en (Angarita Vivas & Tabares Isaza, 2015) y replicar el análisis y los resultados obtenidos en dicho proyecto, se usó la misma definición de las variables de entradas y salida del sistema. Ver sección 5.1.1 Selección de variables de entrada y salida

Definición de las funciones de membresía

En esta etapa se debe modelar por cada variable de entrada y salida, los rangos numéricos en los que está definida cada una de las etiquetas lingüísticas de cada variable. Para el caso de este proyecto, las etiquetas definidas en (Angarita Vivas & Tabares Isaza, 2015) son:

Tabla 13. Etiquetas lingüísticas para variables de entrada

Etiquetas lingüísticas de las variables de entrada
Insignificante
Bajo
Medio
Alto

Tabla 14. Etiquetas lingüísticas para variables de salida

Etiquetas lingüísticas de las variables de Salida
Bajo
Medio
Alto

Una vez se introducen los rangos de valores a cada etiqueta lingüística, quedan definidas como se ve a continuación, donde a cada una de ellas se le define su función de pertenencia o membresía. En esta ilustración del desarrollo del sistema que se está implementando, se observa como para la variable de entrada: AMENAZADF, le quedaron definidas sus etiquetas: INSIGNIFICANTE,

BAJO, MEDIO Y ALTO. Este mismo procedimiento quedó implementado para las demás variables de entrada, pues poseen las mismas etiquetas.

El experto deberá ingresar los valores a cada variable de entrada, según los rangos ya definidos para cada una de ellas y al hacer clic sobre el botón "Evaluar Nivel Riesgo", observará la salida de manera numérica y gráfica.

Para la implementación del sistema difuso en Python, se utilizó la librería SkFuzzy.py, la cual es una caja de herramientas o toolbox de lógica difusa para SciPy[11]. Esta librería permite definir las variables de entrada y salida que tendrá el sistema, las etiquetas lingüísticas de cada variable dentro del motor de inferencia y el motor de reglas o base de conocimiento. (Team, 2016)

Lo primero que se hace en el proyecto es definir el universo de cada uno de los conjuntos difusos y el rango de valores en donde se encuentran definidos:

```
1  import numpy as np
2  import skfuzzy as fuzz
3  from skfuzzy import control as ctrl
4  import matplotlib.pyplot as plt
5
6  # New Antecedent/Consequent objects hold universe variables and membership
7  # functions
8
9
10 # universo de los conjuntos difusos
11 # crisp set
12 amenaza_df = ctrl.Antecedent(np.arange(0, 13, 1), 'amenaza_df')
13 amenaza_ci = ctrl.Antecedent(np.arange(0, 13, 1), 'amenaza_ci')
14 amenaza_ft = ctrl.Antecedent(np.arange(0, 13, 1), 'amenaza_ft')
15 amenaza_ana = ctrl.Antecedent(np.arange(0, 13, 1), 'amenaza_ana')
16 impacto = ctrl.Antecedent(np.arange(0, 13, 1), 'impacto')
17 nivelriesgo = ctrl.Consequent(np.arange(0, 17, 1), 'nivelriesgo')
18
```

Figura 16. Definición de conjuntos difusos

Definición de las funciones de membresía

[11] Biblioteca open source de herramientas y algoritmos matemáticos y de ingeniería escrito en Python

Una vez se han creado los conjuntos difusos, se procede a definir las funciones de membresía para cada etiqueta lingüística dentro del conjunto. Para ello se usamos la función trimf, la cual genera una función de membresía triangular. Se eligió esta porque fue la misma que usaron los autores en la definición de su modelo y para efectos comparativos es ideal que sean las mismas.

```
etiquetas_entrada = ['ninguno', 'insignificante', 'bajo', 'medio', 'alto']


# fuzzyfication
amenaza_df.automf(names=etiquetas_entrada)
amenaza_df['ninguno'] = fuzz.trimf(amenaza_df.universe, [0, 0, 0])
amenaza_df['insignificante'] = fuzz.trimf(amenaza_df.universe, [0, 0, 4])
amenaza_df['bajo'] = fuzz.trimf(amenaza_df.universe, [0, 4, 8])
amenaza_df['medio'] = fuzz.trimf(amenaza_df.universe, [4, 8, 12])
amenaza_df['alto'] = fuzz.trimf(amenaza_df.universe, [8, 12, 12])

amenaza_ci.automf(names=etiquetas_entrada)
amenaza_ci['ninguno'] = fuzz.trimf(amenaza_ci.universe, [0, 0, 0])
amenaza_ci['insignificante'] = fuzz.trimf(amenaza_ci.universe, [0, 0, 4])
amenaza_ci['bajo'] = fuzz.trimf(amenaza_ci.universe, [0, 4, 8])
amenaza_ci['medio'] = fuzz.trimf(amenaza_ci.universe, [4, 8, 12])
amenaza_ci['alto'] = fuzz.trimf(amenaza_ci.universe, [8, 12, 12])

amenaza_ft.automf(names=etiquetas_entrada)
amenaza_ft['ninguno'] = fuzz.trimf(amenaza_ft.universe, [0, 0, 0])
amenaza_ft['insignificante'] = fuzz.trimf(amenaza_ft.universe, [0, 0, 4])
amenaza_ft['bajo'] = fuzz.trimf(amenaza_ft.universe, [0, 4, 8])
amenaza_ft['medio'] = fuzz.trimf(amenaza_ft.universe, [4, 8, 12])
amenaza_ft['alto'] = fuzz.trimf(amenaza_ft.universe, [8, 12, 12])

amenaza_ana.automf(names=etiquetas_entrada)
amenaza_ana['ninguno'] = fuzz.trimf(amenaza_ana.universe, [0, 0, 0])
amenaza_ana['insignificante'] = fuzz.trimf(amenaza_ana.universe, [0, 0, 4])
amenaza_ana['bajo'] = fuzz.trimf(amenaza_ana.universe, [0, 4, 8])
amenaza_ana['medio'] = fuzz.trimf(amenaza_ana.universe, [4, 8, 12])
amenaza_ana['alto'] = fuzz.trimf(amenaza_ana.universe, [8, 12, 12])

impacto.automf(names=etiquetas_entrada)
impacto['ninguno'] = fuzz.trimf(impacto.universe, [0, 0, 0])
impacto['insignificante'] = fuzz.trimf(impacto.universe, [0, 0, 4])
impacto['bajo'] = fuzz.trimf(impacto.universe, [0, 4, 8])
impacto['medio'] = fuzz.trimf(impacto.universe, [4, 8, 12])
impacto['alto'] = fuzz.trimf(impacto.universe, [8, 12, 12])

# Custom membership functions can be built interactively with a familiar,
# Pythonic API
etiquetas_riesgo = ['bajo', 'medio', 'alto']
nivelriesgo.automf(names=etiquetas_riesgo)
nivelriesgo['bajo'] = fuzz.trimf(nivelriesgo.universe, [0, 0, 8])
nivelriesgo['medio'] = fuzz.trimf(nivelriesgo.universe, [0, 8, 16])
nivelriesgo['alto'] = fuzz.trimf(nivelriesgo.universe, [8, 16, 16])
```

Figura 17. Definición del motor FIS en Python

Cuando se definen las funciones de membresía, se puede visualizar como la función triangular actúa para graficarlas en los rangos del nivel de pertenencia de cada etiqueta lingüística para cada variable de entrada o salida:

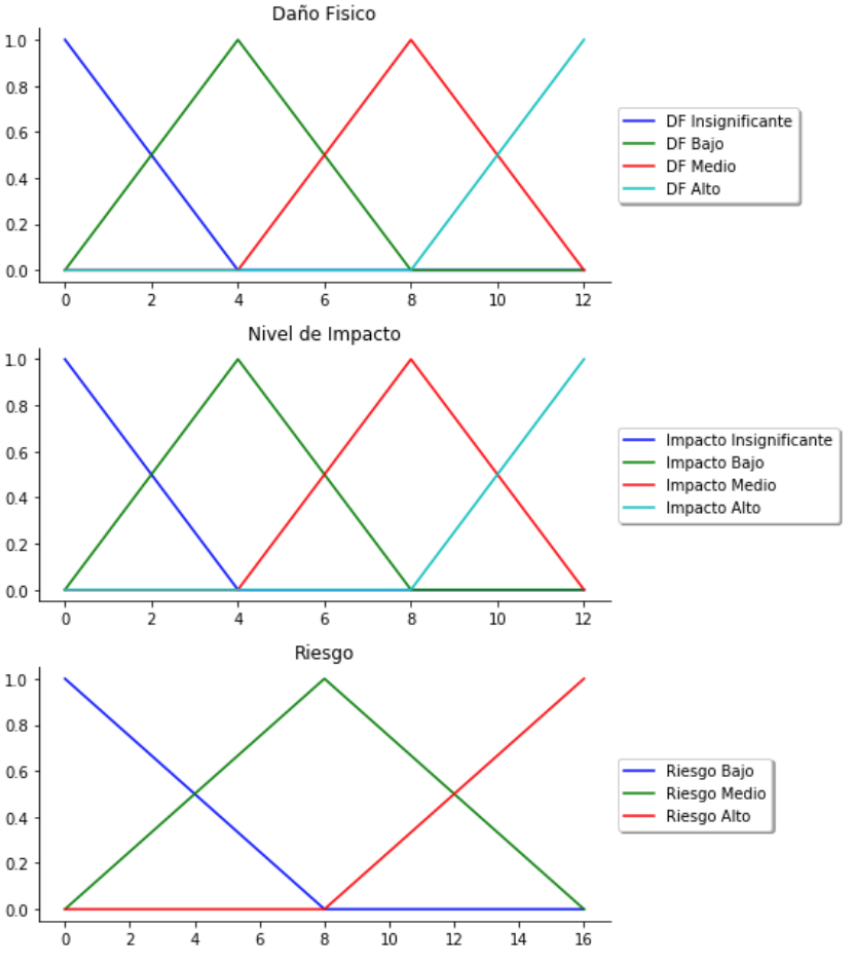

Figura 18. Definición de funciones de membresía para la variable de entrada y salida

Definición de las Reglas de Inferencia

Se describen en el sistema las reglas de inferencia que servirán para que el sistema se comporte de manera adecuada, según el modelo de referencia planteado:

Si Antecedente Entonces Consecuente

```
#Reglas de inferencia de la Base de Conocimiento con Nivel de Riesgo: Alto

regla1 = ctrl.Rule(amenaza_df['alto'] & amenaza_ci['alto'] & amenaza_ft['alto'] & amenaza_ana['alto'] & impacto['alto'], nive
regla8 = ctrl.Rule(amenaza_df['alto'] & amenaza_ci['alto'] & amenaza_ft['insignificante'] & amenaza_ana['alto'] & impacto['a
regla9 = ctrl.Rule(amenaza_df['alto'] & amenaza_ci['alto'] & amenaza_ft['alto'] & amenaza_ana['insignificante'] & impacto['a
regla15 = ctrl.Rule(amenaza_df['alto'] & amenaza_ci['alto'] & amenaza_ft['bajo'] & amenaza_ana['alto'] & impacto['alto'], niv
regla16 = ctrl.Rule(amenaza_df['alto'] & amenaza_ci['alto'] & amenaza_ft['alto'] & amenaza_ana['bajo'] & impacto['alto'], niv
regla17 = ctrl.Rule(amenaza_df['alto'] & amenaza_ci['alto'] & amenaza_ft['medio'] & amenaza_ana['alto'] & impacto['alto'], ni
regla18 = ctrl.Rule(amenaza_df['alto'] & amenaza_ci['alto'] & amenaza_ft['alto'] & amenaza_ana['medio'] & impacto['alto'], ni
regla20 = ctrl.Rule(amenaza_df['alto'] & amenaza_ci['alto'] & amenaza_ft['insignificante'] & amenaza_ana['insignificante'] &
regla21 = ctrl.Rule(amenaza_df['alto'] & amenaza_ci['alto'] & amenaza_ft['bajo'] & amenaza_ana['bajo'] & impacto['alto'], niv
regla22 = ctrl.Rule(amenaza_df['alto'] & amenaza_ci['alto'] & amenaza_ft['bajo'] & amenaza_ana['medio'] & impacto['alto'], ni
regla23 = ctrl.Rule(amenaza_df['alto'] & amenaza_ci['alto'] & amenaza_ft['medio'] & amenaza_ana['medio'] & impacto['alto'], r
regla59 = ctrl.Rule(amenaza_df['medio'] & amenaza_ci['alto'] & amenaza_ft['alto'] & amenaza_ana['alto'] & impacto['alto'], ni
regla60 = ctrl.Rule(amenaza_df['medio'] & amenaza_ci['alto'] & amenaza_ft['insignificante'] & amenaza_ana['alto'] & impacto[
regla61 = ctrl.Rule(amenaza_df['medio'] & amenaza_ci['alto'] & amenaza_ft['alto'] & amenaza_ana['insignificante'] & impacto[
regla62 = ctrl.Rule(amenaza_df['medio'] & amenaza_ci['alto'] & amenaza_ft['bajo'] & amenaza_ana['alto'] & impacto['alto'], ni
regla63 = ctrl.Rule(amenaza_df['medio'] & amenaza_ci['alto'] & amenaza_ft['alto'] & amenaza_ana['bajo'] & impacto['alto'], ni
regla64 = ctrl.Rule(amenaza_df['medio'] & amenaza_ci['alto'] & amenaza_ft['medio'] & amenaza_ana['alto'] & impacto['alto'], r
regla65 = ctrl.Rule(amenaza_df['medio'] & amenaza_ci['alto'] & amenaza_ft['alto'] & amenaza_ana['medio'] & impacto['alto'], r
regla66 = ctrl.Rule(amenaza_df['medio'] & amenaza_ci['alto'] & amenaza_ft['insignificante'] & amenaza_ana['insignificante'] &
regla67 = ctrl.Rule(amenaza_df['medio'] & amenaza_ci['alto'] & amenaza_ft['bajo'] & amenaza_ana['bajo'] & impacto['alto'], ni
regla68 = ctrl.Rule(amenaza_df['medio'] & amenaza_ci['alto'] & amenaza_ft['bajo'] & amenaza_ana['medio'] & impacto['alto'], r
regla69 = ctrl.Rule(amenaza_df['medio'] & amenaza_ci['alto'] & amenaza_ft['medio'] & amenaza_ana['medio'] & impacto['alto'],
regla70 = ctrl.Rule(amenaza_df['bajo'] & amenaza_ci['alto'] & amenaza_ft['alto'] & amenaza_ana['alto'] & impacto['alto'], niv
regla74 = ctrl.Rule(amenaza_df['bajo'] & amenaza_ci['alto'] & amenaza_ft['insignificante'] & amenaza_ana['alto'] & impacto['a
regla75 = ctrl.Rule(amenaza_df['bajo'] & amenaza_ci['alto'] & amenaza_ft['alto'] & amenaza_ana['insignificante'] & impacto['a
regla78 = ctrl.Rule(amenaza_df['bajo'] & amenaza_ci['alto'] & amenaza_ft['bajo'] & amenaza_ana['alto'] & impacto['alto'], niv
regla79 = ctrl.Rule(amenaza_df['bajo'] & amenaza_ci['alto'] & amenaza_ft['alto'] & amenaza_ana['bajo'] & impacto['alto'], niv
regla82 = ctrl.Rule(amenaza_df['bajo'] & amenaza_ci['alto'] & amenaza_ft['medio'] & amenaza_ana['alto'] & impacto['alto'], ni
regla83 = ctrl.Rule(amenaza_df['bajo'] & amenaza_ci['alto'] & amenaza_ft['alto'] & amenaza_ana['medio'] & impacto['alto'], ni
regla84 = ctrl.Rule(amenaza_df['bajo'] & amenaza_ci['alto'] & amenaza_ft['insignificante'] & amenaza_ana['insignificante'] &
regla85 = ctrl.Rule(amenaza_df['bajo'] & amenaza_ci['alto'] & amenaza_ft['bajo'] & amenaza_ana['bajo'] & impacto['alto'], niv
regla86 = ctrl.Rule(amenaza_df['bajo'] & amenaza_ci['alto'] & amenaza_ft['bajo'] & amenaza_ana['medio'] & impacto['alto'], ni
regla87 = ctrl.Rule(amenaza_df['bajo'] & amenaza_ci['alto'] & amenaza_ft['medio'] & amenaza_ana['medio'] & impacto['alto'], r
regla97 = ctrl.Rule(amenaza_df['insignificante'] & amenaza_ci['alto'] & amenaza_ft['alto'] & amenaza_ana['alto'] & impacto['a
regla98 = ctrl.Rule(amenaza_df['insignificante'] & amenaza_ci['alto'] & amenaza_ft['insignificante'] & amenaza_ana['alto'] &
regla99 = ctrl.Rule(amenaza_df['insignificante'] & amenaza_ci['alto'] & amenaza_ft['alto'] & amenaza_ana['insignificante'] &
regla100 = ctrl.Rule(amenaza_df['insignificante'] & amenaza_ci['alto'] & amenaza_ft['bajo'] & amenaza_ana['alto'] & impacto[
```

Figura 19. Definición del conjunto de reglas como base de conocimiento

Posteriormente las reglas creadas son agregadas al controlador difuso que se encargará de activar algunas de ellas dependiendo los valores de las variables de entrada:

```
nivelriesgo_ctrl = ctrl.ControlSystem([regla1, regla2, regla3, regla4, regla5, regla6, regla7, regla8, regla9, regla10,
                                       regla11, regla12, regla13, regla14, regla15, regla16, regla17, regla18, regla19, regla
                                       regla21, regla22, regla23, regla24, regla25, regla26, regla27, regla28, regla29, regla
                                       regla31, regla32, regla33, regla34, regla35, regla36, regla37, regla38, regla39, regla
                                       regla41, regla42, regla43, regla44, regla45, regla46, regla47, regla48, regla49, regla
                                       regla51, regla52, regla53, regla54, regla55, regla56, regla57, regla58, regla59, regla
                                       regla61, regla62, regla63, regla64, regla65, regla66, regla67, regla68, regla69, regla
                                       regla71, regla72, regla73, regla74, regla75, regla76, regla77, regla78, regla79, regla
                                       regla81, regla82, regla83, regla84, regla85, regla86, regla87, regla88, regla89, regla
                                       regla91, regla92, regla93, regla94, regla95, regla96, regla97, regla98, regla99, regla
                                       regla101, regla102, regla103, regla104, regla105, regla105, regla107])
```

Figura 20. Agregación de reglas al controlador difuso

Para finalizar el modelo en Python, se construye una interfaz gráfica de usuario para que el experto pueda interactuar con el sistema difuso y pueda realizar simulaciones con diferentes valores para las variables de entrada.

La interfaz gráfica de usuario que se implementó y que se ilustra a continuación, se diseñó utilizando la librería TKinter, la cual es la librería por defecto que trae Python para construir aplicaciones con interfaz gráfica y que aporta múltiples componentes con los cuales el desarrollador puede diseñar una interfaz amigable para el usuario final.

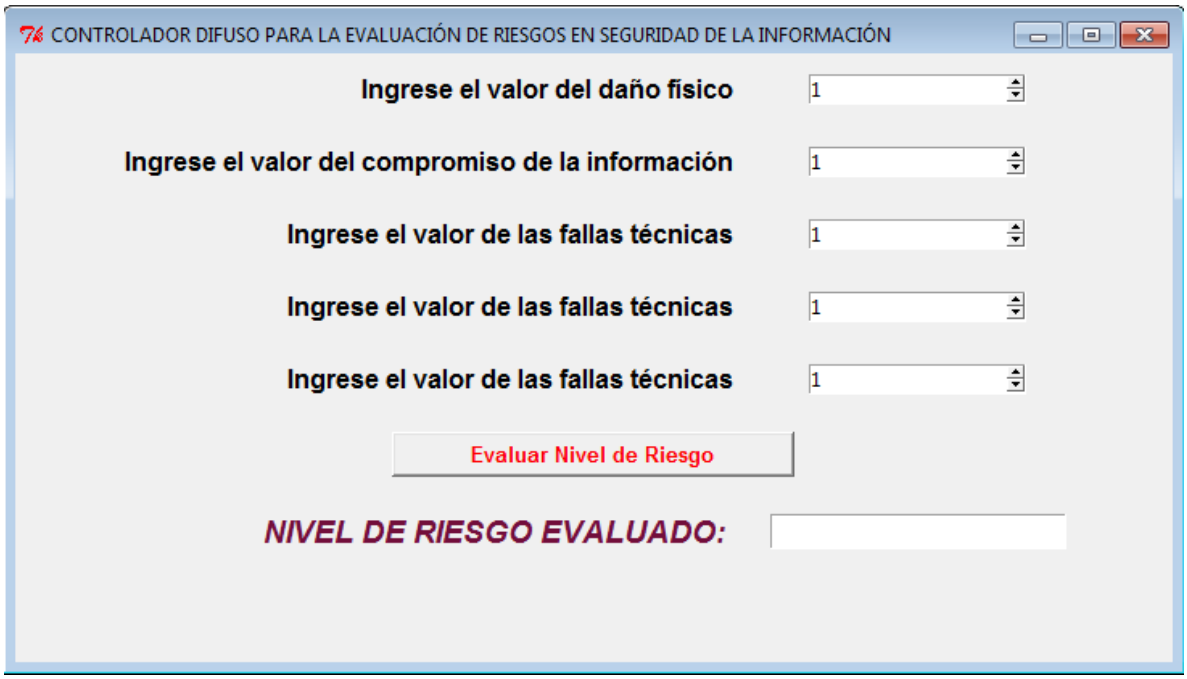

Figura 21. Visualización de la interfaz gráfica de usuario desarrollada en Python

Con esta interfaz, muy similar a la que se desarrolló en MATLAB para el proyecto de (Angarita Vivas & Tabares Isaza, 2015) , el experto podrá realizar las simulaciones y visualizar los resultados también de manera gráfica una vez evalúe el nivel de riesgo dados los argumentos de entrada al modelo en el sistema.

Anexo 4. Implementación del modelo basado en redes neuronales

En la implementación de los modelos basados tanto en lógica difusa como en redes neuronales, se decidió realizarlos usando Python[12,] debido a que es uno de los lenguajes más utilizados en el campo de la inteligencia artificial, en parte gracias al amplio número de bibliotecas y frameworks desarrollados para este lenguaje.

Para la implementación de dichos modelos, se usó como referencia la matriz de riesgos y reglas de inferencia usadas en (Angarita Vivas & Tabares Isaza, 2015), con el objetivo de realizar posteriormente el análisis comparativo entre el modelo basado en lógica difusa y el modelo basado en redes neuronales.

Implementación del modelo

Para implementar el modelo en Python se hace necesario la inclusión de una librería especializada para el manejo de redes neuronales como es sklearn[13], la cual dentro de su conjunto de funcionalidades posee los mecanismos necesarios para crear, entrenar y consultar una red.

El primer paso en la creación de la red neuronal es definir las librerías necesarias para el desarrollo del modelo, entre ellas tenemos librerías para el manejo de matrices y operaciones entre ellas (numpy), operaciones matemáticas (math), manejo de datasets (pandas), división de los datos de entrada en un conjunto de datos de entrenamiento y otro de pruebas (train_test_split), el perceptrón clasificador multicapa (MLPClassifier), la matriz de confusión (confusión_matrix) y el nivel de precisión (accuracy_score).

[12] Es un lenguaje de programación multiparadigma e interpretado cuya filosofía hace hincapié en una sintaxis que favorezca un código legible.

[13] https://scikit-learn.org/

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.neural_network import MLPClassifier
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from random import randint
```

Figura 22. Importación de librerías en Python para el manejo de redes neuronales

Luego de cargar las librerías se procede a leer los patrones de entrada, los cuales harán parte de los datos de entrenamiento y posterior prueba de la red neuronal junto con las salidas esperadas. En el archivo RedNeuronal.csv (ver anexo 1) se definen los datos a cargar, este archivo está se compone de 6 columnas donde las primeras 5 son las variables de entrada y la sexta es la salida esperada.

```
data = pd.read_csv('RedNeuronal.csv', sep=';')
data = data.values
entradas = data[:,[0,1,2,3,4]]
salidas  = data[:,5]
YTrain, YTest, XTrain, XTest = train_test_split(entradas,salidas,train_size=0.8)
```

Figura 23. Carga de datos para entrenar la red neuronal

Entrenamiento de la red neuronal

Esta tarea se realiza empleando la función train_test_split, la cual recibe como parámetros las matrices a dividir y la cantidad de datos requeridos en cada grupo (entrenamiento y pruebas). De esta manera se separan los datos de prueba de los de entrenamiento permitiendo que la red sea entrenada y consultada con datos distintos para tener una mejor estimación del comportamiento de la red. En la figura 14 se puede observar la creación de la red neuronal con sus parámetros, así como su entrenamiento y posterior prueba.

La función MLPClassifier recibe una serie de parámetros para crear la red neuronal para luego ser entrenada mediante la función fit, teniendo como entradas los datos de entrenamiento y almacenando los pesos correspondientes a las conexiones entre la capa de entrada, las capas intermedias y la de salida. Los parámetros de la red se pueden ver en el anexo 2 y los pesos de las conexiones con la opción coefs_ de la red neuronal.

En este sentido, existe un amplio abanico de medidas de rendimiento: media cuadrática del error, funciones de coste, matrices de confusión, índices de sensibilidad y especificidad, etc. Para nuestro estudio, la evaluación de rendimiento se realizó a partir de la matriz de confusión, ya que las librerías usadas cuentan con una función que la calcula dados los parámetros de entrada. Esta matriz indica cuantos elementos fueron clasificados correctamente.

```
red = MLPClassifier(solver='lbfgs',
                    alpha=0.0000001,
                    max_iter=1000000,
                    hidden_layer_sizes=(70,200,150),
                    random_state=1,
                    activation='logistic')
red.fit(YTrain,XTrain)
YPred = red.predict(entradas)
score = accuracy_score(salidas,YPred)
cMat = confusion_matrix(salidas,YPred)
print cMat
print score
```

```
[[ 54   6   0   0   0   0   0   0]
 [  4 106   0   1   0   0   0   0]
 [  0   0   1   0   0   0   0   0]
 [  0   2   1 146   0   0   1   0]
 [  0   0   0   1   0   0   0   0]
 [  0   0   0   0   0   8   0   0]
 [  0   0   0   0   0   0  39   2]
 [  0   0   0   0   0   0   4  35]]
0.9464720194647201
```

Figura 24. Entrenamiento y prueba de la red neuronal

Anexo 5. Valores obtenidos en cada una de las pruebas de la red neuronal

Prueba de la red neuronal con 1 capa oculta de 292 neuronas:

Tabla 15. Resultado para solver lbfgs con activación logistic y 1 capa oculta.

Iteración	Exactitud	Iteración	Exactitud
1	0,966363636363636	6	0,962727272727272
2	0,960909090909090	7	0,966363636363636
3	0,968181818181818	8	0,967272727272727
4	0,964545454545454	9	0,966363636363636
5	0,972727272727272	10	0,966363636363636

$$\bar{x} = 0{,}966181818, \qquad \sigma = 0{,}003172425$$

Tabla 16. Resultado para solver lbfgs con activación tanh y 1 capa oculta.

Iteración	Exactitud	Iteración	Exactitud
1	0,968181818181818	6	0,963636363636363
2	0,969090909090909	7	0,969090909090909
3	0,969090909090909	8	0,969090909090909
4	0,961818181818181	9	0,965454545454545
5	0,965454545454545	10	0,967272727272727

$$\bar{x} = 0{,}966818182, \qquad \sigma = 0{,}002615557$$

Tabla 17. Resultado para solver lbfgs con activación relu y 1 capa oculta.

Iteración	Exactitud	Iteración	Exactitud
1	0,960909090909090	6	0,963636363636363
2	0,970909090909090	7	0,969090909090909
3	0,954545454545454	8	0,963636363636363
4	0,966363636363636	9	0,955454545454545
5	0,964545454545454	10	0,970000000000000

$$\bar{x} = 0{,}963909091, \qquad \sigma = 0{,}005653769$$

Tabla 18. Resultado para solver sgd con activación logistic y 1 capa oculta.

Iteración	Exactitud	Iteración	Exactitud
1	0,361818181818181	6	0,337272727272727
2	0,348181818181818	7	0,348181818181818
3	0,340909090909090	8	0,340000000000000
4	0,352727272727272	9	0,341818181818181
5	0,325454545454545	10	0,345454545454545

$$\bar{x} = 0{,}344181818, \qquad \sigma = 0{,}009717781$$

Tabla 19. Resultado para solver sgd con activación tanh y 1 capa oculta.

Iteración	Exactitud	Iteración	Exactitud
1	0,403636363636363	6	0,410000000000000
2	0,428181818181818	7	0,416363636363636
3	0,390909090909090	8	0,381818181818181
4	0,417272727272727	9	0,401818181818181
5	0,382727272727272	10	0,415454545454545

$$\bar{x} = 0{,}404818182, \qquad \sigma = 0{,}015617383$$

Tabla 20. Resultado para solver sgd con activación relu y 1 capa oculta.

Iteración	Exactitud	Iteración	Exactitud
1	0,346363636363636	6	0,333636363636363
2	0,340000000000000	7	0,320909090909090
3	0,326363636363636	8	0,324545454545454
4	0,327272727272727	9	0,330909090909090
5	0,313636363636363	10	0,340000000000000

$$\bar{x} = 0{,}330363636, \qquad \sigma = 0{,}009895691$$

Tabla 21. Resultado para solver adam con activación logistic y 1 capa oculta.

Iteración	Exactitud	Iteración	Exactitud
1	0,390000000000000	6	0,355454545454545
2	0,362727272727272	7	0,391818181818181
3	0,362727272727272	8	0,367272727272727
4	0,356363636363636	9	0,379090909090909
5	0,356363636363636	10	0,374545454545454

$$\bar{x} = 0{,}369636364, \qquad \sigma = 0{,}013641077$$

Tabla 22. Resultado para solver adam con activación tanh y 1 capa oculta.

Iteración	Exactitud	Iteración	Exactitud
1	0,486363636363636	6	0,477272727272727
2	0,422727272727272	7	0,458181818181818
3	0,410909090909090	8	0,398181818181818
4	0,463636363636363	9	0,443636363636363
5	0,491818181818181	10	0,465454545454545

$$\bar{x} = 0{,}451818182, \qquad \sigma = 0{,}032095473$$

Tabla 23. Resultado para solver adam con activación relu y 1 capa oculta.

Iteración	Exactitud	Iteración	Exactitud
1	0,501818181818181	6	0,448181818181818
2	0,519090909090909	7	0,456363636363636
3	0,442727272727272	8	0,489090909090909
4	0,479090909090909	9	0,413636363636363
5	0,499090909090909	10	0,477272727272727

$$\bar{x} = 0{,}472636364, \qquad \sigma = 0{,}032113921$$

Prueba de la red neuronal con 2 capas oculta de 197 y 179 neuronas:

Tabla 24. Resultado para solver lbfgs con activación logistic y 2 capas ocultas.

Iteración	Exactitud	Iteración	Exactitud
1	0,966363636363636	6	0,966363636363636
2	0,967272727272727	7	0,970000000000000
3	0,967272727272727	8	0,968181818181818
4	0,963636363636363	9	0,964545454545454
5	0,967272727272727	10	0,968181818181818

$$\bar{x} = 0{,}966909091, \qquad \sigma = 0{,}001828255$$

Tabla 25. Resultado para solver lbfgs con activación tanh y 2 capas ocultas.

Iteración	Exactitud	Iteración	Exactitud
1	0,958181818181818	6	0,964545454545454
2	0,964545454545454	7	0,970000000000000
3	0,966363636363636	8	0,966363636363636
4	0,965454545454545	9	0,973636363636363
5	0,961818181818181	10	0,968181818181818

$$\bar{x} = 0{,}965909091, \qquad \sigma = 0{,}004247832$$

Tabla 26. Resultado para solver lbfgs con activación relu y 2 capas ocultas.

Iteración	Exactitud	Iteración	Exactitud
1	0,964545454545454	6	0,964545454545454
2	0,969090909090909	7	0,968181818181818
3	0,968181818181818	8	0,960909090909090
4	0,968181818181818	9	0,969090909090909
5	0,967272727272727	10	0,968181818181818

$$\bar{x} = 0{,}966818182, \qquad \sigma = 0{,}002650433$$

Tabla 27. Resultado para solver sgd con activación logistic y 2 capas ocultas.

Iteración	Exactitud	Iteración	Exactitud
1	0,100909090909090	6	0,331818181818181
2	0,306363636363636	7	0,091818181818182
3	0,332727272727272	8	0,167272727272727
4	0,088181818181818	9	0,317272727272727
5	0,310909090909090	10	0,337272727272727

$$\bar{x} = 0{,}238454545, \qquad \sigma = 0{,}111311614$$

Tabla 28. Resultado para solver sgd con activación tanh y 2 capas ocultas.

Iteración	Exactitud	Iteración	Exactitud
1	0,507272727272727	6	0,487272727272727
2	0,467272727272727	7	0,456363636363636
3	0,482727272727272	8	0,470000000000000
4	0,519090909090909	9	0,485454545454545
5	0,480909090909090	10	0,450000000000000

$$\bar{x} = 0{,}480636364, \qquad \sigma = 0{,}021290116$$

Tabla 29. Resultado para solver sgd con activación relu y 2 capas ocultas.

Iteración	Exactitud	Iteración	Exactitud
1	0,410909090909090	6	0,426363636363636
2	0,362727272727272	7	0,396363636363636
3	0,418181818181818	8	0,385454545454545
4	0,388181818181818	9	0,413636363636363
5	0,401818181818181	10	0,405454545454545

$$\bar{x} = 0{,}400909091, \qquad \sigma = 0{,}018586408$$

Tabla 30. Resultado para solver adam con activación logistic y 2 capas ocultas.

Iteración	Exactitud	Iteración	Exactitud
1	0,334545454545454	6	0,360909090909090
2	0,372727272727272	7	0,381818181818181
3	0,360000000000000	8	0,379090909090909
4	0,361818181818181	9	0,381818181818181
5	0,375454545454545	10	0,351818181818181

$$\bar{x} = 0{,}366000000, \qquad \sigma = 0{,}015189044$$

Tabla 31. Resultado para solver adam con activación tanh y 2 capas ocultas.

Iteración	Exactitud	Iteración	Exactitud
1	0,520909090909090	6	0,554545454545454
2	0,416363636363636	7	0,530909090909090
3	0,488181818181818	8	0,483636363636363
4	0,482727272727272	9	0,556363636363636
5	0,447272727272727	10	0,481818181818181

$$\bar{x} = 0{,}496272727, \qquad \sigma = 0{,}044990459$$

Tabla 32. Resultado para solver adam con activación relu y 2 capas ocultas.

Iteración	Exactitud	Iteración	Exactitud
1	0,606363636363636	6	0,622727272727272
2	0,622727272727272	7	0,557272727272727
3	0,605454545454545	8	0,573636363636363
4	0,571818181818181	9	0,567272727272727
5	0,661818181818181	10	0,560909090909090

$$\bar{x} = 0{,}595000000, \qquad \sigma = 0{,}034300701$$

Prueba de la red neuronal con 3 capas oculta de 194, 97, 84 neuronas:

Tabla 33. Resultado para solver lbfgs con activación logistic y 3 capas ocultas.

Iteración	Exactitud	Iteración	Exactitud
1	0,968181818181818	6	0,968181818181818
2	0,965454545454545	7	0,961818181818181
3	0,966363636363636	8	0,969090909090909
4	0,965454545454545	9	0,969090909090909
5	0,962727272727272	10	0,974545454545454

$$\bar{x} = 0{,}967090909, \qquad \sigma = 0{,}003631310$$

Tabla 34. Resultado para solver lbfgs con activación tanh y 3 capas ocultas.

Iteración	Exactitud	Iteración	Exactitud
1	0,970909090909090	6	0,971818181818181
2	0,970000000000000	7	0,964545454545454
3	0,968181818181818	8	0,970000000000000
4	0,970000000000000	9	0,969090909090909
5	0,972727272727272	10	0,968181818181818

$$\bar{x} = 0{,}969545455, \qquad \sigma = 0{,}002277772$$

Tabla 35. Resultado para solver lbfgs con activación relu y 3 capas ocultas.

Iteración	Exactitud	Iteración	Exactitud
1	0,967272727272727	6	0,970000000000000
2	0,970909090909090	7	0,967272727272727
3	0,966363636363636	8	0,967272727272727
4	0,970909090909090	9	0,970909090909090
5	0,973636363636363	10	0,968181818181818

$$\bar{x} = 0{,}969272727, \qquad \sigma = 0{,}002339425$$

Tabla 36. Resultado para solver sgd con activación logistic y 3 capas ocultas.

Iteración	Exactitud	Iteración	Exactitud
1	0,0909090909090909	6	0,0909090909090909
2	0,1172727272727270	7	0,0909090909090909
3	0,0927272727272727	8	0,0909090909090909
4	0,0909090909090909	9	0,1145454545454540
5	0,1009090909090900	10	0,0881818181818181

$$\bar{x} = 0{,}096818182, \qquad \sigma = 0{,}010621201$$

Tabla 37. Resultado para solver sgd con activación tanh y 3 capas ocultas.

Iteración	Exactitud	Iteración	Exactitud
1	0,484545454545454	6	0,494545454545454
2	0,520909090909090	7	0,494545454545454
3	0,479090909090909	8	0,525454545454545
4	0,507272727272727	9	0,503636363636363
5	0,494545454545454	10	0,515454545454545

$$\bar{x} = 0{,}502000000, \qquad \sigma = 0{,}015331057$$

Tabla 38. Resultado para solver sgd con activación relu y 3 capas ocultas.

Iteración	Exactitud	Iteración	Exactitud
1	0,424545454545454	6	0,370909090909090
2	0,313636363636363	7	0,369090909090909
3	0,383636363636363	8	0,426363636363636
4	0,498181818181818	9	0,401818181818181
5	0,378181818181818	10	0,405454545454545

$$\bar{x} = 0{,}397181818, \qquad \sigma = 0{,}048183629$$

Tabla 39. Resultado para solver adam con activación logistic y 3 capas ocultas.

Iteración	Exactitud	Iteración	Exactitud
1	0,334545454545454	6	0,333636363636363
2	0,332727272727272	7	0,335454545454545
3	0,300909090909090	8	0,324545454545454
4	0,322727272727272	9	0,333636363636363
5	0,327272727272727	10	0,322727272727272

$$\bar{x} = 0{,}326818182, \qquad \sigma = 0{,}010393984$$

Tabla 40. Resultado para solver adam con activación tanh y 3 capas ocultas.

Iteración	Exactitud	Iteración	Exactitud
1	0,530909090909090	6	0,423636363636363
2	0,474545454545454	7	0,528181818181818
3	0,540000000000000	8	0,612727272727272
4	0,452727272727272	9	0,487272727272727
5	0,483636363636363	10	0,503636363636363

$$\bar{x} = 0{,}503727273, \qquad \sigma = 0{,}052783756$$

Tabla 41. Resultado para solver adam con activación relu y 3 capas ocultas.

Iteración	Exactitud	Iteración	Exactitud
1	0,512727272727272	6	0,601818181818181
2	0,501818181818181	7	0,576363636363636
3	0,505454545454545	8	0,552727272727272
4	0,506363636363636	9	0,525454545454545
5	0,580000000000000	10	0,510909090909090

$$\bar{x} = 0{,}537363636, \qquad \sigma = 0{,}037149216$$

Prueba de la red neuronal con 4 capas oculta de 88, 75, 160 y 187 neuronas:

Tabla 42. Resultado para solver lbfgs con activación logistic y 4 capas ocultas.

Iteración	Exactitud	Iteración	Exactitud
1	0,0745454545454545	6	0,0909090909090909
2	0,0909090909090909	7	0,0945454545454545
3	0,0909090909090909	8	0,0872727272727272
4	0,0909090909090909	9	0,0909090909090909
5	0,0909090909090909	10	0,0909090909090909

$$\bar{x} = 0{,}089272727, \qquad \sigma = 0{,}005451177$$

Tabla 43. Resultado para solver lbfgs con activación tanh y 4 capas ocultas.

Iteración	Exactitud	Iteración	Exactitud
1	0,9690909090909090	6	0,9718181818181810
2	0,9709090909090900	7	0,9718181818181810
3	0,9754545454545450	8	0,9654545454545450
4	0,9672727272727270	9	0,9700000000000000
5	0,9681818181818180	10	0,9654545454545450

$$\bar{x} = 0{,}969545455, \qquad \sigma = 0{,}003127237$$

Tabla 44. Resultado para solver lbfgs con activación relu y 4 capas ocultas.

Iteración	Exactitud	Iteración	Exactitud
1	0,9654545454545450	6	0,9681818181818180
2	0,9681818181818180	7	0,9700000000000000
3	0,9672727272727270	8	0,9681818181818180
4	0,9690909090909090	9	0,9672727272727270
5	0,9690909090909090	10	0,9700000000000000

$$\bar{x} = 0{,}968272727, \qquad \sigma = 0{,}001385349$$

Tabla 45. Resultado para solver sgd con activación logistic y 4 capas ocultas.

Iteración	Exactitud	Iteración	Exactitud
1	0,0909090909090909	6	0,0909090909090909
2	0,0909090909090909	7	0,0909090909090909
3	0,0909090909090909	8	0,0827272727272727
4	0,0909090909090909	9	0,0909090909090909
5	0,0909090909090909	10	0,0909090909090909

$$\bar{x} = 0{,}090090909, \qquad \sigma = 0{,}002587318$$

Tabla 46. Resultado para solver sgd con activación tanh y 4 capas ocultas.

Iteración	Exactitud	Iteración	Exactitud
1	0,4609090909090900	6	0,4472727272727270
2	0,4081818181818180	7	0,4754545454545450
3	0,4718181818181810	8	0,4245454545454540
4	0,4800000000000000	9	0,4527272727272720
5	0,4463636363636360	10	0,4536363636363630

$$\bar{x} = 0{,}452090909, \qquad \sigma = 0{,}022453094$$

Tabla 47. Resultado para solver sgd con activación relu y 4 capas ocultas.

Iteración	Exactitud	Iteración	Exactitud
1	0,4700000000000000	6	0,4381818181818180
2	0,4509090909090900	7	0,4700000000000000
3	0,3763636363636360	8	0,4700000000000000
4	0,4354545454545450	9	0,4100000000000000
5	0,4827272727272720	10	0,4727272727272720

$$\bar{x} = 0{,}447636364, \qquad \sigma = 0{,}033509726$$

Tabla 48. Resultado para solver adam con activación logistic y 4 capas ocultas.

Iteración	Exactitud	Iteración	Exactitud
1	0,0909090909090909	6	0,0909090909090909
2	0,3209090909090900	7	0,3118181818181810
3	0,0909090909090909	8	0,0909090909090909
4	0,0909090909090909	9	0,0909090909090909
5	0,0909090909090909	10	0,3154545454545450

$$\bar{x} = 0{,}158454545, \qquad \sigma = 0{,}108779902$$

Tabla 49. Resultado para solver adam con activación tanh y 4 capas ocultas.

Iteración	Exactitud	Iteración	Exactitud
1	0,5427272727272720	6	0,5463636363636360
2	0,5418181818181810	7	0,4681818181818180
3	0,4727272727272720	8	0,5581818181818180
4	0,5018181818181810	9	0,5372727272727270
5	0,5145454545454540	10	0,4736363636363630

$$\bar{x} = 0{,}515727273, \qquad \sigma = 0{,}034465751$$

Tabla 50. Resultado para solver adam con activación relu y 4 capas ocultas.

Iteración	Exactitud	Iteración	Exactitud
1	0,5890909090909090	6	0,6263636363636360
2	0,5754545454545450	7	0,5227272727272720
3	0,6481818181818180	8	0,5936363636363630
4	0,5900000000000000	9	0,6372727272727270
5	0,5900000000000000	10	0,6136363636363630

$$\bar{x} = 0{,}598636364, \qquad \sigma = 0{,}035755650$$

Prueba de la red neuronal con 5 capas oculta de 134, 247, 211, 163 y 157 neuronas:

Tabla 51. Resultado para solver lbfgs con activación logistic y 5 capas ocultas.

Iteración	Exactitud	Iteración	Exactitud
1	0,0909090909090909	6	0,0963636363636363
2	0,0909090909090909	7	0,0909090909090909
3	0,0909090909090909	8	0,0909090909090909
4	0,0909090909090909	9	0,1136363636363630
5	0,0836363636363636	10	0,0909090909090909

$$\bar{x} = 0{,}093000000, \qquad \sigma = 0{,}007856028$$

Tabla 52. Resultado para solver lbfgs con activación tanh y 5 capas ocultas.

Iteración	Exactitud	Iteración	Exactitud
1	0,9672727272727270	6	0,9709090909090900
2	0,9654545454545450	7	0,9700000000000000
3	0,9618181818181810	8	0,9736363636363630
4	0,9654545454545450	9	0,9672727272727270
5	0,9672727272727270	10	0,9690909090909090

$$\bar{x} = 0{,}967818182, \qquad \sigma = 0{,}003297326$$

Tabla 53. Resultado para solver lbfgs con activación relu y 5 capas ocultas.

Iteración	Exactitud	Iteración	Exactitud
1	0,9627272727272720	6	0,9663636363636360
2	0,9672727272727270	7	0,9654545454545450
3	0,9681818181818180	8	0,9700000000000000
4	0,9736363636363630	9	0,9727272727272720
5	0,9645454545454540	10	0,9690909090909090

$$\bar{x} = 0{,}968000000, \qquad \sigma = 0{,}003476274$$

Tabla 54. Resultado para solver sgd con activación logistic y 5 capas ocultas.

Iteración	Exactitud	Iteración	Exactitud
1	0,0909090909090909	6	0,0909090909090909
2	0,0909090909090909	7	0,0909090909090909
3	0,0909090909090909	8	0,0909090909090909
4	0,0909090909090909	9	0,0909090909090909
5	0,0909090909090909	10	0,0900000000000000

$$\bar{x} = 0{,}090818182, \qquad \sigma = 0{,}00028748$$

Tabla 55. Resultado para solver sgd con activación tanh y 5 capas ocultas.

Iteración	Exactitud	Iteración	Exactitud
1	0,5036363636363630	6	0,5181818181818180
2	0,4918181818181810	7	0,5018181818181810
3	0,5063636363636360	8	0,5036363636363630
4	0,5045454545454540	9	0,4981818181818180
5	0,4872727272727270	10	0,4881818181818180

$$\bar{x} = 0{,}500363636, \qquad \sigma = 0{,}009381223$$

Tabla 56. Resultado para solver sgd con activación relu y 5 capas ocultas.

Iteración	Exactitud	Iteración	Exactitud
1	0,4372727272727270	6	0,4409090909090900
2	0,4281818181818180	7	0,4545454545454540
3	0,4927272727272720	8	0,4581818181818180
4	0,4709090909090900	9	0,4800000000000000
5	0,4900000000000000	10	0,4245454545454540

$$\bar{x} = 0{,}457727273, \qquad \sigma = 0{,}024993112$$

Tabla 57. Resultado para solver adam con activación logistic y 5 capas ocultas.

Iteración	Exactitud	Iteración	Exactitud
1	0,0909090909090909	6	0,3236363636363630
2	0,3136363636363630	7	0,3190909090909090
3	0,0909090909090909	8	0,0909090909090909
4	0,0909090909090909	9	0,2818181818181810
5	0,3345454545454540	10	0,0909090909090909

$$\bar{x} = 0{,}202727273, \qquad \sigma = 0{,}118606850$$

Tabla 58. Resultado para solver adam con activación tanh y 5 capas ocultas.

Iteración	Exactitud	Iteración	Exactitud
1	0,4118181818181810	6	0,5300000000000000
2	0,3972727272727270	7	0,5000000000000000
3	0,5009090909090900	8	0,4936363636363630
4	0,4481818181818180	9	0,4600000000000000
5	0,3900000000000000	10	0,4854545454545450

$$\bar{x} = 0{,}461727273, \qquad \sigma = 0{,}048555725$$

Tabla 59. Resultado para solver adam con activación relu y 5 capas ocultas.

Iteración	Exactitud	Iteración	Exactitud
1	0,5545454545454540	6	0,3709090909090900
2	0,5709090909090900	7	0,5681818181818180
3	0,5909090909090900	8	0,5245454545454540
4	0,5136363636363630	9	0,4836363636363630
5	0,4927272727272720	10	0,5027272727272720

$$\bar{x} = 0{,}517272727, \qquad \sigma = 0{,}062993871$$

Prueba de la red neuronal con 6 capas ocultas de 224, 136, 208, 106, 120 y 143 neuronas:

Tabla 60. Resultado para solver lbfgs con activación logistic y 6 capas ocultas.

Iteración	Exactitud	Iteración	Exactitud
1	0,0909090909090909	6	0,0909090909090909
2	0,0909090909090909	7	0,0909090909090909
3	0,0909090909090909	8	0,0909090909090909
4	0,0909090909090909	9	0,0909090909090909
5	0,0909090909090909	10	0,0800000000000000

$$\bar{x} = 0{,}0898181818181818, \qquad \sigma = 0$$

Tabla 61. Resultado para solver lbfgs con activación tanh y 6 capas ocultas.

Iteración	Exactitud	Iteración	Exactitud
1	0,970000000000000	6	0,965454545454545
2	0,966363636363636	7	0,970909090909090
3	0,965454545454545	8	0,962727272727272
4	0,970000000000000	9	0,966363636363636
5	0,970000000000000	10	0,970000000000000

$$\bar{x} = 0{,}967727272727272, \qquad \sigma = 0{,}00278557216831078$$

Tabla 62. Resultado para solver lbfgs con activación relu y 6 capas ocultas.

Iteración	Exactitud	Iteración	Exactitud
1	0,966363636363636	6	0,970000000000000
2	0,965454545454545	7	0,963636363636363
3	0,964545454545454	8	0,968181818181818
4	0,960000000000000	9	0,968181818181818
5	0,970000000000000	10	0,963636363636363

$$\bar{x} = 0{,}966000000, \ \sigma = 0{,}003184000$$

Tabla 63. Resultado para solver sgd con activación logistic y 6 capas ocultas.

Iteración	Exactitud	Iteración	Exactitud
1	0,0909090909090909	6	0,0909090909090909
2	0,0909090909090909	7	0,0909090909090909
3	0,0909090909090909	8	0,0909090909090909
4	0,0909090909090909	9	0,0909090909090909
5	0,0909090909090909	10	0,0909090909090909

$$\bar{x} = 0{,}090909090, \qquad \sigma = 0$$

Tabla 64. Resultado para solver sgd con activación tanh y 6 capas ocultas.

Iteración	Exactitud	Iteración	Exactitud
1	0,470909090909090	6	0,470909090909090
2	0,423636363636363	7	0,387272727272727
3	0,483636363636363	8	0,467272727272727
4	0,480909090909090	9	0,490909090909090
5	0,449090909090909	10	0,479090909090909

$$\bar{x} = 0{,}460363636, \qquad \sigma = 0{,}032207431$$

Tabla 65. Resultado para solver sgd con activación relu y 6 capas ocultas.

Iteración	Exactitud	Iteración	Exactitud
1	0,291818181818181	6	0,377272727272727
2	0,414545454545454	7	0,414545454545454
3	0,402727272727272	8	0,408181818181818
4	0,418181818181818	9	0,343636363636363
5	0,430909090909090	10	0,389090909090909

$$\bar{x} = 0{,}389090909, \qquad \sigma = 0{,}042307197$$

Tabla 66. Resultado para solver adam con activación logistic y 6 capas ocultas.

Iteración	Exactitud	Iteración	Exactitud
1	0,0909090909090909	6	0,0909090909090909
2	0,0909090909090909	7	0,0909090909090909
3	0,0909090909090909	8	0,0909090909090909
4	0,0909090909090909	9	0,0909090909090909
5	0,0909090909090909	10	0,0909090909090909

$$\bar{x} = 0{,}090909090, \qquad \sigma = 0$$

Tabla 67. Resultado para solver adam con activación tanh y 6 capas ocultas.

Iteración	Exactitud	Iteración	Exactitud
1	0,443636363636363	6	0,440000000000000
2	0,473636363636363	7	0,488181818181818
3	0,514545454545454	8	0,495454545454545
4	0,441818181818181	9	0,448181818181818
5	0,533636363636363	10	0,469090909090909

$$\bar{x} = 0{,}474818181, \qquad \sigma = 0{,}032761065$$

Tabla 68. Resultado para solver adam con activación relu y 6 capas ocultas.

Iteración	Exactitud	Iteración	Exactitud
1	0,485454545454545	6	0,523636363636363
2	0,483636363636363	7	0,551818181818181
3	0,497272727272727	8	0,474545454545454
4	0,447272727272727	9	0,415454545454545
5	0,366363636363636	10	0,544545454545454

$$\bar{x} = 0{,}478999999, \qquad \sigma = 0{,}057483898$$