UNIVERSIDAD AUTÓNOMA DEL ESTADO DE MORELOS

FACULTAD DE CONTADURÍA, ADMINISTRACIÓN E INFORMÁTICA

LICENCIATURA DE INFORMÁTICA

RED NEURONAL PARA PREDECIR VOTOS ELECTORALES CON PERTURBACIÓN

T E S I S

Que para obtener el Grado de

LICENCIATURA DE INFORMÁTICA

Presenta

JOSE ANTONIO GARCIA PACHECO

Director de Tesis:

DRA. YESSICA YAZMIN CALDERON SEGURA

Co-Director:

DR. GENNADIY BURLAK

CUERNAVACA, MORELOS ABRIL, 2022

Cuernavaca, Morelos a 14 de FEBRERO del 2022 .

MTRO. FELIPE DE JESÚS BONILLA SÁNCHEZ
DIRECTOR DE LA FCAeI
DE LA U.A.E.M.
PRESENTE

Por este conducto, me permito informar a usted que la Tesis denominada: "RED NEURONAL PARA PREDECIR VOTOS ELECTORALES CON PERTURBACIÓN" que presenta el **C**. JOSÉ ANTONIO GARCÍA PACHECO , Pasante de la carrera LICENCIATURA EN INFORMÁTICA .

A mi juicio, cumple con los requisitos de metodología y contenido, por lo que no tengo inconveniente en otorgarle mi voto aprobatorio, para que continúe con los trámites de titulación correspondientes.

Sin otro particular, aprovecho la ocasión para enviarle un cordial saludo.

A T E N T A M E N T E

Dra. Mireya Flores Pichardo

L.I. Baltazar López Ramírez

DR. Gustavo Medina Ángel

M en E. Ana Linda Pineda Méndez

Dr. José Alberto Hernández Aguilar

UNIVERSIDAD AUTÓNOMA DEL ESTADO DE MORELOS

Se expide el presente documento firmado electrónicamente de conformidad con el ACUERDO GENERAL PARA LA CONTINUIDAD DEL FUNCIONAMIENTO DE LA UNIVERSIDAD AUTÓNOMA DEL ESTADO DE MORELOS DURANTE LA EMERGENCIA SANITARIA PROVOCADA POR EL VIRUS SARS-COV2 (COVID-19) emitido el 27 de abril del 2020.

El presente documento cuenta con la firma electrónica UAEM del funcionario universitario competente, amparada por un certificado vigente a la fecha de su elaboración y es válido de conformidad con los LINEAMIENTOS EN MATERIA DE FIRMA ELECTRÓNICA PARA LA UNIVERSIDAD AUTÓNOMA DE ESTADO DE MORELOS emitidos el 13 de noviembre del 2019 mediante circular No. 32.

Sello electrónico

MIREYA FLORES PICHARDO | Fecha:2022-02-14 14:52:06 | Firmante

alNnMtWmmKae9IBEDurm1oY9ulO6CdqRIhCwXwUBDis2kpA/n9EALBZ6njb8N2OH1/GYBWKg0lng1YUQTFxNR1BxZk+/UxGkUW3B+kfVBbz3MpT9iNCDJRQnv1a8O41Dgkq
UByI/+bjnow7XtkWkTBbPecqNBj2SfW/KL2Gh81NpWCKoGvfIEheS9gewLGV3EwUfnVvzD37pKrC7fz5YLGEYsY0aURhDpLBXUkTQ6vMSNywukWBAaNnkaCk0DqXL1BCgEpj
7kRUDjPL7t9epAHH0+hDWka7ZOqIoEBVu53arlMfbSiVjKC9jF8m4gGzW6gkxBH8PQpRT/+hGIq9jRA==

Puede verificar la autenticidad del documento en la siguiente dirección electrónica o escaneando el código QR ingresando la siguiente clave:

XAkj1EMbU

https://efirma.uaem.mx/noRepudio/nvk6ykW8zkl1HqmI02hTwuCj5ihCjJHW

UNIVERSIDAD AUTÓNOMA DEL ESTADO DE MORELOS

Se expide el presente documento firmado electrónicamente de conformidad con el ACUERDO GENERAL PARA LA CONTINUIDAD DEL FUNCIONAMIENTO DE LA UNIVERSIDAD AUTÓNOMA DEL ESTADO DE MORELOS DURANTE LA EMERGENCIA SANITARIA PROVOCADA POR EL VIRUS SARS-COV2 (COVID-19) emitido el 27 de abril del 2020.

El presente documento cuenta con la firma electrónica UAEM del funcionario universitario competente, amparada por un certificado vigente a la fecha de su elaboración y es válido de conformidad con los LINEAMIENTOS EN MATERIA DE FIRMA ELECTRÓNICA PARA LA UNIVERSIDAD AUTÓNOMA DE ESTADO DE MORELOS emitidos el 13 de noviembre del 2019 mediante circular No. 32.

Sello electrónico

BALTAZAR LOPEZ RAMIREZ | Fecha:2022-02-23 01:44:44 | Firmante

NS6Tor39gCFQzzMtyXMCSEcd6DI8g4IBjzJb2p9cy6ily8YAq/5S2AbCzTfmp8y5biN/6ivysVq41HKe4473T1XUcT7P5M8Yalvl1WmXAo5J/tgDlqC6SW/Kzgu5PIkWCO+ML4Q2TOki4frySSAN0c7lzfIETgMiIdBnxs/PJmEvPw1OHxdBlYwKKUAYKhXbhAedrtjTZuFMirRmonCbSJGZsjC3Ph8vpLmy2ByQrH3mPbacXrQQRMiVfGqhbQVqVPdakZw07sUcOpWt8jADnALpL+a1LYzcsjJaHq75AcZ+dd0SlSD+fFgcx4/caSLIVlplzr+2ZePcmba3EOR/FA==

Puede verificar la autenticidad del documento en la siguiente dirección electrónica o escaneando el código QR ingresando la siguiente clave:

4leEynxfi

https://efirma.uaem.mx/noRepudio/AgHOgOOoXc3uKpLVoELhl4jn1TpBRbFW

UNIVERSIDAD AUTÓNOMA DEL ESTADO DE MORELOS

Se expide el presente documento firmado electrónicamente de conformidad con el ACUERDO GENERAL PARA LA CONTINUIDAD DEL FUNCIONAMIENTO DE LA UNIVERSIDAD AUTÓNOMA DEL ESTADO DE MORELOS DURANTE LA EMERGENCIA SANITARIA PROVOCADA POR EL VIRUS SARS-COV2 (COVID-19) emitido el 27 de abril del 2020.

El presente documento cuenta con la firma electrónica UAEM del funcionario universitario competente, amparada por un certificado vigente a la fecha de su elaboración y es válido de conformidad con los LINEAMIENTOS EN MATERIA DE FIRMA ELECTRÓNICA PARA LA UNIVERSIDAD AUTÓNOMA DE ESTADO DE MORELOS emitidos el 13 de noviembre del 2019 mediante circular No. 32.

Sello electrónico

GUSTAVO MEDINA ANGEL | Fecha:2022-02-18 15:36:16 | Firmante

NFU9yNaCWKq1PUqwGdq8RmokbVBJ0ENV0e1Y75p+29/e7XSt+Vsht1N97IIS21w0BYRDVSUAW+8SS3vbyohfioqE3tI+jpCj7VTRvAf1nsqG0P/IDyJs/bEw4bZ8jpFczQUXhk1yjThb8FAhM7HnJR22qvxbrKjYXAUsAWYVCV58yYx/50qVqoImBWi3YMOjOjw1BdtVTPdQyszqkcvV9M/iEwJBEyNeT8t3oF55qs2QJ0BC2jz/6NNKTcjLe29ME2xPQKULdFXQgAmmWCZGhW+k1vVMZdEsktKK9iP3a6NTCTZvBY6NAio8kMlt0s8YB0GjFKmjbKpSE19H1zxAOw==

Puede verificar la autenticidad del documento en la siguiente dirección electrónica o escaneando el código QR ingresando la siguiente clave:

jL3cqBeSY

https://efirma.uaem.mx/noRepudio/sipSeL30nItmWeyE4rQf7nWcn2rWnRKW

UNIVERSIDAD AUTÓNOMA DEL ESTADO DE MORELOS

Se expide el presente documento firmado electrónicamente de conformidad con el ACUERDO GENERAL PARA LA CONTINUIDAD DEL FUNCIONAMIENTO DE LA UNIVERSIDAD AUTÓNOMA DEL ESTADO DE MORELOS DURANTE LA EMERGENCIA SANITARIA PROVOCADA POR EL VIRUS SARS-COV2 (COVID-19) emitido el 27 de abril del 2020.

El presente documento cuenta con la firma electrónica UAEM del funcionario universitario competente, amparada por un certificado vigente a la fecha de su elaboración y es válido de conformidad con los LINEAMIENTOS EN MATERIA DE FIRMA ELECTRÓNICA PARA LA UNIVERSIDAD AUTÓNOMA DE ESTADO DE MORELOS emitidos el 13 de noviembre del 2019 mediante circular No. 32.

Sello electrónico

ANA LINDA PINEDA MENDEZ | Fecha:2022-02-14 21:29:27 | Firmante

BXlCLheUZr7XUWC2SuuuuSfhPWBlIWgMTqH3BbDbbz6+pmrQmUZvdotKUYd9YF3FZtvQ42Hu58xcSAxMTnWpfhoom5XSQh74yKPBR6Bo6UZ9GoMBR+tnY1GT8faUYibpQR
eOIFpXfoym+unpbp2y8pbpIQAll1wKk+xO6/C0wJQshGgSTeLHR0cmSEkvuS8GDQVibNxhq7eWr0EQ4lS7SBMfv1+oXemtT27IAnX+5uyKvBOhxeYGSDi039NqrBUIZd4nu/nMtL
5WVTDJ30DoqQTIv0xpIs8YV4ds5cxcCevRm0LB/cN4Duebnfl8DvaI+Hz3ibFhr1Rs94tqHdANzA==

Puede verificar la autenticidad del documento en la siguiente dirección electrónica o escaneando el código QR ingresando la siguiente clave:

ngMZkdGfs

https://efirma.uaem.mx/noRepudio/85OGiQLMjF2rU7ylzvKKd5zQ2EWehWU4

Se expide el presente documento firmado electrónicamente de conformidad con el ACUERDO GENERAL PARA LA CONTINUIDAD DEL FUNCIONAMIENTO DE LA UNIVERSIDAD AUTÓNOMA DEL ESTADO DE MORELOS DURANTE LA EMERGENCIA SANITARIA PROVOCADA POR EL VIRUS SARS-COV2 (COVID-19) emitido el
27 de abril del 2020.

El presente documento cuenta con la firma electrónica UAEM del funcionario universitario competente, amparada por un certificado vigente a la fecha de su elaboración y es válido de conformidad con los LINEAMIENTOS EN MATERIA DE FIRMA ELECTRÓNICA PARA LA UNIVERSIDAD AUTÓNOMA DE
ESTADO DE MORELOS emitidos el 13 de noviembre del 2019 mediante circular No. 32.

Sello electrónico

JOSE ALBERTO HERNANDEZ AGUILAR | Fecha:2022-03-10 18:07:59 | Firmante

UsnC8blets97Gnb6zOO0pjhKBZDitVM0KfTGdf9D1P9DJTlGKGtUC9mnCPxp5HLsslQxbOD9MuguTl6LTiHWCwMXnA/viayfs+iPIFFS73nFq6Q2zISaXKbeWl5Exs1CGFaP+qaQ
2r9UJruLFBX6In/w70ZgDwfFfAyNNZVJrA/AliMzmUBsk4OBY0YYcXiXSC/GZvE6nZOewb4gtSuWcjXtVs2DTBLyZFoXcH3YJVx/KLLzK+OqjyC1vPTIOwvL7ZxBUka9Mozenfcog4YQFG0NNel3ke6Hb2p9T6bSPbx7uPO2Lgk7Ek2Zsqv6px/NCCDHpES+mOZ2s6ga+x5C7Q==

Puede verificar la autenticidad del documento en la siguiente dirección electrónica oescaneando el código QR ingresando la siguiente clave:

IKR4YMI0D

https://efirma.uaem.mx/noRepudio/0CaYg7f11eyLTMU7KUFGrPUlou3F2ARR

Dedicatoria

Esta tesis está dedicada a mi madre Emma Pacheco, por apoyarme cuando más la necesito, por extender su mano en momentos difíciles. Quien siempre me dio esperanzas y tuvo fe en mí.

A mi tío Mauricio García, quién me ha apoyado en cada etapa de mi vida y, facilitó mi investigación compartiendo su hogar conmigo cuando necesité un lugar para quedarme.

A toda mi familia porque con sus oraciones, consejos y palabras de aliento hicieron de mí una mejor persona y de una u otra forma me acompañan en todos mis sueños y metas.

Finalmente quiero dedicar esta tesis en memoria de mi padre Eleuterio García quien me enseñó que el mejor conocimiento que se puede tener es el que se aprende por sí mismo. Su ejemplo me mantuvo soñando cuando quise rendirme.

Agradecimientos

Agradezco a mi asesora de tesis Dra. Yessica Yazmin Calderón Segura por sus consejos, enseñanzas, apoyo y sobre todo amistad brindada en los momentos más difíciles de mi vida.

A mi co-asesor Dr. Gennadiy Burlak quien, con su experiencia, conocimiento y motivación me oriento en la investigación.

Agradezco a los todos docentes que, con su sabiduría, conocimiento y apoyo, motivaron a desarrollarme como persona y profesional en la Facultad de Contaduría Administración e Informática de la Universidad Autónoma del Estado de Morelos.

Índice de contenido

Índice de Figuras

Índice de Tablas

Índice de Ecuaciones

INTRODUCCIÓN

Resumen

El objetivo de este estudio, es buscar los principales factores que puedan influir para predecir los resultados de las encuestas de votación. Se desarrolla un sistema que permita la optimización de Redes Neuronales Artificiales para identificar los factores que inciden en el resultado electoral, a través de un método computacional que permite la evaluación de las características que influyen en una votación electoral exitosa. Se utiliza una Red Neuronal Artificial con tres capas y un algoritmo de aprendizaje con propagación hacia atrás.

Dentro de la primera fase se carga el sistema desarrollando una base de datos sintética aleatoria. Esta contendrá los datos que servirán como entrada a la Red Neuronal Artificial.

En la segunda fase se crea la Red Neuronal Artificial con parámetros al azar, los datos son segmentados en pruebas y entrenamiento, La Red Neuronal hace su computo en cada capa mediante pesos y sesgos, en la capa final se hace uso del algoritmo de la retro propagación, Buscando obtener una precisión final lo más acertada posible.

Al final se sintonizan los parámetros de la Red Neuronal Artificial, y se entrena la red con los parámetros sintonizados previamente, obteniendo así la optimización del resultado, tomando en cuenta los atributos más sobresalientes que afectan una votación.

Antecedentes del proyecto

Las Redes Neuronales Artificiales surgen de la interpretación del funcionamiento de un cerebro humano. Aunque el primero en relacionar la computación con el cerebro humano fue Alan Turing en 1936, fueron Warren McCulloch y Walter Pitts los que crearon la teoría sobre el funcionamiento de una neurona (Hilera & Martinez, 1995).

Joel W. Johnson en el 2012 presenta los principales factores que afectan la desigualdad de votos entre las cohortes titulares (miembros del mismo partido y distrito), lo que indica la fuerte influencia de los incentivos de la división de votos sobre los entornos electorales centrados en los candidatos (Johnson & Hoyo, 2012).

El estudio desarrollado por Ching-Hsing Wang en el 2015 indica que la conciencia y la estabilidad emocional pueden aumentar significativamente la participación femenina en los votos electorales, pero no tienen ningún efecto sobre la participación masculina. Además, la apertura a la experiencia ejerce efectos opuestos en la participación masculina y femenina. A medida que aumenta la apertura a la experiencia, los hombres tienen más probabilidades de votar, mientras que las mujeres tienen menos probabilidades de emitir su voto. Sin embargo, la extraversión y la amabilidad no están asociadas con la participación, independientemente del género (Wang, 2014).

Orlando D’Adamo en julio del 2015, hace el estudio de la utilidad y alcance del uso de las redes sociales durante las campañas electorales. Los autores (D’Adamo, Beaudoux, & Kievsky, 2015) presentan resultados de una investigación que analiza el empleo de las redes sociales efectuado por los candidatos a diputados y senadores por la ciudad de Buenos Aires en las elecciones legislativas.

Dimitrios Xefteris en el 2016 considera el modelo de Hotelling-Downs para analizar la incertidumbre en candidatos sobre la preferencia de los votantes (Xefteris, 2016).

Dimitrios Xefteris en el 2017 Hace el estudio de algunos factores que afectan las votaciones electorales estos son religión, raza, cultura, etc. Haciendo uso de acceso de datos optimizados para Data Warehouse maximiza la participación en el voto diferenciado (Dimitrios, 2017).

Las Redes Neuronales Artificiales pueden ser utilizadas para pronosticar las dificultades del proceso electoral, ya que han sido usadas para distintos problemas complejos con mecanismos adaptativos y cognitivos del aprendizaje humano. La literatura nos indica que entrenar una red neuronal es un problema de optimización NP-hard que representa varias limitaciones teóricas y computacionales (Delgado, Rasúa, & Bello, 2019).

Dat Thanh Tran en el mes de enero 2019, propone una biblioteca para evitar el cuello de botella en Machine Learning haciendo uso del perceptrón (Thanh, Serkan, Moncef, & Alexandros, 2019), también se han publicado Redes Neuronales Recurrentes para Modelado de datos Secuenciales, para el reconocimiento de voz considerando la morfología de las palabras, la sintaxis y la semántica (Tijskens, Roels, & Janssen, 2019).

Diego F. Aranha en el 2019, presento un análisis de seguridad detallado y actualizado del software de votación utilizado en las elecciones brasileñas basado en los resultados obtenidos por los autores en un reciente desafío de piratería organizado por la autoridad electoral nacional (Aranha, Barbosa, Cardoso, Araújo, & Matias, 2019).

Las Redes Neuronales Artificiales nos permiten simular una evaluación probabilística que se tiene en cuenta como incertidumbre lo que permite una evaluación del comportamiento, en este caso se consideran los votos electorales. Se debe resaltar que los votos electorales representan un tema de investigación para interpretar y analizar los procesos sociales y demográficos. La red neuronal proporciona los resultados experimentales que demuestran la efectividad de los datos simulados.

La literatura demuestra que la información que influye para la elección electoral posee incertidumbre. Se tiene claro que influyen diversos factores y atributos para conocer al ganador de las elecciones, se debe resaltar que hacer esto de forma manual representa una carga de trabajo muy laborioso, y conocer los principales atributos que influyen siempre es de gran importancia tanto para los votantes como los candidatos postulados.

Planteamiento del problema

Se genera una base de datos de forma aleatoria con los que la Red Neuronal Artificial pueda evaluar los atributos que afectan una votación. Estos deben ser seguros para evitar inconsistencias ya que los datos deberán ser administrados por un software el cual de acuerdo a las restricciones del problema y a la información de entrada obtenida de la base de datos, mostrarán la eficiencia del algoritmo propuesto.

Este software es un algoritmo que dará una solución para encontrar el atributo con más eficiencia con un método de retro propagación. La base de datos tendrá la información inicial necesaria que es requerida para la Red Neuronal Artificial. Estas representan las diferentes posibilidades para hacer una elección al votar entre las cuales se encuentran la economía, movimientos socio culturales y trabajo. Tomando eso en cuenta, los datos de entrada a la Red neuronal Artificial son los siguientes:

a) Edad.
b) Ingreso.
c) Educación.
d) Deuda.

Y las variables de salida son los partidos políticos a tomar en consideración, en este caso utilizáremos los patrones de ideología política más comunes entre los cuales suelen clasificarse como de Izquierda, derecha y agnóstico o central. Con los cuales crearemos tres partidos políticos ficticios denominados:

1) Conservador (derecha).
2) Moderado (centro).
3) Liberal (izquierda).

Para comprender la manera en la cual se encasilla a los simpatizantes de cada partido se muestra en la Figura 0.1. La distribución esquemática del espectro político.

Figura 0.1 Espectro político

Esta distribución política *Figura 0.1* tiene su origen en 1789 en Francia durante la Asamblea Nacional Constituyente en la cual se discutía la revocación del poder político a la monarquía. Quienes estaban en contra se situaron a la derecha y quienes promovían un cambio, buscando la soberanía nacional se situaron a la izquierda. (Péronnet, 1985).

Dicha distribución se modificó conservando las mismas bases políticas pues a principios del siglo XIX la aristocracia fue suplantada por la burguesía como la clase predominante.

Con lo cual podemos decir que la política liberal o de izquierda busca la igualdad política, para el progreso del pueblo. Sin imponer a los más la ley de los menos. La política de derecha o conservadora representa el mantener el orden político actual, en el cual representa a quienes poseen poder y riquezas buscando el bien individual sin tomar en cuenta a todas las clases sociales. La política moderada ha ganado popularidad en los últimos años pues representa la unión entre la política liberal y la política conservadora tratando de tomar lo mejor de ambas partes.

La idea de afiliarse y discernir el voto en la escala de Izquierda a derecha conlleva la aceptación a la forma de trabajo de ese grupo de personas teniendo en cuenta la forma en la que afrontan los problemas, es decir los medios utilizados para la resolución de conflictos. Generando así la pelea con la otra parte de la escala atribuyéndole connotaciones peyorativas a la identidad de la oposición y viceversa. (Bobbio, 2001).

Recientemente existe una tensión debido a circunstancias no tan claras entre las geometrías políticas dejando fuera el debate al diferenciar el pensamiento político, enfocándose en la desacreditación del adversario. Estos actos causan confusión en los electores frustrando al razonamiento de su voto, esto quiere decir que la decisión electoral en muchos casos es avasallada por la economía, movimientos socio culturales y el trabajo, a la hora de tomar una decisión.

La perturbación en las elecciones electorales representa a la incertidumbre como un elemento clave en la optimización de resultados, gran parte de encuestas electorales no son realizadas con el rigor necesario y en lugar de integrar la incertidumbre en la ecuación se contempla un margen de error estimado, pero no se trata de reducir el error. En consecuencia, pueden tener repercusión en un segmento de la población con respecto a la formación de opinión propia, y así deliberadamente distorsionar el propósito de la realización de las encuestas electorales por los encuestadores, con la difusión en medios de comunicación (Valdez, 2013).

Objetivo general

Probar la eficiencia del uso de redes neuronales artificiales en encuestas electorales, para la toma de decisiones dentro de partidos o elementos relacionados a la política.

Objetivos específicos

- Crear una Red Neuronal Artificial
- Entrenar a la Red Neuronal Artificial con el método de retro propagación
- Sintonizar los parámetros de la Red Neuronal Artificial
- Analizar bases de datos sintéticas dentro de la Red Neuronal Artificial entrenada y con los parámetros óptimos, para comprobar su funcionalidad.

Justificación

Las encuestas solo representan el panorama en el momento que se realiza el estudio por lo cual ya no son vigentes en un futuro, debido a que una gran parte de los encuestadores toman en cuenta un margen de error, pero no contemplan cambios, ni la variación fuera de la muestra del estudio, razón por la cual un gran porcentaje de las encuestas realizadas previas a una votación electoral son deficientes y difieren del resultado de la decisión final en los votantes (Alanís, 2012).

De acuerdo con la revista este país. Tan solo en México en el marco de las elecciones efectuadas el pasado 1 julio del 2018 se estima que durante el mes previo con 25 encuestas realizadas entre el 1 y el 27 de junio. Una de cada cuatro de las predicciones electorales es deficiente (Abitia, 2018).

Con base en el funcionamiento de las redes neuronales artificiales se estima que puede reducirse el tiempo de elaboración de encuestas electorales, con un mejor manejo de información en los cambios que acontecen en torno a las partes involucradas previo a las elecciones. para así poder medir de manera acertada el comportamiento de la población.

Hipótesis

H_1: La aplicación de Redes Neuronales Artificiales a través del método de aprendizaje de propagación hacia atrás, puede optimizar el resultado obtenido en una encuesta electoral.

Hipótesis nula

H_0: La aplicación de Redes Neuronales Artificiales a través del método de aprendizaje de propagación hacia atrás, no presenta un mejor resultado obtenido en una encuesta electoral.

Organización de la tesis

INTRODUCIÓN.

En este capítulo se presentan los antecedentes del proyecto, el planteamiento del problema, el objetivo general, y la justificación del mismo, los alcances y Organización de tesis.

CAPÍTULO 1. MARCO TEORICO.

En este capítulo se presenta la explicación de todo lo que engloban las Redes Neuronales Artificiales, como la partes que las conforman, tipos de redes y su funcionamiento.

CAPÍTULO 2. MODELO MATEMATICO.

En este capítulo se especifica el diseño de la solución, se presentan también la estructura y modelo matemático necesario para el funcionamiento del sistema.

CAPÍTULO 3. IMPLEMENTACIÓN.

En este capítulo se describen las funciones de sistema y todos sus componentes para su uso correcto.

CAPÍTULO 4. RESULTADOS.

En este capítulo se muestran evidencias gráficas de que el sistema cumplió con el objetivo y los requerimientos planteados inicialmente.

CONCLUSIONES.

Se presentan las lecciones aprendidas durante el desarrollo del sistema, las potencialidades de uso de la tecnología empleada para otros casos prácticos. Así como trabajos a futuro.

BIBLIOGRAFÍA.

Utilización de fuentes bibliográficas de libros utilizados para definir términos y conocimientos determinados. También se hace referencia a las páginas Web visitadas.

ANEXOS.

Los anexos agregados hacen referencia al glosario de términos, formulas, y códigos utilizados en la tesis.

Capítulo 1 MARCO TEÓRICO

1.1¿Que son las Redes Neuronales?

Las redes neuronales son un conjunto de neuronas cuya función base es recibir, procesar y transferir información mediante señales, es decir, que se comunican entre sí rápida y eficazmente ya que lo realizan de forma paralela (C. Aldrich, 1995).

1.2 Clasificación de las Redes Neuronales

1. Modelos inspirados en la Biología: Estos comprenden las redes que tratan de simular los sistemas neuronales biológicos, así como ciertas funciones como las auditivas o de visión.

2. Modelos artificiales aplicados: Estos modelos no tienen por qué guardar similitud estricta con los sistemas biológicos. Sus arquitecturas están bastante ligadas a las necesidades de las aplicaciones para las que son diseñados.

1.3 Modelo Biológico

Las neuronas biológicas son únicas, ya que estás pueden comunicarse entre sí, la información de entrada se recibe en las dendritas que la pasan al cuerpo, El cuerpo o soma hace la operación y genera información de salida, El axón pasa la información a los terminales axónicos y estos a su vez comparten la información con otras neuronas para propagar el proceso millones de veces más (MICHELONE, 2019). Como se muestra a continuación en la Figura 1.1.

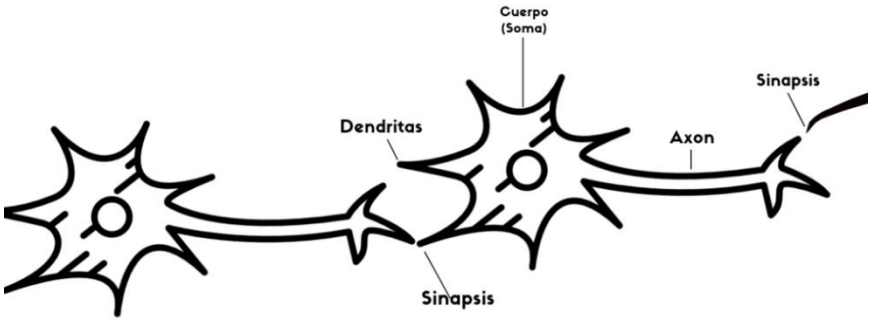

Figura 1.1 Representación de neuronas biológicas (Arbib, 2003)

1.4 Redes Neuronales Artificiales

A continuación, se explica la Historia de las Redes Neuronales Artificiales y se muestran las partes que las componen.

1.4.1 Historia de las Redes Neuronales Artificiales

Desde la invención de las máquinas computacionales se tenía la visión de poder crear una máquina que sea capaz de aprender y tomar decisiones por sí misma, uno de los primeros pasos que se dio, es tener la primera máquina de tal magnitud su creador fue Alan Turing quien es considerado uno de los padres de la ciencia de la computación y la informática moderna.

Incluso al día de hoy a más de 60 años de su muerte, la inteligencia artificial es puesta a prueba en el denominado “Test de Turing” el cual fue publicado en su artículo de 1950 titulado Computing machines and intelligence, en el cual una maquina trata de engañar a un ser humano haciéndose pasar por otro ser humano. (Turing, 1950)

Aunque las Redes Neuronales Artificiales existen desde mediados del siglo XX a causa de sus limitantes fueron abandonas por un periodo, ha sido hasta hace unos años que gracias al avance tecnológico tienen su resurgimiento con una infinidad

de aplicaciones, tal es su impacto que al día de hoy están presentes en casi todos los ámbitos relacionados con la inteligencia artificial (John F. Kolen, 2001).

Las redes neuronales artificiales se encuentran desarrolladas en la estructura y funcionamiento de los sistemas nerviosos, donde la neurona es el elemento principal.

1.4.2 Concepto de las Redes Neuronales Artificiales

"Redes neuronales artificiales son redes interconectadas masivamente en paralelo de elementos simples (usualmente adaptativos) y con organización jerárquica, las cuales intentan interactuar con los objetos del mundo real del mismo modo que lo hace el sistema nervioso biológico" (Kohonen, 1995).

Una Red Neuronal Artificial es una función matemática compleja inspirada en el funcionamiento de su homónimo biológico. Pero también es la interacción de muchas partes más simples llamadas neuronas, trabajando conjuntamente, las cuales que tienen entradas y salidas numéricas. Y su objetivo es resolver problemas de una manera similar al cerebro humano (Kohonen, 1989).

1.4.3 Cómo funcionan las Redes Neuronales Artificiales

La red neuronal es la integración de muchas neuronas en la cual cada neurona realiza una suma ponderada cuya ponderación está dada por el peso que se le asigna a cada una de las conexiones de entrada. Lo que significa que cada conexión que llegue a la neurona tendrá asociado un valor que definirá la intensidad con la que afectara la variable de entrada o input a la neurona y por consiguiente influirá en el resultado que arroje la capa de salida u output (Ballesteros, 2001).

1.5 Tipos de Redes Neuronales Artificiales

A continuación, se explican cuatro tipos de Redes Neurnales Artificiales más comunes.

1.5.1 Red neuronal de propagación hacia atrás

Las redes con propagación hacia atrás (back-propagation) emplean la retroalimentación siendo un método supervisado, consta de tres capas: entrada, oculta y de salida. Teniendo mejor precisión debido a que el error se propaga de forma inversa, es decir, comienza desde la capa de salida pasando entre la capa oculta para llegar a la capa de entrada (Rojas, 1996).

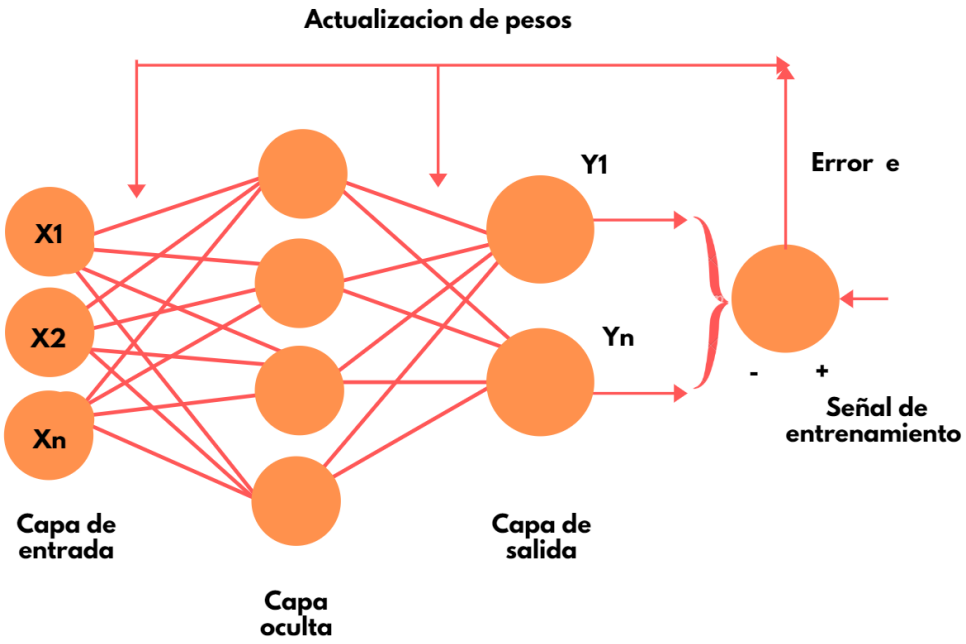

Figura 1.2 Representación de una red neuronal de propagación hacia atrás

Como se muestra en la *Figura 1.2* las variables *X* desde X*1* hasta X*n* representan las entradas a la red, y la variable *Y* desde Y*1* hasta Y**n** representa el resultado obtenido de las neuronas en la capa de salida.

1.5.2 Red neuronal Kohonen

A diferencia de la red Neuronal de propagación hacia atrás la Red Neuronal de Kohonen como se aprecia en la *Figura 1.3* es más simple al tener una sola capa, que utilizan un método no supervisado por lo cual no posee vector específico para ser capacitada, entre otras razones que afectan en la veracidad del resultado en la salida (Albán, 2016).

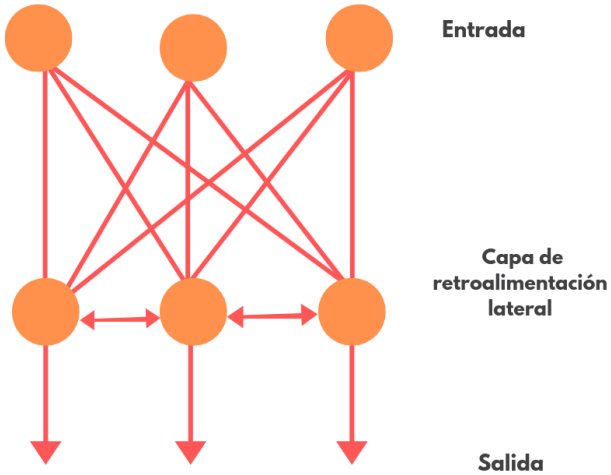

Figura 1.3 Representación de una red neuronal de Kohonen.

1.5.3 Red neuronal de propagación posterior circular

Es similar a la Back Propagation convencional, pero con una modificación para una menor complejidad en el algoritmo de entrenamiento y adecuarla a aplicaciones en tiempo real. (Ridella, Rovetta, & Zunino, 1997). Cambiando también la forma lineal por una forma circular estando la capa de entrada cerca de la capa de salida obteniendo así una conexión inmediata entre ambas capas como se muestra en la *Figura 1.4*.

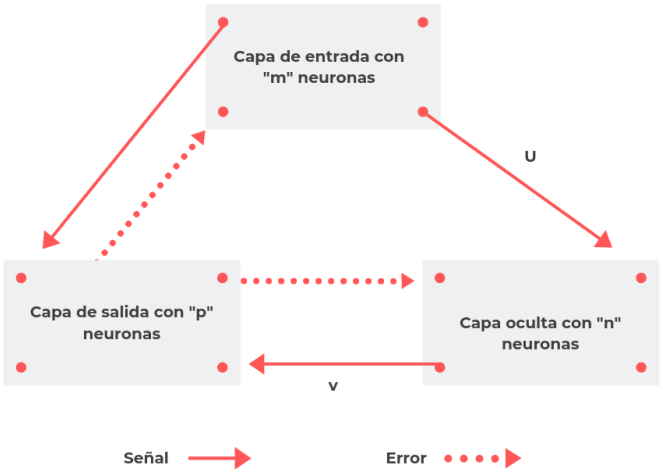

Figura 1.4 Representación de una red neuronal de propagación posterior circular.

1.5.4 Red Neuronal profunda de Kohonen

Presentada para mejorar la precisión de la red convencional de Kohonen con una arquitectura de tres capas: entrada, resumen y salida (JudeHemanth, 2018). La diferencia entre esta red y la convencional de Kohonen es la inserción de la capa abstracta entre la capa de entrada y la capa de salida representada a continuación en la *Figura 1.5*.

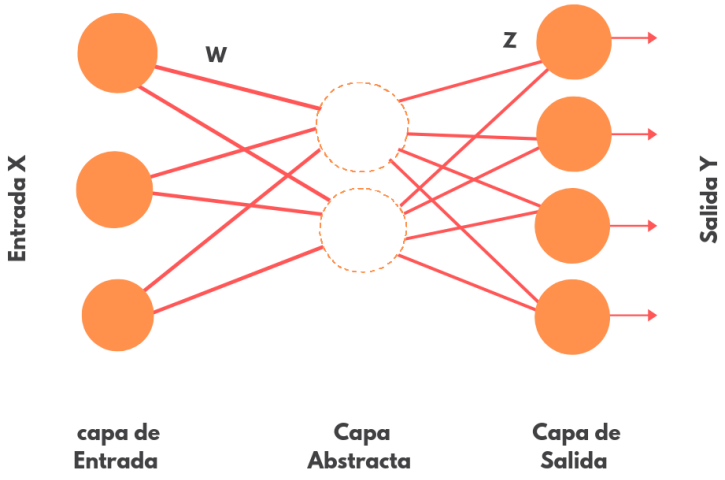

Figura 1.5 Representación de una red neuronal profunda de Kohonen.

Capítulo 2 MODELO MATEMÁTICO

2.1 Descripción de las Redes neuronales Artificiales

A continuación, se describen las características y representación del modelo artificial de un Red neuronal.

2.1.1 Neurona

Es la unidad base de una red neuronal, básicamente es un procesador elemental el cual procesa un vector x y produce una salida resultante de la suma ponderada (Fukushima, 1980).

El modelo de una neurona artificial es una imitación del proceso de una neurona biológica, tal como se ve en la *Figura 2.1*.

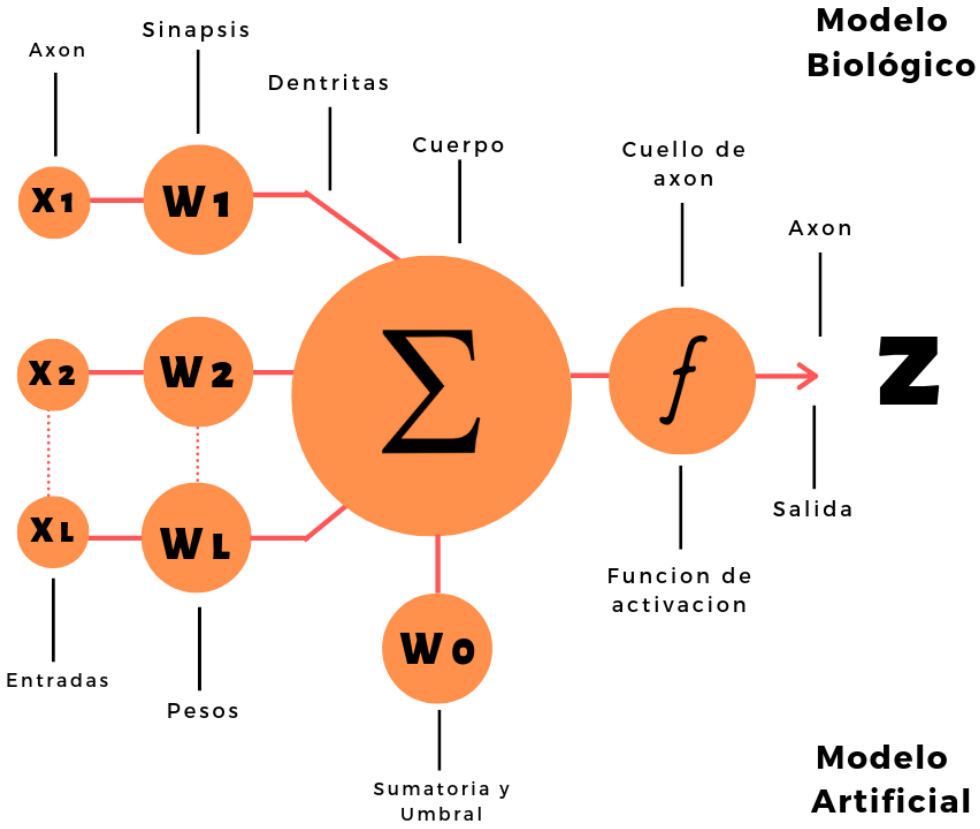

Figura 2.1 Similitud entre redes neuronales biológicas y artificiales (Haykin, 2009).

Donde los x^l son las entradas (por las dendritas) a la neurona. Estas sufren un efecto multiplicador en el peso w^l, por la comunicación de las mismas al núcleo de la neurona, y b es el sesgo. Obteniendo así la siguiente formula como se puede apreciar en la *Figura 2.1.* **z = función de activación [∑ (peso x entrada) + sesgo].**

2.1.2 Capa

Las Redes Neuronales Artificiales normalmente se dividen en tres capas. La primera es la capa de entrada, en medio esta la capa oculta y termina con capa de salida, cada capa posee un cierto número de neuronas como se aprecia en la *Figura 2.2*.

La capa de entrada recibe las variables de entrada o input realiza la operación correspondiente y transfiere sus resultados a la capa oculta esta a su vez procesa la información y la manda a la capa de salida, generando un resultado u output (Cires, Meier, Masci, Gambardella, & Schmidhuber, 2011).

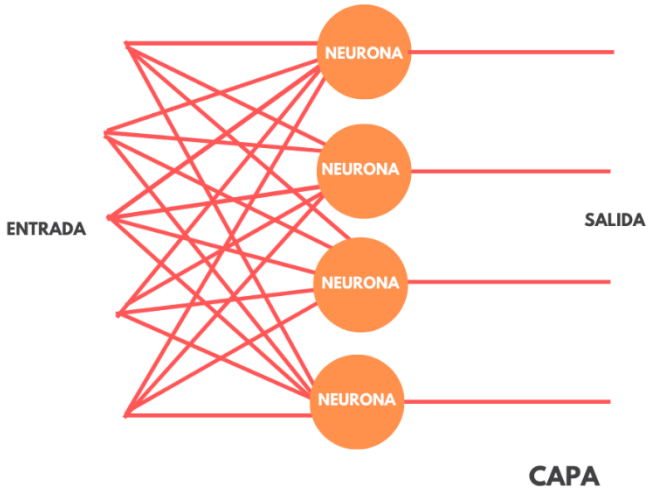

Figura 2.2 Representación de capa en la red neuronal

2.1.3 Red neuronal

La red neuronal es un conjunto que engloba las capas dentro de ella y sus respectivas neuronas obteniendo así la interconexión de todas las neuronas para su funcionamiento óptimo (Martínez LAE, 2001) representada en la *Figura 2.3*.

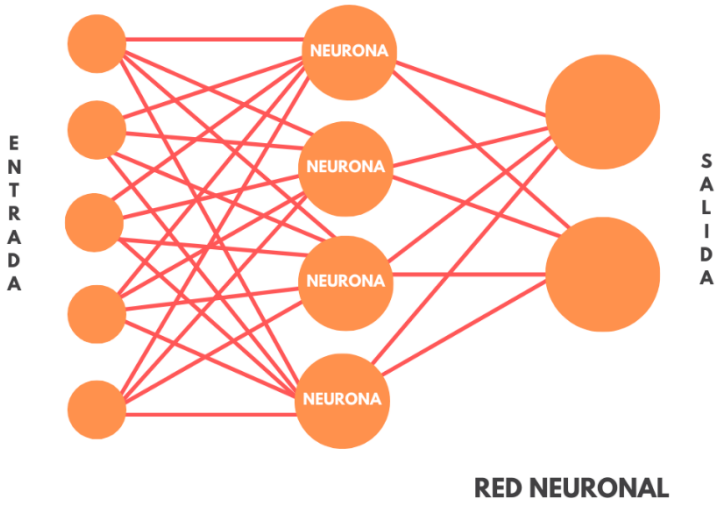

Figura 2.3 Representación de una red neuronal

2.2 Partes del Modelo matemático

La característica básica de una red neuronal es la que está compuesta de tres capas. La capa de entrada se encarga de recibir los valores de entrada y enviar estos valores a la segunda capa denominada capa oculta y estas realizan su proceso y transfieren la información a la capa de salida, la red puede contener más capas si así se requiere pueden ser modificadas agregando o eliminando variables de entrada, de salida o cambiando el proceso de aprendizaje o entrenamiento (Callejas, Piñeros, Rocha, Hernández, & Delgado, 2013).

Una red neuronal convencional está compuesta de tres características:

- El modelo de interconexión entre las diferentes capas de la red.
- Desarrollo de aprendizaje en la variación de pesos entre las interconexiones.
- La función de activación modifica el formato resultado ponderado de la red en el valor de activación de salida.

En este caso se utilizará una Red Neuronal Entrenada con un algoritmo de propagación hacia atrás orientado en el algoritmo del descenso del gradiente y la utilización de la regla de cadena o chain-rule.

2.2.1 Función de activación

La función de activación es utilizada para modificar los datos e introducirlos dentro de un rango más corto para hacer un cálculo más simple (Zhengbing Hu, 2016).

A continuación, tenemos la función de activación que utilizaremos en nuestro algoritmo de retro propagación.

2.2.1.1 Función Sigmoide

Esta función modifica los valores de entrada. donde los valores altos son aproximados a 1 y los valores muy bajos se acercan a 0 y está representada en la ***Ecuación 2:1***.

$$f(x) = \frac{1}{1 - e^{-x}}$$

Ecuación 2:1 Función Sigmoide

2.2.2 Retro Propagación

La retro propagación o Back-propagation es un algoritmo ampliamente utilizado en el entrenamiento de redes neuronales de avance para el aprendizaje supervisado. Funciona calculando el gradiente de la función de pérdida con respecto a cada peso por la regla de la cadena, iterando hacia atrás una capa a la vez desde la última capa para evitar cálculos redundantes de términos intermedios en la regla de la cadena y se basa en derivadas parciales de cálculo. Cada valor de peso y sesgo tiene una derivada parcial asociada. Puede pensar en una derivada parcial como un valor que contiene información sobre cuánto y en qué dirección debe ajustarse un valor de peso para reducir el error. La colección de todas las derivadas parciales se llama gradiente. Sin embargo, por simplicidad, cada derivada parcial se llama comúnmente un gradiente (Toda Caraballo, 2010).

2.2.3 Regla de cadena

Sí una variable y , depende de una segunda variable u , que a la vez depende de una tercera variable x ; entonces, la razón de cambio de y con respecto a x puede ser calculada con el producto de la razón de cambio de y con respecto a u multiplicado por la razón de cambio de u con respecto a x .

$$\frac{dy}{d\mathrm{x}} = \frac{dy}{d\mathrm{u}} \cdot \frac{du}{dx}$$

Ecuación 2:2 Regla de cadena

Si $g(x)$ es diferenciable en el punto x y $f(x)$ es diferenciable en el punto $g(x)$, entonces $f°$ es diferenciable en x. Además, deja $y = f\big(g(x)\big)$ y $u = g(x)$, luego se obtiene la *Ecuación 2:2*.

2.2.4 Función de coste

La función de coste trata de determinar el error entre el valor estimado y el valor real, con el fin de optimizar los parámetros de la red neuronal. En este caso utilizaremos el error cuadrático medio.

2.2.4.1 Error Cuadrático Medio

En el análisis de regresión el Error Cuadrático Medio se refiere a la media de las desviaciones al cuadrado de las predicciones de los verdaderos valores, a lo largo un espacio fuera de la muestra de ensayo, generado por un modelo estimado durante un espacio de muestra particular. Su fórmula se muestra en la *Ecuación 2:3*.

$$c\big(a_j^l\big) = \frac{1}{2}\sum_j (y_j - a_j^l)^2$$

Ecuación 2:3 Error Cuadrático Medio.

2.3 Modelo matemático de Nuestra Red Neuronal Artificial

Inicialmente los valores de la red son generados aleatoriamente. Con esto es muy probable que el error obtenido sea muy elevado por lo cual la red debe ser entrenada para obtener el mínimo error posible.

Empezaremos calculando la derivada de los parámetros en la última capa, en la cual el resultado obtenido de suma ponderada que se muestra a continuación en la *Ecuación 2:4*.

$$z^l = w^l x + b^l$$

Ecuación 2:4 Suma ponderada en la última capa sin f. de activación.

z^l Resultado de la suma ponderada w^l peso b^l sesgo

Posteriormente a ese resultado se le agrega la función de activación y coste dando como resultado el error obtenido de la red representado en la *Ecuación 2:5*.

$$c(a(z^l)) = Error$$

Ecuación 2:5 Error obtenido de la red.

z^l Resultado de la suma ponderada a Función de activación c F. coste

Lo que buscaremos será la derivada parcial del costo respecto a los parámetros de peso y sesgo con lo cual tendremos que calcular dos derivadas.

Como hemos dicho, vamos a empezar a trabajar de atrás hacia adelante por lo tanto comenzamos a calcular la derivada de los parámetros de la última capa. El número de la capa a la que pertenece el parámetro si nuestra red neuronal tiene l capas.

para calcular esta derivada es importante analizar cuál es el camino que conecta el valor del parámetro, y el coste final en la última capa de este camino no es muy largo aunque aun así tiene varios pasos, anteriormente vimos que en funcionamiento de la neurona el parámetro w participaba en una suma ponderada ahora que nos vamos a referir como z que luego sería pasada por la función de activación representada en la *Ecuación 2:1* y el resultado de las activaciones de la neurona en la última capa conformaría en el resultado de la red que luego sería evaluada por la función del coste descrito en Tema 2.2.4. Teniendo c para determinar así el error de la red.

Con esto se forma una composición de funciones y utilizaremos una herramienta de cálculo matemático llamada Regla de cadena representada en la *Ecuación 2:6*. Para calcular la derivada de composición de funciones lo que nos dice es que para calcular la derivada de una composición de funciones simplemente tenemos que multiplicar cada una de las derivadas intermedias. Tomando en cuenta las funciones *Ecuación 2:6* y *Ecuación 2:7*.

$$z^l = w^l a^{l-1} + b^l$$

Ecuación 2:6 Suma ponderada de la última capa.

$$c(a(z^l)) = Error$$

Ecuación 2:7 Error obtenido de la red.

Obtendremos la derivada del peso con respecto al costo *Ecuación 2:8* y la derivada del sesgo con respecto al costo *Ecuación 2:9*.

$$\frac{\partial c}{\partial w^l} = \frac{\partial c}{\partial a^l} \cdot \frac{\partial a^l}{\partial z^l} \cdot \frac{\partial z^l}{\partial w^l}$$

Ecuación 2:8 Derivada del peso con respecto al costo.

$$\frac{\partial c}{\partial b^l} = \frac{\partial c}{\partial a^l} \cdot \frac{\partial a^l}{\partial z^l} \cdot \frac{\partial z^l}{\partial b^l}$$

Ecuación 2:9 Derivada del sesgo con respecto al costo.

Obteniendo así tres derivadas parciales en donde la derivada de la activación con respecto al costo *Ecuación 2:9* para obtener la variación de costo de la red cuando se varía el output la activación de las neuronas en la última capa, es decir que se hace una derivada de la función de costo con respecto al output de la red neuronal.

La función de coste que utilizaremos en este caso será el error cuadrático medio descrita en la *Ecuación 2:3* con los parámetros de nuestra red *Ecuación 2:11*.

$$\frac{\partial c}{\partial a^l}$$

Ecuación 2:10 derivada parcial de la activación con respecto al costo.

$$c\left(a_j^l\right) = \frac{1}{2}\sum_j (y_j - a_j^l)^2$$

Ecuación 2:11 Error cuadrático medio de la red.

Así la derivada de la función con respecto a la salida u output de la red quedaría representada *Ecuación 2:12* de la siguiente forma.

$$\frac{\partial c}{\partial a_j^l} = \ (a_j^l - y_i)$$

Ecuación 2:12 derivada de la función de activación con respecto a la salida de la red.

Continuamos con la función activación *Ecuación 2:13* y su derivada respecto a la suma ponderada *Ecuación 2:14.* Lo que nos revela la variación de salida de la neurona cuando se varia la suma ponderada de la neurona calculando así la función de activación.

Esta derivada se calcula dependiendo del tipo de función de activación, en este caso utilizaremos la función sigmoide *Ecuación 2:1*.

$$a^l(z^l) = \frac{1}{1+e^{-z^l}}$$

Ecuación 2:13 Función de activación de la suma ponderada.

$$\frac{\partial a^l}{\partial z^l} = a^l(z^l) \cdot (1 - a^l(z^l))$$

Ecuación 2:14 Derivada de la función de activación con respecto a la suma ponderada.

con todo eso solo nos faltarían dos derivadas parciales con respecto a sesgo *Ecuación 2:16* y peso *Ecuación 2:17*. Estas se obtienen derivando la suma ponderada de la neurona como se muestra a continuación en la *Ecuación 2:15*.

$$z^l = \sum_i a_i^{l-1} \; w_i^l + b^l$$

Ecuación 2:15 Derivación de la suma ponderada de la neurona.

$$\frac{\partial z^l}{\partial b^l} = 1$$

Ecuación 2:16 Derivada parcial con respecto a sesgo.

$$\frac{\partial z^l}{\partial w^l} = a_i^{l-1}$$

Ecuación 2:17 Derivada parcial con respecto al peso.

$$\frac{\partial c}{\partial b^l} = \frac{\partial c}{\partial a^l} \cdot \frac{\partial a^l}{\partial z^l} \cdot \frac{\partial z^l}{\partial b^l}$$

Ecuación 2:18 Derivada del sesgo con respecto al costo.

$$\frac{\partial c}{\partial z^l} = \delta^l$$ Error imputado a la neurona

Ecuación 2:19 Error en función del valor de z

Al aplicar la regla de cadena y utilizar dos derivadas parciales en la derivada del sesgo con respecto al costo *Ecuación 2:18* se obtiene el error en función del valor de z representado en la *Ecuación 2:19*. Que es la suma ponderada calculada dentro de la neurona ,es decir, lo que nos cuenta esta derivada es en qué grado se modifica el error del coste cuando se produce un pequeño cambio en la suma de la neurona si esta derivada es grande, ante un pequeño cambio en el valor de la neurona éste se verá reflejado en el resultado final y por el contrario si la derivada es pequeña da igual como variamos el valor de la suma ya que éste no afectará al error de la red, es decir, la derivada de aquí es la que nos va a contar qué responsabilidad tiene la neurona en el resultado final y por tanto en el error, esto lo que dijimos antes, si la neurona es una parte responsable del error final entonces deberíamos utilizar esta información para sacarle parte de ese error por esta misma.

Necesitaremos la derivada del sesgo con respecto al costo *Ecuación 2:20* que será el error imputado a la neurona *Ecuación 2:21* la cual se calcula en la *Ecuación 2:22.*

$$\frac{\partial c}{\partial b^l} = \delta^l \cdot \frac{\partial z^l}{\partial b^l}$$

Ecuación 2:20 Derivación del sesgo con respecto al costo con error imputado a la neurona.

$$\frac{\partial c}{\partial b^l} = \delta^l \cdot 1$$

Ecuación 2:21 Derivación del sesgo con respecto al costo con error imputado a la neurona 2.

$$\frac{\partial c}{\partial b^l} = \delta^l$$

Ecuación 2:22 Error imputado a la neurona

Posteriormente haremos lo mismo, pero con la derivada parcial del peso con respecto al costo del error imputado a la neurona *Ecuación 2:23* que se reduce a la *Ecuación 2: 24.*

$$\frac{\partial c}{\partial w^l} = \delta^l \cdot \frac{\partial z^l}{\partial w^l}$$

Ecuación 2:23 Derivada del peso con respecto al costo con error imputado a la neurona 1

$$\frac{\partial c}{\partial w^l} = \delta^l \cdot \boldsymbol{a_i^{l-1}}$$

Ecuación 2:24 derivada del peso con respecto al costo con error imputado a la neurona 2

Hemos deducido tres expresiones diferentes que nos permiten obtener las derivadas parciales que estamos buscando para la última capa, una que nos dice cómo calcular el error de las neuronas en la última capa, y otra para cada una de las derivadas parciales y así obtenemos el resultado de la última capa.

Para obtener el resultado de la capa anterior volvemos a aplicar la Regla de Cadena a la composición siguiente *Ecuación 2:25* ésta genera dos derivadas; derivada del peso con respecto al costo en la penúltima capa w^{l-1} *Ecuación 2:26* y derivada del sesgo con respecto al costo en la penúltima capa b^{l-1} *Ecuación 2:27.*

$$c(a^l(w^l a^{l-1}(w^l a^{l-2} + b^{l-1}) + b^l))$$

Ecuación 2:25 Error en la penúltima capa

$$\frac{\partial c}{\partial w^{l-1}} = \frac{\partial c}{\partial a^l} \cdot \frac{\partial a^l}{\partial z^l} \cdot \frac{\partial z^l}{\partial a^{l-1}} \cdot \frac{\partial a^{l-1}}{\partial z^{l-1}} \cdot \frac{\partial z^{l-1}}{\partial w^{l-1}}$$

Ecuación 2:26 Derivada de peso con respecto al costo en la penúltima capa

$$\frac{\partial c}{\partial b^{l-1}} = \frac{\partial c}{\partial a^l} \cdot \frac{\partial a^l}{\partial z^l} \cdot \frac{\partial z^l}{\partial a^{l-1}} \cdot \frac{\partial a^{l-1}}{\partial z^{l-1}} \cdot \frac{\partial z^{l-1}}{\partial b^{l-1}}$$

Ecuación 2:27 Derivada de sesgo con respecto al costo en la penúltima capa

Calculado el error de la capa l y estas derivadas se operan igual que antes menos 1, y la activación de la capa previa y esta de aquí es la derivada de la función de la activación en esta expresión.

Lo único que haría falta calcular sería esta derivada que nos habla de cómo varía la suma ponderada de una capa cuando se varía el output de una neurona en la capa previa esta derivada también es simple de calcular y es básicamente la matriz de parámetros w *Ecuación 2:28*, que conecta ambas capas, esto lo que hace es mover el

error de una capa a la capa anterior distribuyendo el error en función de cuáles son las ponderaciones de las conexiones, con esto ya tendríamos nuevamente una expresión a partir de la cual obtener las derivadas parciales que estamos buscando.

Nuevamente el bloque remarcado en la *Ecuación 2:28*. Se convierte en esta derivada *Ecuación 2:29*. Que vuelve a representar al error de las neuronas en esta capa

δ^l w^l Derivada de la función de activación a^{l-2}

$$\frac{\partial c}{\partial w^{l-1}} = \frac{\partial c}{\partial a^l}\cdot\frac{\partial a^l}{\partial z^l}\cdot\frac{\partial z^l}{\partial a^{l-1}}\cdot\frac{\partial a^{l-1}}{\partial z^{l-1}}\cdot\frac{\partial z^{l-1}}{\partial w^{l-1}}$$

$$111\frac{\partial c}{\partial b^{l-1}} = \frac{\partial c}{\partial a^l}\cdot\frac{\partial a^l}{\partial z^l}\cdot\frac{\partial z^l}{\partial a^{l-1}}\cdot\frac{\partial a^{l-1}}{\partial z^{l-1}}\cdot\frac{\partial z^{l-1}}{\partial b^{l-1}}$$

Ecuación 2:28 Simplificación de derivadas para las capas ocultas y, de entrada

$$\frac{\partial c}{\partial z^{l-1}} = \delta^{l-1}$$

Ecuación 2:29 Error imputado a la neurona para las capas ocultas y, de entrada

La eficacia del algoritmo de back propagación radica, en que lo que hemos hecho en esta capa ya es extensible al resto de las capas de la red aplicando la misma lógica, tomamos el error de la capa anterior lo multiplicamos por la matriz de peso en una transformación que viene a representar la retro propagación de los errores *Ecuación 2:28*. Calculamos las derivadas parciales respecto a los parámetros y así sucesivamente recorriendo todas las capas de la red hasta el final, con un único paso, calculamos todos los errores y las derivadas parciales de nuestra red haciendo sólo uso de cuatro expresiones.

Con lo que al final obtendremos cuatro expresiones para calcular error empezando con la última capa *Ecuación 2:30*, y posteriormente realizar una retro propagación del error de la capa anterior *Ecuación 2:31*, y el cálculo de las derivadas de cada capa *Ecuación 2:32, Ecuación 2:33* de nuestra red.

$$\delta^l = \frac{\partial c}{\partial a^l} \cdot \frac{\partial a^l}{\partial z^l}$$

Ecuación 2:30 computo del Error en la última capa

$$\delta^{l-1} = w^l \delta^l \cdot \frac{\partial a^{l-1}}{\partial z^{l-1}}$$

Ecuación 2:31 Retro propagación del error a la capa anterior

$$\frac{\partial c}{\partial b^{l-1}} = \delta^{l-1}$$

Ecuación 2:32 Derivada sesgo de la capa usando el error

$$\frac{\partial c}{\partial w^{l-1}} = \delta^{l-1}\, a^{l-2}$$

Ecuación 2:33 Derivada de peso de la capa usando el error

Para calcular las derivadas parciales que estamos buscando en las expresiones *Ecuación 2:32* y *Ecuación* 2:33, es sencillo debido a que son bastante intuitivas, porque simplemente estamos contando cómo tenemos que utilizar el error de la capa anterior para calcular el error en esta capa, hay dos casos diferentes uno es en la última capa donde el error ya pertenece a la función de coste *Ecuación 2:30*, y otros son el resto de capas de nuestra red que dependen de otra capa *Ecuación 2:31*, y claro una vez tenemos estas dos expresiones podemos calcular el error en la capa actual con respecto a la anterior.

Capítulo 3 IMPLEMENTACIÓN

3.1 Implementación de las interfaces del software

En esta sección se describen las herramientas utilizadas para el desarrollo del sistema**.** También se describen las pantallas de la aplicación y las partes de programación más relevantes del mismo.

3.2 Descripción del equipo utilizado

Las pruebas experimentales realizadas para la sintonización y desarrollo de la Red Neuronal Artificial. Fueron realizadas en una computadora de las siguientes características:

- Marca: Acer
- Modelo: Aspire VX5-591G
- Procesador: Intel (R) core (TM) i7-7700HQ CPU 2.80GHz
- Tarjeta Gráfica: NVIDIA (R) GeForce (TM) GTX 1050Ti 4.0GB GDDR5
- Memoria RAM: 16.0GB
- Almacenamiento: SSD 128.0GB y HDD 1TB
- Sistema operativo: Windows 10 Home Single Language

Compilador y software Utilizado:

- Microsoft Office Professional Plus 2016
- Microsoft Visual C# 2008 Express Edition
- Brackets, Release 1.13 versión 1.13.1
- CanvasJS, versión 2.3.2

3.3 Interfaz de inicio al sistema

La figura 3.1 muestra la pantalla principal donde el usuario ingresa al sistema para definir los parámetros con los que trabajara la Red Neuronal Artificial.

Figura 3.1 Pantalla de inicio

3.3 Descripción del menú principal

A continuación, se explican campos de la interfaz:

Inputs:

a) NumImput: Representa el número de entradas a la red
b) NumHidden: Es el número de nodos (neuronas) que tendrá la capa oculta de la red.
c) NumOutput: Es el número de salidas que tendrá la red.
d) NumRows: es el número de líneas de información en la red.
e) NumCols: Numero de columnas que tendrá cada línea de información en la red.
f) File: se elige en archivo de la base de datos (artif es generado por el programa).
g) Clear Console: borra la información desplegada en la ventana de consola

Parámetros:

a) Seed: Determina la aleatoriedad en la creación de los datos artificiales (dejar en 0 para generar por completo aleatoriamente).

   Los siguientes parámetros se utilizan para optimizar nuestra Red Neuronal:

b) MaxEpochs: Determina el tiempo de entrenamiento de la red.

c) LearnRate: Tasa de aprendizaje de la red.

d) Momentum: Es el impulso, parámetro necesario para el aprendizaje.

e) Show perturb: Se selecciona para agregar perturbación en forma de ruido a los resultados de la red.

f) Amp noise: determina la amplitud del ruido en la perturbación.

Los parámetros “Renormalize” y” Compare with Py” no se utilizaron durante este proyecto, debido a que son utilizados para obtener valores de entrenamiento en una Red Neuronal Artificial con datos fijos, el inconveniente de utilizarlos seria, que los resultados tienden a ser erróneos con datos nuevos.

3.4 Presentación de la base de datos sintéticos "*artif*" de entrada al algoritmo

En esta sección se presenta la descripción del archivo generado artificialmente con la representación simbólica de los datos para que la Red Neuronal Artificial trabaje adecuadamente, como se muestra en la *Figura 3.2*.

3.5 Funcionamiento

El programa crea aleatoriamente una base de datos artificial, esta comienza generando 1,000 elementos de datos sintéticos de forma aleatoria.

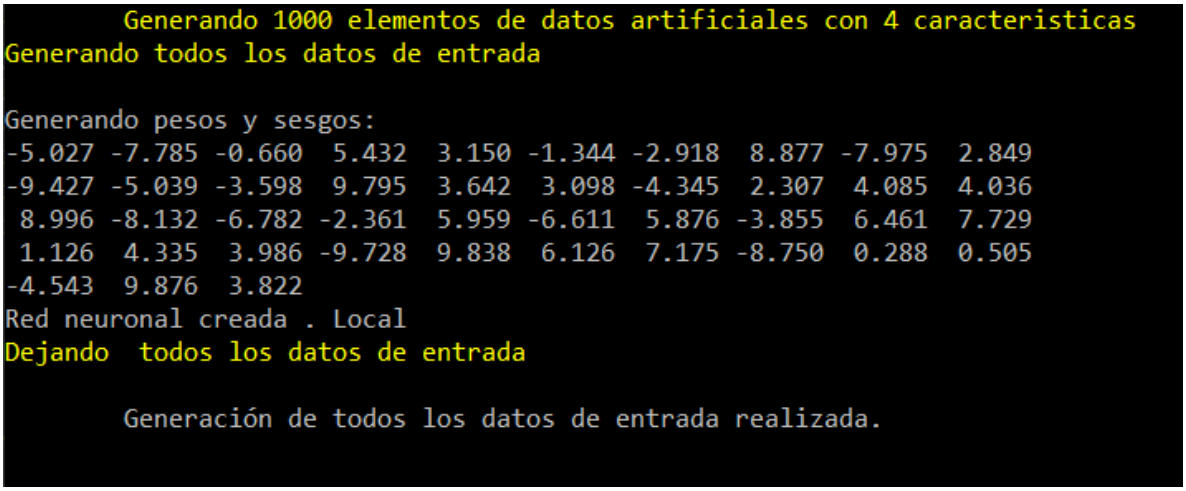

Figura 3.2 Generación de datos artificiales

Cada elemento de datos tiene cuatro valores de entrada y tres valores de salida como se puede observar en la *Figura 3.3*.

- Los cuatro valores de entrada que representan las características de la población de estudio, están todos entre -10.0 y +10.0. Corresponden a valores de predicción que se han normalizado, de modo que los valores por debajo de cero son menores que el promedio y los valores por encima de cero son mayores que el promedio.

- Los tres valores de salida corresponden a una variable para predecir que puede tomar uno de los tres valores categóricos. Para predecir la inclinación política de una persona: **conservadora, moderada o liberal.**

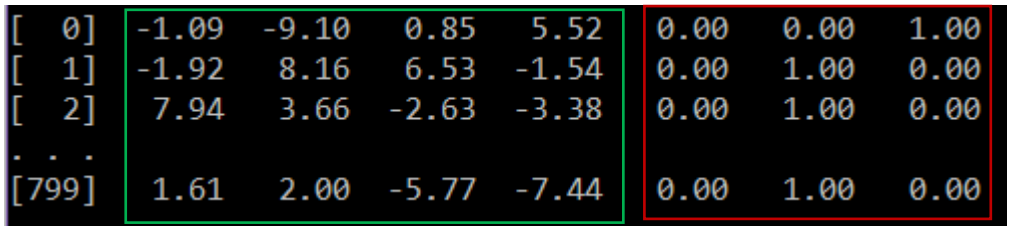

Figura 3.3 Valores de entrada y salida a la red neuronal artificial

Como se aprecia en la *Figura 3.4.* El programa divide los datos al azar, en un conjunto de entrenamiento de 800 elementos y un conjunto de prueba de 200 elementos. El conjunto de entrenamiento se utiliza para crear el modelo de red neuronal, y el conjunto de pruebas se utiliza para estimar la precisión del modelo.

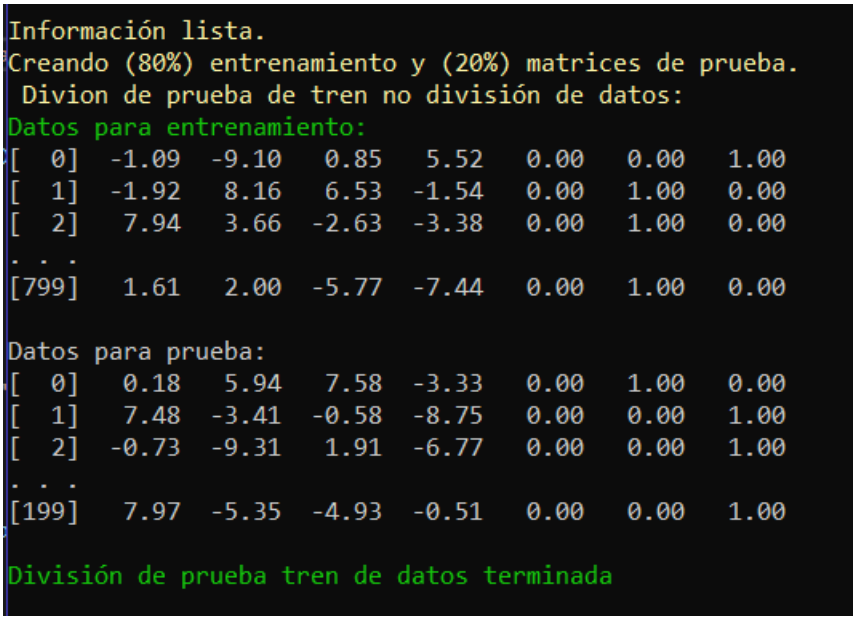

Figura 3.4 División de prueba tren de datos

Una vez divididos los datos, el programa crea una instancia de una red neuronal con n nodos ocultos. El número de nodos ocultos es arbitrario y debe determinarse por prueba y error. A continuación, en la *Figura 3.5*, se establecen los valores de los parámetros maxEpochs, learnRate y momentum, que se deben sintonizar

posteriormente para el funcionamiento óptimo de la Red Neuronal Artificial de propagación hacia atrás.

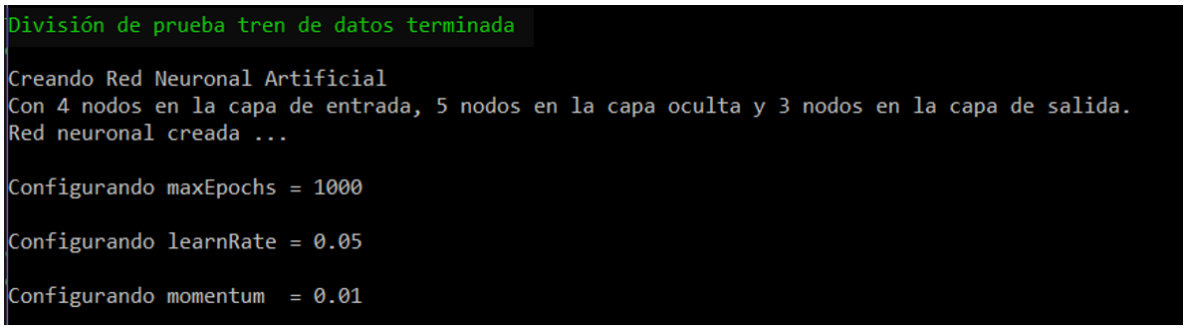

Figura 3.5 Optimización de los parámetros

Para terminar el programa genera una red neuronal final con los pesos y sesgo óptimos, con valores generados durante el entrenamiento previo de la red para obtener así el resultado final.

Capítulo 4 RESULTADOS

4.1 Pruebas de funcionalidad

En este capítulo se presentan los resultados de las pruebas experimentales, realizados a la Red Neuronal Artificial aplicada a votos electorales. Estos resultados son sintonizados con el objetivo de conocer si, la dependencia de los parámetros de la Red Neuronal Artificial encuentra el valor promedio normalizado. Y conocer si el número de las Capas ocultas afecta a los resultados de las votaciones.

4.1.2 Sintonización de parámetros

La red neuronal tiene cuatro entradas (una para cada característica) y tres salidas (porque la variable Y puede ser uno de los tres valores categóricos). La elección de las unidades de procesamiento ocultas para la red neuronal es la misma que la cantidad de unidades ocultas utilizadas para generar los datos sintéticos.

Para encontrar una cantidad óptima de unidades ocultas, así como los parámetros de optimización como: MaxEpoch, LearnRate y Momentum, requieren prueba y error por lo cual realizaremos una sintonización de parámetros de estos cuatro valores.

4.1.2.1 Numero de capas ocultas - NumHidden

El parámetro NumHiden indica el número de capas ocultas de la Red Neuronal Artificial. Se utilizaron varias capas ocultas 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 generadas de forma aleatoria como se observa en la *Figura 4.1*. fueron ejecutadas 30 veces con varios tamaños de capas ocultas y se registraron los resultados con mejor dependencia.

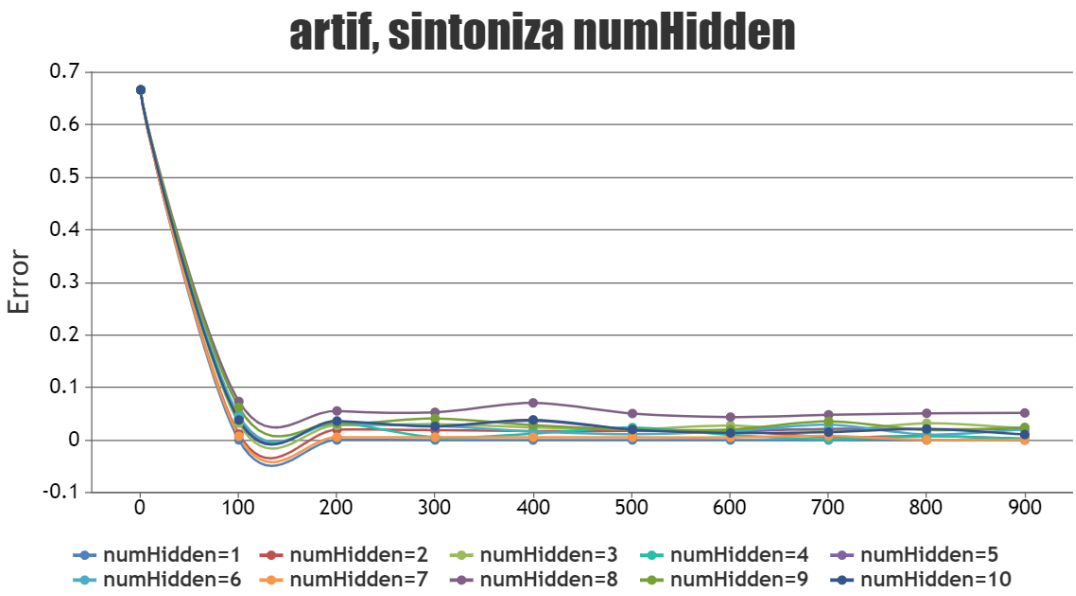

Figura 4.1 Sintonización de numero de capas ocultas

En la *Figura 4.1*. Se muestran los resultados de 30 ejecuciones para la Red Neuronal Artificial. Demostrando que un numero de 5 capas ocultas en la Red Neuronal Artificial está dentro del promedio normalizado.

4.1.2.2 Epocas Maximas - MaxEpoch

En la *figura 4.2*. Se presenta la sintonización de parámetros para el máximo de épocas para determinar el tiempo de entrenamiento de la Red Neuronal Artificial.

Con un Rango de 100,000, podemos observar un mejor comportamiento en el sistema. Demostrando que la Red Neuronal Artificial obtiene mejores resultados.

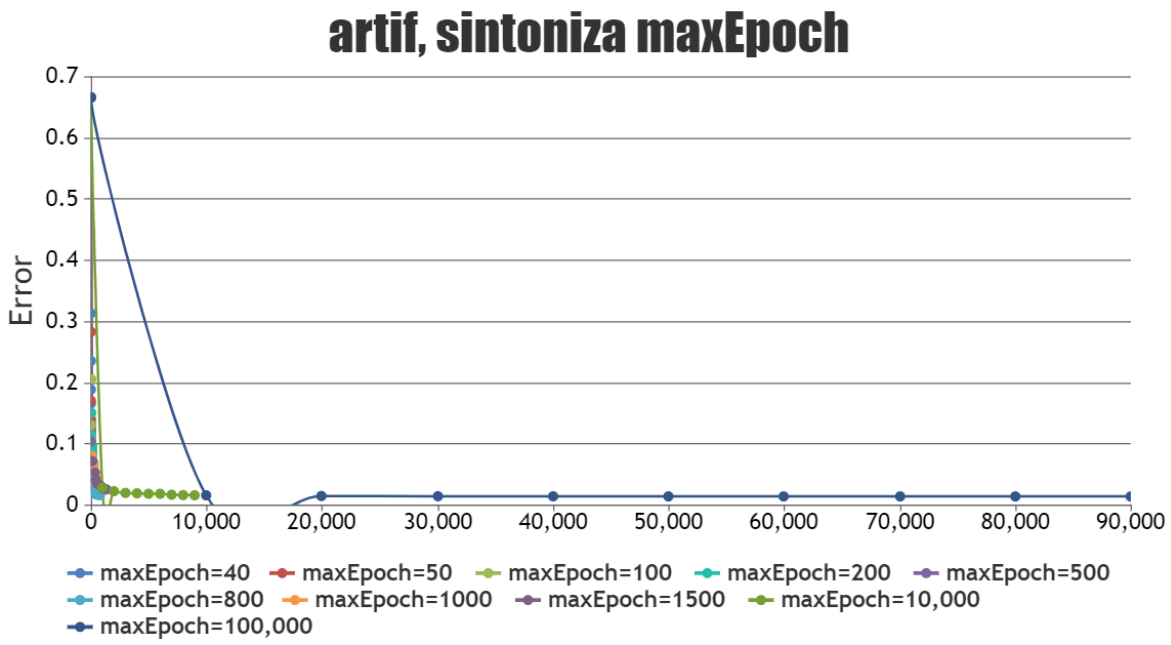

Figura 4.2 Sintonización de Épocas máximas

4.1.2.3 Tasa de aprendizaje LearnRate

En la *Figura 4.3*. Se muestran los resultados obtenidos de la mejor solución para obtener la mejor tasa de aprendizaje en la Red Neuronal Artificial. Se puede observar que, para este parámetro, la mejor solución encontrada de las 30 ejecuciones. Demostrando, que con un rango de aprendizaje del 1e-2 al 0.30. La Red Neuronal Artificial es más eficiente al tener un valor de 0.01 ya que el error tiene un valor menor al resto.

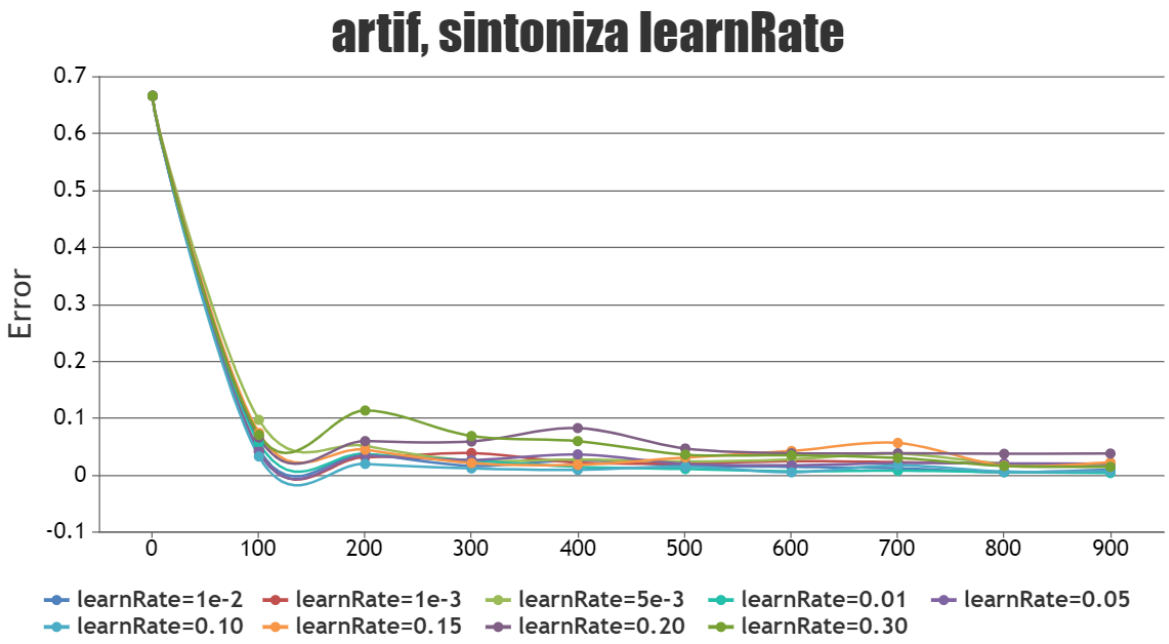

Figura 4.3 sintonización de tasa de aprendizaje

4.1.2.4 Impulso - Momentum

En la *Figura 4.4*. Se muestran los resultados obtenidos de 30 ejecuciones realizadas para la Red Neuronal Artificial. La función de coste muestra que el parámetro Momentum es necesario para el aprendizaje de la Red Neuronal Artificial. Sintonizando en un intervalo 1e-3, 5e-3, 0.01, 0.10, 0.70, 0.80, 0.90, 1.00. El aprendizaje de la Red Neuronal Artificial es de calidad cuando tiene un valor de 0.8, al tener un decrecimiento del error más cercano a cero antes que los otros valores en el rango de estudio.

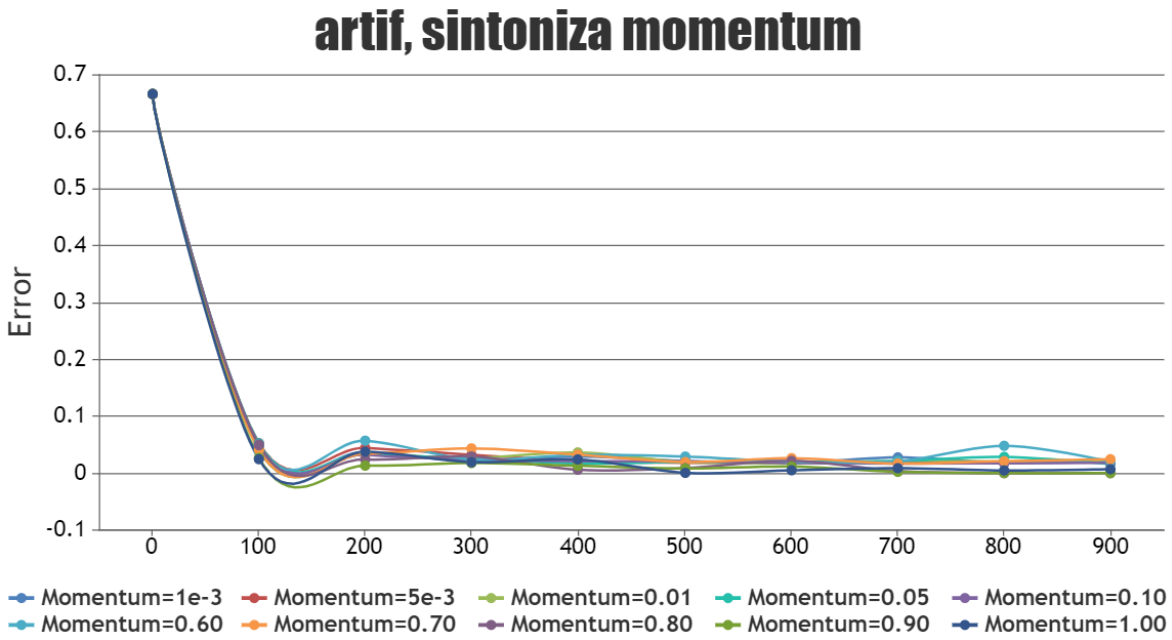

Figura 4.4 sintonización de Momentum

4.2 Resultados Experimentales

Podemos observar en la *Tabla 4-1* el número de capas ocultas (NumHidden) óptimo, se obtiene cuando hay una capa oculta, debido a que es el valor más pequeño con una media de 0. 0667240612695. También observamos el peor resultado al tener 8 capas ocultas en la red, con una media de 0.116324855868.

Tabla 4-1 Resultados en el parámetro de capas ocultas

Parametro	Media	Varianza σ2	DesvEst σ
1	0.066724061269486	0.044457338648618	0.21084908975051
2	0.078388657565341	0.042771050500891	0.20681163047781
3	0.090033319857606	0.041068010963467	0.20265243882931
4	0.080320990205377	0.042636895162207	0.20648703388399
5	0.090428669427933	0.041056037753971	0.20262289543379
6	0.087554960068494	0.041524859910155	0.20377649498938
7	0.070790801802586	0.043838701151205	0.20937693557602
8	0.11632485586751	0.037476119064442	0.19358749718007
9	0.094901702846709	0.040526770337977	0.2013126184271
10	0.088486127323551	0.041378803004582	0.20341780405014

Como se aprecia en la *Tabla 4-2.* de resultados de épocas máximas (MaxEpoch). El parámetro que muestra un mejor resultado es con un valor de 100,000 ya que tiene un promedio de 0.798494450216 y en contra parte el parámetro con el error más grande es el que tiene un valor de 40 épocas máximas con una media de 0.224202170011.

Tabla 4-2 Resultados en épocas máximas

Parametro	Media	Varianza σ2	DesvEst σ
40	0.224202170011	0.0277420584356	0.166559474169
50	0.209185017943	0.028611787768	0.169150192929
100	0.172781369022	0.0315228596581	0.177546781604
200	0.148263632496	0.0338120226826	0.183880457587
500	0.123442242176	0.0367977942228	0.191827511642
800	0.0897819548235	0.0411527016245	0.202861286658
1000	0.107724253636	0.0388386594634	0.197075263449
1500	0.101125245798	0.0396901730603	0.199223926927
10,000	0.084461978995	0.0418555028643	0.204586174666
100,000	0.0798494450216	0.0425078791082	0.20617439004

La *Tabla4-3* de resultados de tasa de aprendizaje LearnRate, demostró que la Red Neuronal Artificial es más eficiente cuando se sintoniza el parámetro con valor de 0.01 ya que obtenemos el error con valor más pequeño que tiene una media de 0.0836027360417. También podemos determinar que el parámetro con el peor resultado es el que tiene un valor de 0.20 ya que tiene una media de 0.113403211026.

Tabla 4-3 Tabla de resultados en el parámetro tasa de aprendizaje (LearnRate)

Parametro	Media	Varianza σ2	DesvEst σ
1e-2	0.0850195353349	0.041921161028	0.204746577573
1e-3	0.0905500691342	0.0410421799061	0.202588696393
5e-3	0.0998065627265	0.0402099146334	0.200524099882
0.01	0.0836027360417	0.0422532436613	0.205555938035
0.05	0.0904286694279	0.041056037754	0.202622895434
0.10	0.0791508165578	0.0426770631625	0.206584276174
0.15	0.0994677791877	0.0400582886811	0.200145668654
0.20	0.113403211026	0.0380145381562	0.194973172914
0.30	0.111204881944	0.0389908375477	0.197460977278

Para deducir el mejor parámetro de impulso se toma el que tenga un mejor valor de aprendizaje como podemos observar en la *Tabla 4-4.* El parámetro con mayor a aprendizaje es el que tiene un valor de 0.9 con una media de 0.076041633024302**,** y el que tiene el peor valor de aprendizaje es el parámetro con un valor de 0.6 con una media de 0.0967532830532.

Tabla 4-4 Tabla de resultados en el parámetro de Impulso (Momentum)

Parametro	Media	Varianza σ2	DesvEst σ
1e-3	0.0906561843099	0.041034982529	0.202570932093
5e-3	0.0913720339043	0.041003261404	0.202492620616
0.01	0.0904286694279	0.041056037754	0.202622895434
0.05	0.0902908411014	0.0411243211438	0.202791324133
0.10	0.0886126639154	0.041353053962	0.203354503176
0.60	0.0967532830532	0.0402667621109	0.200665797063
0.70	0.0927851492455	0.0407362663507	0.201832272818
0.80	0.0810643859019	0.0425758175751	0.206339083974
0.90	0.0760416330243	0.0431300117029	0.207677662985
1.00	0.0799201489704	0.0426337429667	0.206479400829

CONCLUSIONES

En conclusión, los parámetros con mejor y peor rendimiento en nuestra Red Neuronal Artificial son los expresados a continuación en la *Tabla 5-1.*

Parámetro	Mejor	peor	Promedio
Momentum	**0.9**	**0.6**	**0.44587224101**
MaxEpoch	**100,000**	**40**	**11419**
LearnRate	**0.01**	**0.20**	**0.1382286326**
NumHidden	**1**	**8**	**5.5**

Tabla 0-1 Parámetros sintonizados con mejor rendimiento

Los cuales son demostrados con la gráfica en la Figura 5.1 realizada con los valores obtenidos de esta sintonización de resultados óptimos en nuestra red neuronal artificial entrenada con el algoritmo de retro propagación.

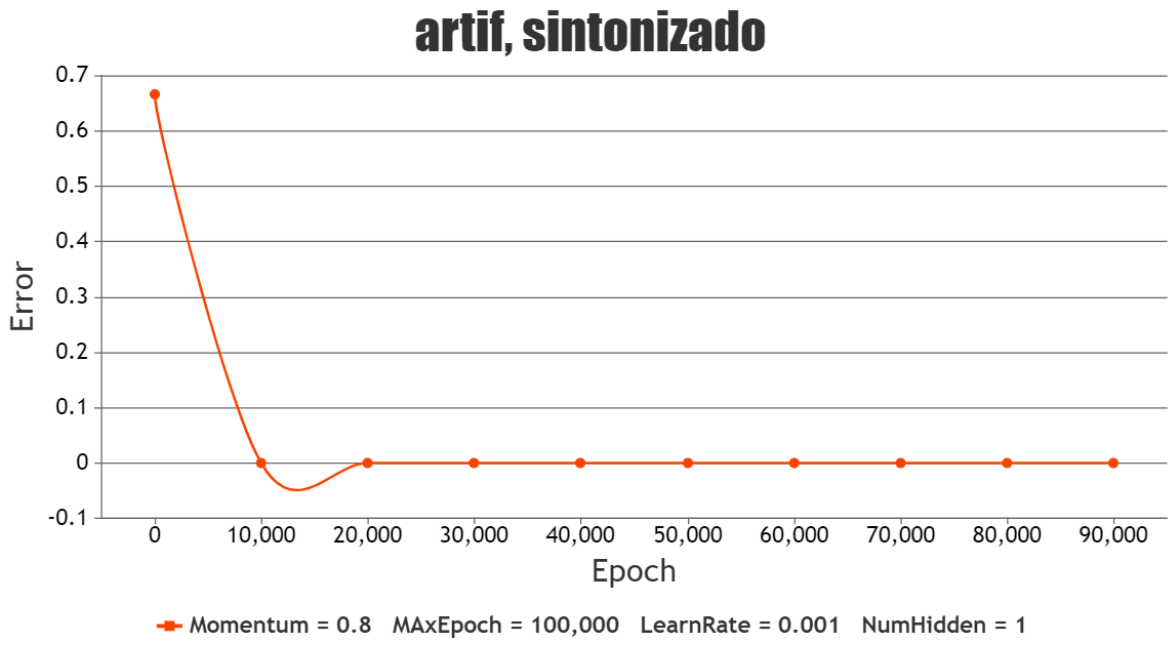

Figura 0.1 Resultados de la sintonización de parámetros de la red.

Solo nos resta tomar el enfoque basado en votaciones electorales, con datos experimentales, y la perturbación como un evento determinante en la alteración, o no, del resultado final. Tenemos tres posibles resultados los cuales en escala de 0 a 10. Si se representa a una persona que es más joven que el promedio, tiene un ingreso mucho menor que el promedio, es algo más educado que el promedio y tiene mayor deuda que el promedio. La persona tiene una visión política liberal. De igual manera si otra persona, en la misma escala es de edad mayor que el promedio, tiene un ingreso mayor al promedio, es un poco menor, igual o un poco mayor educado al promedio y tiene una deuda menor al promedio, dicha persona tiene una visión política conservadora. Pero cuando una persona se encuentra dentro del

promedio o un poco por debajo o un poco por arriba en todos los parámetros se encuentra en una ideología moderada.

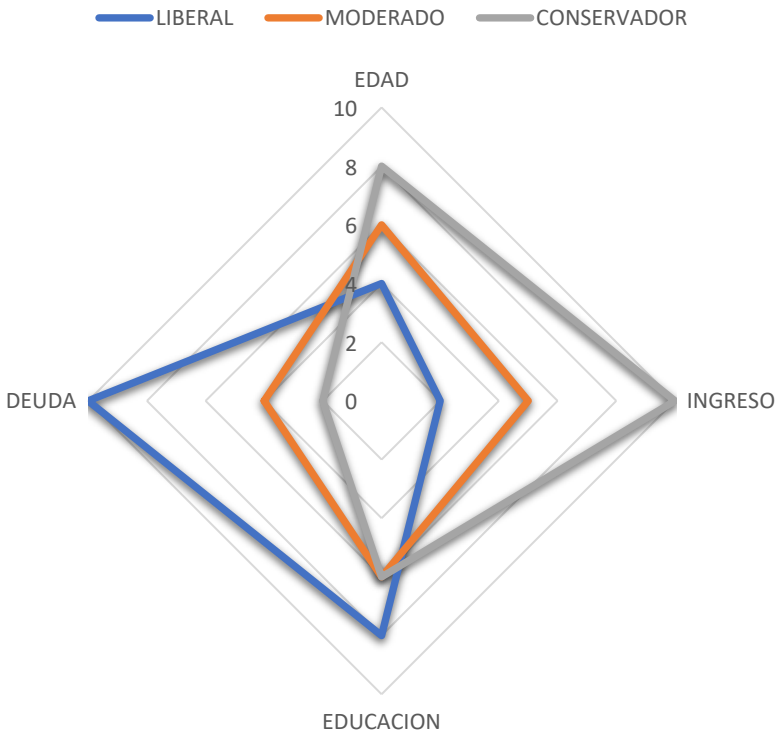

Figura 0.2 Enfoque para la predicción de votaciones Electorales

Trabajos futuros

Las redes neuronales artificiales tienen una extensa gama de aplicación en proyectos de investigación, en los cuales se destacan: La optimización de recursos, toma de decisiones, interpretación de datos y predicción de resultados.

Con lo cual la Red Neuronal Artificial desarrollada en este proyecto, podría ser utilizada en otro proyecto con un enfoque diferente o de otra área de estudio, e incluso implantarla en un proyecto ya implementado con el fin de optimizar procesos o añadir nodos.

Se espera la continuidad de esta línea de desarrollo experimental en el tema electoral para la obtención de resultados más precisos al proceso electoral con datos reales, en la cual nuestra Red Neuronal Artificial podría utilizarse como base por otros investigadores para la comprobación de datos estructurados y seguridad en el manejo de la información más compleja.

Bibliografía

Abitia, E. A. (6 de agosto de 2018). *Tan solo en México en el marco de las elecciones a efectuadas el pasado 1 julio del 2018 se estima que durante el mes.* Obtenido de Este Pais: https://anterior.estepais.com/articulo.php?id=1662&t=la-ciencia-del-error-acierta-evaluacion-de-las-encuestas-electorales-2018

Alanís, M. d. (2012). *Las encuestas en la elección presidencial.* Obtenido de Este País: https://archivo.estepais.com/site/2012/las-encuestas-en-la-eleccion-presidencial/

Albán, H. M. (2016). APPLICATION OF KOHONEN MAPS FOR THE PRIORITIZATION. *EIA*, 157-169.

Aranha, D. F., Barbosa, P. Y., Cardoso, T. N., Araújo, C. L., & Matias, P. (2019). The return of software vulnerabilities in the Brazilian voting machine. *Computers & Security*, 335-349.

Ballesteros, A. (2001). *Neural Network Framework*. Obtenido de http://www.redes-neuronales.com.es/tutorial-redes-neuronales/tutorial-redes.htm

Bobbio, N. (2001). *Derecha e Izquierda.* Madrid: Taurus.

C. Aldrich, J. S. (1995). *Comparison of different artificial neural nets for the detection and location of gross errors in process systems.*

Callejas, I., Piñeros, J., Rocha, J., Hernández, F., & Delgado, F. (2013). Implementación de una red neuronal artificial tipo SOM en una FPGA para la resolución de trayectorias tipo laberinto. *2013 II International Congress of Engineering Mechatronics and Automation (CIIMA).* Bogota, Colombia: IEEE.

Cires, D., Meier, U., Masci, J., Gambardella, L., & Schmidhuber, J. (2011). Flexible, High Performance Convolutional Neural Networks for Image Classification. *Manno-Lugano*.

D'Adamo, O., Beaudoux, V. G., & Kievsky, T. (2015). Comunicación política y redes sociales: análisis de las campañas para las elecciones legislativas de 2013 en la ciudad de Buenos Aires. *Revista Mexicana de Opinión Pública, 19*, 107-126. Obtenido de http://www.sciencedirect.com/science/article/pii/S187073001500006X

Delgado, J. R., Rasúa, R. T., & Bello, R. (2019). A continuation approach for training Artificial Neural Networks with meta-heuristics. *Pattern Recognition Letters, 125*, 373-380. Obtenido de http://www.sciencedirect.com/science/article/pii/S0167865519301667

Dimitrios, X. (2017). Multidimensional electoral competition between differentiated candidates. *Games and Economic Behavior, 105*, 112-121. Obtenido de https://www.sciencedirect.com/science/article/pii/S0899825617301203

Fukushima, K. (1980). *Neocognitron: A Self-organizing Neural Network Model.* Tokyo, Japan: Springer-Verla.

Hilera, J., & Martinez, H. (1995). *Redes neuronales artificiales : fundamentos, modelos y aplicaciones.* Madrid: RA-MA Editorial.

John F. Kolen, S. C. (2001). *A Field Guide to Dynamical Recurrent Networks.* New York: IEEE PRESS.

Johnson, J. W., & Hoyo, V. (2012). Beyond personal vote incentives: Dividing the vote in preferential electoral systems. *Electoral Studies, 31*, 131-142. Obtenido de https://www.sciencedirect.com/science/article/pii/S0261379411001089

JudeHemanth, J. H. (2018). Brain signal based human emotion analysis by circular back propagation and Deep Kohonen Neural Networks. *Computers & Electrical Engineering*, 170-180.

Kohonen, T. (1989). *Self-organization and associative memory.* New York: Springer Verlag.

Kohonen, T. (1995). *Self-Organizing Maps.* Heidelberg: Springer-Verlag Berlin Heidelberg.

Martínez LAE, G. C. (2001). Definición de una red neuronal para clasificación por medio de un programa. *Revista Mexicana de Ingeniería Biomédica*, 4-11.

MICHELONE, M. L. (14 de julio de 2019). *Proceso*. Obtenido de https://www.proceso.com.mx/592365/que-son-y-como-funcionan-las-redes-neuronales

N.L., G. (21 de Octubre de 2020). *https://www.nl.gob.mx*. Obtenido de https://www.nl.gob.mx: https://www.nl.gob.mx/boletines-comunicados-y-avisos/presenta-icv-nueva-licencia-para-conducir-digital-del-nuevo-leon

Péronnet, M. (1985). *Vocabulario básico de la Revolución Francesa.* Barcelona: Critica.

Ramírez, F. (20 de julio de 2018). *LUCA Telefonica Data Unit*. Obtenido de https://empresas.blogthinkbig.com/historia-de-la-ia-frank-rosenblatt-y-e/

RF López, J. F. (2008). *Las redes neuronales artificiales.* Oleiros, España: Gesbiblo s.l.

Ridella, S., Rovetta, S., & Zunino, R. (1997). Circular backpropagation networks for classification. *IEEE Transactions on Neural Networks*, 84 - 97.

Rojas, R. (1996). *Neural Networks: A Systematic Introduction.* Springer.

Thanh, T. D., Serkan, K., Moncef, G., & Alexandros, I. (2019). A Python library for Generalized Operational Perceptron algorithms. *Knowledge-Based Systems*, 855-863. Obtenido de https://reader.elsevier.com/reader/sd/pii/S0360132319304925?token=4728E3EE843E532B4B7332A07F58A34CFDB178A8C6A76843F890DFCAF41CD3304D45519B9044F3FC5AAA0DF5BBB363DA

Tijskens, A., Roels, S., & Janssen, H. (2019). Neural networks for metamodelling the hygrothermal behaviour of building components. *Building and*

Environment, 106,282. Obtenido de http://www.sciencedirect.com/science/article/pii/S0360132319304925

Toda Caraballo, I. (2010). Diseño de redes neuronales con aprendizaje combinado de retropropagación y búsqueda aleatoria progresiva aplicado a la determinación de austenita retenida en aceros. *Revista de Metalurgia*.

Turing, A. M. (1950). Computing Machinery and Intelligence. *Mind, 59*(236), 436-460.

Valdez, A. H. (11 de 03 de 2013). *http://www.scielo.org.mx/*. Obtenido de http://www.scielo.org.mx/scielo.php?script=sci_arttext&pid=S1870-35692013000200003

Wang, C.-H. (2014). Gender differences in the effects of personality traits on voter turnout. *Electoral Studies*, 167-176. Obtenido de http://www.sciencedirect.com/science/article/pii/S0261379413001613

Xefteris, D. (2016). Stability in electoral competition: A case for multiple votes. *Journal of Economic Theory, 161*, 76-102. Obtenido de https://www.sciencedirect.com/science/article/pii/S0022053115001908

Zhengbing Hu, Y. V. (2016). A cascade deep neuro-fuzzy system for high-dimensional online possibilistic fuzzy clustering. *2016 XIth International Scientific and Technical Conference Computer Sciences and Information Technologies (CSIT)* (págs. 801-805). Lviv, Ukraine: IEEE.

Anexo A. Glosario

Iteración: Repetir varias veces un proceso con la intención de alcanzar una meta deseada, objetivo o resultado.

Algoritmo: Conjunto ordenado de operaciones sistemáticas que permite hacer un cálculo y hallar la solución de un tipo de problemas.

Peyorativo: Palabra o expresión que transmite una connotación negativa de desprecio o poco respeto.

Avasallada: Sujetar o someter a alguien a obediencia.

Perturbación: Alteración o trastorno que se produce en el orden o en las características permanentes que conforman una cosa o en el desarrollo normal de un proceso.

Demográfica: Estudia las poblaciones humanas; su dimensión, estructura, evolución y características generales.

Machine Learning: Es una disciplina científica del ámbito de la Inteligencia Artificial que crea sistemas que aprenden automáticamente. Aprender en este contexto quiere decir identificar patrones complejos en millones de datos.

Perceptrón: La neurona artificial o unidad básica de inferencia en forma de discriminador lineal, a partir de lo cual se desarrolla un algoritmo capaz de generar un criterio para seleccionar un subgrupo a partir de un grupo de componentes más grande.

Cohorte: Conjunto de cosas o personas que acompañan o siguen a otra cosa o persona.

Morfología: Parte de la lingüística que estudia las reglas que rigen la flexión, la composición y la derivación de las palabras.

Data Warehouse: Es una colección de datos orientada a un determinado ámbito, integrado, no volátil y variable en el tiempo, que ayuda a la toma de decisiones en la entidad en la que se utiliza.

Software: Soporte lógico de un sistema informático

Monarquía: Forma de gobierno en la que la jefatura del Estado reside en una persona, un rey o una reina, cargo habitualmente vitalicio al que se accede por derecho y de forma hereditaria.

Revocación: Modo de extinguir una relación jurídica o una causa de ineficacia del acto jurídico

Aristocracia: Clase social formada por las personas que poseen títulos nobiliarios concedidos por el rey o heredados de sus antepasados.

Burguesía: Clase social formada por las personas acomodadas que logran tener propiedades y capital.

Esquemática: Que está explicado o hecho de manera simple, únicamente con sus rasgos esenciales.

Espectro Político: Ordenamiento visual de grupos u organizaciones políticas de acuerdo con ciertos ejes conceptuales.

Geometrías políticas: Posición de ideas políticas e ideologías dentro de simples mapas o dimensiones.

Paralelo: Es una relación que se establece entre cualquier variedad lineal de dimensión mayor o igual que 1.

Soma: Es el cuerpo de una neurona.

Dendritas: son prolongaciones protoplásmicas ramificadas, bastante cortas de la neurona, dedicadas principalmente a la recepción de estímulos y, secundariamente, a la alimentación celular.

Axón: Prolongación que arranca del cuerpo de la neurona y termina en una ramificación que está en contacto con células musculares, glandulares, etc., o con otras células nerviosas, y por la cual circulan los impulsos nerviosos.

Input: Conjunto de datos que se introducen en un sistema o un programa informáticos.

Output: Información que proporciona una computadora después de procesar un conjunto de datos determinados.

Retroalimentación: es un mecanismo por el cual una cierta proporción de la salida de un sistema se redirige a la entrada, con señales de controlar su comportamiento.

Vector: Un vector se representa mediante un segmento de recta, orientado dentro del espacio euclidiano tridimensional.

Anexo B. Acrónimos

LearnRate: Término en inglés que hace referencia a la tasa de aprendizaje o curva de aprendizaje.

NumHidden: Término en inglés que hace referencia al número de capas ocultas en una red neuronal.

MaxEpoch: Término en inglés que hace referencia al número de épocas de ejecución en una red neuronal.

Momentum: Término en inglés que hace referencia a una magnitud física derivada de tipo vectorial que describe el movimiento de un cuerpo en cualquier teoría mecánica.

ANN: Artificial Neural network - Redes Neuronales Artificiales

RNNLM: Recurrent Neural Network Language - Modelado de Lenguaje de Red Neuronal Recurrente.

PeMS: Performance Measurement System - Sistema de medición de rendimiento

Anexo C. Código

Código para la sintonización de parámetros en html Javascript y php con uso de CanvaJS.

```
<!DOCTYPE HTML>

<html>

<head>

<meta charset="UTF-8">


      <?php

   $archivo = file( "hiddenLayers/ARTHiddenLayer10.dat" );

   $lineas = count( $archivo );

         for( $i = 0; $i < $lineas; $i++ ){

           for( $j = 0; $j < 3;  $j++ )

                 $ar10[$i] = explode(" ", $archivo[$i]);

         }
```

```
$archivo = file( "hiddenLayers/ARTHiddenLayer9.dat" );

$lineas = count( $archivo );

                    for( $i = 0; $i < $lineas; $i++ ){

        for( $j = 0; $j < 3;  $j++ )

            $ar9[$i] = explode(" ", $archivo[$i]);

            }



$archivo = file( "hiddenLayers/ARTHiddenLayer8.dat" );

$lineas = count( $archivo );

                    for( $i = 0; $i < $lineas; $i++ ){

        for( $j = 0; $j < 3;  $j++ )

            $ar8[$i] = explode(" ", $archivo[$i]);

                    }



$archivo = file( "hiddenLayers/ARTHiddenLayer7.dat" );

$lineas = count( $archivo );

                    for( $i = 0; $i < $lineas; $i++ ){

        for( $j = 0; $j < 3;  $j++ )
```

```
                        $ar7[$i] = explode(" ", $archivo[$i]);
        }


$archivo = file( "hiddenLayers/ARTHiddenLayer6.dat" );
$lineas = count( $archivo );
                    for( $i = 0; $i < $lineas; $i++ ){
          for( $j = 0; $j < 3;  $j++ )
                $ar6[$i] = explode(" ", $archivo[$i]);
        }


$archivo = file( "hiddenLayers/ARTHiddenLayer5.dat" );
$lineas = count( $archivo );
        for( $i = 0; $i < $lineas; $i++ ){
            for( $j = 0; $j < 3;  $j++ )
                                $ar5[$i] = explode(" ", $archivo[$i]);
        }
      $archivo = file( "hiddenLayers/ARTHiddenLayer4.dat" );
$lineas = count( $archivo );
```

```
        for( $i = 0; $i < $lineas; $i++ ){

            for( $j = 0; $j < 3;  $j++ )

                                        $ar4[$i] = explode(" ", $archivo[$i]);

        }



       $archivo = file( "hiddenLayers/ARTHiddenLayer3.dat" );

$lineas = count( $archivo );

        for( $i = 0; $i < $lineas; $i++ ){

            for( $j = 0; $j < 3;  $j++ )

                                        $ar3[$i] = explode(" ", $archivo[$i]);

        }



        $archivo = file( "hiddenLayers/ARTHiddenLayer2.dat" );

$lineas = count( $archivo );

        for( $i = 0; $i < $lineas; $i++ ){

            for( $j = 0; $j < 3;  $j++ )

                                        $ar2[$i] = explode(" ", $archivo[$i]);

        }
```

```
        $archivo = file( "hiddenLayers/ARTHiddenLayer1.dat" );
   $lineas = count( $archivo );
        for( $i = 0; $i < $lineas; $i++ ){
            for( $j = 0; $j < 3;  $j++ )
                                        $ar1[$i] = explode(" ", $archivo[$i]);
        } //var_dump($ar1);
 ?>
```

```
   <script> //grafica
window.onload = function () {

var chart = new CanvasJS.Chart("chartContainer", {
        animationEnabled: true,
        exportEnabled: true,
```

```
title:{
        text: "artif, sintoniza numHidden"
},
axisY:{
        title: "Error"
},
toolTip: {
        shared: true
},
legend:{
        cursor:"pointer",
        itemclick: toggleDataSeries
},
data: [{
name: "numHidden=1",
        type: "spline",
        yValueFormatString: "#0.### error",
        showInLegend: true,
```

```
                dataPoints: [
        <?php
        for( $i = 0; $i < 10; $i++ ){
            if($i<9){
                                    echo '{ x: '.$ar1[$i][0].' , y: '.$ar1[$i][2].' },';
                            }else{
                        echo '{ x: '.$ar1[$i][0].' , y: '.$ar1[$i][2].'}';}
                                                $art1[$i]=$ar1[$i][2];
                        }
                                $mean1 = array_sum($art1)/count($art1) ;
                            $sd1=sd($art1);
                                $var1=pow($sd1, 2);
                        ?>
                ]
},
{
            name: "numHidden=2",
                        type: "spline",
```

```
            yValueFormatString: "#0.### error",
            showInLegend: true,
             dataPoints: [
        <?php
        for( $i = 0; $i < 10; $i++ ){
           if($i<9){
                                echo '{ x: '.$ar2[$i][0].' , y: '.$ar2[$i][2].' },';
                        }else{
            echo '{ x: '.$ar2[$i][0].' , y: '.$ar2[$i][2].'}';}
                                            $art2[$i]=$ar2[$i][2];
            }
                         $mean2 = array_sum($art2)/count($art2) ;
                        $sd2=sd($art2);
                         $var2=pow($sd2, 2);
            ?>
         ]
},
{
```

```
  name: "numHidden=3",
            type: "spline",
            yValueFormatString: "#0.### error",
            showInLegend: true,
             dataPoints: [
<?php
for( $i = 0; $i < 10; $i++ ){
   if($i<9){
                        echo '{ x: '.$ar3[$i][0].' , y: '.$ar3[$i][2].' },';
                }else{
            echo '{ x: '.$ar3[$i][0].' , y: '.$ar3[$i][2].'}';}
                                $art3[$i]=$ar3[$i][2];
            }
                 $mean3 = array_sum($art3)/count($art3) ;
                $sd3=sd($art3);
                  $var3=pow($sd3, 2);
            ?>
  ]
```

```
},
{
        name: "numHidden=4",
                type: "spline",
                yValueFormatString: "#0.### error",
                showInLegend: true,
                 dataPoints: [
      <?php
      for( $i = 0; $i < 10; $i++ ){
         if($i<9){
                                echo '{ x: '.$ar4[$i][0].' , y: '.$ar4[$i][2].' },';
                        }else{
                echo '{ x: '.$ar4[$i][0].' , y: '.$ar4[$i][2].'}';}
                                $art4[$i]=$ar4[$i][2];
                }
                         $mean4 = array_sum($art4)/count($art4) ;
                        $sd4=sd($art4);
                          $var4=pow($sd4, 2);
```

```
                ?>
            ]
},
{
        name: "numHidden=5",
                type: "spline",
                yValueFormatString: "#0.### error",
                showInLegend: true,
                 dataPoints: [
        <?php
        for( $i = 0; $i < 10; $i++ ){
            if($i<9){
                                echo '{ x: '.$ar5[$i][0].' , y: '.$ar5[$i][2].' },';
                        }else{
                echo '{ x: '.$ar5[$i][0].' , y: '.$ar5[$i][2].'}';}
                                                $art5[$i]=$ar5[$i][2];
                }
                        $mean5 = array_sum($art5)/count($art5) ;
```

```
                        $sd5=sd($art5);
                          $var5=pow($sd5, 2);
                    ?>
            ]
},
{
          name: "numHidden=6",
                    type: "spline",
                    yValueFormatString: "#0.### error",
                    showInLegend: true,
                     dataPoints: [
        <?php
        for( $i = 0; $i < 10; $i++ ){
            if($i<9){
                                echo '{ x:  '.$ar6[$i][0].' , y: '.$ar6[$i][2].' },';
                        }else{
                                echo '{ x: '.$ar6[$i][0].' , y: '.$ar6[$i][2].'}';}
                                        $art6[$i]=$ar6[$i][2];
```

```
                }
                        $mean6 = array_sum($art6)/count($art6) ;
                       $sd6=sd($art6);
                        $var6=pow($sd6, 2);
                ?>
        ]
},
{
        name: "numHidden=7",
                type: "spline",
                yValueFormatString: "#0.### error",
                showInLegend: true,
                 dataPoints: [
      <?php
      for( $i = 0; $i < 10; $i++ ){
         if($i<9){
                                echo '{ x:  '.$ar7[$i][0].' , y: '.$ar7[$i][2].' },';}
        else{
```

```
                echo '{ x:  '.$ar7[$i][0].' , y: '.$ar7[$i][2].' }'; }
                $art7[$i]=$ar7[$i][2];
                }
                        $mean7 = array_sum($art7)/count($art7) ;
                       $sd7=sd($art7);
                        $var7=pow($sd7, 2);
                ?>
        ]
},
{
        name: "numHidden=8",
                type: "spline",
                yValueFormatString: "#0.### error",
                showInLegend: true,
                 dataPoints: [
      <?php
      for( $i = 0; $i < 10; $i++ ){
         if($i<9){
```

```
                    echo '{ x:  '.$ar8[$i][0].' , y: '.$ar8[$i][2].' },';}
                else{
            echo '{ x:  '.$ar8[$i][0].' , y: '.$ar8[$i][2].'}';}

                $art8[$i]=$ar8[$i][2];
            }
                 $mean8 = array_sum($art8)/count($art8) ;
                $sd8=sd($art8);
                  $var8=pow($sd8, 2);
            ?>
        ]
},
{
        name: "numHidden=9",
            type: "spline",
            yValueFormatString: "#0.### error",
            showInLegend: true,
             dataPoints: [
```

```
<?php
for( $i = 0; $i < 10; $i++ ){
   if($i<9){
                        echo ' { x:  '.$ar9[$i][0].' , y: '.$ar9[$i][2].' },';}
                else{
            echo '{ x:  '.$ar9[$i][0].' , y: '.$ar9[$i][2].' }';}
                        $art9[$i]=$ar9[$i][2];
            }
                 $mean9 = array_sum($art9)/count($art9) ;
                $sd9=sd($art9);
                  $var9=pow($sd9, 2);
            ?>
        ]
},
{
      name: "numHidden=10",
            type: "spline",
            yValueFormatString: "#0.### error",
```

```
                showInLegend: true,
                 dataPoints: [
        <?php
        for( $i = 0; $i < 10; $i++ ){
           if($i<9){
                                echo '{ x:  '.$ar10[$i][0].' , y: '.$ar10[$i][2].' },';}
          else{
                            echo '{ x:  '.$ar10[$i][0].' , y: '.$ar10[$i][2].' }'; }
                                $art10[$i]=$ar10[$i][2];
                }
                             $mean10 = array_sum($art10)/count($art10) ;
                            $sd10=sd($art10);
                              $var10=pow($sd10, 2);
                ?>
         ]
}
         ]
});
```

```
chart.render();


function toggleDataSeries(e) {
	if(typeof(e.dataSeries.visible) === "undefined" || e.dataSeries.visible) {
		e.dataSeries.visible = false;
	}
	else {
		e.dataSeries.visible = true;
	}
	chart.render();
}
	}
</script>


 <title>Neural Network CIICAP</title>


  <!-- Bootstrap core CSS -->
```

```
  <link href="vendor/bootstrap/css/bootstrap.min.css" rel="stylesheet">


  <!-- Custom styles for this template -->
  <link href="css/modern-business.css" rel="stylesheet">
        <link rel="shortcut icon" href="gosoft.png" />
</head>
<body>
        <!-- Navigation -->
  <nav class="navbar fixed-top navbar-expand-lg navbar-light bg-light fixed-top">
    <div class="container">
      <a class="navbar-brand" href="index.html"><img src="logo-uaem.png"
style="width: 80px; ma" ><img src="logoCiicap.png" style="width: 90px; margin-left:
10%;" ></a>
Sintonizacion de parametros
    </div>
  </nav>
<div id="chartContainer" style="height: 500px; max-width: 920px; margin: 0px auto;
margin-top: 30px;"></div>
<script src="canvasjs.min.js"></script>
```

```
 <div class="container">

<br><br>

    <!-- Marketing Icons Section -->

    <div class="row">

            <a href="epochs.php" class="btn btn-primary" style="margin:
auto;">MaxEpochs</a>

            <a href="learnRate.php" class="btn btn-primary"
style="margin:auto;">LearnRate</a>

          <a href="momentum.php" class="btn btn-primary"
style="margin:auto;">Momentum</a>

            <a href="hiddenlayers.php" class="btn btn-primary" style="margin:
auto;">numHidden</a>

            </div>

    <!-- /.row -->

  <br/><br/>

  </div>

  <!-- /.container -->

                  <div class="container">

  <h2 style="margin-left:30%;">Tabla de Resultados NumHidden</h2>
```

```
<p></p>

<table class="table">

  <thead>

    <tr>

      <th>Parametro</th>

      <th>Media</th>

      <th>Varianza &sigma;2</th>

                <th>DesvEst &sigma;</th>

    </tr>

  </thead>

  <tbody>

     <tr>

      <td>1</td>

      <td><?php echo $mean1; ?></td>

      <td><?php echo $var1; ?></td>

              <td><?php echo $sd1; ?></td>

    </tr>

                 <tr>
```

```
  <td>2</td>
  <td><?php echo $mean2; ?></td>
  <td><?php echo $var2; ?></td>
            <td><?php echo $sd2; ?></td>
</tr>
            <tr>
  <td>3</td>
  <td><?php echo $mean3; ?></td>
  <td><?php echo $var3; ?></td>
            <td><?php echo $sd3; ?></td>
</tr>
            <tr>
  <td>4</td>
  <td><?php echo $mean4; ?></td>
  <td><?php echo $var4; ?></td>
            <td><?php echo $sd4; ?></td>
</tr>
                  <tr>
```

```
  <td>5</td>
  <td><?php echo $mean5; ?></td>
  <td><?php echo $var5; ?></td>
            <td><?php echo $sd5; ?></td>
</tr>
                    <tr>
  <td>6</td>
  <td><?php echo $mean6; ?></td>
  <td><?php echo $var6; ?></td>
            <td><?php echo $sd6; ?></td>
</tr>
                    <tr>
  <td>7</td>
  <td><?php echo $mean7; ?></td>
  <td><?php echo $var7; ?></td>
            <td><?php echo $sd7; ?></td>
</tr>
                    <tr>
```

```
  <td>8</td>
  <td><?php echo $mean8; ?></td>
  <td><?php echo $var8; ?></td>
        <td><?php echo $sd8; ?></td>
 </tr>
    <tr>
  <td>8</td>
  <td><?php echo $mean9; ?></td>
  <td><?php echo $var9; ?></td>
        <td><?php echo $sd9; ?></td>
 </tr>
              <tr>
  <td>10</td>
  <td><?php echo $mean10; ?></td>
  <td><?php echo $var10; ?></td>
        <td><?php echo $sd10; ?></td>
 </tr>
</tbody>
```

```
  </table>
</div>
        <?php
//Function to calculate square of value - mean
function sd_square($x, $mean) { return pow($x - $mean,2); }

//Function to calculate standard deviation (uses sd_square)
function sd($ar) {
    //square root of sum of squares devided by N-1
    return sqrt(array_sum(array_map("sd_square", $ar, array_fill(0,count($ar),
(array_sum($ar) / count($ar)) ) ) ) / (count($ar)-1) );
}

         ?>

  <!-- Footer -->
  <footer class="py-5 bg-light">
    <div class="container">
```

```
      <p class="m-0 text-center text-dark">Copyright &copy; Gosoft Solutions
2019<img src="gosoft.png" width="60px"></p>
    </div>
    <!-- /.container -->
  </footer>
  <!-- Bootstrap core JavaScript -->
  <script src="vendor/jquery/jquery.min.js"></script>
  <script src="vendor/bootstrap/js/bootstrap.bundle.min.js"></script>
<script src="canvasjs.min.js"></script>
</body>
</html>
```