UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE ESTUDIOS SUPERIORES ACATLÁN

METODOLOGÍA PARA IMPLEMENTAR REDES NEURONALES ARTIFICIALES RECURRENTES PREDICTIVAS.

T E S I S

QUE PARA OBTENER EL TÍTULO DE:

LIC. EN MATEMÁTICAS APLICADAS Y COMPUTACIÓN

P R E S E N T A :

ALI ALDERETE PERALTA

DIRECTOR DE TESIS:
M.C. JAVIER ROSAS HERNÁNDEZ
2014

A mi madre Lilia, a mi padre Josué y a mi hermano Isay.

A mis amigos quienes son parte de mí y de mi vida.

A todas las personas que en algún momento caminaron a mi lado.

A la vida por brindarme un día más de vida y con esté la oportunidad de poder ser mejor cada vez.

Agradecimientos

Quisiera que estas palabras llegaran a todas las personas que directa o indirectamente contribuyeron a la realización no sólo de este trabajo, sino a todos los que aportaron algo a mi formación como persona y profesionista, gracias.

Me gustaría agradecer explícitamente a la Facultad de Estudios Superiores Acatlán por ser una institución en la que las personas pueden crecer y desarrollarse plenamente en un ambiente de alta calidad académica, pero además de equidad, respeto y tolerancia.

También me gustaría agradecer a las personas que estuvieron más de cerca durante este proceso, en específico a mi asesor M.C. Javier Rosas Hernandez y al Lic. Fernando Israel Gonzalez Trejo, quienes además de su apoyo incondicional me brindaron una palmada en la espalda cuando más se necesitó, gracias por confiar en mí.

Antes de concluir con estos agradecimientos, me gustaría también agradecer a mis amigos, quienes juntos hemos crecido como profesionistas, en especial gracias por todo a ti Fany

Y finalmente me gustaría agradecer a mi familia: gracias Lilia, gracias Josué, gracias Isay.

"De qué sirve ser más fuerte
sin saber ser mejor."

Contenido

Lista de Figuras

Introducción

En la actualidad existen problemas computacionales cuya solución es muy costosa en tiempo de operación, es decir que tomaría mucho tiempo realizar todos los cálculos necesarios para llegar a una solución óptima. En la teoría de la complejidad estos problemas son denominados problemas $NPC(NP-completos)$ tal como el problema del Árbol de Steiner [Woegingerv & Gerhard, 2000]; son problemas de decisión que no es posible resolver en tiempo polinomial. En los últimos años se ha investigado mucho en la solución de estos problemas con nuevas e innovadoras técnicas y heurísticas. Una de las tendencias en estos estudios es la imitación de conductas, comportamientos y funcionalidades específicas de los seres vivos.

En en este trabajo de investigación se presentarán modelos que solucionan problemas imitando el funcionamiento neuronal de un individuo y como éste genera su propio conocimiento.

En primera instancia y como parte medular de este trabajo se hablará de las Redes neuronales artificiales, que basan su funcionamiento en el de neuronas, se hará énfasis en la solución de problemas de predicción.

En segudo término se abordará el problema de predecir el precio del jitomate en México, se tomará como referente un modelo estadístico que ataca el mismo problema con la intensión de identificar áreas de oportunidad en la predicción del modelo neuronal.

El objetivo principal de la tesis es proponer una metodología para la implementación y optimización de una red neuronal artificial recurrente, y se trabajará sobre la hipótesis del mejor funcionamiento de las Redes Neuronales Artificiales con elementos de recurrencia.

El trabajo se encuentra dividio en cuatro capítulos,

1. Capítulo I- Redes neuronales artificiales

   Se plantean las características y funcionamiento de una red neuronal artificial así como las diferentes arquitecturas, tipos de aprendizaje y entrenamientos. Además, se

mostrarán las ventajas de las arquitecturas multicapa y las arquitecturas con elementos de recurrencia sobre las arquitecturas monocapa.

2. Capítulo II- Metodoloía de implementación

   Se hará una recapititulación de contenido del Capítulo I y se planteará una metodología de implementación previa a la aplicación, en el contexto de un problema de predicción.

3. Capítulo III- Caso de aplicación, predicción del precio del jitomate en México

   La elección del caso de aplicación es justificada y se proponen las variables que intervendrán en el modelo. Finalmente se hará la propuesta de modelos de redes neuronales artificiales que den solución a este problema de pronósticos.

4. Capítulo IV- Entrenamiento y resultados

   Para concluir el trabajo y con la finalidad de probar la hipótesis se compararán los resultados obtenidos entre los modelos propuestos con recurrencia y sin elementos recurrentes, empleando datos reales en el problema y se compararán los resultados con los del modelo estadístico para concluir sobre la eficiencia de los modelos.

Capítulo 1

Redes neuronales artificiales

En este primer capítulo se presentará las principales características y el funcionamiento de una Red neuronal artificial como parte de la investigación en el campo de la Inteligencia artificial, ésto se retomará en el Capítulo II donde se propondrá una metodología para implementar estos modelos neuronales. Esta investigación se desarrolla desde el punto de vista de la inteligencia artificial.

También se plantea como la recurrencia es una alternativa que permite sortear las limitantes del modelo.

1.1 Antecedentes

Las Redes neuronales artificiales, RNA o ANN (por sus siglas en inglés), son un modelo computacional simplificado del cerebro humano. Este modelo es un sistema de procesamiento de información que comparte características con un cerebro humano y tiene como principal objetivo el generar conocimiento.

Una RNA, análogamente a su símil biológico, es capaz de generar y almacenar conocimiento a través de refuerzo y la experiencia. Su unidad básica de procesamiento semeja a la célula fundamental del sistema nervioso, la neurona.

Se analizará las características de una neurona biológica, que es el precedente del funcionamiento de una RNA, después se expondrá la analogía con las neuronas artificiales.

Las neuronas son las responsables de todas las funciones vitales del cuerpo humano, al igual que cualquier otra célula tiene una estructura fisiológica y a su vez son parte de una estructura funcional, en este caso la estructura funcional es el sistema nervioso.

En cuanto a su estrucutra, a continuación son presentadas las características y propiedades que son de interés para la investigación.

1.2 Modelo neuronal

Existen millones de neuronas en el cerebro y éstas se interconectan para formar redes neuronales que a su vez forman parte del sistema nervioso. Las neuronas interactuan entre sí para realizar cada una de las funciones voluntarias o involuntarias del cuerpo humano. Cada una de las neuronas en una red neuronal necesita comunicarse con las demás y ser capaz de procesar las señales que le son transmitidas para finalmente hacer su función en el proceso de transmición de esta señal. Para realizar el proceso mencionado anteriormente son necesarios tres elemento fisiológicos elementales, la figura 1.1 muestra el diagrama fisiológico de una neurona,

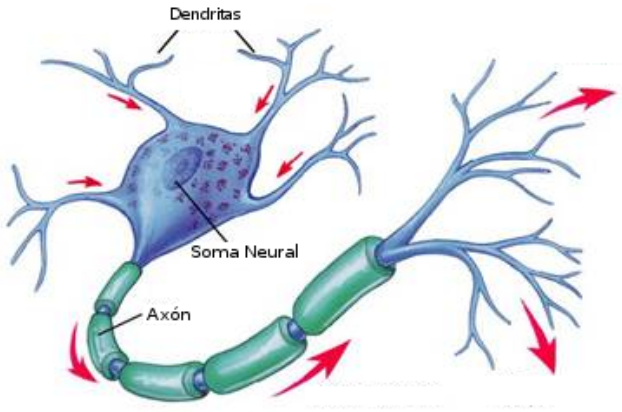

Figura 1.1: Fisiología de una neurona, obtenido de https://www.wikisaber.es

i) Dendritas:

Las dendritas tienen como función principal el recibir impulsos eléctricos y transmitirlos a su soma neuronal, estos impulsos eléctricos son producidos por el mismo sistema nervioso. Si los impulsos provienen del mundo exterior se les llamará estímulos. Pero si éstos provienen de otra neurona se les denominará señales. Este proceso de comunicación es llamado sinapsis.

Una neurona puede estar conectada con más de una neurona al mismo tiempo de manera que se reciben y transmiten señales a más de una neurona a la vez. Durante

la transmición y recepción de señales éstas se modifican, ésta última característica también es emulada por las RNA.

ii) Soma neuronal:

El soma neuronal o el cuerpo de la neurona se encarga de colectar todas las señales, si estos estímulos superan un cierto umbral de activación entonces la neurona propaga la señal a las neuronas con las que esté conectada, esta señal se propaga por el axón de la neurona a la dendritas de las demás neuronas.

iii) Axón

El axón al contrario de las dendritas se encarga de transferir los impulsos eléctricos del soma neuronal a las dendritas de otra neurona, sí y sólo si es que se supera su umbral de activación.

Las RNA, al homologar el modelo neuronal, además de los elementos fisiológicos pretenden emular tres aspectos básicos de los sistemas biológicos, ésto es lo que coloca a las redes neuronales dentro del campo inteligencia artificial, a continuación se listan estas tres características.

1. Procesamiento en paralelo:

   A un cerebro humano le toma al rededor de $20ms$ preprocesar una imagen de millones de píxeles, analizarla e interpretarla. Aún no existe ningún sistema artificial desarrollado por el ser humano que sea capaz de hacer algo parecido. El punto escencial recae en que durante este proceso se activan e interactuan millones de neuronas al mismo tiempo, millones de neuronas operan paralelamente en el procesamiento de la imagen.

2. Memoria distribuida:

   Además de procesar la imagen, el cerebro es capaz de asociarla con algún recuerdo, persona, sentimiento etc. En los sistemas tradicionales de computación los datos se alojan en localidades de memoria bien definidas, mientras que en el cerebro la memoria se encuentra distribuida a través de las neuronas. En una RNA cada una de las neuronas guarda una parte del conocimiento. De modo que si una neurona es dañada,

tanto en el sistema biológico como en el artificial, no se pierde toda la información sino sólo una porción de la información. Otro concepto relacionado a la memoria distribuida es la redundancia, ésto significa que varias neuronas pueden tener una función y/o información similar, estas características le permiten al sistema ser tolerante a fallos.

Las RNA son los primeros modelos computacionales con inherente tolerancia a fallos, ésto hace que las RNA puedan aprender a reconocer señales distorsionadas o incompletas.

La memoria distribuida se asocia con la capacidad de funcionar, incluso si se perdiera alguna neurona, el cerebro hace esto mismo por medio de la Plasticidad Funcional Compensatoria, como se expone en la investigación [Blázquez, et al., 2006], que minimiza los efectos de la modificación estructural a causa de daño en los tejidos. En una red neuronal si se continua con el funcionamiento bajo estas circunstancias implicará una disminución en la precisión de los resultados.

3. Aprendizaje adaptativo

   Esta característica es la de mayor importancia en las RNA y se refiere a la capacidad del cerebro de generar y retener un aprendizaje mediante un entrenamiento a través de estímulos y señales que ejemplifiquen los escenarios y los resultados esperados. Una RNA no requiere que se le especifique el algoritmo para obtener los datos de salida, ya que por si misma puede generar las reglas de aprendizaje mediante un entrenamiento las RNA son capaces de autogenerar su propio conocimiento.

1.3 Modelo neuronal artificial

Las características fisiológicas y funcionales de las neuronas, que son mencionadas en §1.2, tienen su símil en una neurona artificial y se muestran en la figura 1.2.

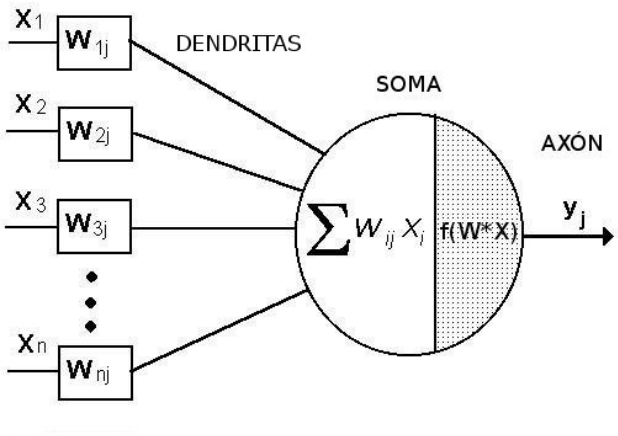

Figura 1.2: Diagrama de una neurona artificial

Las señales de entrada o estímulos provienen del mundo exterior y son propagadas al siguiente nivel del que se encuentra la neurona artificial, ésto hace referencia a las dendritas y soma de una neurona.

Una vez que todas las señales son sumadas dentro del soma neuronal de la capa de procesamiento la suma es evaluada por una función de activación y después es propagada a la siguiente capa de procesamiento o finalmente emitida como salida del modelo.

Cada vez que el modelo genera una salida, también se calcula el error asociado con base en la diferencia entre la entrada y la salida, es aquí donde el aprendizaje adaptativo tiene lugar.

Ahora se supone más de una neurona, organizadas un vector de neuronas en la capa de entrada y del mismo modo un vector de parámetros que se llamarán pesos sinápticos, son estos pesos sinápticos los responsables de almacenar el conocimiento, al mismo tiempo el conocimiento estará distribuido en estos pesos. La modificación de estos pesos se lleva a cabo con cada señal de entrada y aquí tiene lugar al aprendizaje de la red neuronal.

Se analizará primero los diferentes diseños o arquitecturas en las que se basan las redes neuronales artificiales para establecer sus características y limitantes.

1.4 Arquitectura del modelo

Las RNA tienen características operativas que las diferencían unas de otras, dos aspectos principales: su arquitectura y su método de aprendizaje.

Se considerará conjuntos o vectores independientes de neuronas que trabajan en distintos

niveles de la RNA, estos conjuntos se acomodan en tres tipos de niveles de operación de una red neuronal.

1. Nivel de entrada

   En este nivel las neuronas no realizarán ningún tipo de cálculo. Estas neuronas sólo se encargan de recibir estímulos del mundo exterior, se hará referencia a estas señales como Inputs; el nivel de entrada propaga las señales al siguiente nivel de procesamiento.

2. Nivel de procesamiento

   El nivel de procesamiento estará constituido por neuronas que se comportan de manera similiar en su función de activación, además reciben las señales de una capa anterior y deben llevar a cabo el proceso de evaluación y propagación de las señales al siguiente nivel de la red neuronal.

   Este nivel tiene el papel de capa de salida, siempre que la RNA solo conste de dos niveles, el nivel de entrada y el nivel de salida.

3. Nivel de salida

   El nivel de salida es una capa de procesamiento, con la única diferencia de no propagar las señales sino emitir un resultado, el Output del modelo.

En lo general una RNA contará con el nivel de entrada, el cual no será considerado como capa, y el nivel de salida. Si éste es el caso se referirá a una RNA monocapa (Single layer network), ya que en la red sólo se considerará el nivel de salida.

En caso de que exista uno o más niveles de procesamiento entonces se tratará de una RNA multicapa (Multilayer network), en la literatura se le llama a estas capas de procesamiento capas ocultas (Hidden layer).

En resumen, el modelo neuronal artificial está formado por neuronas que se agrupan en niveles, estos niveles están conectados hacia adelante con el siguiente nivel, sea este un nivel de procesamiento o de salida.

Las neuronas que compartan un nivel en la red también compartirán la misma configuración en su función de activación. La conexión entre distintos niveles se lleva a cabo por medio de los pesos sinápticos los cuales se ven modificados con base en los errores producidos durante un entrenamiento.

Las distintas arquitecturas o diseños están fuertemente relacionadas con el método de aprendizaje y en general con el funcionamiento de la RNA. Ahora, se abordará el diseño de la estructura en una red, su arquitectura.

1.4.1 Redes monocapa

Esta arquitectura será referida como Single layer network, que solo están formadas por las neuronas que reciben los Inputs del mundo exterior y las neuronas que realizan el procesamiento y finalmente arrojan un Output.

Las neuronas de entrada se conectan con las neuronas de la capa de salida pero no interaccionan con alguna otra de la capa de entrada, esto hace que los pesos de cada unidad de entrada sean independientes con respecto a los pesos de las demás. Esta arquitectura es capaz de resolver una gran variedad de problemas, todo depende de la interpretación que se le dé a los Outputs de la red, mientras que para un problema de clasificación, cada neurona de la capa de salida representa una de las categorías, en un problema de reconocimiento de patrones todo el conjunto de Outputs es la solución al problema.

Como se presentará en la siguiente sección, §1.4.2, el entrenamiento de esta arquitectura muestra limitantes ante ciertos problemas por lo que son requeridas otras arquitecturas.

Para estudirar un ejemplo de Single layer network se muestra la figura 1.3. Se asumirá un caso básico de reconocimiento de patrones con tantas neuronas de entrada como características tengan los individuos, en la literatura se llama patrones a estos conjuntos de características, y una sola característica.

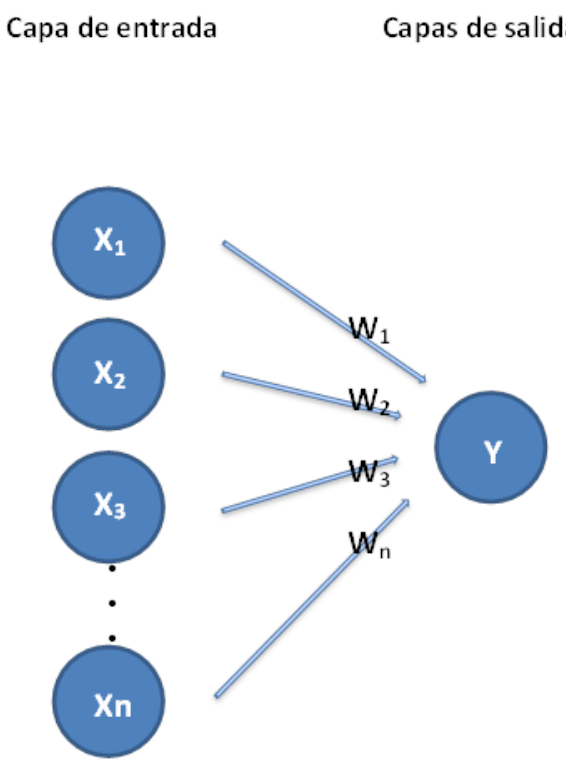

Figura 1.3: Red monocapa (simple)

Un ejemplo es el de la detección de operaciones fraudulentas hechas sobre cuentas bancarias, los Inputs son todas las características de cualquier operación bancaria ya clasificada como fraudulenta. Se considera la información relevante con respecto a las transacciones para entrenar a la red neuronal, después del entrenamiento ésta será capaz de predecir para cualquier operación si ésta es fraudulenta o no (con un cierto nivel de confianza). La salida final de esta arquitectura será si el patrón pertenece o no a la categoría, si es o no una transacción fraudulenta.

Ahora, será analizado el caso en el que existe más de una categoría para los patrones entonces se debe añadir más unidades de procesamiento a la capa de salida, cada una correspondiente a las distintas categorías:

Para ejemplificar esta red, será abordado el caso de la clasificación de caracteres dibujados en una cuadrícula, cada neurona de la capa de entrada recibirá información de cada porción sobre la cuadrícula. Después del entrenamiento la red será capaz de distinguir entre una letra y otra.

El Output de esta red dependerá principalmente de los patrones utilizados para el entrenamiento, después de dicha fase la red es capaz de clasificar correctamente cada Input, la figura 1.4 ejemplifica una arquitectura con vectores de neuronas tanto en la capa de entrada

como en la capa de salida.

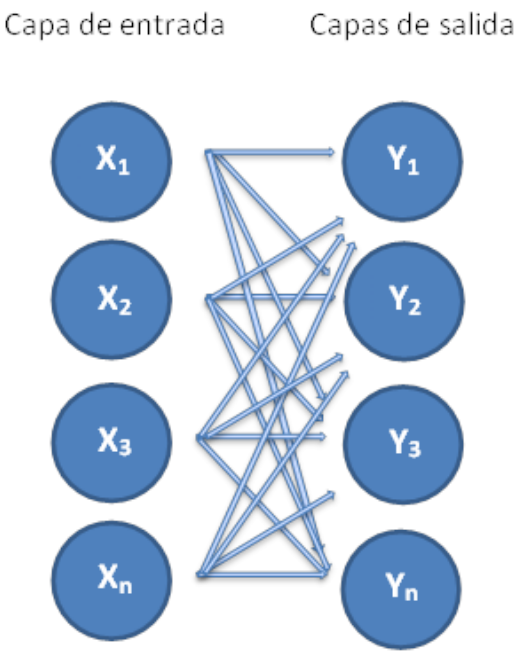

Figura 1.4: Red monocapa

1.4.2 Implementación de XOR, OR exclusivo

Un ejercicio que expuso las limitantes de la arquitectura Single Layer es la implementación de la función lógica XOR, esta implementación se expone en [Minsky & Papert 1969] . Esta función lógica devuelve "true" si y sólo sí sólo uno de sus Inputs es "true" o devuelve "false" en cualquier otro caso.

La función XOR se define como:

X_1 XOR X_2 := (X_1 AND NOT X_2) OR (X_2 AND NOT X_1)

Tabla 1.1: Tabla de verdad XOR

X_1	X_2	XOR
0	0	0
0	1	1
1	0	1
1	1	0

En la tabla 1.1 se listan los valores de la tabla de verdad con las distintas combinaciones de Inputs y sus respectivos Outputs, donde el valor "true" se define como 1 y el valor "false"

como 0.

Se puede implementar esta función con base en una disyunción (OR) entre dos conjunciones (AND) que tienen como parámetros ambos Inputs. El Output requiere al mismo tiempo el valor de los Inputs y la negación de los mismos. Se necesita una capa más de procesamiento que reciba como Input la resolución de cada miembro de la disyunción.

Tabla 1.2: Tabla de verdad implementación XOR

X_1	X_2	(X_1 AND NOT X_2)	(X_2 AND NOT X_1)	(X_1 AND NOT X_2) OR (X_2 AND NOT X_1)
0	0	0	0	0
0	1	0	1	1
1	0	1	0	1
1	1	0	0	0

En la tabla 1.2 se ejemplifica la tabla de verdad de la función implementada con base en conjunciones y disyunciones. Una red multicapa sería capaz de recibir en la primer capa los Inputs X_1 y X_2, mientras que la capa oculta procesaría el resultado de las conjunciones.

1.4.3 Redes multicapa

Su principal diferenciador es la existencia de dos o más capas de procesamiento y es lo que le permite a esta arquitectura resolver problemas de mayor complejidad. Por otra parte su algoritmo de entrenamiento es más complejo debido a que se generan errores asociados a cada capa y a su vez implica la existencia de más vectores de pesos sinápticos.

Se abordará el problema de la implementación de la función lógica XOR, expuesto en §1.4.2.

Se parte de la expresión,

X_1 XOR X_2 := (X_1 AND NOT X_2) OR (X_2 AND NOT X_1)

Cada miembro de la disyunción se representa como un problema independiente como se muestra en la tabla 1.1 y 1.2,

$Y_1 = (X_1$ AND NOT $X_2)$

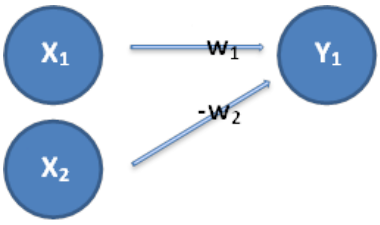

Figura 1.5: Primer conjunción

y

$Y_2 = (X_2 \text{ AND NOT } X_1)$

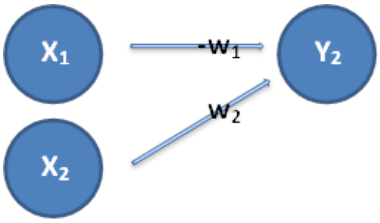

Figura 1.6: Segunda conjunción

Y_1, Y_2 serán renombradas como Z_1, Z_2 ,sus Outoputs serán las entradas para la disyunción y se tendrá, como se muestra en la figura 1.7, el modelo neuronal para la función XOR.

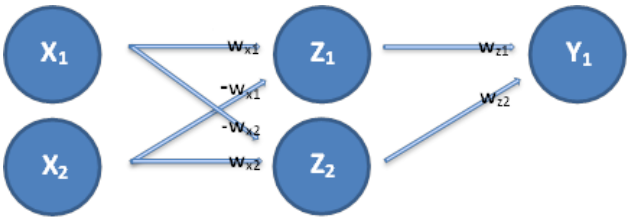

Figura 1.7: Implementación de red multicapa para la función XOR

Este modelo producirá un Output correcto para la función XOR y ejemplifica como las redes Multilayer son capaces de implementar problemas más complejos que las redes Single Layer.

1.4.4 Redes competitivas

Ésta es una variante de la Multilayer Network consiste en la introducción de una semi-recurrencia con respecto a las conexiones de las neuronas de las capas de salida. La capa oculta propaga su señal a la capa de salida y cada una de las señales afecta por un cierto

coeficiente, en función de las demás señales de salida, dicho de otra forma, cada una de las salidas tendrá un efecto de inhibición sobre las demás, con la intención de llegar a ser la "ganadora" y tener la mayor intensidad. El objetivo primordial de esta arquitectura es el que la neurona ganadora sea la que tenga mayor aprendizaje de acuerdo con el Input de tal forma que cada neurona tendrá el aprendizaje específico de cada categoría.

Su aplicación es comparable con el Análisis Cluster, técnica estadística la cual pretende formar conjuntos de patrones con características similares, la solución considera a cada una de las neuronas de salida como propietarias de cada categoría, la figura 1.8 muestra la arquitectura para esta red.

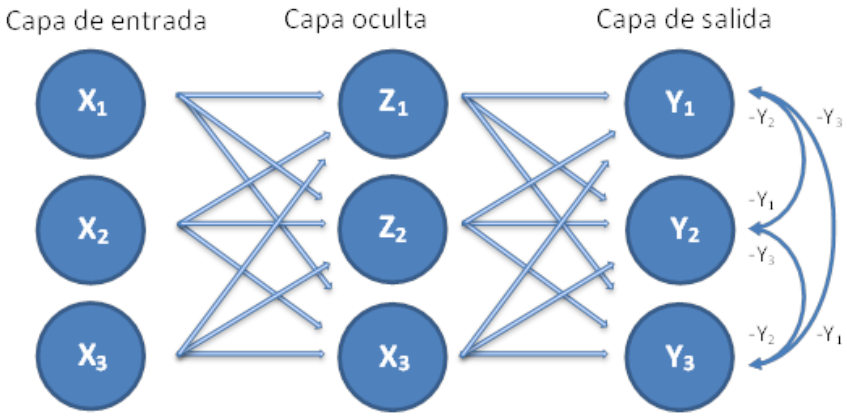

Figura 1.8: Red competitiva

Los ejemplos más usuales de esta arquitectura son las redes de Kohonen y los Mapas auto-organizados.

1.4.5 Redes recurrentes

Esta arquitectura tiene retroalimentación en las sinapsis de las capas de procesamiento, y están diseñadas para aprender patrones secuenciales o dependientes del tiempo, ésto las lleva a ser sujetos de investigación en aplicaciones de predicción.

Existen dos variantes de esta arquitectura:

1. Red completamente recurrente

   Dicha arquitectura no hace diferencia entre las capas de entrada y de salida, cada unidad tiene señales de entrada para las demás y es posible tener señales hacia sí misma, figura 1.9.

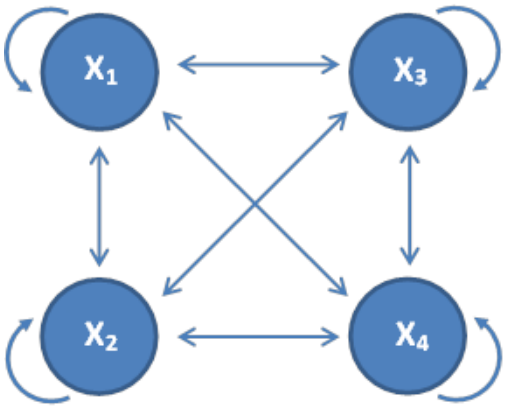

Figura 1.9: Red completamente recurrente

2. Red parcialmente recurrente

   Con una estructura de retroalimentación sólo sobre ciertas unidades, se hace la diferencia entra la capa de entrada y salida, dichas unidades reciben Inputs de su capa inmediata posterior, y se interpreta como un Input con retraso.

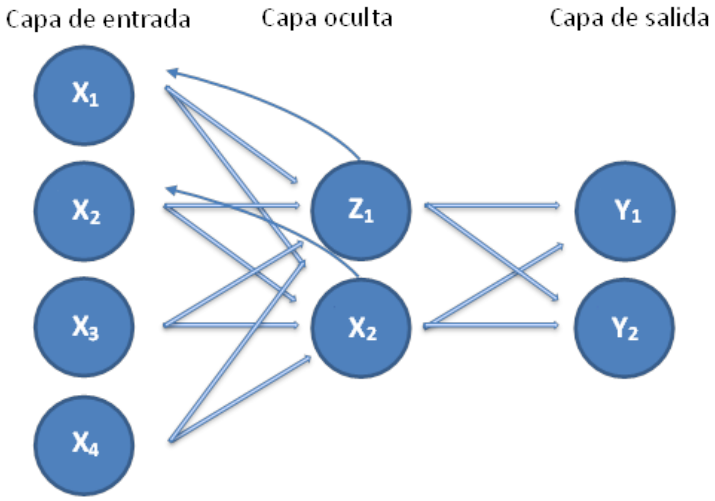

Figura 1.10: Red parcialmente recurrente

   Si es el caso en el que el Output de cada neurona de la capa oculta retroalimenta a las neuronas de la capa de entrada, como se muestra en la figura 1.10, entonces se estará hablando de una Red localmente recurrente, Local feedback. Si es el Output final de la red quien intervine en la retroalimentación, se referirá a una Red recurrente retroalimentada por la salida (Output-Feedback).

Con lo hasta ahora estudiado se utilizará una nueva clasificación de las RNA, y se clasificarán en función al carácter de sus Outputs: redes neuronales estáticas y redes

neuronales dinámicas, las primeras serán todas aquellas redes no recurrentes dejando a las recurrentes dentro del segundo grupo.

Esta nueva clasificación es por la remarcable cualidad de la recurrencia de ser capaz de crear una memoria de los datos, lo anterior permite obtener resultados distintos para el mismo Input, a diferencia de las redes estáticas en donde para el mismo Input siempre se obtendrá el mismo Output. De manera que los Outputs dependerán del estado actual de la red y éste cambiará a través del paso del tiempo.

La siguiente característica a estudiar es el método en el que se llevará a cabo el entrenamiento del modelo neuronal.

1.5 Aprendizaje

El siguiente aspecto a considerar en el funcionamiento de las redes es el método de aprendizaje, pero antes se debe diferenciar dos tipos diferentes de generar conocimiento: el aprendizaje supervisado y el aprendizaje no supervisado.

1. Aprendizaje supervisado

   La primer forma de generar conocimiento es el aprendizaje supervisado y se refiere básicamente a utilizar vectores o patrones de entrada junto con sus patrones asociados de salida, entonces el algoritmo creará internamente una categoría que describa al patrón de entrada y pueda reconocerlo o algún otro patrón similar. Al conocer tanto los patrones de entrada como los de salida, es posible saber el nivel de error al asociar un patrón de entrada con su correspondiente patrón de salida, entonces se puede pensar en modelar una función de error y minimizar dicha función.

   El aprendizaje supervisado se compara con métodos de clasificación como el Análisis Discriminante, donde es necesario que los sujetos se encuentren previamente clasificados para la correcta generación de las categorías.

2. Aprendizaje no supervisado

   El aprendizaje no supervisado, no cuenta con la posibilidad de comprobar los resultados en relación de las salidas correctas, el aprendizaje de la red se basa solamente en las semejanzas entre los patrones de entrada. Al no conocer la correcta clasificación de los patrones la red creará las categorías.

Este método de aprendizaje se compara con métodos estadísticos como el Análisis Cluster que forma clusters o conjuntos de sujetos altamente parecidos (correlacionados).

Cual sea el método de aprendizaje, el proceso de entrenamiento se lleva a cabo en dos fases, la fase de entrenamiento y la fase de prueba. Existe la posibilidad de realizar los procesos por separado o en conjunto. El tiempo es uno de los factores que intervienen en decidir si llevar a cabo las fases por separado. Si se cuenta con tiempo suficiente entonces es recomendable realizar las fases de manera disjunta, mientras que si se cuenta con menos tiempo se recomienda realizarlas a la par.

A continuación son descritos ambos tipos de aprendizaje para dar pie al entrenamiento de la red.

1. Aprendizaje fuera de línea

   A la primer opción mencionada en §1.5 se le denomina aprendizaje fuera de línea (Offline). Se lleva a cabo la fase de entrenamiento en donde se toma una parte de los patrones de entrada y se deja otra para la segunda fase. Una vez que se terminan las iteraciones con la primera parte de patrones se tiene una solución estática al problema, es decir, las reglas de clasificación permanecerán con sus respectivas magnitudes. Después se utiliza el resto de los patrones para comprobar el funcionamiento de la RNA. Dada su naturaleza estática no habrá problemas de estabilidad una vez obtenida la solución.

2. Aprendizaje en línea

   En la segunda opción donde se supone que el tiempo apremia, no es posible diferenciar la fase de entrenamiento de la de prueba ya que la red es entrenada y empleada al mismo tiempo. Aquí no existe como tal un punto de término de las iteraciones, simplemente la red se pone en funcionamiento.

1.6 Entrenamiento

La generación de conocimiento en una RNA se reduce a minimizar el error asociado a una salida deseada a partir de una entrada dada. De modo que el entrenamiento se reflejará con un algoritmo que sea capaz de modificar las reglas de clasificación.

Se utilizará el algoritmo Backpropagation por su capacidad de propagar el error asociado a las capas ocultas de la red y ajustar los pesos sinápticos. El ajuste se lleva a cabo en función de este error que existe entre el patrón de salida y el Output, este error se ve propagado reflejado en cada una de las Hidden layer.

Algorithm 1 Backpropagation Algorithm

1: Inicializar los pesos sinápticos
2: **while** Condición de paro = NO **do**
3: **for** Para cada Input **do**
4: ▷ Feedforward
5: Cada neurona de entrada recibe las señales externas y las propaga a la siguiente capa.
6: Cada neurona oculta suma y evalúa las señales en la función de activación y propaga esta señal a la siguiente capa.
7: Cada neurona de salida suma y evalúa las señales para generar el Output de la red.
8: ▷ Backpropagation
9: Cada neurona de salida calcula el error entre el Output de la red y el Output asociado con el Input, y calcula el ajuste para los pesos sinápticos.
10: Cada neurona oculta calcula el ajuste de los pesos sinápticos de acuerdo con el error propagado por la capa anterior.
11: ▷ Actualización de parámetros
12: Cada capa oculta y de salida actualiza los pesos sinápticos.
13: Actualizar condición de paro.
14: **end for**
15: **end while**

El algoritmo de Backpropagation consta de dos etapas que iterativamente evalúan una función de activación y minimizan su función de error asociada.

El algoritmo Backpropagation fue diseñado por distintos investigadores durante el mismo periodo de tiempo pero fue a Hinton Rumelhart y Williams Rumelhart en [Rumelhart, Hinton & Williams, 1986], a quienes se les atribuye su publicación y divulgación. Otos investigadores que trabajaron en este descubrimiento fueron David Rumelhart, Geoffrey Hinton, Ronal Williams y Yann Le Cun entre otros.

El descubrimiento se dió en 1986 tras un gran abandono en este campo de investigación, en los años 70's no se tuvo actividad significativa en la investigación de las Redes neuronales artificiales. El abandono que se había presentado en los últimos años se debió a las limitantes de las arquitecturas monocapa, fue entonces que el algoritmo Backpropagation logró que una red asociara correctamente patrones de entrada con sus respectivas categorías, además representó más avances para las redes neuronales como la naturaleza paralela de las neuronas, con lo que se podía aprovechar al máximo la capacidad de los procesadores.

Como se mencionó en la sección §1.4.3 las arquitecturas multicapa son capaces de resolver problemas más complejos y fue el descubrimieto del algorimo Backpropagation lo que permitió abordar su entrenamiento.

La teoría que respalda este funcionamiento es el de la derivada y el gradiente de una función, cabe mencionar que se debe tener presente las limitaciones de cómputo de todas las operaciones, hablando específicamente de la factibilidad de una solución analítica de todas las funciones implicadas en el entrenamiento.

En el Algoritmo 1 "Backpropagation Algorithm" se lista de forma breve el funcionamiento de éste, también llamado entrenamiento de doble pasada. Previo a dar inicio a las iteraciones que generan el conocimiento, es necesario inicializar los pesos sinápticos de la red para los que asignamos un peso entre $[-0.5, 0.5]$ aleatoriamente, de tal modo que v_i y w_j, los pesos de la capa de entrada hacia la capa oculta y de la capa oculta a la capa de salida, tendrán pesos en este rango.

Para iniciar el algoritmo y en la etapa de primera pasada (Feedforward) la red es alimentada con los Inputs, en este caso serán vectores de números reales y cada uno de los elementos del arreglo será recibido por una de las neuronas entrada, X_i. Inmediatamente las señales son propagadas a la siguiente capa en donde cada neurona oculta, suma estas señales multiplicadas por su respectivo peso sináptico v_i y se evalúa en la función de activación de tal forma que el Input para esa neurona está dado por:

$$Z_{jl} = f(\sum_{i}^{n} x_i v_{ij}) \tag{1.1}$$

Sea Z_{jl} la j-ésima neurona de la l-ésima capa oculta, de la misma manera cada señal de esta capa es enviada, sumada y procesada por la siguiente capa, hasta que finalmente estas señales sean propagadas hasta la capa de salida donde el Input se define como:

$$Y_k = f(\sum_j^m z_{jl} w_{jk}) \tag{1.2}$$

Sea Y_k la k-ésima neurona de la capa de salida, en este último proceso se genera un Output y damos por terminada la primer iteración o pasada del algoritmo.

Recibe el nombre de primer pasada debido a la dirección en que se mueven las señales a través de la arquitectura, en la primer fase la dirección es del nivel de entrada hacia el nivel de salida, en la segunda fase cambia del nivel de salida hacia el nivel de entrada. Se toma el error entre el patrón de salida y el patrón de entrada como Input para la capa oculta y se calcula el factor de corrección δ tal que:

$$\delta_k = (t_k - y_k) f'(\sum_j^m z_{jl} w_j) \tag{1.3}$$

Donde sea t_k el k-ésimo elemento del patrón correcto asociado al Input, con la expresión anterior se obtiene el error asociado a la capa de salida y es también un factor en el ajuste de los pesos sinápticos definidos como:

$$\Delta w_{jk} = \delta_k Z_j \tag{1.4}$$

El factor Δw_{jk} se emplea para ajustar los pesos sinápticos en la fase final del algoritmo, una vez propagado el error a la capa de salida también se propaga a las l capas de procesamiento mediante el factor δ_j como Input,

$$\delta_j = (t_k - y_k) f'(\sum_j^m z_{jl} w_j) \tag{1.5}$$

El proceso para calcular el ajuste de los pesos es el mismo para las capas ocultas. Una vez que se tiene el ajuste para los pesos sinápticos, el algoritmo actualiza los pesos sinápticos de la siguiente manera,

$$v_{ij} = v_{ij} + \Delta v_{ij} \tag{1.6}$$

$$w_{jk} = w_{jk} + \Delta w_{jk} \tag{1.7}$$

Para finalizar, el algoritmo se debe evaluar si se ha alcanzado la condición de paro del algoritmo. Se puede tomar como criterio el nivel de error que tenga la red o el número de

iteraciones a las que se someten los Inputs.

1.6.1 Entrenamiento con aprendizaje no supervisado

El entrenamiento con aprendizaje no supervisado carece de patrones de salida asociados a los Inputs por lo que su funcionamiento se basa en comparar las señales de activación entre entradas distintas.

Al evaluar entradas parecidas se refuerzan las sinapsis de las neuronas activadas por el Input. La creación de las categorías se da a partir del concepto de aprendizaje asociativo, donde a estímulos parecidos se presentan reacciones similares.

Este tipo de aprendizaje presenta sus propias dificultades, tales como el que no existe una magnitud fija para el incremento en los pesos sinápticos por lo que los pesos pueden seguir incrementandose mientras más Inputs se presenten, o la falta de una función que inhiban los pesos sinápticos.

El funcionamiento de las redes, pero en específico la activación de las neuronas y la corrección de los pesos, está íntimamente ligado a la función de activación que se define para las neuronas. La función de activación también define características específicas en el funcionamiento de la red neuronal, como el tipo de Inputs que se deben presentar o el tipo de Output que arrojará el modelo neuronal.

1.7 La Función de activación

En las secciones anteriores fue planteado el algoritmo de entrenamiento para una red neuronal, ahora abordaré a la función de activación y la función del error que se requiere en el algoritmo.

Se mencionará algunas de las funciones utilizadas en el entrenamiento de la red y la función del error. Después será introducido ciertas características que se le piden a la función de activación con el fin de ser de utilidad al algoritmo.

La función de activación actúa como el umbral de activación de una neurona en un ser vivo, al evaluar las señales de entrada, ésta las tranforma en un coeficiente de activación. La función de error es el medio por el cual calculamos la veracidad de las señales de salida, es decir el error entre un Input y su Output asociado.

1.7.1 Función identidad

La función identidad no realiza ningún cambio a los Inputs de la neurona, simplemente suma las señales provenientes de las neuronas en la capa inmediata anterior y las propaga a la siguiente capa. La función se define de la siguiente forma:

$$f(x) = x \text{ para toda } x \tag{1.8}$$

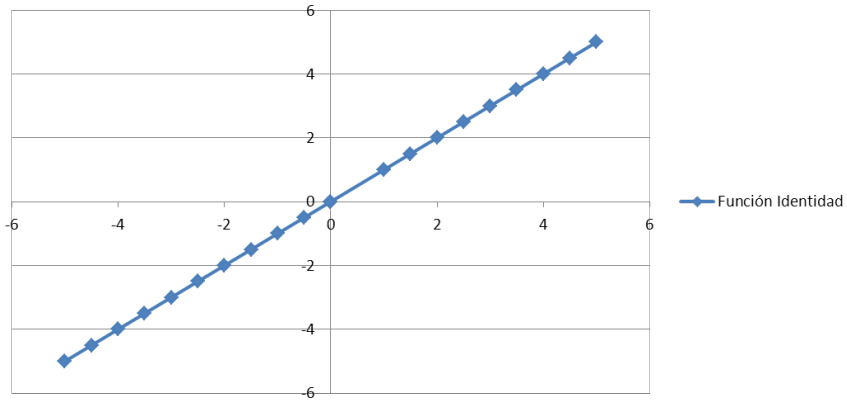

Figura 1.11: Gráfica de la función identidad

Se asume que esta función está implícita para la capa de entrada, ya que la capa de entrada sólo transfiere los estímulos a la siguiente capa.

1.7.2 Función escalón

La principal característica de esta función es su capacidad de clasificar las entradas en dos categorías, estas dos categorías dependen del valor de los Inputs de forma siguiente,

$$f(x) = \Big\{1 \text{ si } x > 0 \quad 0 \text{ si } x \leq 0\Big\} \tag{1.9}$$

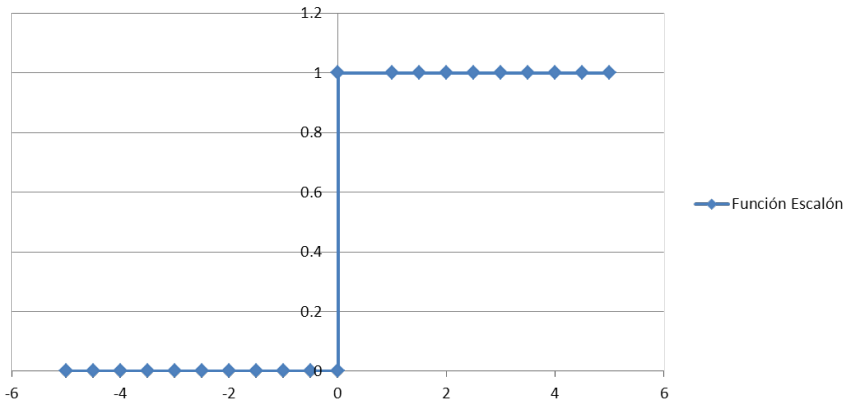

Figura 1.12: Gráfica de la función escalón

Existe también una función simétrica llamada Función de Tranferencia Hardim, dicha función sufre una transformación y una traslación para volver bipolar el Output de la función original, la siguiente expresión muestra dicha función.

$$f(x) = \Big\{1 \text{ si } x > 0 \quad -1 \text{ si } x \leq 0\Big\} \tag{1.10}$$

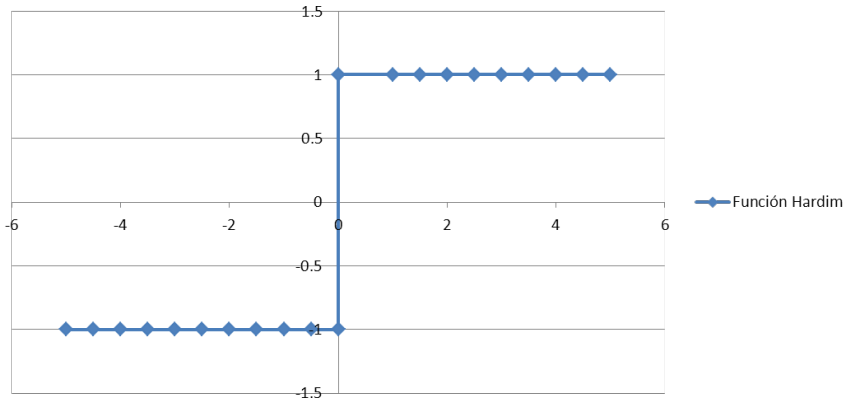

Figura 1.13: Gráfica de la función escalón simétrica

1.7.3 Función sigmoidal

La función sigmoidal es una función continua a diferencia de la función escalón, pero tienen el mismo dominio. Su principal propiedad es que su derivada está dada en términos de la misma función, tal como se muestra a continuación:

$$f(x) = \frac{1}{1 + e^{-x}} \tag{1.11}$$

$$f'(x) = f(x)(1 - f(x)) \tag{1.12}$$

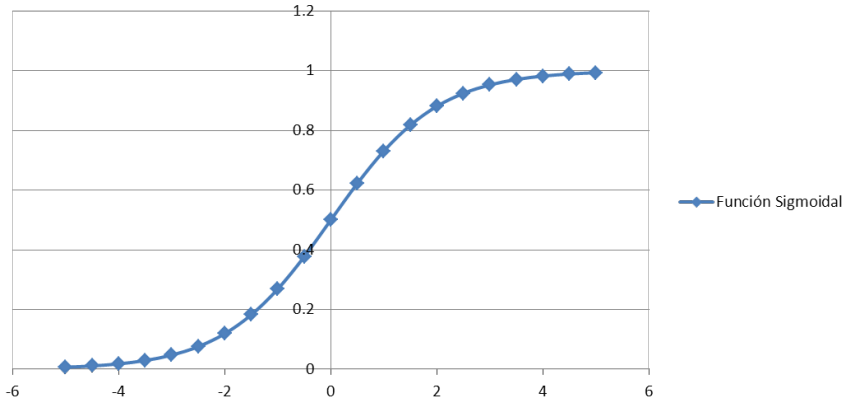

Figura 1.14: Gráfica de la función sigmoidal

De modo que al calcular un Output con esta función estaremos calculando también el valor de su derivada, ésto vuelve a la sigmoidal función en una buena candidata a ser utilizada en el algoritmo ya que el costo en tiempo de su cálculo es mínimo.

La continuidad de la función permite tener más de dos clasificaciones. De la misma forma que para la función escalón existe una función simétrica para la función sigmoidal.

$$f(x) = \frac{1 - e^{-x}}{1 + e^{-x}} \tag{1.13}$$

$$f'(x) = \frac{1}{2}[1 + f(x)][(1 - f(x))] \tag{1.14}$$

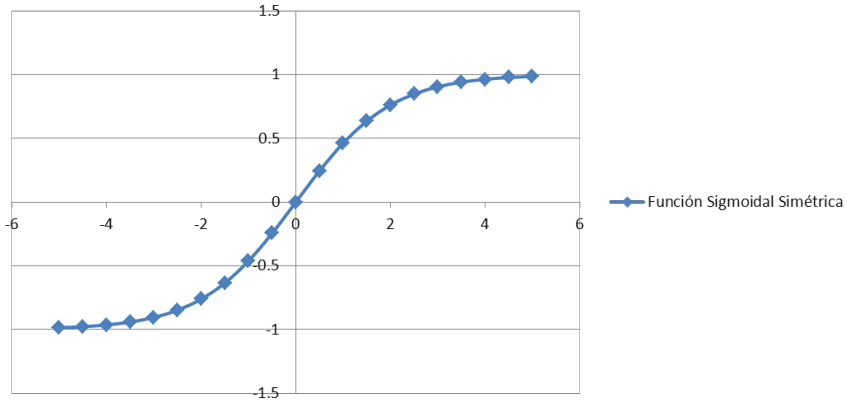

Figura 1.15: Gráfica de la función sigmoidal simétrica

Las ecuaciones anteriores muestran la función sigmoidal bipolar y su derivada la cual tiene la misma característica que la función sigmoidal no simétrica. Otra función con las mismas características es la tangente hiperbólica.

1.7.4 Función tangente hiperbólica

$$f(x) = \frac{1 - e^{-2x}}{1 + e^{-2x}} \tag{1.15}$$

$$f'(x) = [1 + f(x)][(1 - f(x))] \tag{1.16}$$

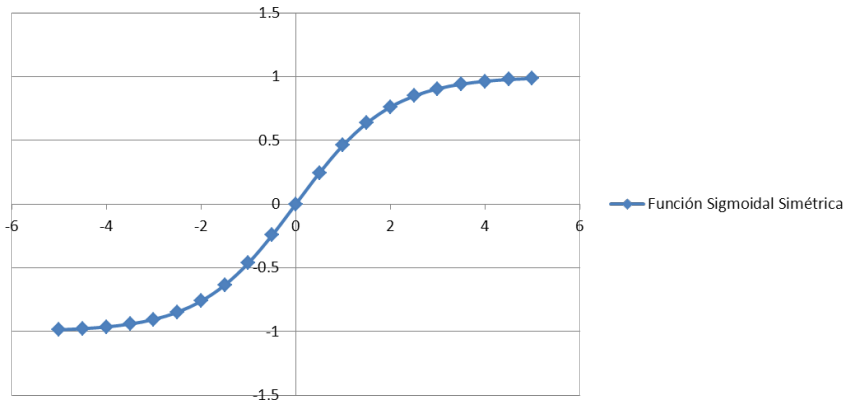

Figura 1.16: Gráfica de la función tangente hiperbólica

En las expresiones anteriores se observan las definiciones de la función y su derivada.

En la próxima sección se hablará formalmente de las propiedades que se le piden a una función de activación y el por qué son de relevancia.

1.7.5 Propiedades de la función de activación

La función de activación juega un papel muy importante en el algoritmo de entrenamiento como generadora del Output en cada una de las capas de procesamiento.

En un proceso de reconocimiento de patrones, el funcionamiento de la red se traduce en el proceso de relacionar Inputs con Outputs deseados, esta relación hace referencia al concepto matemático de función.

Función Una función f es una regla que asigna a cada elemento x de un conjunto A exactamente sólo un elemento, llamado $f(x)$, de un conjunto B.

Ésto no es más que el mapeo del espacio de Inputs sobre el espacio de Outputs, entonces la función de activación será responsable de asignar o producir Outputs a partir de los Inputs.

Las características deseables en la función de activación que se aplique en el entrenamiento se describen a continuación, se retomará en la sección §1.8.4 cuando exponga la relación de la derivada y el gradiente en el entrenamiento.

1.7.6 Continuidad

Se dice que un proceso es continuo si tiene lugar gradualmente, sin interrupciones ni cambio abrupto. Y formalmente,

Continuidad Una función f es continua en un número a si

$$\lim_{x \to a} f(x) = f(a)$$

De tal modo que la continuidad se asegura que la función de activación arroje para todo Input una correspondencia en el espacio de Outputs.

1.7.7 Diferenciable

Diferenciable Una función f es diferenciable en a si $f'(a)$ existe. Es derivable en un intervalo abierto (a,b) o $[(a,\infty)$ o $(-\infty,a)$, $(-\infty,\infty)]$ si es diferenciable en todo número del intervalo.

Después de asegurarse que todos los Inputs sean mapeados se debe asegurar que la derivada de la función existe para todo Input, ya que es necesario su cómputo para generar la información asociada al error, dicha información es requerida para calcular el ajuste en los pesos sinápticos.

1.7.8 Monótonamente no-decreciente

Una función se dice monótonamente no-decreciente en un intervalo,

Si $f(x_1) \leq f(x_2)$ siempre que $x_1 \leq x_2$ en el intervalo $[a, b]$, $\forall\ x_1, x_2 \in [a, b]$.

La diferenciabilidad junto con la monotoriedad no-decreciente son características extras que toman sentido cuando sea calculado el error asociado. El cálculo del error se realiza por medio de una función que se debe definir para la red, en la siguiente sección hablaré de algunas de estas funciones prestando mayor detalle al método del gradiente.

1.8 Función del error

Medir la eficiencia de la red es de suma importancia para llevar a cabo la implementación. Para el modelo neuronal la eficiencia del modelo se mide a través del error entre el Output generado por la red y los Outputs conocidos (asociados al Input). Los pesos sinápticos se modifican con base a esta medida de error empleando el gradiente.

Ahora, se hablará del gradiente como método de minimización del error mediante la derivada de la función de activación.

1.8.1 Error cuadrático medio

El error es una medida de eficiencia en el desempeño de la red neuronal, el error está dado en función de todos los patrones y sus correspondientes salidas asociadas. Es por ésto que la suma de los errores es una medida para saber que tan buena es la predicción, el error cuadrático se define de la siguiente manera.

$$e = \frac{\sum (t - y_i)^2}{N} \tag{1.17}$$

Donde t es el Output asociado al i-ésimo Input y y_i el i-ésimo Output calculado por la red.

Otras expresiones que involucran la suma del error se muestran a continuación, en listo algunas de ellas.

1. Error cuadrado
$$e = \sum (t - y_i)^2 \tag{1.18}$$
2. Error absoluto medio
$$e = \sum |(t - y_i)| \tag{1.19}$$
3. Error medio absoluto
$$e = \frac{1}{N} \sum \left| \frac{(t - y_i}{y_i}) \right| \tag{1.20}$$

1.8.2 Distancia de Mahalanobis

La distancia de Mahalanobis fue introducida por Prasanta Mahalanobis en 1936. Es una función que mide la diferencia entre dos vectores multivariantes valiéndose de su correlación

y se define de la siguiente manera:

$$d(\overrightarrow{x}, \overrightarrow{y}) = \sqrt{(\overrightarrow{x} - \overrightarrow{y})^T S^{-1} (\overrightarrow{x} - \overrightarrow{y})} \tag{1.21}$$

Así se define a esta diferencia entre patrones, donde $(\overrightarrow{x} - \overrightarrow{y})$ es el vector traspuesto de la diferencia de cada uno de los elementos de los vectores y S^{-1} es la inversa de la matriz de covarianza.

La distancia de Mahalanobis así como la Distancia de Voronoi y la Distancia de Manhattan, expuestas en [Paul,1998], son una alternativa para medir los errores asociados.

La función del error también forma parte de la condición de paro, ya que estableciendo el nivel de error para la red, será la manera en que el algoritmo decida cuando parar.

1.8.3 Condición de paro

El algoritmo de entrenamiento se detendrá cuando se cumpla algún criterio de eficiencia, basado en el nivel de error que la red aún produce al procesar los patrones.

Existen varios criterios, que pueden estar basados en el error o basados en el número de iteraciones. La diferencia entre estos dos criterios es el tiempo de procesamiento que implicará. En el caso de seleccionar algún nivel de confianza el número de iteraciones no está definido y se iterará hasta conseguir alcanzar el criterio o cuando se haya estabilizado el nivel de error.

La naturaleza aleatoria de los parámetros de la red al inicio del entrenamiento hace que el número de iteraciones requeridas para alcanzar el criterio del error varíe, así que si se entrenan dos redes idénticas (excepto por los parámetros iniciales) con los mismos patrones, nada asegura que el número de iteraciones sea el mismo.

Por otra parte, el criterio de paro definido como un cierto número de iteraciones es utilizado con la intención de fijar el tiempo de entrenamiento de la red. De manera contraria al criterio anterior, al fijar el número de iteraciones no se garantiza que se alcance un nivel de error específico. En esta ocasión si se entrenan dos redes idénticas (excepto por los parámetros iniciales) con los mismos patrones nada asegura que al terminar el número de iteraciones ambas redes tengan el mismo nivel de acierto en sus clasificaciones.

La función de activación así como la condición de paro son propiedades de una red que intervendrá en el entrenamiento, pero el cómo se lleve a cabo el aprendizaje y ajuste de los pesos sinápticos, es propio del algoritmo. En la siguiente sección será analizada la

forma en que el algoritmo Backpropagation aprovecha las propiedades que se expusieron en §1.7 a través del gradiente, también será estudiada una alternativa al uso del algoritmo Backpropagation.

1.8.4 El gradiente

El problema del entrenamiento se puede abordar como la búsqueda de una combinación de pesos sinápticos que arrojen un error mínimo entre el Output generado por la red y el Ouput asociado al Input. El algoritmo Backpropagation emplea el método del gradiente negativo, que es un método numérico para encontrar los mínimos de una función. Entonces tienen sentido las características que le exigimos a la función de activación.

El gradiente está basado en la derivada de una función y trabaja bajo la teoría de la recta tangente, la cual proporciona la dirección de más rápido cambio en la función, este vector tiene la misma dirección que la función de activación.

Si se considera una caminata en una montaña, la trayectoria describirá una serie de curvas que se pueden representar con una función. Si se derivara esta función se obtendría la recta tangente a la caminata y esta recta tendría la dirección de más rápido avance a la punta de la montaña. Si se cambiara la dirección de la recta se obtendría la recta de mayor avance hacia las faldas de la montaña.

Ahora, en caso de que no sea una caminata sino la función asociada a los errores del modelo neuronal, aplicando el gradiente negativo llegaríamos a los mínimos de la función.

Retomado el concepto del gradiente y la analogía con la caminata en la montaña, puedo presentar las características deseables en la función de activación de la siguiente manera:

i) Continuidad y diferenciabilidad

   Estas características son necesarias para ser capaces de "caminar" por toda la función y calcular en cada punto la dirección de más rápido cambio.

ii) Monótonamente No-decreciente

   Se debe asegurar que en todo punto del intervalo, el gradiente negativo dirige lleva hacia el mínimo. Si la función tuviera signo negativo en algún punto del dominio de la derivada la recta tangente, enviaría en dirección del máximo de la función.

1.8.5 Tasa de aprendizaje

Un parámetro extra en la arquitectura de una red neuronal es la tasa de aprendizaje, este parámetro define la proporción del cambio en los pesos sinápticos de las capas del modelo neuronal. La tasa de aprendizaje puede verse como la magnitud de los pasos en la caminata sobre la montaña, el valor para este parámetro se define entre 0 y 1.

Una tasa de aprendizaje igual a 1 no tendrá ningún efecto sobre δ para los pesos sinápticos, mientras que un parámetro menor a 1 disminuirá el cambio en los pesos sinápticos. Acorde con cada problema el parámetro puede variar ya sea modelo a modelo o dinámicamente dentro del mismo modelo.

La principal utilidad de este parámetro es evitar la convergencia prematura de la red neuronal, esto quiere decir que los pesos sinápticos se estabilizarían en un valor que pudiera no ser el óptimo (para minimizar el error) debido a tener cambios de tal magnitud que no se pueda representar correctamente el comportamiento de los datos.

Capítulo 2

Metodología de implementación

En este capítulo se hablará del proceso analítico previo a la implementacíon del modelo y será planteada una serie de pasos que involucran a cada uno de los conceptos trabajados en el Capítulo I. También se introducirán nuevos conceptos relacionados con el comportamiento de los datos, así como términos relacionados con la clasificación de los modelos según sus catacterísticas.

2.1 Antecedentes

El funcionamiento de una RNA recurrente permite abordar problemas cuyo comportamiento está ligado al tiempo o a una sucesión estructurada de los datos, en otras palabras pueden abordar los problemas de Series de tiempo y Pronósticos.

Existen diferentes metodologías para atacar problemas de series de tiempo y pronóstico como lo es la metodología Box-Jenkins [Box, Jenkins, 1994] y el Suavizamiento de curvas [Brown, 1963] los cuales están basados en la aplicación de indicadores estadísticos o ecuaciones matemáticas de aproximación.

Por otra parte las RNA son un modelo computacional que soluciona de forma alterna este tipo de problemas.

En la siguiente sección se propondrá una serie de etapas que inician con la selección de las Redes Neuronales como solución al problema hasta el desempeño en el funcionamiento de la red.

2.2 ¿Por qué una red neuronal artificial?

Cuando se enfrenta un problema de series de tiempo o pronósticos se piensa principalmente en metodologías estadísticas para la solución del problema. Sin embargo los métodos estadísticos mencionados en §2.1 tienen una limitante para proporcionar una correcta solución, es requerimiento satisfacer supuestos específicos de cada metodología, estos supuestos son también llamados pruebas de diagnóstico.

En específico para las metodologías Box-Jenkings, (i.e.) el supuesto de forma funcional lineal, significa que los datos deben de comportarse de manera lineal, de modelo contrario las metodologías anteriores no serán adecuadas para describir su comportamiento no lineal.

Cabe mencionar que es necesario el conocimiento suficiente de Estadística y Probabilidad para realizar e interpretar tanto las pruebas de diagnóstico así como para completar el método en general.

La principal ventaja de las redes neuronales sobre los métodos mencionados anteriormente es el principio de aproximación a una forma lineal o no lineal que describa el comportamiento de los datos. Las redes neuronales sortean esta limitante al aproximarse a cualquier forma funcional [Hornik,1989] , y entonces pueden abordar cualquier comportamiento en los datos.

Además, para dar solución a un problema de series de tiempo o pronósticos como mencioné en §1.3 no es necesario especificar el algoritmo de entrenamiento a la red neuronal ya que ella misma generará su conocimiento y reglas de aprendizaje y no es necesario mayor conocimiento a priori de los parámetros de la solución o necesidad de condicionar el comportamiento de los datos.

También de ser un aproximador universal y no necesitar mayor conocimiento del comportamiento de los datos, existen ya herramientas y software que permiten de manera intuitiva implementar una red neuronal. Entre estas herramientas se encuentran softwares matemáticos como MathLab y Mathematica o neurosimuladores como DataEngine y Pythia.

Para contestar finalmente a la pregunta, consideraré una red neuronal no como principal solución a problemas de esta naturaleza sino como una excelente alternativa que aborde problemas que las soluciones tradicionales no puedan abordar o no tengan una solución adecuada.

En las siguientes secciones se encuentran los pasos que propongo, previos a la implementación del redes neuronales.

2.3 Análisis de los datos

Una vez seleccionada una red neuronal como solución a mi problema, es momento de abordar los datos, para que con base en ellos defina los Inputs. En el apartado anterior mencioné que no es necesario conocer el comportamiento de los parámetros de la red, sin embargo ésto es diferente con respecto a los datos, puesto que es necesario un conocimiento previo del comportamiento al interior de los datos.

2.3.1 Estructura de los datos

Las variables son cada una de las características o atributos que formen parte de los Inputs. A continuación se hablará de la estructura de los Inputs y de la relación entre variables, de la relación de cada variable consigo misma y de cada una de las variables unas con otras.

Los Inputs estarán formados por una o más características, si está formado por sólo una variable entonces diré que es un Input univariado, en modelo de tener más de una, me referiré a él como un Input multivariado. En la literatura también se refiere al número de variables como dimensión, así de manera general un Input de una sola variable es un vector de dimensión 1 y de manera progresiva un vector de dimensión 2 es un vector de dos características y así sucesivamente.

La o las variables de las que esté compuesto el Input suponen tener una relación estructurada a través del tiempo o una secuencia estructurada. Tomaré el modelo de un Input con una sola variable X, la cual tendrá una relación consigo misma y gracias a ésto se puede conocer el valor a futuro de la variable. La relación de una variable consigo misma a través del tiempo permite modelar un problema univariado autoexplicativo de la variable.

Si el vector del Input está formado por más de una variable, y bajo las mismas condiciones de secuencia lógica a través del tiempo, se puede modelar un problema de explicación multivariado, en este modelo la relación que interesa es la explicación de una variable con el resto de ellas.

Cuando se habla de un modelo explicativo se refiere a poner el valor de una variable en función de las demás, la variable a explicar se le llama en la literatura, variable dependiente y a las variables que explican su comportamiento se les denomina independientes. En los subsecuente se utilizará estos términos para hacer referencia al tipo de variables y al tipo de problemas.

2.3.2 Temporalidad

El término temporalidad se refiere a conocer en específico al periodo de tiempo en el que fue medida cada una de las observaciones de cada variable, tanto la variable a predecir (dependiente) como las variables predictoras (independientes), con el fin de determinar el número de Inputs, i.e. una serie de observaciones semanales de los valores de una acción en el mercado podría tener un tamaño de 4 o 5 observaciones consecutivas dentro del mismo Input para reflejar un comportamiento semanal, mientras que observaciones mensuales tendrían 3 observaciones para reflejar un comportamiento trimestral, 4 para uno cuatrimestral, etc.

A las observaciones anteriores de una variable serán llamadas retrasos, Lags. Entonces de vuelta al ejemplo, el valor futuro de una acción estaría en función de las 4 (o 5) observaciones anteriores, expresión 2.1.

$$X(t+1) = f(X(t), X(t-1), X(t-2), X(t-3)) \tag{2.1}$$

Y de manera general el valor de una variable X en el tiempo t estará definido como una función de la variable X en uno o más periodos anteriores, como en la expresión 2.2.

$$X(t+1) = f(X(t), X(t-1), X(t-2), ..., X(t-n), ..., X(0)) \tag{2.2}$$

La temporalidad será vista como a las métricas o unidades de tiempo: segundo, minuto, semana, etc., se propone que el número de observaciones en el patrón de entrada debe abarcar los suficientes elementos para cubrir a la siguiente unidad de medida y que esta cobertura sea de 95% del total del comportamiento a reflejar.

En otras palabras la granularidad de las observaciones a través del tiempo marcará un indicador del número de variables en la función que describe a los datos y se hará de nuevo incapié en que no le especificaré dicha función a la red sino que simplemente el número de variables en ella.

2.3.3 Espacio de estados

Esta característica de los datos refiere al tipo de datos, si éstos son cualitativos o cuantitativos, si éstos son discretos o continuos.

Las RNA son capaces de procesar cualquier tipo de datos siempre y cuando se haga la codificación correcta de las variables. En el modelo específico de las series de tiempo y

pronósticos se empleará solamente variables cuantitativas continuas ya que éste es el formato que las fuentes de información manejan para sus bases de datos.

Los datos históricos ocupados el este estudio provienen de tres órganos, dos nacionales y uno internacional: INEGI, SIAP, FAO.

1. INEGI

   El Instituto Nacional de Estadística y Geografía coordina y armoniza la generación y administración de información geográfica de interés nacional de calidad, ésto acorde a la normatividad establecida por el Sistema Nacional de Información Estadística y Geografía, también conserva y promueve su disponibilidad para brindar el Servicio Público de Información.

2. SIAP

   El Servicio de Información Agroalimentaria y Pesca es una fuente de información estadística y geográfica del Sector Agroalimentario y Pesquero, que proveé de información a productores y agentes económicos para la oportuna toma de decisiones que contribuyan al desarrollo rural sustentable.

3. FAO

   La "Food and Agricultere Organization of the United Nations" es una organización internacional que busca alcanzar la seguridad alimentaria para todos y asegurar que las personas tengan acceso a alimentos de buena calidad que les permitan llevar una vida activa y saludable.

Un aspecto más que se debe tener en cuenta, relacionado con el espacio de estados, es el número de variables que utilizaré para explicar el comportamiento de la variable objetivo, cuidando de no confundir con el número de observaciones que serán utilizadas de la misma variable.

Para el modelo univariado la variable objetivo se explicará a sí misma con sus valores anteriores, y en el modelo multivariado habrá una o más variables explicando el comportamiento de la variable objetivo.

Para definir de manera general el valor futuro de la variable explicada se utilizará la siguiente expresión,

$$X(t+1) = f(X(t), X(t-1), .., X(0), Y(t), Y(t-1), .., Y(0), Z(t), Z(t-1), .., Z(0), ..) \quad (2.3)$$

ó

$$X(t+1) = f(X(t), Y(t), Z(t), ...) \quad (2.4)$$

Las expresiones 2.3 y 2.4 son el modelo general para el valor de la variable dependiente y corresponde el modelo multivariado, el modelo univariado es un modelo particular con una sola variable, como se muestra en las siguientes expresiones 2.5 y 2.6,

$$X(t+1) = f(X(t), X(t-1), X(t-2), .., X(0)) \quad (2.5)$$

ó

$$X(t+1) = f(X(t)) \quad (2.6)$$

Sin importar las características de los datos es necesario tener un conocimiento a priori del problema, pues el cómo se aborde, definirá tanto los imputs como la arquitectura del modelo.

2.4 Definición de los Inputs

El siguiente paso en la implementacíon, es definir la manera de como será alimentada la RNA, el formato en que se presenten los Inputs reflejará el comportamiento de los datos, se examinará el modelo multivariado y univariado, ambos empleando recurrencia local y recurrencia por el final (Out-Feedback), ésto con la intención que de forma experimental sea capaz de comparar los modelos.

2.4.1 Modelo univariado

El modelo base de los modelos univariados es una red simple con entrenamiento supervisado que emplea al algoritmo Backpropagation y se muestra en la figura 2.1. Mas adelante en §3.5 se presentará este modelo junto con los demás ya que será con base a estos resultados que se retomará la hipótesis acerca del mejor funcionamiento de las redes recurrentes para este tipo de ploblemas.

2.4.2 Modelo univariado Output-Feedback

Como fue mencionado anteriormente en §2.3.3 la expresión (2.3) define el valor del Output para modelo univariado :

$$X(t+1) = f(X(t), X(t-1), X(t-2), X(t-3)) \tag{2.7}$$

Para un comportamiento temporal de 4 periodos, se retomará la idea de la temporalidad para definir el número de neuronas en el layer de entrada, de forma que para este ejemplo la RNA tendrá 4 unidades de entrada y los inputs estarán contenidos en un vector de 4 elementos.

Como se indica en el Algoritmo 1 (§1.6) cada neurona X_i recibirá un elemento del patrón de entrada como se muestra en la figura 2.1.

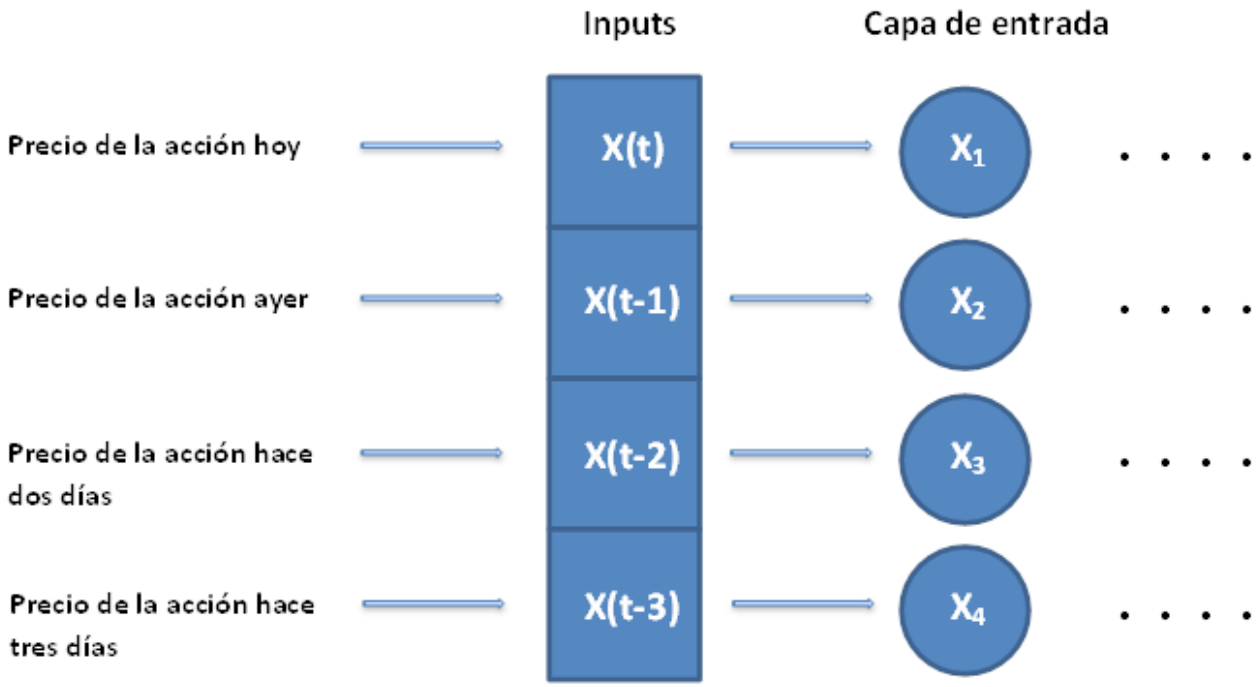

Figura 2.1: Inputs univariado

Con ésto el valor de la variable estará dado por su valor actual y su valor con uno, dos y tres retrasos. Si el elemento de la recurrencia no estuviera presente, esta implementacíon sólo estaría aprendiendo escenarios de condiciones pasadas para dar la predicción, por lo que el siguiente paso es aplicar la recurrencia de tal modo que la red reciba nuevas condiciones a partir de los mismos inputs. De tal modo que tendrá lugar una primer ejecución del algoritmo

Backpropagation, una vez que la red arroja el primer Output se utilizará este resultado para sustituir uno de los Inputs de modo que,

$$X_n = X_{n-1}, X_{n-1} = X_{n-2}, ..., X_2 = X_1, X_1 = Y \tag{2.8}$$

Para el ejemplo anterior los Inputs de la red en la segunda ejecución se observan en la figura 2.2 y están definidos de la siguiente manera,

$$X_4 = X_3, X_3 = X_2, X_2 = X_1, X_1 = Y \tag{2.9}$$

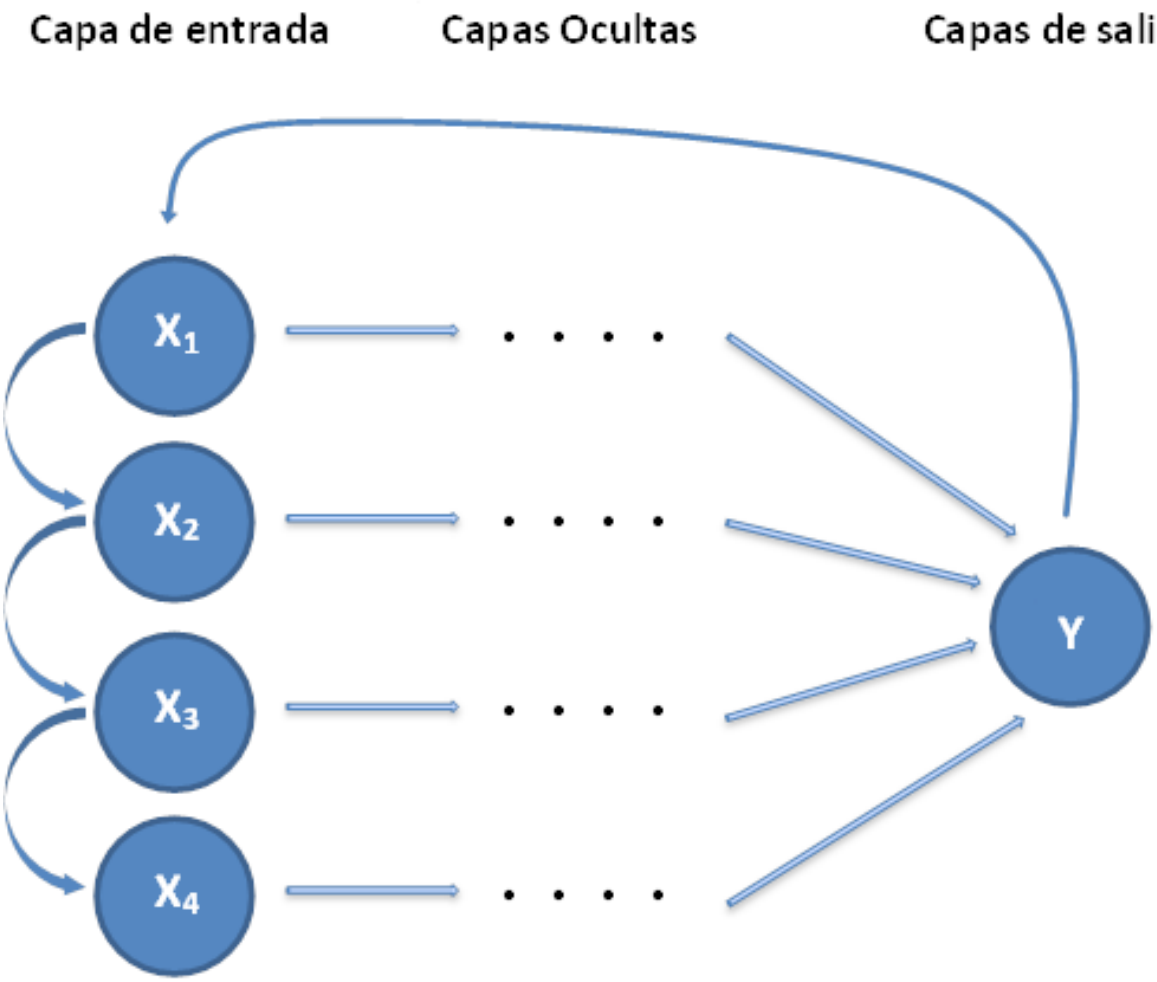

Figura 2.2: Sustitución de Inputs en recurrencia

La interpretación de esta actualización en los Inputs es que el algoritmo ha creado nuevas condiciones a partir de las anteriores, a las que se les pueden aplicar el algoritmo para evaluarlas, así que no sólo la red arrojará nuevos Inputs sino también un error asociado a

estas nuevas condiciones. El número de etapas en las que el algoritmo trabaja sobre las mismas condiciones iniciales dependerá del número de parámetros en el patrón, es decir, que se iterará hasta que cada uno de los parámetros haya sido sustituido por nuevas condiciones. Para el modelo de nuestro ejemplo se realizarán cuatro sustituciones hasta que la observación X_4 tenga un valor generado por la red, la figura 2.2 muestra el orden de la sustitución.

2.4.3 Modelo multivariado Output-Feedback

Para la recurrencia por el final en el modelo multivariado se repetirá la misma lógica de modelo univariado pero con dos vertientes, la primera es considerar que el comportamiento de las variables predictoras es independiente unas de las otras, y la segunda vertiente es considerar que el comportamiento de las variables predictoras no es independiente y se ven afectadas por el comportamiento de las demás.

Esta característica también es estudiada en la metodología Box-Jenkins, ya que la metodología tiene otro supuesto que exige a las variables independientes no ser explicativas entre ellas. En Estadística esto es la multicolinealidad y se interpreta como dependencia funcional entre dos variables independientes , por lo que una de ellas debe salir del modelo para no generar falsas relaciones de causalidad con la variable dependiente.

Para ambos modelos los datos se presentan en vectores de la misma dimensión, los vectores contendrán las observaciones de cada variable para el mismo intervalo de tiempo.

El primer modelo en el que se supone una independencia entre las variables, basta con implementar tantas copias sean necesarias del problema univariado, después un nuevo Layer que opere las salidas para cada variable. En la figura 2.3 se ejemplifica esta implementación para el modelo de tres variables y dos retrasos. Siguiendo con el mismo modelo del precio de una acción, la variable Y será el valor de otra acción en el mercado, o la oferta y demanda de algún producto o incluso el precio de algún otro producto como puede ser el precio de consumibles como la carne de res, la leche o el jitomate. La selección de las variables depende del conocimiento del problema y de la disponibilidad de información.

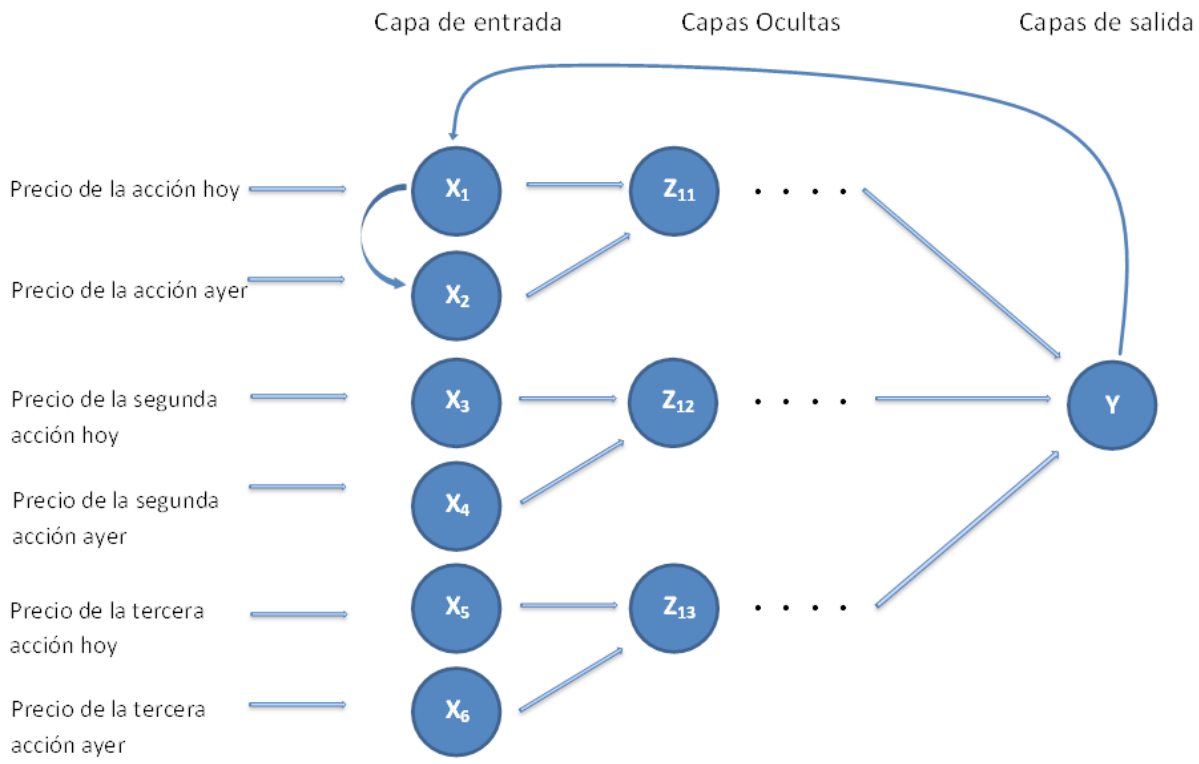

Figura 2.3: RNA Out-Feedback con Inputs multivariados independientes

Retomada la definición para el valor de la variable en el modelo multivariado de la expresión 2.3, y se presentará de tal forma que sea la composición del problema univariado,

$$X(t+1) = f(f(X(t)), f(Y(t)), f(Z(t))) \tag{2.10}$$

cabe hacer la anotación de la presencia de la misma variable objetivo en la ecuación, $f(X(t))$.

Con esta última propiedad se hace referencia al estado actual de la red y al mismo tiempo es el miembro sobre el cual recae la recurrencia. En la primer etapa de aplicación del algoritmo, la red operará con los Inputs originales y después sustituirá todos los Inputs relacionados a la variable X_i, la regla de sustitución es la misma que en los modelos univariados. Las demás variables no serán sustituidas y permanecerán con las mismas condiciones.

En el modelo que se suponga no independencia entre las variables, todos los Inputs concurrirán en la única capa oculta, como se muestra en la figura 2.4.

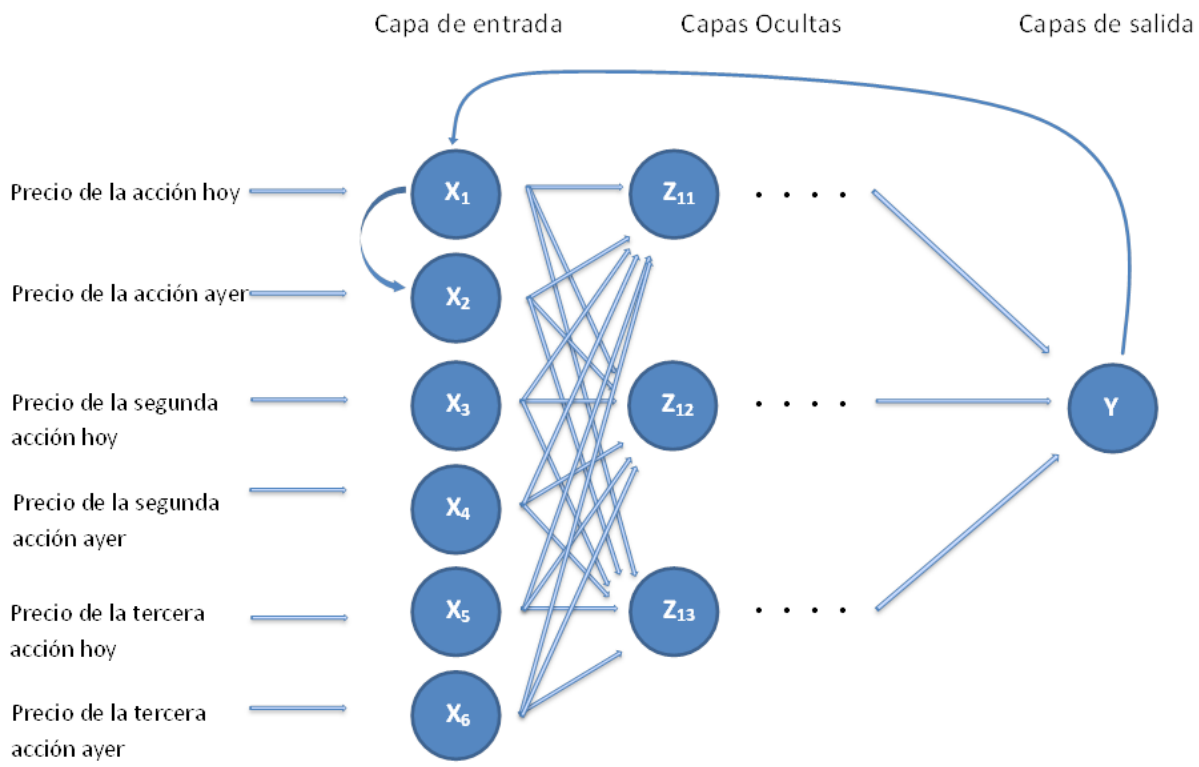

Figura 2.4: RNA Out-Feedback con Inputs multivariados no independientes

La sustitución de los Inputs de la variable X_i se realiza del mismo modo que para el modelo univariado.

En los dos siguientes modelos la recurrencia se presenta de forma local.

2.4.4 Modelo univariado localmente recurrente

Como fue explicado en §2.4.3 la recurrencia Out-Feedback reemplaza un Input por un Output calculado por la red, por otra parte la recurrencia local sustituirá a cada uno de los Inputs por uno calculado. Se retomará a la definición para el valor del Output, expresión 2.7.

$$X(t+1) = f(X(t), X(t-1), X(t-2), X(t-3)) \tag{2.11}$$

En la figura 2.5 se presenta la dirección de la recurrencia, ahora se hará énfasis en la capa oculta de la red, donde el valor de cada input está dado por:

$$Z_j = f(\sum_{i}^{n} x_i v_i) \tag{2.12}$$

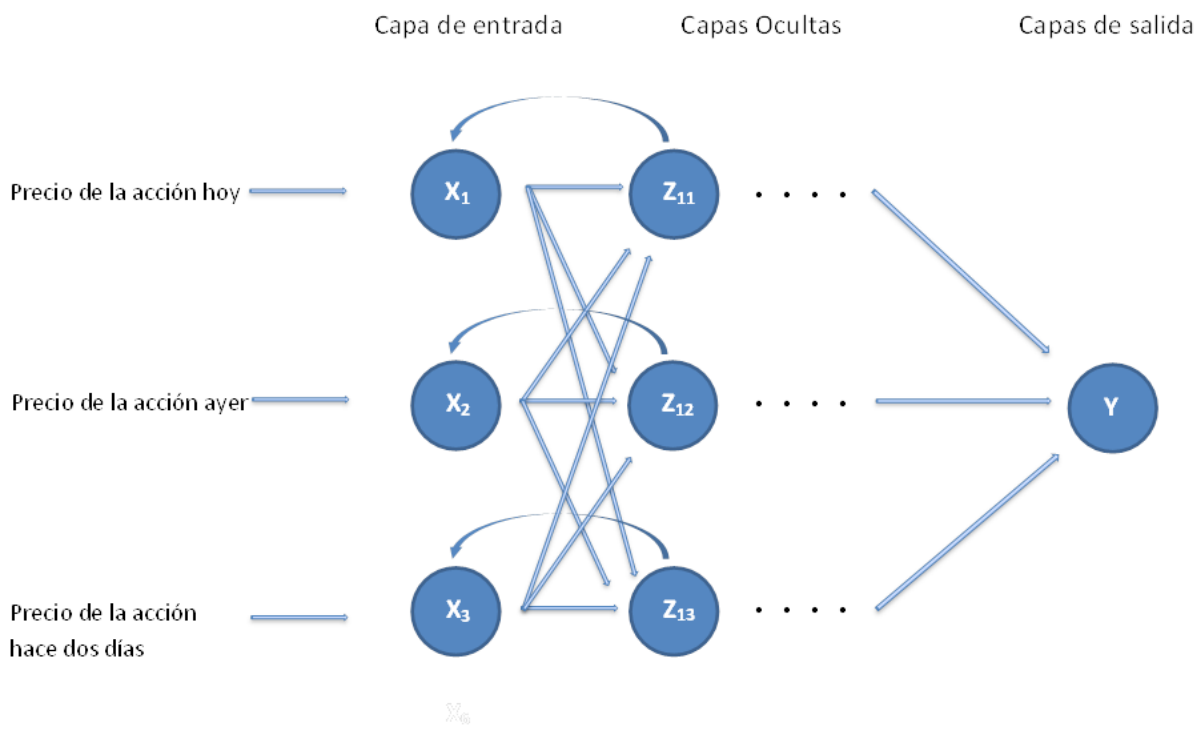

Figura 2.5: RNA localmente recurrente con Inputs univariados

La expresión 2.10 es la suma ponderada de los Inputs que posteriormente se evaluarán, ponderarán y sumarán sucesivamente hasta generar el Ouput de la red. Si se hace una agrupación en descenso comenzando por las conexiones hacia Y_i se tiene una especie de árbol en el que a cada rama se le asociará con un input en particular, así que la conexión representada por w_1 representará al árbol relacionado con X_1 y así sucesivamente.

Cada una de las ramas forma parte del output final en una cierta proporción, por lo que el Output de cada neurona oculta se considerará como el Output asociado a cada Input de la siguiente manera:

Z_1 será la porción de X_{t+1} relacionada con X_t

Z_2 será la porción de X_{t+1} relacionada con X_{t-1}

Z_3 será la porción de X_{t+1} relacionada con X_{t-2}

Y al aplicar la recurrencia los Inputs serán sustituidos de la misma manera

$X_t = Z_1$, la porción de X_{t+1} relacionada con X_t

$X_{t-1} = Z_2$, la porción de X_{t+1} relacionada con X_{t-1}

$X_{t-2} = Z_3$, la porción de X_{t+1} relacionada con X_{t-2}

De manera similar en los modelos Out-Feedback, los Inputs son reemplazados sólo por una fracción de predicción actualizando las condiciones iniciales.

2.4.5 Modelo multivariado localmente recurrente

Por último el modelo multivariado con recurrencia local se basa en la solución al problema univariado localmente recurrente en múltiples repeticiones para cada una de las variables, y como una variante del primer modelo multivariado también existen los mismos dos escenarios.

Para el modelo donde se presume independencia entre las variables predictoras la figura 2.6 presenta el diagrama de recurrencia donde los Inputs son sustituidos uno a uno en cada ejecución del algoritmo.

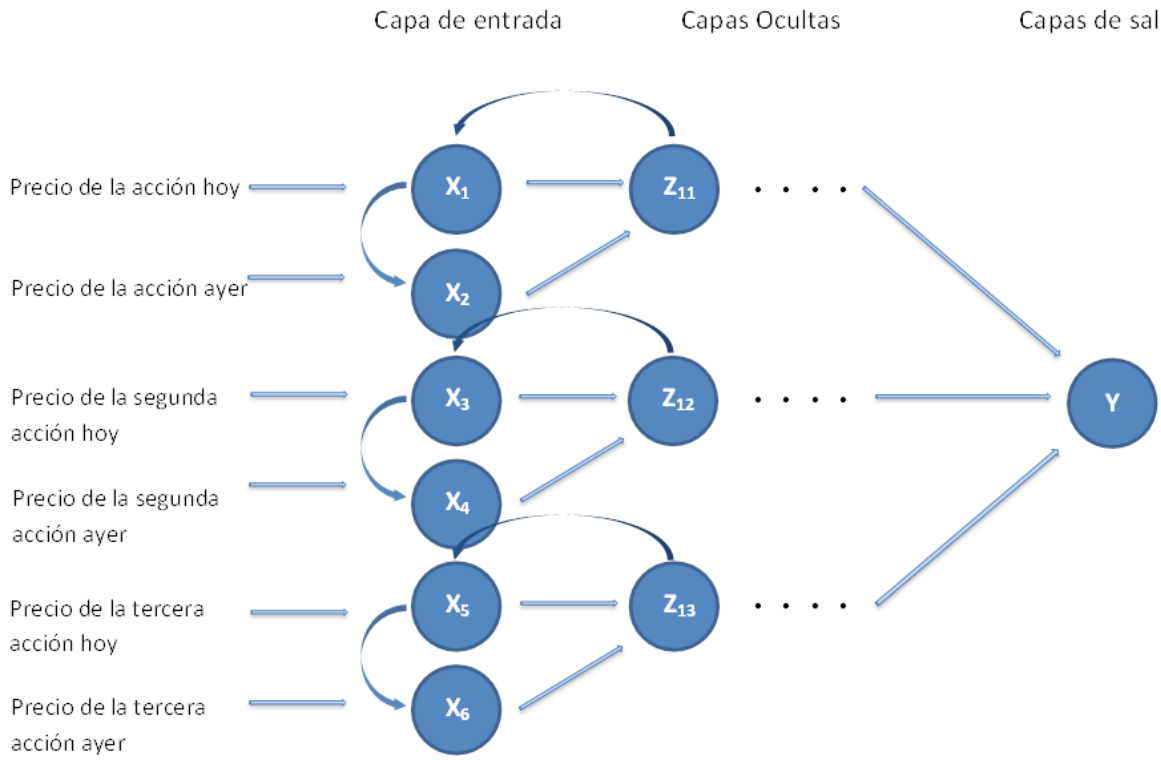

Figura 2.6: Tabla resumén de la arquitectura de una red neuronal

En la figura 2.7 se ejemplifica el modelo anterior considerando no independencia entre las variables independientes.

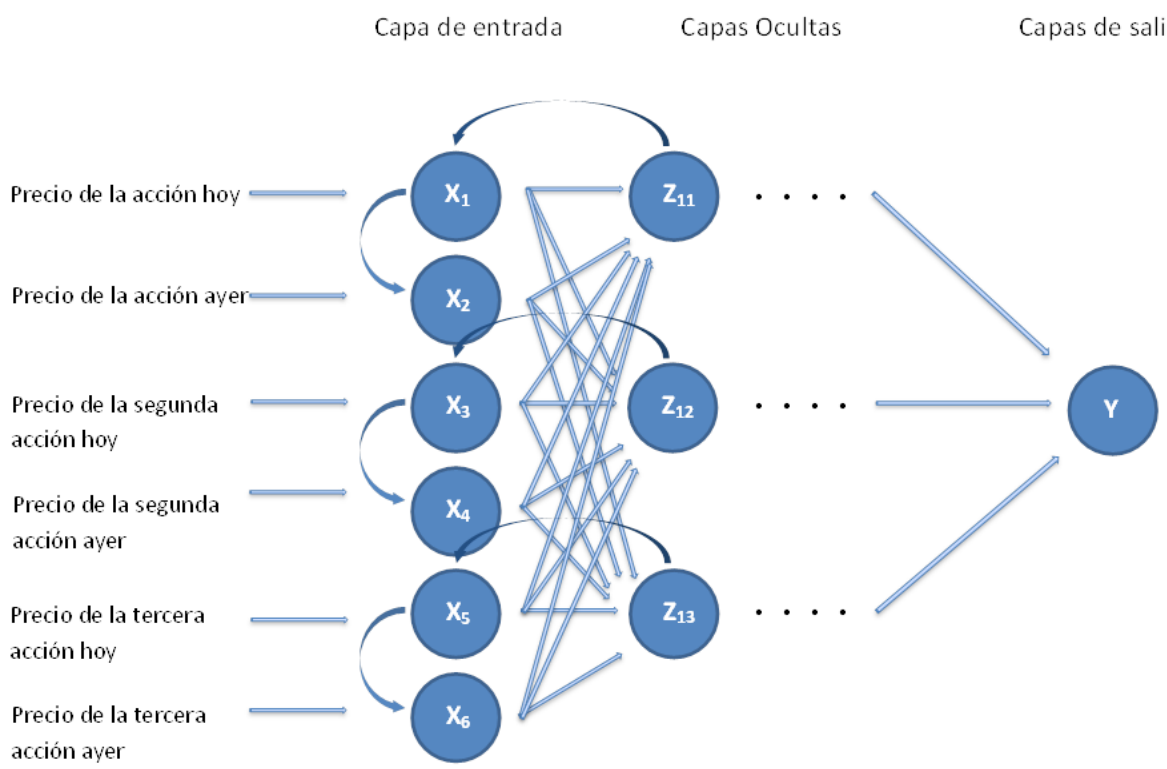

Figura 2.7: RNA localmente recurrente con Inputs univariados

Se han presentado distintos tipos de formato para los Inputs y distintas formas de aplicar la recurrencia, ahora será descrita la forma en que el algortimo Backpropagation aborda la recurrencia.

2.5 Entrenamiento

Se ha definido el valor de los Outputs de manera distinta en este capítulo que en comparación con el Capítulo 1, por ende también es necesario cambiar la definición del algoritmo de Backpropagation, el algoritmo alternativo que se propone usar es el llamado "Backpropagation Through Time", el algoritmo Backpropagation a través del tiempo. Este algoritmo es con frecuencia utilizado en el reconocimiento de patrones ya que permite implementar redes más complejas.

A continuación se expone el pseudocódigo del Algoritmo Backpropagation a través del tiempo, Algoritmo 2.

Este algoritmo permite manejar las sustituciones en los Inputs y al mismo tiempo todas las demás características del algorimo original expuesto en §1.6. El Algortimo

Algorithm 2 Algoritmo Backpropagation a través del tiempo

Inicializar los pesos sinápticos
while Condición de paro = NO **do**
  for Para cada Input **do**
    ▷ Feedforward
    Cada neurona de entrada recibe las señales externas y las propaga a la siguiente capa.
    Cada neurona oculta suma y evalua las señales en la función de activación y propaga esta señal a la siguiente capa.
    Cada neurona de salida suma y evalua las señales para generar el Output de la red.
    while Condición recurrente de paro = NO **do**
      ▷ Backpropagation
      Cada neurona de salida calcula el error entre el Output de la red y el Output asociado con el Input, y calcula el ajuste para los pesos sinápticos.
      Cada neurona oculta calcula el ajuste de los pesos sinápticos de acuerdo con el error propagado por la capa anterior.
      ▷ Recurrencia
      Sustituir los Inputs de la recurrencia con los Outputs calculados por la red.
      Actualizar la condición recurrente
    end while
    ▷ Actualización de parámetros
    Cada capa oculta y de salida actualiza los pesos sinápticos.
    Actualizar condición de paro.
  end for
end while

Backpropagation Through Time es un ajuste al algoritmo básico sumamente estudiado y se ha vuelto muy popular en el reconocimiento de patrones, [Werbos, 1990], [Watrous & Shastri, 1987]

Sus dos principales diferencias con respecto al algoritmo básico son la sustitución de los inputs dentro de la misma iteración de un Input y el cálculo de las actualizaciones en los pesos sinápticos.

Como fue mencionado en §2.4 habrá una sustitución en los Inputs en la red, cada modelo expuesto lo hace de manera distinta, pero la forma de implementar cada uno de éstos, no interviene en la estructura general del algoritmo. A pesar de que se calculan los errores asociados en cada una de las sustituciones, la actualización de los pesos se realizará una sola vez al final de todas ellas. El algoritmo calculará el ajuste a los pesos acorde con los errores los cuales se acumulan para posteriormente al final de las iteraciones se genere un promedio de estos ajustes y sea el que se aplique a los pesos de la red.

2.6 Desempeño

Lo mencionado en §1.6 han cubierto las principales características de una red neuronal y estas características son completamente modificables para ajustar el funcionamiento, las modificaciones que hagan en estos aspectos se verán reflejadas directamente en el desempeño de la red.

El criterio de paro del algoritmo es un indicador que se define para establecer el nivel de precisión de la clasificación o predicción. Este criterio está íntimamente relacionado con la función del error. El criterio estará construido con los errores asociados a cada uno de los patrones, por lo que el ajuste varía entre una iteración y otra.

Debido a lo anterior, se debe estar consciente que los errores se estabilizarán en un cierto número de iteraciones.

Puesto que se entrena a la red con el total de los patrones en cada una de las iteraciones los errores seguirán generándose de manera obvia, es por ésto que se propone establecer un intervalo para el error, normalmente entre 5% y 1%. Establecer este intervalo implicará que una vez los errores se encuentren dentro de él, el entrenamiento habrá terminado.

Por último se debe hacer la aclaración de la variabilidad en los resultados que dos implementaciones totalmente iguales puedan llegan a tener, esto provocado por la naturaleza

aleatorea de los valores de los parámetros al inicio del entrenamiento. Recordando de §1.8.4 la teoria del gradiente y la analogía de la caminata en la montaña, no importa el lugar en el que se comience la caminata siempre nos dirigirá hacia la cima en la mejor dirección.

2.7 Definición de la arquitectura

Se ha presentado propuesta de proceso previo a la implementación de la red neuronal y a manera de resumen se recopilará lo expuesto en §1 y también a lo largo de este capítulo para presentar en la siguiente tabla, 2.1, las principales características de una red. Desde el tipo de arquitectura , las características de cada capa de la red, el modo de entrenamiento, aprendizaje y condición de paro, también presenta las características de los inputs, las características anteriores se enlistarán según las necesidades de cada modelo.

Tabla 2.1: Tabla resumen de las características de una red neuronal artificial

Arquitectura	Monocapa/Multicapa
Capa de entrada	Numero de Neuronas
Capa oculta (1,..,n)	3 neuronas
Función de activación	Sigmoidal/Escalón (i.e)
Capa de salida	Numero de Neuronas
Función de activación	Sigmoidal/Escalón (i.e)
Entrenamiento	Backpropagation/ BP a través del tiempo
Aprendizaje	Supervisado/No supervisado
Función del error	ECM/Distancia Mahalanobis
Condición de paro	% de error / Número de Iteraciones
Temporalidad	Intervalo de tiempo (Dimensión del Imput)
Inputs	Univariado/Multivariado

En la tabla 2.2 se ejemplifica uno de los modelos que se retomarán en §3 y §4 en el formato anterior, la tabla contiene las características de una red neuronal multicapa con 2 capas de procesamiento. La red, como se lista en la tabla, está constituída por 3 neuronas en la capa de entrada, 3 en la única capa de procesamiento y una en su capa de salida, la función de activación para la capa oculta es la función Sigmoidal y la función Identidad para la capa de salida. Está entrenada de forma supervisada con el algoritmo Backpropagation y utiliza el error cuadrático medio como función del error. La condición de paro de la red está basada en 1000 iteraciones. Los Inputs son univariados con 3 retrasos y finalmente emplea la recurrencia de forma local.

Tabla 2.2: RNA sin elementos recurrentes con Inputs univariados independientes

Modelo	I
Arquitectura	Multicapa
Capa de entrada	3 neuronas
Capa oculta	3 neuronas
Función de activación	Sigmoidal
Capa de salida	1 neurona
Función de activación	Identidad
Entrenamiento	Backpropagation
Aprendizaje	Supervisado
Función del error	Error cuadrático medio
Condición de paro	1000 iteraciones
Temporalidad	Trimestral
Inputs	Univariados

En este punto se ha planteado una propuesta para el proceso previo a la implementación de la red neuronal con el fin de contemplar las características primordiales de una red neuronal así como los conceptos básicos del funcionamiento de la misma. Lo cual se puede definir en los pasos enlistados a lo largo del capítulo y sintetizar en una tabla (i.e. tabla 2.2). Se espera que la implementación de estos pasos y el uso de esta forma de resumen en tabla sea más sencillo el manejo de la información, elaboración del análisis y presentación de los resultados.

En el siguiente capítulo, §3, se empleará este modo de resumir la arquitectura de la red para ejemplificar los modelos de redes neuronales referentes al modelo de aplicación, éstos modelos corresponden al comportamiento del precio del jitomate en el mercado mexicano, debido a su importancia en la economía mexicana, así como en el aspecto alimenticio. Y como parte final de este trabajo en §4 se llevará a cabo el entrenamiento de las redes neuronales artificiales y serán presentados los resultados y conclusiones.

Capítulo 3

Caso de aplicación, predicción del precio del jitomate en México

En este capítulo serán expuestas la justificación y especificaciones del caso de aplicación de este trabajo de investigación. Se retomarán las ideas comentadas en el Capítulo II y se adecuarán las expresiones al comportamiento de los datos, del mismo modo se propondrá diferentes modelos para dar solución al problema de series de tiempo relacionado a la predicción del precio del jitomate en México.

3.1 Antecedentes

Hoy más que nunca la certidumbre acerca del comportamiento de la información es de vital importancia para las organizaciones, debido a que con base en información confiable se lleva acabo la planeación. Esta necesidad de certidumbre se refleja en la inversión de tiempo y recursos en la investigación, una tendencia en esta área es el modelado del comportamiento de los datos con el fin de encontrar patrones de conducta o tendencias que auxilien en el proceso de toma de decisiones. Un ejemplo de este modelado es la proyección de valores futuros por medio de series de tiempo o pronósticos, estos modelos proporcionarían valores a futuro con un cierto nivel de confianza que después la organización emplearía para tomar decisiones con mayor certeza.

Otra necesidad que cubre el modelado tanto en las organizaciones como en los órganos gubernamentales es la formulación de modelos económicos. El modelado económico pretende modelar la periodicidad en los negocios, tasas de inflación, flujo de capital y recursos,

planificación de oferta/demanda, impacto de estrategias de marketing, valor de indicadores, etc. Por citar algún ejemplo de ésto, es la proyección de la inflación y el crecimiento económico de un país.

El cálculo de la inflación se realiza con base en diferentes indicadores, por lo que para su cálculo se modela cada uno de estos indicadores por separado, uno de estos indicadores proviene del jitomate, que junto con otros productos agropecuarios como la zanahoria, cebolla, frijol o nopal también intervienen en el cálculo de la inflación. Es por esto que se tomó la decisión de elegir al jitomate como caso de aplicación.

A continuación expongo brevemente características del jitomate, su importancia en la economía mexicana y una propuesta para modelar su comportamiento.

3.2 El jitomate

El jitomate es una planta sudamericana originaria de las costas occidentales, forma parte de las hortalizas y es uno de los principales productos agrícolas en el mundo. Su fruto es de suma importancia para la sociedad contemporánea tanto en su economía como en el aspecto alimenticio [Díez, et al.,1996].

En la tabla 3.1 se muestra la composición química y nutrimental del jitomate, una más de las razones de su importancia [SAGARPA, 2010].

Tabla 3.1: Composición química y nutricional del jitomate

Composición química	
Agua	94%
Carbohidratos	3%
Fibra	1%
Proteínas	1%
Lípidos	0.30%
Otros	0.7%
Potasio	258 mg/100 g
Sodio	3 mg/100 g
Calcio	10 mg/100 g
Hierro	0.6 mg/100 g
Fósforo	24 mg/100 g
Vitamina C	26 mg/100 g
Vitamina A	207 mg/ 100 g
Tiamina	0006 mg/100 g
Riboflavina	0.04 mg/100 g
Niacina	0.028 mg/100 g

3.2.1 El jitomate en el mundo

El jitomate es el segundo producto en importancia dentro del grupo de las hortalizas, sólo por debajo de la papa, que está formado por: hortalizas frescas, tomates, sandías, coles, cebolla seca, pepino, berenjenas, melones y papas, entre otras. La producción del jitomate ha ido en aumento, mientras el número de hectáreas cultivadas disminuye, el total de producción aumenta debido a la optimización del área de cultivo y nuevas tecnologías en la producción.

El país con mayor producción de tomate en el mundo es China. México se encuentra 10 lugares por debajo de este país, sin embargo México se posiciona como el segundo exportador de jitomate en el mundo. En las figuras 3.1 y 3.2 se muestras las gráficas de superficie cosechada y producción de jitomate para los tres principales productores de jitomate: China, United States y e India.

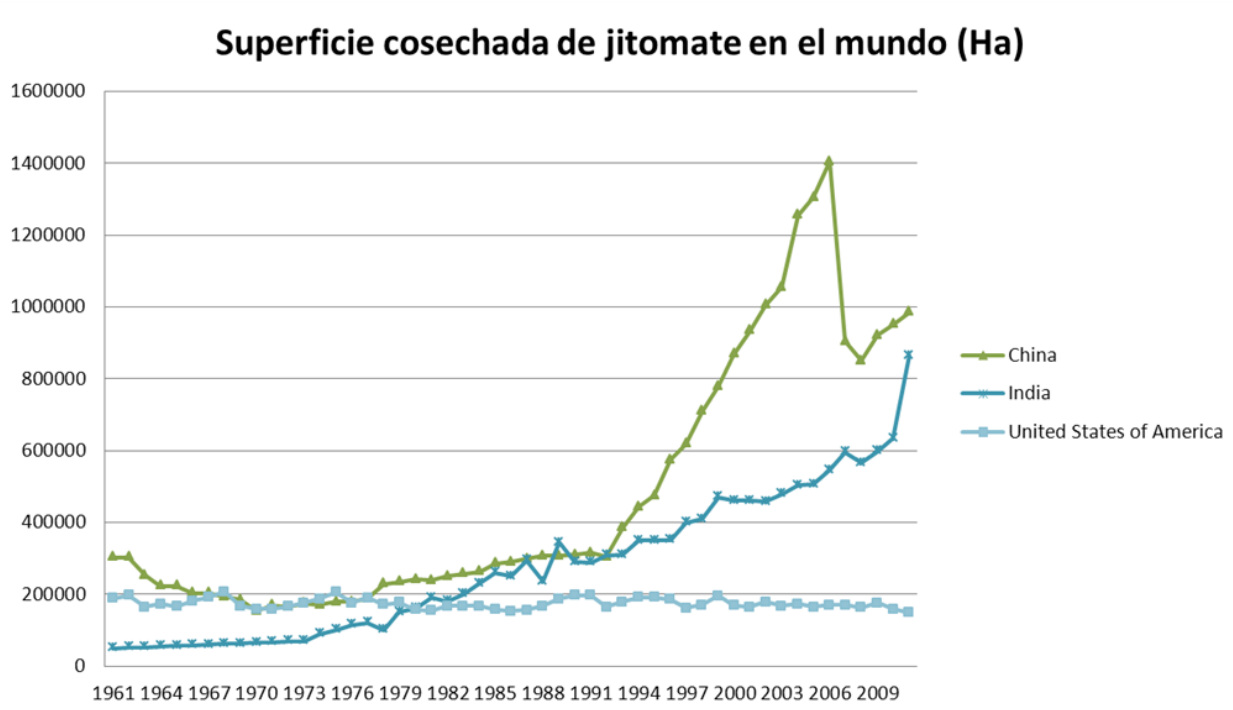

Figura 3.1: Histograma de superficie cosechada

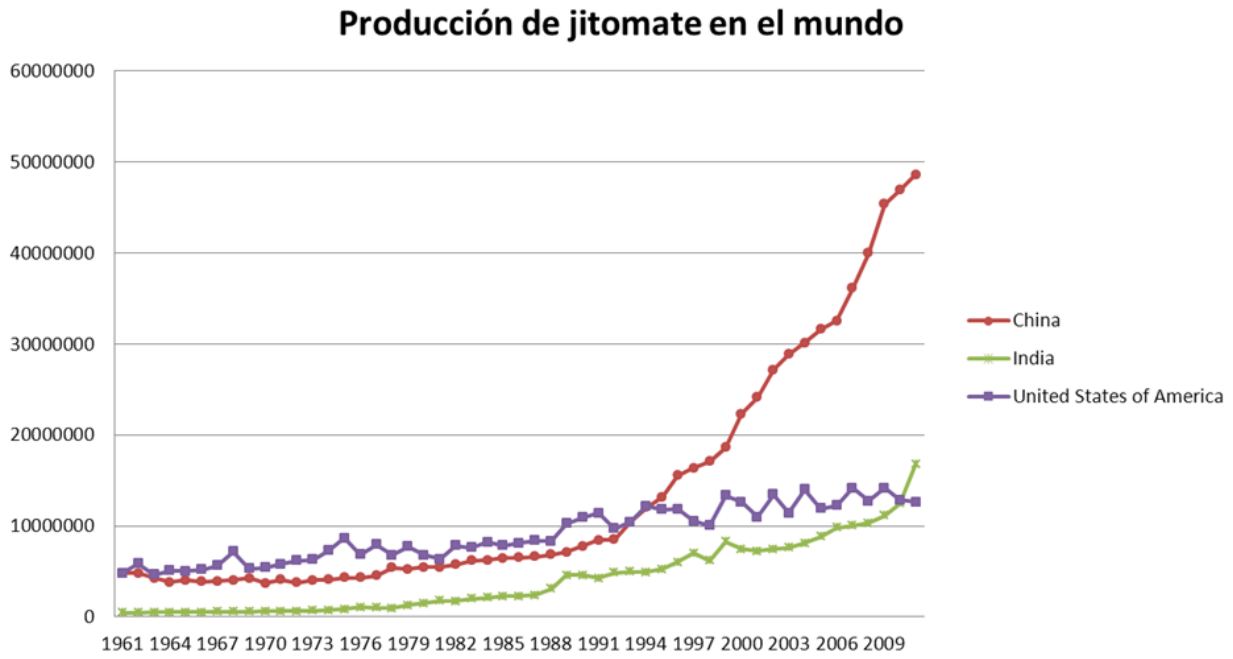

Figura 3.2: Histograma de los mayores productores de jitomate

3.2.2 El jitomate en México

Como se mencionó en la sección anterior, México está en el segundo lugar de exportación de jitomate, su principal destino de exportación es EE.UU. y su principal competidor nacional de EE.UU. son los estados de Florida y California, y en los competidores extranjeros figura Canadá en gran cantidad y Holanda, Guatemala y República Dominicana en porciones no tan significativas.

Una característica más del jitomate relacionada a México es la domesticación, fue aquí que se le nombra xictli del náhuatl ombligo y tomatl para tomate, que significa "tomate de ombligo". En México el jitomate juega un gran papel en la vida diaria del país. En principio, en el ámbito alimenticio el consumo percapita de jitomate es superior a los 25.0 Kg. al año, además de formar parte de la canasta básica y ser un pilar en la gastronomía nacional.

Por otra parte y como pieza clave en la actividad socioeconómica de México, se calcula que una producción de 75,000 hectáreas proveé empleo para 172 mil trabajadores de campo. Otro factor importante es la exportación de este producto agrícola, donde el jitomate abarca el 22% de las exportaciones agrícolas y genera más de 460 millones de dólares en divisas, datos estadísticos de [INEGI, 2002].

En México se produce jitomate durante todo el año, siendo en los meses de diciembre, enero, julio y agosto donde se registran los niveles más altos, ésto debido a la estacionalidad del jitomate en las diferentes regiones productoras en México.

Las fluctuaciones en el precio del jitomate afectan directamente en la economía de la familia mexicana, se mencionó también que el jitomate forma parte de la canasta básica y es con base a los productos que la conforman que se calcula el INPC, Índice Nacional de Precios al Consumidor.

Dentro del cáculo del Índice Nacional de Precios al Consumidor del Banco de México, el cual mide los cambios promedio en los precios a través del tiempo (INPC) el jitomate tiene una ponderación de 0.5647% con lo que se coloca en la primer posición de la clasificación "Legumbres y hortalizas" y tan sólo se encuentra en la posición 11 de la clasificación "Alimentos, Bebidas y Tabaco" del INPC, en lo general el jitomate se encuentra en la posición 42 de 292 artículos que forman parte de la canasta básica [BANXICO, 2011].

"En 2013 INEGI detalló que los productos con precios al alza en abril fueron transporte colectivo, tomate verde, huevo, gasolina de bajo octanaje, pollo, cerveza, vivienda propia, limón, papa y otros tubérculos y chile serrano.

En contraste, los que estuvieron a la baja son electricidad, servicios turísticos en paquete, cebolla, jitomate, transporte aéreo, pepino, productos para el cabello, calabacita, manzana y carne de cerdo, agregó en un comunicado." [Notimex, 2013]

Este comportamiento reafirma la prioridad y peso de ciertos artículos de la canasta básica, a pesar de tener alza en los precios de artículos con bajo peso en el INPC mientras los artículos con mayor peso se mantengan estables la inflación también lo hará.

"Apuntó que el índice de precios de la canasta básica de consumo presentó una caída de 0.17 por ciento en abril pasado, situando su tasa de crecimiento anual en 5.13 por ciento; mientras que en igual mes de 2012 fueron de 0.85 menos y 5.78 por ciento, en ese orden." [Notimex, 2013].

Lo anterior provocó que los INPC aumentaran 0.07% en abril de este año, pero a tasa anual la inflación se ubicó en 4.65 por ciento, por segundo mes por arriba de 4.0 por ciento, que es el nivel más significativo para el Banco de México como indicador de la economía.

La principal causa de estas fluctuaciones fue el aumento de las tarifas en el trasporte público y de productos de la canasta básica [15].

Además el jitomate forma parte de grandes negociaciones y tratados de los que México se ve beneficiado, en 2011 las divisas generadas por la exportaciones de jitomate a EE.UU fue de más de 1.8 millones de dólares, lo que provocó que en febrero de 2013, México y el vecino del norte firmaran un acuerdo de los precios mínimos al mayoreo del jitomate. El principal objetivo fue proveer condiciones de estabilidad y disminuir la competencia desleal entre los agricultores de ambas naciones [Secretaría de Economía, 2013].

3.3 El jitomate a través del tiempo

A continuación se muestran cifras en gráficas del comportamiento del jitomate en específico del precio promedio mensual, el área cultivada, volumen de producción y volumen de exportación, en específico del total de México, Figuras 3.3, 3.4, 3.5 y 3.6 respectivamente, junto con los comportamientos de cada una de las variables también se presentará el precio del jitomate con la intención de hacer algunas conjeturas simples acerca del las relaciones causales entre variables.

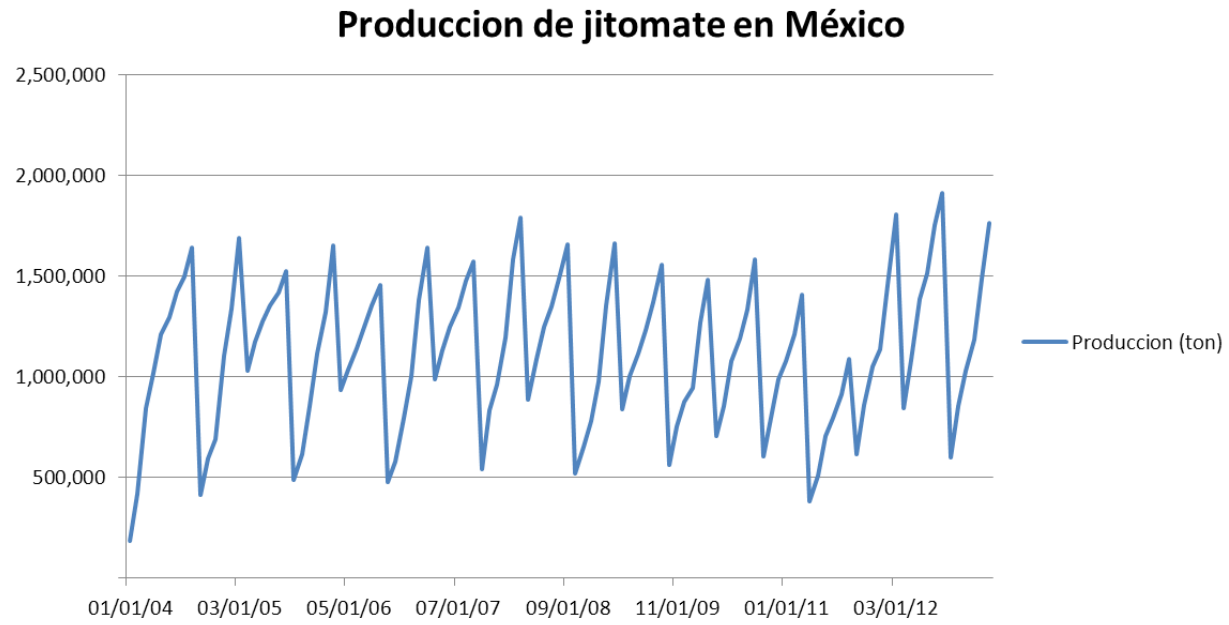

Figura 3.3: Histograma de producción de jitomate

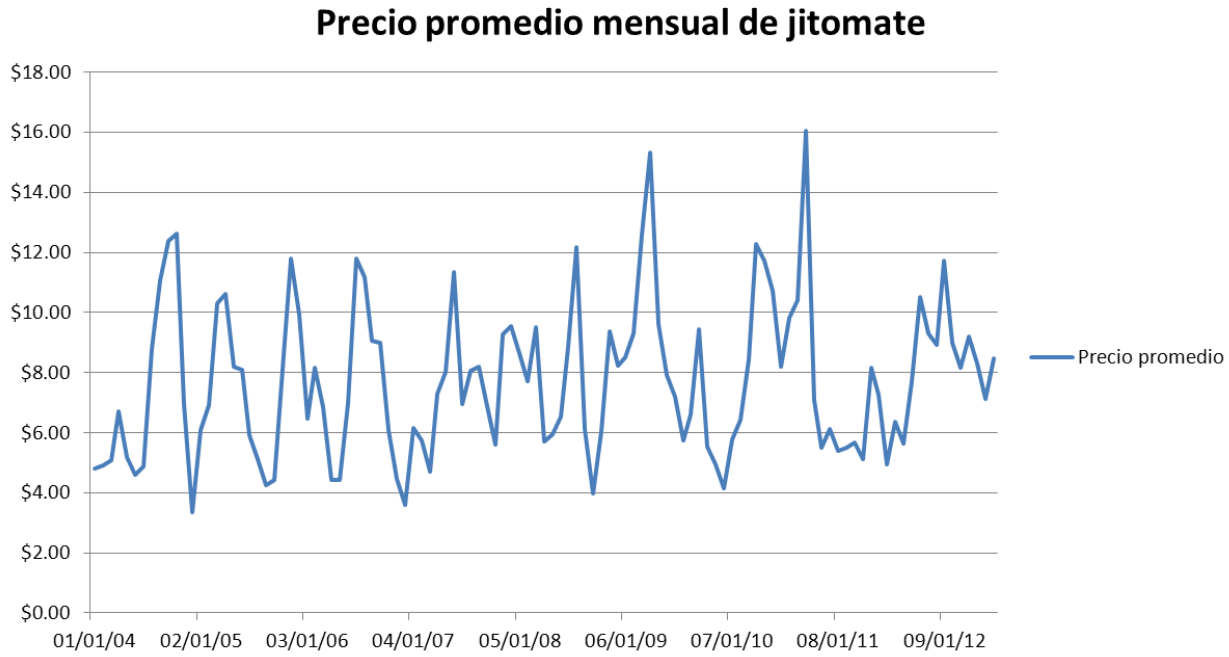

Figura 3.4: Histograma del precio de jitomate en la central de abastos D.F.

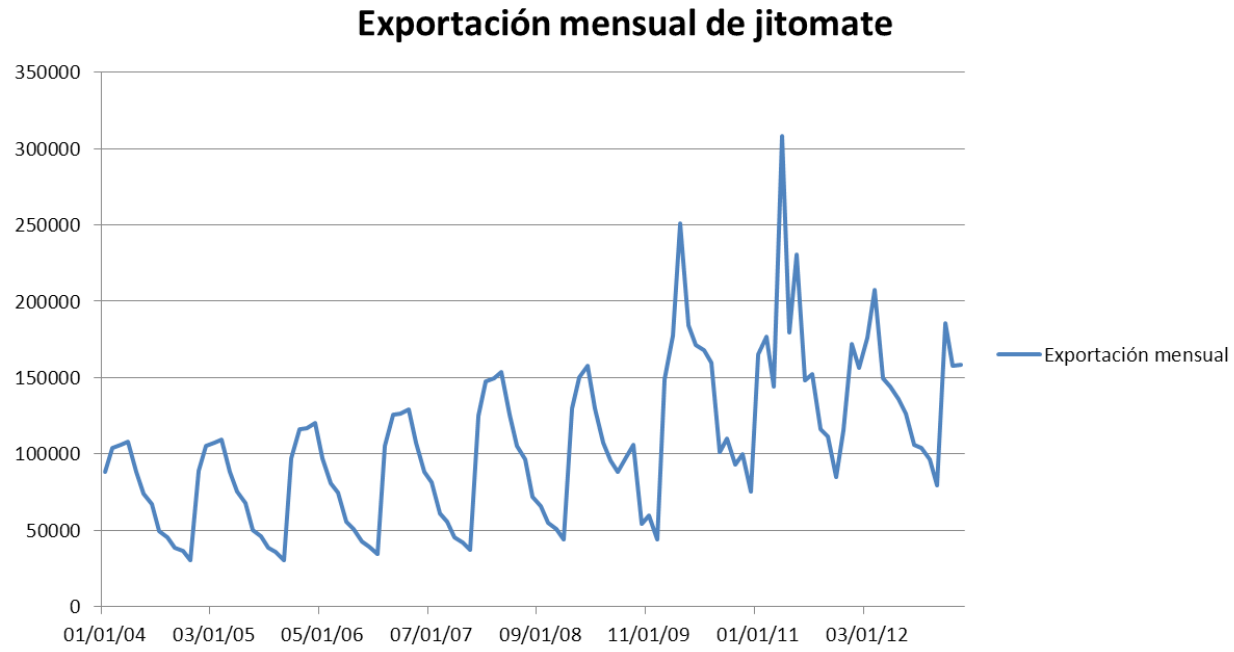

Figura 3.5: Histograma de exportación de jitomate

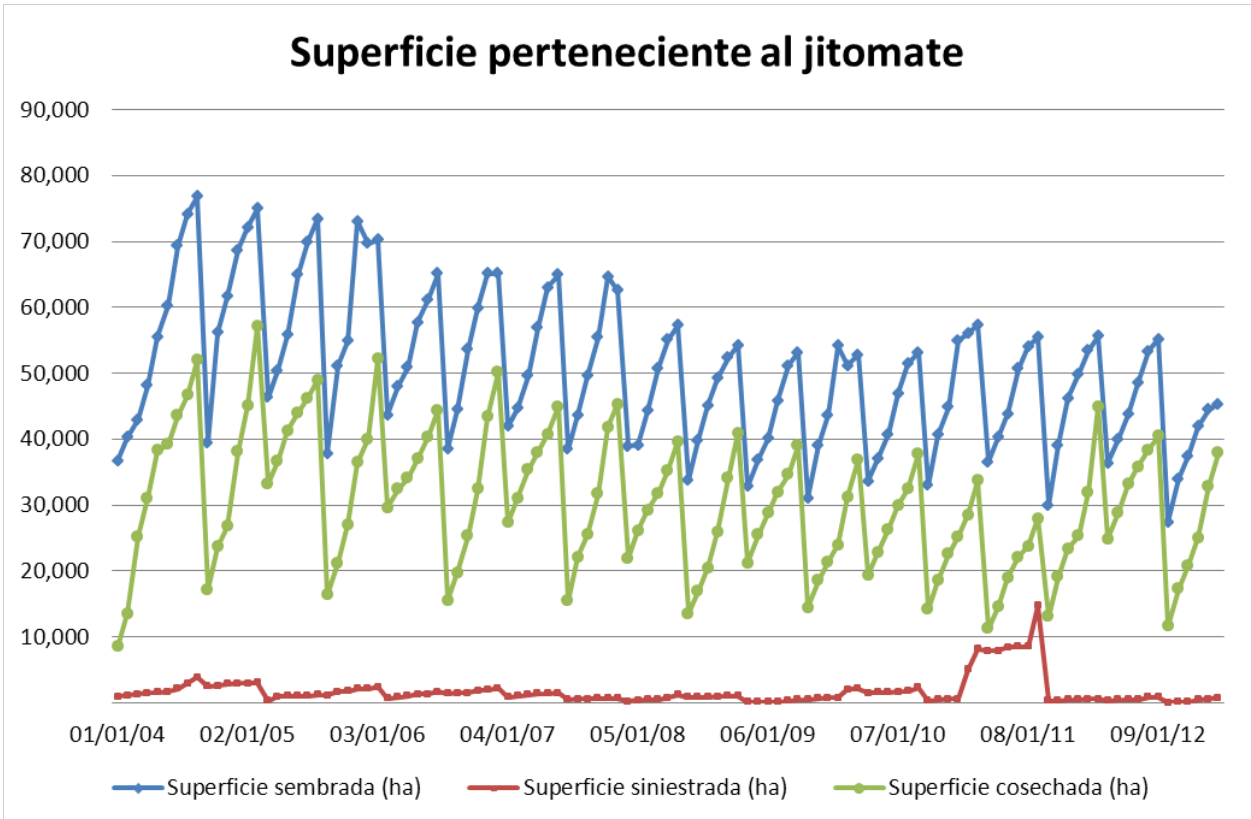

Figura 3.6: Histograma de superficie sembrada, siniestrada y cosechada

3.4 Preliminares

Si observamos las gráficas 3.4, 3.5 y 3.6 podemos tener una idea, a priori, del comportamiento de los datos, tendencias si es que las hay, valores máximos y valores mínimos.

Es evidente que se tiene una naturaleza cíclica anual con los puntos máximos en los meses de enero y febrero para el precio del jitomate, en general este comportamiento se repite en las demás características. Los datos tienen una tendencia de incremento a través del tiempo y la variabilidad de los datos aumenta del mismo modo.

3.5 Modelos para predecir el precio del jitomate en México

Se tiene en cuenta los factores que vuelven importante al jitomate para la economía mexicana y las capacidades de las redes neuronales expuestas en §2, hace sentido que aborde el problema de predecir el precio del jitomate.

En el pasado se han realizado distintos trabajos para dar solución a este problema empleando la metodología Box-Jenkins, como lo es el trabajo de Estrada y Priego [Estrada, et al., 2006] que propuso un modelo AR(1) como solución al problema o el trabajo de Gaspar Marroquín que implementa un modelo ARIMA(23,0,1) [Marroquín, et al., 2011].

Ambos trabajos están basados en un modelo autoexplicativo, es decir, que sólo utilizan el comportamiento del precio del jitomate en el pasado para modelar una ecuación que arroje valores a futuro.

Se utilizará más de un aspecto relacionado con el jitomate para la predicción del precio del jitomate. Se han utilizado las redes neuronales en otros aspectos del jitomate tales como la medición de ciertos indicadores del jitomate, por ejemplo en la estimación de la fotosíntesis foliar [Vargas, 2012].

La propuesta está basada en redes neuronales y toma dos tipos de recurrencia expuestos en §2.4. Los casos univariados serán de manera general equivalentes al modelo autoexplicativo de Estrada y Priego. Para los modelos multivariados se empleará la producción en toneladas, el volumen de exportación y la superficie sembrada de jitomate como variables explicativas.

Como se presentó en las gráficas de los comportamientos estas variables proporcionan

mayor cantidad de información para describir el comportamiento de la variable objetivo ya que suponemos una relacion de causalidad, además de ser las principales variables a disposición.

3.5.1 Modelos univariados

Tanto para los modelos univariados como para los modelos multivariados primero se propone un modelo base el cual no contiene el elemento de recurrencia con el fin de tener una referencia acerca del ajuste de todos los modelos.

El modelo univariado con recurrencia local utilizará el comportamiento anterior del precio del jitomate para predecir los valores futuros. Debido a que las observaciones son mensuales, se utilizarán 3 observaciones con el fin de reflejar un comportamiento trimestral. De modo que el valor de $X(t+1)$ estará dado por la siguiente expresión,

$$X(t+1) = f(X(t), X(t-1), X(t-2)) \tag{3.1}$$

donde $X(t)$ es el precio promedio futuro del jitomate estará en función por los valores del trimestre inmediato anterior.

Tabla 3.2: Tabla resumen del modelo univariado con recurrencia local

Modelo	II
Arquitectura	Multicapa
Capa de entrada	3 neuronas
Capa oculta	3 neuronas
Función de activación	Sigmoidal
Capa de salida	1 neurona
Función de activación	Identidad
Entrenamiento	Backpropagation a través del tiempo
Aprendizaje	Supervisado
Función del error	Error cuadrático medio
Condición de paro	1000 iteraciones
Temporalidad	Trimestral
Inputs	Univariados con recurrencia local

En la tabla 2.2 se listan las características del modelo base con Inputs univariados mientras que en las tablas 3.2 y 3.3 el mismo modelo neuronal con recurrencia local y por el final, respectivamente.

Tabla 3.3: Tabla resumen del modelo univariado con recurrencia por el final

Modelo	III
Arquitectura	Multicapa
Capa de entrada	3 neuronas
Capa oculta	3 neuronas
Función de activación	Sigmoidal
Capa de salida	1 neurona
Función de activación	Identidad
Entrenamiento	Backpropagation a través del tiempo
Aprendizaje	Supervisado
Función del error	Error cuadrático medio
Condición de paro	1000 iteraciones
Temporalidad	Trimestral
Inputs	Univariados con recurrencia por el final

3.5.2 Modelos multivariados

En el caso del modelo multivariado se emplearán tres retrasos de cada una de las variables precio promedio, producción en millones toneladas, el volumen de exportación en millones de toneladas y la superficie sembrada de jitomate, de forma que el valor de $X(t+1)$ esta dado por,

$$\begin{aligned} X(t+1) = f(&X(t), X(t-1), X(t-2), Y(t), Y(t-1), Y(t-2), \\ &Z(t), Z(t-1), Z(t-2), W(t), W(t-1), W(t-2)) \end{aligned} \tag{3.2}$$

Siendo $X(t)$ los valores para el precio del jitomate, $Y(t)$ el volumen de producción, $Z(t)$ el volumen exportado y $W(t)$ la superficie sembrada.

Para enlistar las características de los modelos con Inputs independientes se hace referencia a las tablas 3.4, 3.5 y 3.6

Tabla 3.4: Tabla resumen del modelo multivariado sin elementos de recurrencia con independencia entre las variables

Modelo	IV
Arquitectura	Multicapa
Capa de entrada	12 neuronas
Capa oculta	3 neuronas
Función de activación	Sigmoidal
Capa de salida	1 neurona
Función de activación	Identidad
Entrenamiento	Backpropagation
Aprendizaje	Supervisado
Función del error	Error cuadrático medio
Condición de paro	1000 iteraciones
Temporalidad	Trimestral
Inputs	Multivariados independientes

Tabla 3.5: Tabla resumen del modelo multivariado con recurrencia local con independencia entre las variables

Modelo	V
Arquitectura	Multicapa
Capa de entrada	3 neuronas
Capa oculta	3 neuronas
Función de activación	Sigmoidal
Capa de salida	1 neurona
Función de activación	Identidad
Entrenamiento	Backpropagation a través del tiempo
Aprendizaje	Supervisado
Función del error	Error cuadrático medio
Condición de paro	1000 iteraciones
Temporalidad	Trimestral
Inputs	Multivariados independientes con recurrencia local

Tabla 3.6: Tabla resumen del modelo multivariado con recurrencia local

Modelo	VI
Arquitectura	Multicapa
Capa de entrada	3 neuronas
Capa oculta	3 neuronas
Función de activación	Sigmoidal
Capa de salida	1 neurona
Función de activación	Identidad
Entrenamiento	Backpropagation a través del tiempo
Aprendizaje	Supervisado
Función del error	Error cuadrático medio
Condición de paro	1000 iteraciones
Temporalidad	Trimestral
Inputs	Multivariados independientes con recurrencia por el final

Los últimos modelos en donde se presupone Inputs no independientes y sus características se muestran en las tablas 3.7, 3.8 y 3.9.

Tabla 3.7: Tabla resumen del modelo multivariado sin elementos de recurrencia con independencia entre las variables

Modelo	VII
Arquitectura	Multicapa
Capa de entrada	3 neuronas
Capa oculta	3 neuronas
Función de activación	Sigmoidal
Capa de salida	1 neurona
Función de activación	Identidad
Entrenamiento	Backpropagation a través del tiempo
Aprendizaje	Supervisado
Función del error	Error cuadrático medio
Condición de paro	1000 iteraciones
Temporalidad	Trimestral
Inputs	Multivariados no independientes

Tabla 3.8: Tabla resumen del modelo multivariado con recurrencia por el final con independencia entre las variables

Modelo	VIII
Arquitectura	Multicapa
Capa de entrada	3 neuronas
Capa oculta	3 neuronas
Función de activación	Sigmoidal
Capa de salida	1 neurona
Función de activación	Identidad
Entrenamiento	Backpropagation a través del tiempo
Aprendizaje	Supervisado
Función del error	Error cuadrático medio
Condición de paro	1000 iteraciones
Temporalidad	Trimestral
Inputs	Multivariados no independientes con recurrencia local

Tabla 3.9: Tabla resumen del modelo multivariado con recurrencia por el final

Modelo	IX
Arquitectura	Multicapa
Capa de entrada	3 neuronas
Capa oculta	3 neuronas
Función de activación	Sigmoidal
Capa de salida	1 neurona
Función de activación	Identidad
Entrenamiento	Backpropagation a través del tiempo
Aprendizaje	Supervisado
Función del error	Error cuadrático medio
Condición de paro	1000 iteraciones
Temporalidad	Trimestral
Inputs	Multivariados no independientes con recurrencia por el final

Estos nueve modelos son diseños de redes neuronales para abordar el problema de la predicción del valor futuro del jitomate, el cual tiene un impacto en la economía mexicana. En el capítulo siguiente se llevará a cabo el entrenamiento de los modelos neuronales

anteriormente propuestos y se presentarán los resultados. Serán tomados como referencia los modelos sin elementos de recurrencia y el modelo ARIMA (de Estrada y Priego mencionados en §3.4) como referente.

Capítulo 4

Entrenamiento y resultados

En este último capítulo se llevará a cabo el entrenamiento de los modelos propuestos en el capítulo III y se concluirá con base en los resultados. El entrenamiento se realizará repetidas veces con el mismo modelo para obtener mayor información de la eficiencia de los modelos propuestos en §3 y serán tomados como referencia el modelo ARIMA de Estrada y Priego mencionado en §3.4.

4.1 Antecedentes

En el Capítulo 1 se mencionó el modelo ARIMA del trabajo de Estrada y Priego [Estrada & Priego, 2006] , el cual atiende al mismo problema de predecir el valor para el precio del jitomate, este modelo tiene una confiabilidad del 76% para el primer pronóstico.

Mientras que en el capítulo 3 se propusieron 9 modelos neuronales que al ser entrenados se ajustan a los datos históricos del precio del jitomate y son capaces de predecir los precios futuros de éste. Es de esperar que la propuesta hecha de los nueve modelos, y habiendo tomado en cuenta los conceptos presentados en §1 y §2, al llevar a cabo el entrenamiento de las redes neuronales se obtengan resultados parecidos o mejores.

4.2 Entrenamiento de la red

Se implementó el algoritmo Backpropagation y Backpropagation a través del tiempo utilizando el lenguaje de programación VisualBasic para Aplicaciones (VBA 7) para Microsoft Office Excel 2010. Esta implementación permitió diseñar las nueve arquitecturas

de los modelos de redes neuronales y llevar acabo su entrenamiento.

A continuación se presentan los nueve modelos por grupos del tipo de Inputs. En la tabla 4.1 se muestran los modelos con Inputs univariados, en la tabla 4.2 los modelos con Inputs multivariados donde se presume independencia funcional y por último en la tabla 4.3 los modelos multivariados sin independencia funcional.

Tabla 4.1: Modelos univariados

Modelo	I	II	III
Arquitectura	Multicapa	Multicapa	Multicapa
Capa de entrada	3 neuronas	3 neuronas	3 neuronas
Capa Oculta	3 neuronas	3 neuronas	3 neuronas
Función de activación	Sigmoidal	Sigmoidal	Sigmoidal
Capa de salida	1 neurona	1 neurona	1 neurona
Función de activación	Identidad	Identidad	Identidad
Entrenamiento	**Backpropagation**	**Backpropagation a través del tiempo**	**Backpropagation a través del tiempo**
Aprendizaje	Supervisado	Supervisado	Supervisado
Función del error	Error cuadrático medio	Error cuadrático medio	Error cuadrático medio
Condición de paro	1000 iteraciones	1000 iteraciones	1000 iteraciones
Temporalidad	Trimestral	Trimestral	Trimestral
Inputs	**Univariados**	**Univariados con recurrencia local**	**Univariados con recurrencia por el final**

Tabla 4.2: Modelos multivariados independientes

Modelo	IV	V	VI
Arquitectura	Multicapa	Multicapa	Multicapa
Capa de entrada	12 neuronas	12 neuronas	12 neuronas
Capa Oculta (1)	12 neuronas	12 neuronas	12 neuronas
Función de activación	Sigmoidal	Sigmoidal	Sigmoidal
Capa Oculta (2)	3 neuronas	3 neuronas	3 neuronas
Función de activación	Sigmoidal	Sigmoidal	Sigmoidal
Capa de salida	1 neurona	1 neurona	1 neurona
Función de activación	Identidad	Identidad	Identidad
Entrenamiento	**Backpropagation**	**Backpropagation a través del tiempo**	**Backpropagation a través del tiempo**
Aprendizaje	Supervisado	Supervisado	Supervisado
Función del error	Error cuadrático medio	Error cuadrático medio	Error cuadrático medio
Condición de paro	1000 iteraciones	1000 iteraciones	1000 iteraciones
Temporalidad	Trimestral	Trimestral	Trimestral
Inputs	**Multivariados independientes**	**Multivariados independientes con recurrencia local**	**Multivariados independientes con recurrencia por el final**

Tabla 4.3: Modelos multivariados no independientes

Modelo	VII	VIII	IX
Arquitectura	Multicapa	Multicapa	Multicapa
Capa de entrada	12 neuronas	12 neuronas	12 neuronas
Capa Oculta	3 neuronas	3 neuronas	3 neuronas
Función de activación	Sigmoidal	Sigmoidal	Sigmoidal
Capa de salida	1 neurona	1 neurona	1 neurona
Función de activación	Identidad	Identidad	Identidad
Entrenamiento	**Backpropagation**	**Backpropagation a través del tiempo**	**Backpropagation a través del tiempo**
Aprendizaje	Supervisado	Supervisado	Supervisado
Función del error	Error cuadrático medio	Error cuadrático medio	Error cuadrático medio
Condición de paro	1000 iteraciones	1000 iteraciones	1000 iteraciones
Temporalidad	Trimestral	Trimestral	Trimestral
Inputs	**Multivariados no independientes**	**Multivariados no independientes con recurrencia local**	**Multivariados no independientes con recurrencia por el final**

Los datos del precio promedio del jitomate fueron utilizados para esto, en la Central de abastos del Distrito Federal, de enero del 2004 a septiembre del 2012 para un total de 105 observaciones.

Estas 105 observaciones se agruparon en Inputs de tres datos, el primer grupo de observaciones estuvo formado por enero, febrero y marzo del 2004, el segundo grupo se conformó por febrero, marzo y abril del 2004 y así sucesivamente.

El primer grupo de observaciones tuvo como Output asociado el precio del jitomate en abril del 2004, el segundo grupo con el precio en mayo y asi sucesivamente, de manera general se tuvo que los Inputs correspondían a los tres meses anteriores al Output.

La misma dinámica aplica para los modelos multivariados donde los Inputs se componen del precio promedio, volumen de producción y volumen de exportación, el Outoput asociado es el precio del jitomate.

4.3 Simulación del modelo

Con la intención de obtener mejores resultados en el entrenamiento fueron realizadas distintas simulaciones ajustando la tasa de aprendizaje, Learning Rate. Se llevo a cabo el entrenamiento treinta ocasiones para tener suficiente información acerca del ajuste a los datos verdaderos, también se calculó el error asociado entre el Ouput de la red y el dato real, este error se calculó a través del error cuadrático medio.

Las 105 observaciones se sometieron a 1000 iteraciones para llevar acabo el entrenamiento, a continuación se muestran los errores promedio de cada modelo y su respectiva tasa de aprendizaje.

Tabla 4.4: Errores de modelos univariados

Learning Rate\Modelo	I	II	III
0.01	26.53%	29.25%	32.96%
0.1	30.63%	21.47%	**25.39%**
0.3	27.09%	19.62%	32.66%
0.5	29.09%	**18.83%**	41.29%
0.7	25.96%	20.91%	52.82%
0.9	**25.83%**	20.30%	66.96%
1.0	25.83%	21.28%	71.23%

Como se observa en la tabla 4.4, los errores mínimos para los modelos univariados, respectivamente, son: 25.83%, 18.83%, 25.39%.

Tabla 4.5: Errores de modelos multivariados independientes

Learning Rate\Modelo	IV	V	VI
0.01	29.56%	27.19%	25.97%
0.1	27.41%	24.09%	**23.72%**
0.3	28.17%	24.25%	23.97%
0.5	25.87%	24.60%	25.29%
0.7	25.56%	25.11%	25.80%
0.9	25.23%	**23.11%**	26.11%
1.0	**25.07%**	25.52%	26.91%

Los errores mínimos para los modelos multivariados donde se presume independencia entre las variables explicativas son: 25.07%, 23.11%, 23.72%, tabla 4.5.

En resumen, se identifica un patrón dentro de los modelos que habla del mejor funcionamiento de las redes con recurrencia local, en comparación con la recurrencia por el final o sin recurrencia.

Tabla 4.6: Errores de modelos multivariados no independientes

Learning Rate\Modelo	VII	VIII	IX
0.01	25.86%	27.35%	27.20%
0.1	25.73%	22.73%	22.72%
0.3	26.12%	**20.57%**	21.26%
0.5	25.65%	21.18%	**21.24%**
0.7	25.46%	21.92%	23.39%
0.9	**25.45%**	23.88%	24.30%
1.0	25.45%	23.13%	28.26%

Por último, para los modelos donde se presume no independencia entre variables los errores son: 25.45%, 20.57%, 21.24%, como se observa en la tabla 4.6.

4.4 Resultados

El modelo con el menor error es el modelo univariado con recurrencia local, modelo II. Este modelo permite la predicción del precio del jitomate al 81.17% de confianza, el comportamiento del precio del jitomate y los valores predichos por el modelo de red neuronal se muestran en la figura 4.1. Así mismo los valores dados por el modelo ARIMA se muestran en la figura 4.2.

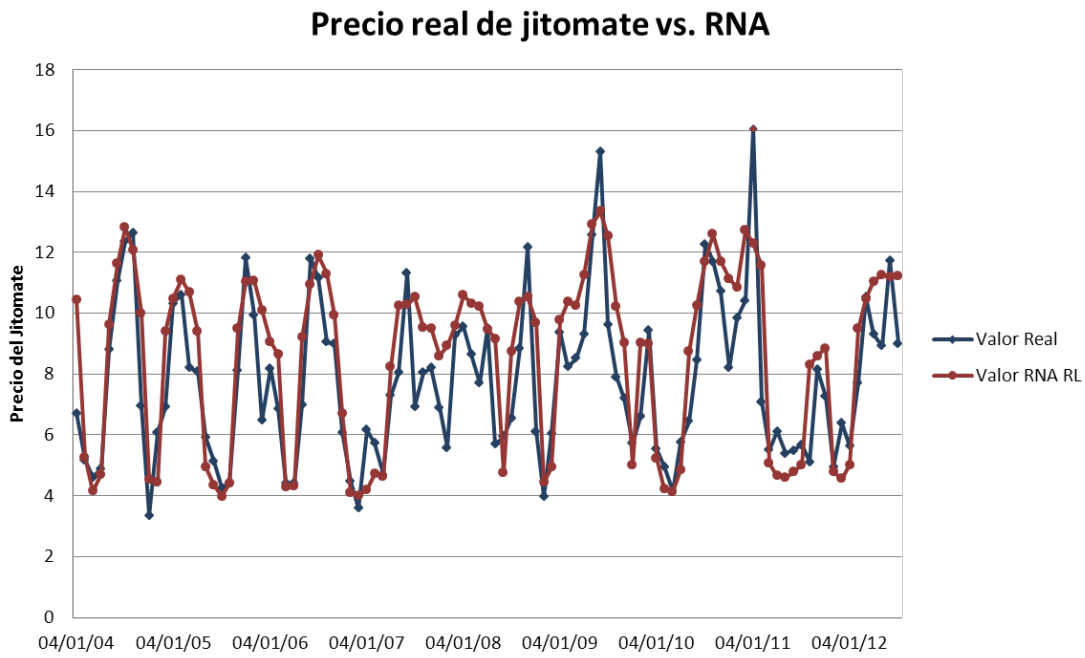

Figura 4.1: Valores reales del precio del jitomate y valores predichos por el modelo neuronal.

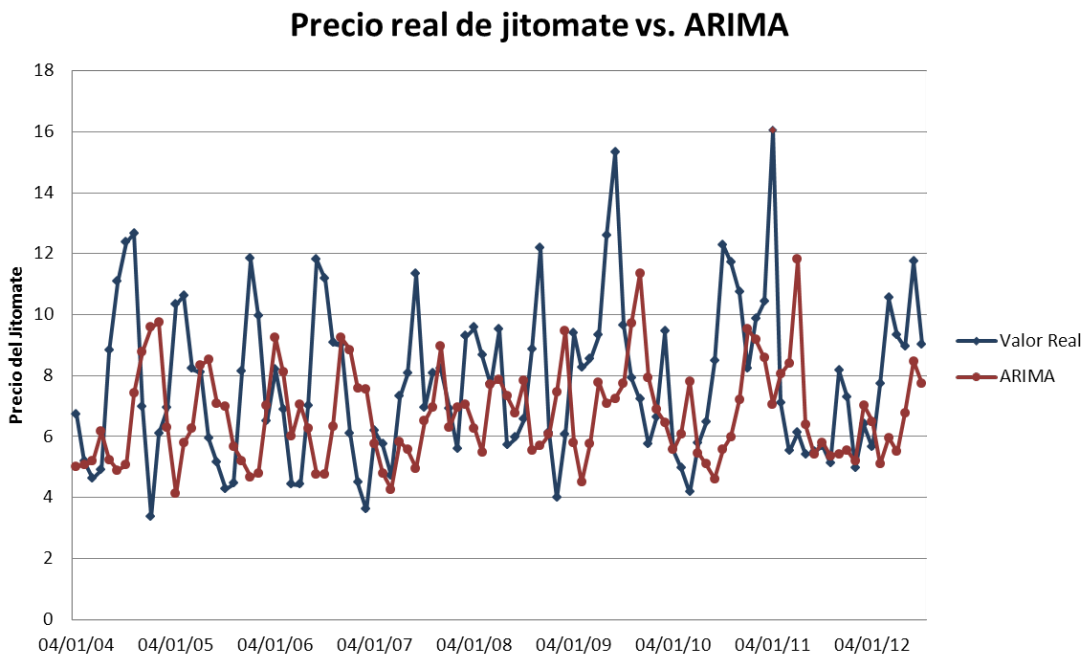

Figura 4.2: Valores reales actualizados y valores predichos por el modelo ARIMA.

En la figura 4.3 se observan las tres series de valores: Precio del jitomate, Valores predichos por la Red Neuronal y Valores predichos por el modelo ARIMA, y se puede concluir de manera gráfica que el comportamiento que más asemeja a los valores reales es el modelo neuronal.

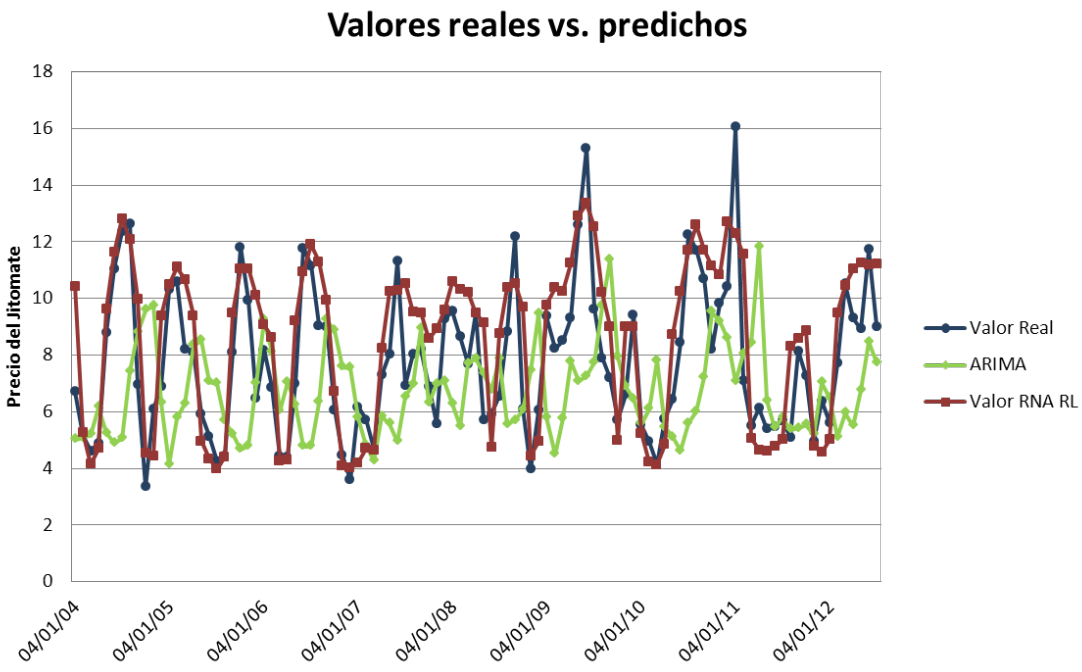

Figura 4.3: Valores reales actualizados y valores predichos.

El la figura 4.4 se muestra la distribución de los errores con respecto a ambos modelos, se observa que el denso de los errores para el modelo de red neuronal se encuentra entre 10% y 30% mientras que los errores para el modelo ARIMA los decienden lentamente hasta un 70% inclusive se tienen errores del 186%, ésto quiere decir que los errores producidos por la red neuronal se estabilizan en cierto punto del entrenamiento.

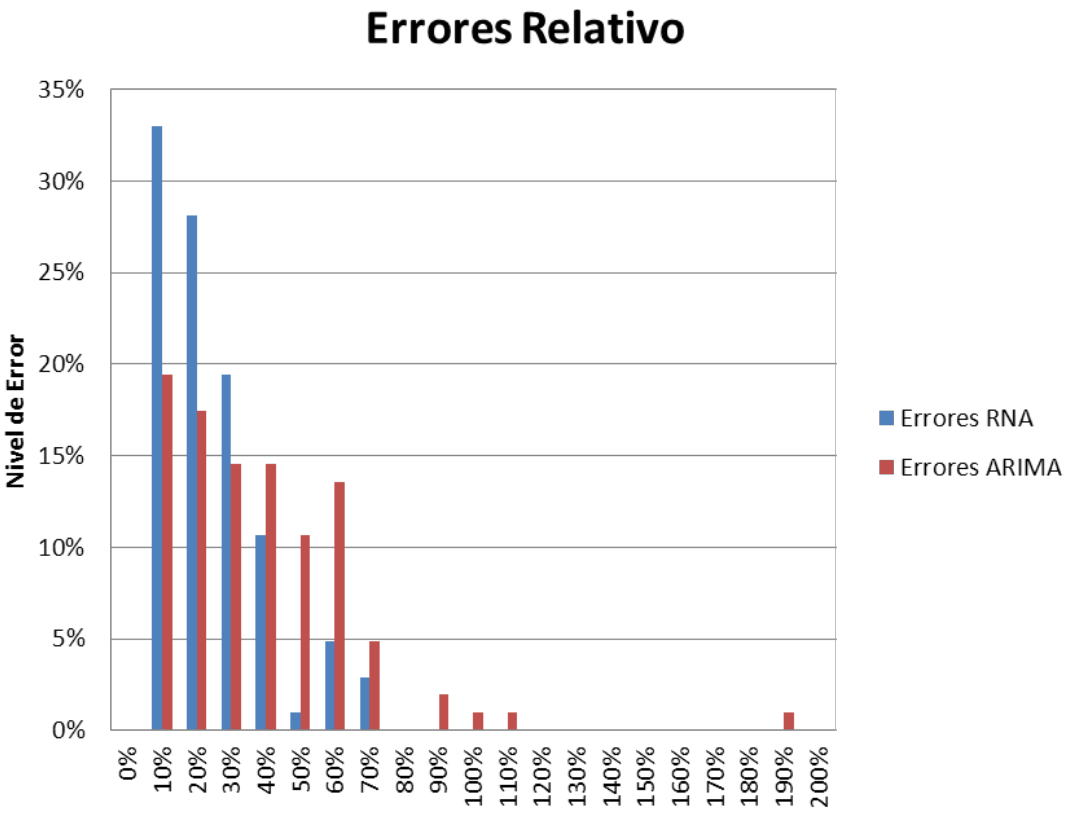

Figura 4.4: Distribución de los errores

Los modelos con recurrencia presentaron un mejor ajuste a los datos que los modelos sin recurrencia en las tres variedades presentadas, incluso presentaron un mejor ajuste que el modelo ARIMA, siendo la recurrencia local el mejor modelo de los tres escenarios.

A diferencia de la recurrencia por el final donde se sustituye un solo parámetro en los Inputs de la red, la recurrencia local reemplazó todos los inputs de la red lo que proporcionó más información predicha a corregir.

El principal objetivo de los modelos neuronales propuestos es la predicción del precio del jitomate a futuro por lo que es lógico que se predigan valores que no se hayan empleado durante el entrenamiento. Se predijeron los valores de Octubre 2012 a Julio 2013, los datos que se obtuvieron en base al modelo neuronal y al modelo ARIMA se muestran en la figura 4.5, de la misma manera el modelo neuronal se ajusta más al comportamiento real incluso a los picos en la serie de valores reales.

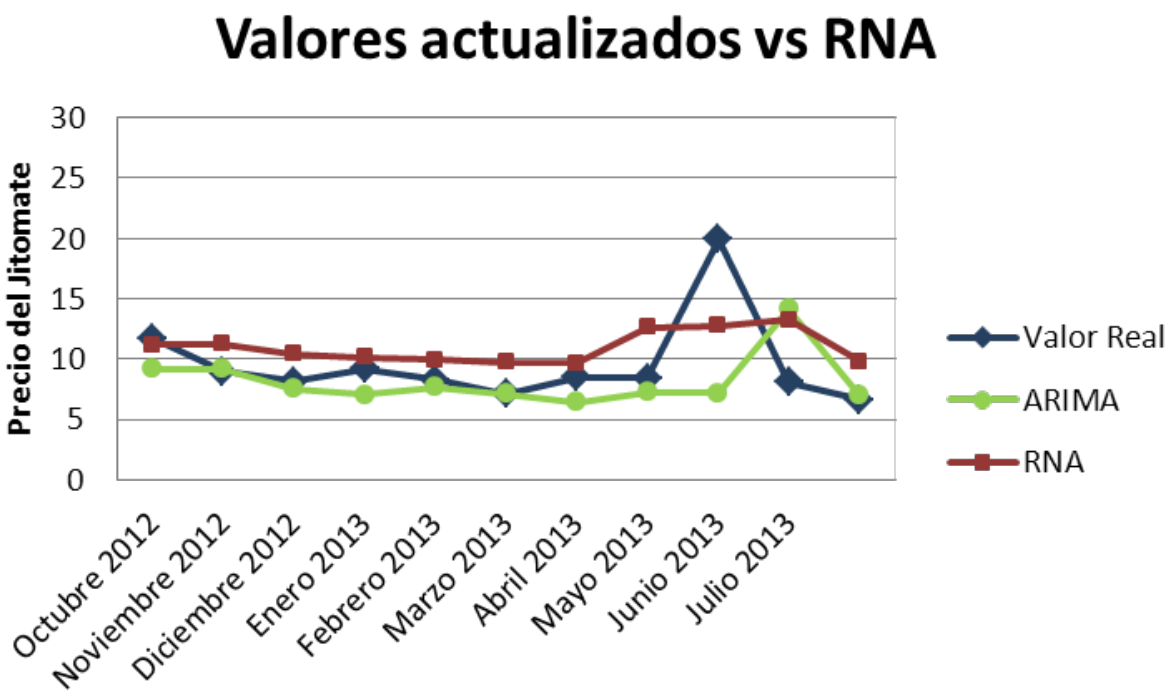

Figura 4.5: Pronóstico del precio actualizado del jitomate

En la figura 4.6 se presentan los valores reales, los valores generados por el modelo neuronal y los errores asociados a cada uno de los valores predichos.

Tabla 4.7: Errores en valores pronosticados

Mes	Valor Real	Output Red Neuronal	Error Absoluto
Septiembre 2012	$ 11.74	$ 10.21	13%
Octubre 2012	$ 9.01	$ 10.24	14%
Noviembre 2012	$ 8.18	$ 9.45	16%
Diciembre 2012	$ 9.19	$ 9.16	0%
Enero 2013	$ 8.30	$ 8.97	8%
Febrero 2013	$ 7.14	$ 8.74	22%
Marzo 2013	$ 8.47	$ 8.66	2%
Abril 2013	$ 8.44	$ 11.65	38%
Mayo 2013	$ 19.90	$ 11.78	41%
Junio 2013	$ 8.10	$ 12.23	51%
Julio 2013	$ 6.63	$ 8.79	33%
		ECM	22%

4.5 Discusión

Ha sido mostrado en las secciones anteriores los errores asociados a los modelos de redes neuronales propuestos y su eficacia para predecir valores futuros fuera del entrenamiento. Sin embargo se debe aclarar algunas consideraciones que deben tener en cuenta sobre la solución que aportan los modelos propuestos.

Los modelos con recurrencia que se propusieron mostraron de manera experimental tener un mejor ajuste para un problema de predicción. Los modelos recurrentes fueron capaces de modificar los pesos sinápticos y generar conocimiento con base en la evidencia de los datos, pero también ajustaron este conocimiento a partir de valores predichos, es decir se ajustó con datos desconocidos.

Por lo que se concluye con base en los resultados experimentales y específicos para el problema que las arquitecturas localmente recurrentes tienen un mejor ajuste a los datos, en comparación con los modelos sin recurrencia. Inclusive si se toma a los modelos ARIMA como modelos no recurrentes los modelos neuronales propuestos tienen un mejor desempeño para este problema en específico.

Conclusiones

El objetivo de este trabajo de investigación fue proponer una metodología para la implementación de redes neuronales, esta metodología fue presentada a lo largo de §2. El desarrollo de los pasos descritos en el Capítulo II permitió conocer más a fondo el problema de la predicción del precio del jitomate así como tener una imagen más amplia y clara del comportamiento de los datos al comparar gráficamente las variables.

Por otra parte el uso de las tablas resumen propició el mejor uso de los datos y modelos, de tal forma que la comparación entre tipos de Inputs y tipos de arquitactura fue más sencillo e interpretativo, i.e. el resumen de los resultados en tablas permitió la agrupación por características en específico. Por lo que identificar el modelo con el mejor ajuste fue más sencillo.

Además, se demuestra la validez de la hipótesis planteada de manera experimental con los resultados expuestos en §4 muestran que, para el caso de aplicación, las redes neuronales con elementos de recurrencia presentan un mejor ajuste a los datos y por ende una mejor predicción.

Si bien el problema de predecir el precio del jitomate se vio justificado, dada su importancia en la economía mexicana, los modelos neuronales recurrentes se pueden aplicar a otros productos de la canasta básica o incluso a productos bursátiles (i.e. FOREX), siendo así que se puede pasar de un campo de aplicación a uno totalmente diferente.

Trabajo Futuro

Se debe tener en mente las siguientes consideraciones que pueden llegar a ser factores que incrementen la efectividad de los modelos neuronales.

1. Flexibilidad en la arquitectura

   Los nueve modelos presentados son sólo una porción ínfima de todo el universo de posibles arquitecturas de redes neuronales. Podría existir un modelo con un número distinto de neuronas de entrada, por ejemplo un modelo que refleje un comportamiento semestral con seis Inputs; del mismo modo se puede modificar el número de capas ocultas y la función de activación.

2. Optimización de parámetros

   Así como se realizó el entrenamiento con distintos niveles de la tasa de aprendizaje para tener mejores resultados es posible que la tasa de aprendizaje se modifique de manera dinámica a lo largo del entrenamiento dependiendo de la respuesta en cada una de las iteraciones.

3. Técnicas auxiliares

   Una de las posibilidades es el uso de los algoritmos genéticos para optimizar los parámetros del modelo o la arquitectura misma del modelo neuronal. La flexibilidad del modelo neuronal permite implementar técnicas como los algoritmos genéticos o estrategias evolutivas para mejorar el desempeño de las redes neuronales.

Tomando en cuenta las consideraciones expuestas anteriormente, se puede pensar que un paso lógico para este trabajo es la implementación de técnicas auxiliares para optimizar el modelo que mejor se ajuste al problema así como los parámetros de la solución como algoritmos genéticos o incluso programación dinámica.

Además, existe la posibilidad que el tiempo de procesamiento se vuelva un inconveniente cuando se trabaja con demasiados datos o aplicando técnicas auxiliares, entonces una posibilidad para mejorar o evadir este problema es utilizar la programación paralela y después distribuir este proceso en diferentes ordenadores.

Anexos

Anexo A

Tablas de datos empleados en el análisis exploratorio y en el entrenamiento de la red.

Fecha	Superficie sembrada (Ha)	Superficie siniestrada (Ha)	Superficie cosechada (Ha)	Fecha	Superficie sembrada (Ha)	Superficie siniestrada (Ha)	Superficie cosechada (Ha)
01/01/2004	36,769	942	8,556	01/01/2009	49,259	954	25,925
01/02/2004	40,262	1,128	13,584	01/02/2009	52,367	1,006	34,088
01/03/2004	42,933	1,276	25,200	01/03/2009	54,250	1,064	40,908
01/04/2004	48,171	1,470	31,087	01/04/2009	32,812	93	21,265
01/05/2004	55,571	1,576	38,255	01/05/2009	36,969	105	25,516
01/06/2004	60,267	1,645	39,253	01/06/2009	40,080	117	28,815
01/07/2004	69,348	2,116	43,607	01/07/2009	45,900	124	31,968
01/08/2004	74,222	2,917	46,675	01/08/2009	51,209	340	34,678
01/09/2004	76,801	3,859	52,018	01/09/2009	53,164	535	39,041
01/10/2004	39,529	2,488	17,167	01/10/2009	31,053	474	14,422
01/11/2004	56,230	2,598	23,684	01/11/2009	39,135	720	18,593
01/12/2004	61,713	2,878	26,817	01/12/2009	43,659	778	21,396
01/01/2005	68,690	2,919	38,133	01/01/2010	54,188	760	23,974
01/02/2005	72,177	2,952	45,115	01/02/2010	51,160	2,074	31,310
01/03/2005	74,992	3,009	57,170	01/03/2010	52,853	2,119	36,971
01/04/2005	46,333	289	33,238	01/04/2010	33,635	1,517	19,425
01/05/2005	50,470	987	36,700	01/05/2010	36,989	1,545	22,766
01/06/2005	55,800	1,037	41,352	01/06/2010	40,807	1,577	26,318
01/07/2005	65,036	1,038	44,050	01/07/2010	47,008	1,643	29,957
01/08/2005	70,003	1,073	46,263	01/08/2010	51,492	1,781	32,498
01/09/2005	73,496	1,246	48,996	01/09/2010	53,206	2,278	37,842
01/10/2005	37,803	1,125	16,351	01/10/2010	33,023	347	14,230
01/11/2005	51,172	1,660	21,231	01/11/2010	40,634	512	18,572
01/12/2005	54,897	1,773	27,042	01/12/2010	44,843	582	22,621
01/01/2006	73,050	2,148	36,444	01/01/2011	54,994	586	25,236
01/02/2006	69,690	2,153	40,036	01/02/2011	56,033	5,072	28,544
01/03/2006	70,276	2,455	52,305	01/03/2011	57,381	8,206	33,759
01/04/2006	43,694	720	29,624	01/04/2011	36,566	7,872	11,301
01/05/2006	48,032	815	32,419	01/05/2011	40,352	7,865	14,594
01/06/2006	50,863	1,075	34,059	01/06/2011	43,829	8,453	19,035
01/07/2006	57,760	1,340	36,986	01/07/2011	50,737	8,516	22,128
01/08/2006	61,236	1,352	40,354	01/08/2011	54,137	8,488	23,807
01/09/2006	65,132	1,638	44,430	01/09/2011	55,433	14,783	27,985
01/10/2006	38,615	1,380	15,563	01/10/2011	29,946	274	13,103
01/11/2006	44,535	1,460	19,763	01/11/2011	39,136	317	19,155
01/12/2006	53,680	1,504	25,300	01/12/2011	46,247	451	23,448
01/01/2007	59,903	1,902	32,462	01/01/2012	49,855	451	25,352
01/02/2007	65,120	1,978	43,406	01/02/2012	53,498	546	32,029
01/03/2007	65,173	2,249	50,209	01/03/2012	55,604	561	44,968
01/04/2007	41,946	836	27,352	01/04/2012	36,385	327	24,909
01/05/2007	44,677	1,040	31,075	01/05/2012	39,992	466	28,925
01/06/2007	49,602	1,202	35,335	01/06/2012	43,807	466	33,186
01/07/2007	56,967	1,385	37,992	01/07/2012	48,482	536	35,802
01/08/2007	63,037	1,416	40,796	01/08/2012	53,387	849	38,361
01/09/2007	64,971	1,540	44,985	01/09/2012	55,185	872	40,534
01/10/2007	38,515	483	15,498	01/10/2012	27,445	63	11,618
01/11/2007	43,712	565	22,146	01/11/2012	34,026	83	17,298
01/12/2007	49,656	606	25,526	01/12/2012	37,448	83	20,820
01/01/2008	55,510	659	31,696	01/01/2013	41,934	460	25,040
01/02/2008	64,664	663	41,889	01/02/2013	44,545	586	32,893
01/03/2008	62,557	683	45,330	01/03/2013	45,287	803	37,912
01/04/2008	38,954	225	21,912				
01/05/2008	39,165	372	26,184				
01/06/2008	44,360	514	29,268				
01/07/2008	50,771	535	31,816				
01/08/2008	55,096	751	35,295				
01/09/2008	57,288	1,205	39,697				
01/10/2008	33,787	836	13,542				
01/11/2008	39,861	883	16,966				
01/12/2008	45,144	883	20,525				

Datos referentes al cultivo

Fecha	Producción (Ton)	Exportación mensual (Ton)	Fecha	Producción (Ton)	Exportación mensual (Ton)
01/01/2004	184,103	87,608	01/01/2009	975,544	130,049
01/02/2004	417,130	103,584	01/02/2009	1,356,794	150,286
01/03/2004	843,763	105,157	01/03/2009	1,663,847	157,727
01/04/2004	1,017,881	107,618	01/04/2009	837,343	129,729
01/05/2004	1,212,375	88,154	01/05/2009	1,005,351	107,219
01/06/2004	1,296,486	73,782	01/06/2009	1,116,792	96,123
01/07/2004	1,422,599	67,875	01/07/2009	1,235,658	88,015
01/08/2004	1,500,406	50,302	01/08/2009	1,368,886	97,677
01/09/2004	1,639,887	46,274	01/09/2009	1,559,685	105,692
01/10/2004	414,500	38,473	01/10/2009	564,976	54,581
01/11/2004	592,698	35,757	01/11/2009	754,484	60,001
01/12/2004	690,861	30,865	01/12/2009	874,695	43,739
01/01/2005	1,104,331	88,790	01/01/2010	945,811	149,039
01/02/2005	1,339,503	104,983	01/02/2010	1,267,732	177,391
01/03/2005	1,688,506	106,576	01/03/2010	1,482,856	251,247
01/04/2005	1,031,705	109,070	01/04/2010	708,170	184,412
01/05/2005	1,176,198	89,344	01/05/2010	853,623	171,509
01/06/2005	1,273,977	74,778	01/06/2010	1,080,446	168,065
01/07/2005	1,356,584	68,791	01/07/2010	1,190,538	159,635
01/08/2005	1,417,087	50,981	01/08/2010	1,334,022	101,517
01/09/2005	1,527,754	46,898	01/09/2010	1,584,531	110,340
01/10/2005	487,553	38,993	01/10/2010	603,631	92,965
01/11/2005	618,623	36,240	01/11/2010	799,634	99,848
01/12/2005	856,593	31,282	01/12/2010	991,098	75,512
01/01/2006	1,118,186	97,244	01/01/2011	1,080,535	165,427
01/02/2006	1,325,608	114,978	01/02/2011	1,212,366	176,795
01/03/2006	1,651,655	116,723	01/03/2011	1,406,907	144,002
01/04/2006	937,488	119,455	01/04/2011	383,168	307,936
01/05/2006	1,036,088	97,851	01/05/2011	507,994	179,674
01/06/2006	1,142,486	81,898	01/06/2011	705,055	230,940
01/07/2006	1,249,828	75,341	01/07/2011	799,244	148,403
01/08/2006	1,355,676	55,835	01/08/2011	912,718	152,331
01/09/2006	1,454,230	51,364	01/09/2011	1,089,693	116,243
01/10/2006	479,073	42,705	01/10/2011	615,072	111,701
01/11/2006	580,385	39,690	01/11/2011	862,485	85,047
01/12/2006	796,691	34,260	01/12/2011	1,052,488	115,795
01/01/2007	1,006,907	105,489	01/01/2012	1,138,259	171,878
01/02/2007	1,379,939	124,726	01/02/2012	1,457,527	156,397
01/03/2007	1,640,660	126,619	01/03/2012	1,806,910	176,293
01/04/2007	989,142	129,583	01/04/2012	846,548	207,800
01/05/2007	1,133,248	106,147	01/05/2012	1,084,259	149,390
01/06/2007	1,246,946	88,841	01/06/2012	1,387,763	144,080
01/07/2007	1,342,418	81,728	01/07/2012	1,513,263	136,247
01/08/2007	1,476,302	60,568	01/08/2012	1,752,616	126,440
01/09/2007	1,575,571	55,718	01/09/2012	1,913,924	105,942
01/10/2007	544,378	46,326	01/10/2012	600,578	104,081
01/11/2007	832,338	43,055	01/11/2012	858,187	96,480
01/12/2007	961,891	37,165	01/12/2012	1,032,960	79,343
01/01/2008	1,193,754	124,855	01/01/2013	1,185,701	185,932
01/02/2008	1,586,013	147,625	01/02/2013	1,489,622	157,702
01/03/2008	1,793,566	149,866	01/03/2013	1,764,099	158,353
01/04/2008	888,003	153,373			
01/05/2008	1,087,537	125,634			
01/06/2008	1,249,444	105,151			
01/07/2008	1,348,125	96,733			
01/08/2008	1,507,606	71,688			
01/09/2008	1,657,061	65,947			
01/10/2008	521,841	54,831			
01/11/2008	655,227	50,959			
01/12/2008	782,291	43,987			

Datos referentes a la producción

Fecha	Precio promedio mensual (MXN)	Fecha	Precio promedio mensual (MXN)	Fecha	Precio promedio mensual (MXN)
01/01/2004	$ 4.81	01/01/2007	$ 6.08	01/01/2010	$ 5.73
01/02/2004	$ 4.90	01/02/2007	$ 4.47	01/02/2010	$ 6.60
01/03/2004	$ 5.10	01/03/2007	$ 3.59	01/03/2010	$ 9.43
01/04/2004	$ 6.71	01/04/2007	$ 6.17	01/04/2010	$ 5.54
01/05/2004	$ 5.18	01/05/2007	$ 5.73	01/05/2010	$ 4.95
01/06/2004	$ 4.61	01/06/2007	$ 4.71	01/06/2010	$ 4.15
01/07/2004	$ 4.89	01/07/2007	$ 7.30	01/07/2010	$ 5.76
01/08/2004	$ 8.80	01/08/2007	$ 8.04	01/08/2010	$ 6.44
01/09/2004	$ 11.06	01/09/2007	$ 11.33	01/09/2010	$ 8.45
01/10/2004	$ 12.37	01/10/2007	$ 6.94	01/10/2010	$ 12.27
01/11/2004	$ 12.64	01/11/2007	$ 8.05	01/11/2010	$ 11.71
01/12/2004	$ 6.95	01/12/2007	$ 8.20	01/12/2010	$ 10.71
01/01/2005	$ 3.35	01/01/2008	$ 6.90	01/01/2011	$ 8.20
01/02/2005	$ 6.09	01/02/2008	$ 5.59	01/02/2011	$ 9.83
01/03/2005	$ 6.91	01/03/2008	$ 9.27	01/03/2011	$ 10.42
01/04/2005	$ 10.32	01/04/2008	$ 9.55	01/04/2011	$ 16.06
01/05/2005	$ 10.61	01/05/2008	$ 8.65	01/05/2011	$ 7.09
01/06/2005	$ 8.21	01/06/2008	$ 7.70	01/06/2011	$ 5.51
01/07/2005	$ 8.09	01/07/2008	$ 9.50	01/07/2011	$ 6.12
01/08/2005	$ 5.92	01/08/2008	$ 5.70	01/08/2011	$ 5.39
01/09/2005	$ 5.13	01/09/2008	$ 5.94	01/09/2011	$ 5.49
01/10/2005	$ 4.26	01/10/2008	$ 6.55	01/10/2011	$ 5.68
01/11/2005	$ 4.44	01/11/2008	$ 8.84	01/11/2011	$ 5.11
01/12/2005	$ 8.12	01/12/2008	$ 12.18	01/12/2011	$ 8.15
01/01/2006	$ 11.81	01/01/2009	$ 6.11	01/01/2012	$ 7.26
01/02/2006	$ 9.94	01/02/2009	$ 3.97	01/02/2012	$ 4.96
01/03/2006	$ 6.47	01/03/2009	$ 6.06	01/03/2012	$ 6.38
01/04/2006	$ 8.18	01/04/2009	$ 9.38	01/04/2012	$ 5.63
01/05/2006	$ 6.86	01/05/2009	$ 8.24	01/05/2012	$ 7.72
01/06/2006	$ 4.42	01/06/2009	$ 8.52	01/06/2012	$ 10.52
01/07/2006	$ 4.41	01/07/2009	$ 9.31	01/07/2012	$ 9.31
01/08/2006	$ 7.00	01/08/2009	$ 12.59	01/08/2012	$ 8.93
01/09/2006	$ 11.79	01/09/2009	$ 15.31	01/09/2012	$ 11.74
01/10/2006	$ 11.16	01/10/2009	$ 9.63	01/10/2012	$ 9.01
01/11/2006	$ 9.05	01/11/2009	$ 7.91	01/11/2012	$ 8.18
01/12/2006	$ 8.99	01/12/2009	$ 7.20	01/12/2012	$ 9.19
				01/01/2013	$ 8.30
				01/02/2013	$ 7.14
				01/03/2013	$ 8.47
				01/04/2013	$ 8.44
				01/05/2013	$ 19.90
				01/06/2013	$ 8.10
				01/07/2013	$ 6.63
				01/08/2013	$ 8.39

Datos refrentes al precio

Anexo B

Tablas de datos arrojados durante la simulación.

Fecha	Target	Output Red Neuronal	Erro Absoluto	Output ARIMA	Erro Absoluto	Fecha	Target	Output Red Neuronal	Erro Absoluto	Output ARIMA	Erro Absoluto
01/03/2004	$ 6.71	$ 10.44	56%	$ 5.01	25%	01/01/2008	$ 5.59	$ 8.93	60%	$ 6.97	25%
01/04/2004	$ 5.18	$ 5.27	2%	$ 5.06	2%	01/02/2008	$ 9.27	$ 9.58	3%	$ 7.06	24%
01/05/2004	$ 4.61	$ 4.15	10%	$ 5.18	12%	01/03/2008	$ 9.55	$ 10.58	11%	$ 6.27	34%
01/06/2004	$ 4.89	$ 4.71	4%	$ 6.16	26%	01/04/2008	$ 8.65	$ 10.33	19%	$ 5.48	37%
01/07/2004	$ 8.80	$ 9.64	9%	$ 5.23	41%	01/05/2008	$ 7.70	$ 10.23	33%	$ 7.71	0%
01/08/2004	$ 11.06	$ 11.63	5%	$ 4.89	56%	01/06/2008	$ 9.50	$ 9.48	0%	$ 7.88	17%
01/09/2004	$ 12.37	$ 12.82	4%	$ 5.06	59%	01/07/2008	$ 5.70	$ 9.16	61%	$ 7.33	29%
01/10/2004	$ 12.64	$ 12.07	4%	$ 7.42	41%	01/08/2008	$ 5.94	$ 4.74	20%	$ 6.76	14%
01/11/2004	$ 6.95	$ 9.99	44%	$ 8.79	26%	01/09/2008	$ 6.55	$ 8.75	34%	$ 7.85	20%
01/12/2004	$ 3.35	$ 4.54	35%	$ 9.58	186%	01/10/2008	$ 8.84	$ 10.39	18%	$ 5.55	37%
01/01/2005	$ 6.09	$ 4.45	27%	$ 9.75	60%	01/11/2008	$ 12.18	$ 10.53	14%	$ 5.69	53%
01/02/2005	$ 6.91	$ 9.40	36%	$ 6.30	9%	01/12/2008	$ 6.11	$ 9.70	59%	$ 6.06	1%
01/03/2005	$ 10.32	$ 10.48	2%	$ 4.13	60%	01/01/2009	$ 3.97	$ 4.44	12%	$ 7.45	88%
01/04/2005	$ 10.61	$ 11.11	5%	$ 5.78	45%	01/02/2009	$ 6.06	$ 4.94	18%	$ 9.47	56%
01/05/2005	$ 8.21	$ 10.68	30%	$ 6.28	24%	01/03/2009	$ 9.38	$ 9.78	4%	$ 5.80	38%
01/06/2005	$ 8.09	$ 9.40	16%	$ 8.34	3%	01/04/2009	$ 8.24	$ 10.38	26%	$ 4.50	45%
01/07/2005	$ 5.92	$ 4.96	16%	$ 8.52	44%	01/05/2009	$ 8.52	$ 10.25	20%	$ 5.77	32%
01/08/2005	$ 5.13	$ 4.34	15%	$ 7.07	38%	01/06/2009	$ 9.31	$ 11.25	21%	$ 7.77	16%
01/09/2005	$ 4.26	$ 3.98	7%	$ 6.99	64%	01/07/2009	$ 12.59	$ 12.92	3%	$ 7.08	44%
01/10/2005	$ 4.44	$ 4.40	1%	$ 5.68	28%	01/08/2009	$ 15.31	$ 13.36	13%	$ 7.25	53%
01/11/2005	$ 8.12	$ 9.49	17%	$ 5.20	36%	01/09/2009	$ 9.63	$ 12.53	30%	$ 7.73	20%
01/12/2005	$ 11.81	$ 11.03	7%	$ 4.68	60%	01/10/2009	$ 7.91	$ 10.21	29%	$ 9.72	23%
01/01/2006	$ 9.94	$ 11.05	11%	$ 4.79	52%	01/11/2009	$ 7.20	$ 9.01	25%	$ 11.36	58%
01/02/2006	$ 6.47	$ 10.10	56%	$ 7.01	8%	01/12/2009	$ 5.73	$ 5.00	13%	$ 7.93	38%
01/03/2006	$ 8.18	$ 9.06	11%	$ 9.24	13%	01/01/2010	$ 6.60	$ 9.01	37%	$ 6.88	4%
01/04/2006	$ 6.86	$ 8.64	26%	$ 8.11	18%	01/02/2010	$ 9.43	$ 8.99	5%	$ 6.46	32%
01/05/2006	$ 4.42	$ 4.28	3%	$ 6.01	36%	01/03/2010	$ 5.54	$ 5.24	5%	$ 5.57	0%
01/06/2006	$ 4.41	$ 4.31	2%	$ 7.05	60%	01/04/2010	$ 4.95	$ 4.22	15%	$ 6.09	23%
01/07/2006	$ 7.00	$ 9.21	32%	$ 6.25	11%	01/05/2010	$ 4.15	$ 4.14	0%	$ 7.80	88%
01/08/2006	$ 11.79	$ 10.95	7%	$ 4.77	60%	01/06/2010	$ 5.76	$ 4.84	16%	$ 5.45	5%
01/09/2006	$ 11.16	$ 11.90	7%	$ 4.77	57%	01/07/2010	$ 6.44	$ 8.74	36%	$ 5.09	21%
01/10/2006	$ 9.05	$ 11.30	25%	$ 6.33	30%	01/08/2010	$ 8.45	$ 10.24	21%	$ 4.61	45%
01/11/2006	$ 8.99	$ 9.93	11%	$ 9.23	3%	01/09/2010	$ 12.27	$ 11.70	5%	$ 5.58	54%
01/12/2006	$ 6.08	$ 6.71	10%	$ 8.85	46%	01/10/2010	$ 11.71	$ 12.61	8%	$ 6.00	49%
01/01/2007	$ 4.47	$ 4.10	8%	$ 7.57	69%	01/11/2010	$ 10.71	$ 11.71	9%	$ 7.21	33%
01/02/2007	$ 3.59	$ 4.01	12%	$ 7.54	110%	01/12/2010	$ 8.20	$ 11.14	36%	$ 9.52	16%
01/03/2007	$ 6.17	$ 4.20	32%	$ 5.78	6%	01/01/2011	$ 9.83	$ 10.84	10%	$ 9.18	7%
01/04/2007	$ 5.73	$ 4.71	18%	$ 4.80	16%	01/02/2011	$ 10.42	$ 12.72	22%	$ 8.58	18%
01/05/2007	$ 4.71	$ 4.64	2%	$ 4.27	9%	01/03/2011	$ 16.06	$ 12.28	24%	$ 7.06	56%
01/06/2007	$ 7.30	$ 8.23	13%	$ 5.83	20%	01/04/2011	$ 7.09	$ 11.56	63%	$ 8.05	13%
01/07/2007	$ 8.04	$ 10.26	28%	$ 5.57	31%	01/05/2011	$ 5.51	$ 5.07	8%	$ 8.40	53%
01/08/2007	$ 11.33	$ 10.28	9%	$ 4.95	56%	01/06/2011	$ 6.12	$ 4.65	24%	$ 11.81	93%
01/09/2007	$ 6.94	$ 10.53	52%	$ 6.52	6%	01/07/2011	$ 5.39	$ 4.59	15%	$ 6.39	19%
01/10/2007	$ 8.05	$ 9.52	18%	$ 6.96	13%	01/08/2011	$ 5.49	$ 4.78	13%	$ 5.43	1%
01/11/2007	$ 8.20	$ 9.50	16%	$ 8.95	9%	01/09/2011	$ 5.68	$ 5.02	12%	$ 5.80	2%
01/12/2007	$ 6.90	$ 8.59	24%	$ 6.30	9%	01/10/2011	$ 5.11	$ 8.30	62%	$ 5.36	5%
						01/11/2011	$ 8.15	$ 8.60	5%	$ 5.42	33%
						01/12/2011	$ 7.26	$ 8.85	22%	$ 5.54	24%
						01/01/2012	$ 4.96	$ 4.79	3%	$ 5.19	5%
						01/02/2012	$ 6.38	$ 4.57	28%	$ 7.03	10%
						01/03/2012	$ 5.63	$ 5.02	11%	$ 6.49	15%
						01/04/2012	$ 7.72	$ 9.50	23%	$ 5.10	34%
						01/05/2012	$ 10.52	$ 10.47	0%	$ 5.96	43%
						01/06/2012	$ 9.31	$ 11.04	19%	$ 5.51	41%
						01/07/2012	$ 8.93	$ 11.27	26%	$ 6.77	24%
						01/08/2012	$ 11.74	$ 11.18	5%	$ 8.46	28%
								Red Neuronal		ARIMA	
								ECM	18.77%	ECM	33.25%

EMC para el mejor modelo neuronal y para el modelo ARIMA

Bibliografía

L. Fausett. *Fundamentals of Neural Networks. Architectures, Algorithms, And Applications.*
Prentice Hall, 1993.

S. Haykin. *Neural Networks: A Comprehensive Foundation.*
Pearson (segunda edición), 1997.

K. Mehrotra,et. al. *Elements of Artificial Neural NEtworks.*
Bradford , 1996.

BOX and JENKINS. *Time series analysis. Forecasting and control.*
Holden-Day, E.U.A. , 1994.

Referencias

1. E. Muñoz, J. L. Blázquez, N. Galpasoro, B. González. *Estimulación cognitiva y rehabilitación neuropsicológica.*
   CAtaluña; Ed. UCO, 2011.

2. M. Minsky and S. A. Papert. *Perceptrons: An Introduction to Computational Geometry.*
   Massachusetts ; The MIT Press, 1987.

3. D. E. Rumelhart, G. E. Hinton and R. J. Williams. *Learning representations by back-propagating errors.*

en Nature vol. 323, Octubre 1986.

4. Dictionary of Algorithms and Data Structures, disponible en:
   http://[www.xlinux.nist.gov]/dads//

5. G. E. P. Box, G. M. Jenkins, G. C. Reinsel . *Time Series Analysis: Forecasting and Control.*
   San Francisco; Ed. Wiley, 4a. ed, 2008.

6. R.G. Brown. *Smoothing, Forecasting and Prediction of Discrete Time Series.*
   New York; Dover Publications, 1963.

7. K. Hornik, M. Stinchcombe and H. White. *Multilayer feedforward networks are universal approximators.*
   en Neural Networks vol. 2 issue 5, 1989.

8. P. J. Werbos. *Backpropagation through time: what it does and how to do it.*
   en Proceedings of the IEEE vol. 78 issue 10, 1990.

9. R. Watrous and L. Shastri. *Learning phonetic features using connectionist networks: An experiment in speech recognition.*
   en 1st. IEEE Int. Conf. Neural Networks, June 1987.

10. Catálogo de semillas de tomate.
    1ra edición. Instituto Nacional de Investigación y Tecnología Agraria y Alimentaria Ministerio de Agricultura, Pesca y Alimentación. Madrid, 1996.

11. Instituto Nacional de Estadística y Geografía disponible en:
    http://[www.inegi.org.mx]

12. Documento metodológico INPC INEGI disponible en:
    http://[www.inegi.org.mx]/est/contenidos/Proyectos/INP/

13. Banco de México disponible en:
    http://[www.inegi.org.mx]

14. Agencia de Noticias del Estado Mexicano disponible en:
    http://[www.notimex.com.mx]

15. Canasta INPP disponible en:
http://[www.inegi.org.mx]

16. Secretaría de Economía disponible en:
http://[www.economia.gob.mx]

17. Sr. [FISZELEW] Abel.
Generación automática de redes neuronales con ajuste de parámetros basado en algoritmos genéticos.
Tesis para optar al grado de Ingeniero Informático: Universidad de Buenos Aires, Facultad de Ingeniería, 2002, 262 p.

18. [ESTRADA] Benigno, [PRIEGO] José Luis.
Pronóstico de Precios del Jitomate Saladette Usando un Modelo ARIMA.
Tesis para optar al grado de licenciatura: Universidad Autonoma de Chapingo, Div. Ciencias Forestales, 2006, 146 p.

19. [Rev. Mex. Cienc. Agrí vol.2 no.4].
Aplicación de la metodología Box-Jenkins para pronósticos de precios de jitomate.
Revista mexicana de ciencias agrícolas
México, 2011.

Referencias electrónicas

http://[www.inegi.org.mx]/est/contenidos/proyectos/inp/
Instituto Nacional de Estadística y Geografía.
2011
Fecha de consulta: Julio 2013

http://[www.siap.gob.mx].
Servicio de Información Agroalimentaria y Pesquera.
2010
Fecha de consulta: Julio 2013

http://[www.economia-sniim.gob.mx].
Secretaría de Agricultura, Ganadería, Desarrollo Rural, Pesca y Alimentación.
2010
Fecha de consulta: Julio 2013

http://[www.fao.org]/home/.
Organización de las Naciones Unidas para la Alimentación y la Agricultura.
2013
Fecha de consulta: Julio 2013

http://[www.campomexicano.gob.mx]/campo/index.php
Sistema Nacional de Información e Integración de Mercados.
2010
Fecha de consulta: Julio 2013

http://[www.banxico.org.mx]
Banco de México.
Fecha de consulta: Julio 2013