Humanidades, Tecnología y Ciencia, del Instituto Politécnico Nacional
Certificado de Reserva de Derechos al Uso Exclusivo del Título: 04-2010-0326124-14000-203 ISSN: 2007-1957

CLASIFICADOR DEL HÍGADO GRASO NO ALCOHÓLICO BASADO EN REDES NEURONALES BAYESIANAS USANDO TORCH

Israel Rivera Zárate
Instituto Politécnico Nacional-CIDETEC
irivera@ipn.mx

Miguel Hernández Bolaños
Instituto Politécnico Nacional-CIDETEC
mbolanos@ipn.mx

Patricia Pérez Romero
Instituto Politécnico Nacional-CIDETEC
promerop@ipn.mx

Resumen

The problems that expert systems can deal with can be classified into two types: deterministic problems and stochastic problems. In the medical field, the relationships between symptoms and diseases are known only with certain degree of certainty (the presence of a set of symptoms does not always imply the presence of a disease). Deterministic problems can be formulated using a set of rules that relate well-defined objects. Some expert systems use the same structure as rule-based systems, but introduce a measure associated with the uncertainty of the rules and their premises. Another intuitive measure of uncertainty in probability, in which the joint distribution of a set of variables is used to describe the dependency relationships between them, and conclusions are drawn using well-known formulas from probability theory. This work proposes the development of a system that allows classifying degree of progression of liver damage based on Bayesian neural networks using Torch language.

Key words: Nonalcoholic fatty liver, Bayesian neural network, probability theory, Torch.

Los sistemas de aprendizaje automático pueden modelarse a partir de una función de probabilidad de sus parámetros. El objetivo será maximizar esa probabilidad con respecto a los parámetros en un proceso que ha sido denominado como estimación de máxima verosimilitud (MLE). En el modelado bayesiano, además de la función de verosimilitud, se debe definir también distribuciones previas para los parámetros del modelo. La regla de Bayes en su forma convencional sirve para encontrar la distribución de parámetros a posteriori. El cálculo de la distribución posterior a menudo es intratable o extremadamente difícil analíticamente. Además, incluso si existe una

Humanidades, Tecnología y Ciencia, del Instituto Politécnico Nacional
Certificado de Reserva de Derechos al Uso Exclusivo del Título: 04-2010-0326124-14000-203 ISSN: 2007-1957

forma analítica cerrada para el posterior, el cálculo de la integral sobre todos los parámetros es básicamente imposible para modelos de complejidad razonable.

Las formas más empleadas de enfrentar esta dificultad son:

I. Estimación apostólica máxima (MAP) donde se ubica el pico de la distribución posterior y es usado como una estimación puntual para el modelo (es mejor que usar solo la probabilidad, pero no brinda una medida adecuada de incertidumbre para nuestras predicciones).

II. Markov Chain Monte Carlo (MCMC) para modelos y conjuntos de datos grande resulta ser muy lento, y no funciona bien con distribuciones posteriores altamente multimodales.

III. La inferencia variacional (VI) se aproxima al posterior con una distribución más simple y de "buen comportamiento".

1. Clasificación Bayesiana

La propuesta bayesiana que es un enfoque de clasificación supervisada que consiste en asignar a un objeto de atributos, $X1$, $X2$, … , Xn, una de m clases posibles, $c1$, $c2$, … , cm, de modo tal que considerando como: $X = \{X1, X2, \dots, Xn\}$, la probabilidad de la clase se maximiza como se indica en (1):

$$\mathrm{Arg_C}[MaxP(C|X)] \quad (1)$$

La idea del clasificador bayesiano tiene como base el uso de la regla de Bayes para calcular la probabilidad posterior de la clase dados los atributos, tal como indica la expresión (2):

$$P(C|X1,X2, \dots, Xn) = P(C)P(X1,X2, \dots, Xn|C)\ P(X1,X2, \dots, Xn) \quad (2)$$

Que se puede escribir de la forma:

$$(C|X) = (C)(X|C)\ P(X) \quad (3)$$

Entonces el problema se puede expresar como se muestra en (4):

$$Arg_C \left\{ Max \left[\mathrm{P(C|X)} = \frac{\mathrm{P(C)P(X|C)}}{\mathrm{P(X)}} \right] \right\} \quad (4)$$

En la expresión se observa que el denominador (X), no varía para las diferentes clases, por lo que se puede considerar como una constante, de modo que, si lo que interesa es maximizar la probabilidad de la clase se tiene en (5):

$$Arg_C \{\mathrm{Max[P(C|X) = \alpha P(C)P(X|C)]}\} \quad (5)$$

De esta forma resolver un problema de clasificación bajo el enfoque bayesiano hace necesario contar con la probabilidad a priori de cada clase, $P(C)$, la probabilidad de los atributos dada la clase $P(X|C)$, conocida como verosimilitud; y así poder obtener la probabilidad posterior $P(C|A)$. Ver expresión (6).

$$\mathrm{Posterior} = \frac{\text{a priori} * \text{verosimilitud}}{\text{evidencia}} \quad (6)$$

1.1 Red Neuronal Bayesiana

Una Red Neuronal (RNN) convencional cuenta con una etapa de entrada donde se reciben los datos por evaluar x que opera como un vector D dimensional con H neuronas ocultas por capa. Los pesos sinápticos ω se aprenden durante el entrenamiento. Debido a que la RNN es una regresión logística que corresponde a una línea recta, b es el sesgo, el bias o la ordenada en el origen. Adicionalmente se emplea una función de

Humanidades, Tecnología y Ciencia, del Instituto Politécnico Nacional
Certificado de Reserva de Derechos al Uso Exclusivo del Título: 04-2010-0326124-14000-203 ISSN: 2007-1957

activación no lineal (por ejemplo, la tangente hiperbólica). Ver figura 1.

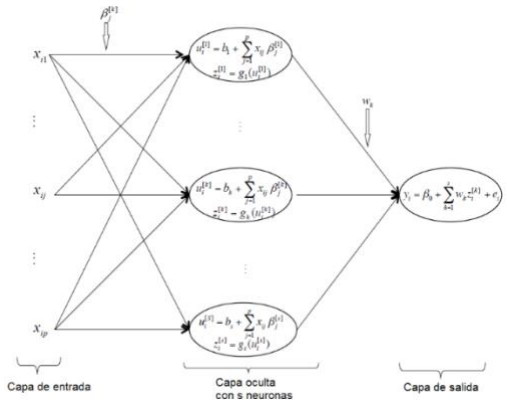

Figura 1. Diagrama de una red neuronal de una sola capa oculta. Single Hidden Layer Neural Network. Guzmán Eduardo et al. (2018).

Como lo indica Graves Alex et al. (2014), para entrenar una red neuronal se deben alimentar datos en la capa de entrada y evaluar el error de los resultados a la salida respecto del valor esperado. Se hace necesario repetir el proceso muchas veces (épocas). Para esto se requiere definir una función de error (típicamente error cuadrático medio) y un proceso de optimización que lleve el error a cero gradualmente para lo que se emplea una tasa de aprendizaje; lo que se conoce como algoritmo del descenso del gradiente. Ver ecuación 7. El vector de parámetros $\theta = (\hat{b}, \hat{\omega}, b, \omega)$ contiene todos los pesos y los sesgos del modelo.

$$f_\theta(x) = \hat{b} + \sum_{j=1}^{H} \hat{w}_j h_j = \hat{b} + \sum_{j=1}^{H} \hat{w}_j \tanh\left(b_j + \sum_{d=1}^{D} w_{jd} x_d\right) \quad (7)$$

Para extender el concepto de las RNN hacia las redes neuronales Bayesianas (RNB) se requiere proponer una probabilidad a priori, que es típicamente una normal centrada en cero con un cierto tipo de covarianza. Ver ecuación 7. Una verosimilitud (likelihood) que depende del problema a resolver; si se hace regresión continua se propone una distribución gaussiana, en el caso de clasificación binaria será distribución Bernoulli, o bien, si es clasificación de múltiples clases se propone una distribución categórica o multinomial. Contando ya con ambos elementos se puede escribir la probabilidad a posteriori. Ver ecuación 8.

$$\theta \sim \mathcal{N}(\theta|0, \Sigma_\theta) \quad (8)$$

$$p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}|\theta)p(\theta)}{p(\mathcal{D})} = \frac{1}{p(\mathcal{D})} \prod_n \mathcal{N}(y^{(n)}|f(x^{(n)}), \sigma^2)\mathcal{N}(\theta|0, \Sigma_\theta) \quad (9)$$

Sin embargo, se debe decir que esta expresión encierra una estructura anidada de funciones no lineales que impide la obtención de una solución analítica. Para obtener entonces la distribución a posteriori y más aún, lo que se conoce como posteriori predictivo se requiere el uso de técnicas matemáticas tales como el MCMC o método Monte Carlo basado en cadenas de Markov, o también el VI que es la inferencia variacional.

1.2 Inferencia Variacional

De acuerdo con Wainwright & Jordan (2008), si se considera una familia de densidades aproximadas de las variables latentes $q(z, \Phi)$, parametrizadas por un vector $\Phi \in \Phi$ se observa que la Inferencia Variacional es capaz de encontrar los parámetros que minimizan la divergencia hacia la posterior. Ver ecuación (10).

Humanidades, Tecnología y Ciencia, del Instituto Politécnico Nacional
Certificado de Reserva de Derechos al Uso Exclusivo del Título: 04-2010-0326124-14000-203 ISSN: 2007-1957

$$\dot{\Phi} = \arg_{\Phi\in\Phi}\min \mathrm{KL}\{q(z,\Phi) \,||\, p(z|x)\} \quad (10)$$

Se sabe que los objetivos variacionales importantes son los que muestren dos propiedades fundamentales: primera, optimizan la función produciendo una buena aproximación posterior; y segunda, el problema es tratable cuando la distribución posterior se conoce hasta una constante. Por lo tanto, el límite inferior de evidencia (ELBO) queda expresado como se indica en la expresión (11).

$$L(\Phi) = E_{q(x)}[\log p(x,z) - \log q(z,\Phi)] \quad (11)$$

La expresión se maximiza cuando q(z)=p(z | x) y solo depende de la distribución posterior hasta una constante manejable, $\log p\ (x,z)$. El ELBO ha sido el foco de gran parte de la literatura clásica y de acuerdo con Ghahramani & Beal (2001), maximizar el ELBO equivale a minimizar la divergencia hacia el posterior, y las expectativas son analíticas para una gran clase de modelos. Optimizar la divergencia implica una restricción de que el soporte de la aproximación se encuentra dentro del soporte de la posterior p(z | x). Con esta restricción explícita, el problema de optimización de la ecuación 9 se convierte en la expresión de la ecuación 12.

$$\dot{\Phi} = \arg_{\Phi\in\Phi}\ \max L(\Phi) \text{tal que } \sup\{q(z,\Phi)\} \subseteq \sup\{p(z|x)\} \quad (12)$$

Sin embargo, el soporte de la parte posterior también puede ser desconocido. Entonces, se asume además que el soporte del posterior es igual al del anterior. Ver ecuación 13.

$$\sup\{q(z,\Phi)\} = \sup\{p(z|x)\} \quad (13)$$

2. Desarrollo

La metodología adoptada en el presente trabajo consiste de los siguientes pasos:

[Paso 1:] Preprocesamiento.

[Paso 2:] Implementación del Algoritmo.

[Paso 3:]. Entrenamiento.

2.1. Preprocesamiento

Desde 1990 los laboratorios de análisis clínicos y microbiológicos Montecristo, en Chalco municipio del Estado de México, han llevado a cabo una base de datos de sus pacientes sobre la determinación de diversas sustancias a efecto de prevenir, detectar o diagnosticar diferentes enfermedades. Como lo indica Aguilera Méndez et al. (2017). Se puede dar lugar en el hígado a una condición metabólica donde sin necesidad de consumo crónico de alcohol se logran niveles elevados de los lípidos, a esta enfermedad se le denomina enfermedad del hígado graso no alcohólico (HGNA) o en sus siglas en inglés como NAFLD (Nonalcoholic Fatty Liver Desease). Este padecimiento exhibe varias fases y diferentes grados de severidad: esteatosis simple, esteatohepatitis, fibrosis, cirrosis y, en algunas ocasiones, cáncer hepático. Además, esta enfermedad es difícil de diagnosticar y pasa inadvertida hasta que presenta complicaciones. También está relacionada con otros padecimientos metabólicos como la obesidad, diabetes, dislipidemias, resistencia a la insulina y síndrome metabólico. Tomando como referencia estos valores es posible establecer criterios de análisis para desarrollar un sistema de apoyo médico para la clasificación de la enfermedad de hígado graso no alcohólico.

Para la variable de entrada (X):

Humanidades, Tecnología y Ciencia, del Instituto Politécnico Nacional
Certificado de Reserva de Derechos al Uso Exclusivo del Título: 04-2010-0326124-14000-203 ISSN: 2007-1957

1. Edad (años).
2. Índice de masa corporal (kg/m2).
3. Presión arterial (mm Hg).
4. Concentración de colesterol (mg/dL).
5. Concentración de triglicéridos (mg/dL).
6. Concentración de glucosa en plasma (mg/dL).
7. Concentración de alanina aminotransferasa ALT/GPT (U/L).
8. Concentración de aspartato aminotransferasa AST/GOT (U/L).

Para la variable de salida (Y):

1. Clase: se clasifica en enfermedad de hígado graso “Grave”, “Moderado” y “Sano”.

Debido a que es un problema de clasificación multiclase se requiere un conjunto de entrenamiento y uno de validación. Se tomó la información de 700 pacientes para el entrenamiento y 300 para evaluación.

Todos los parámetros del modelo se pueden aproximar con frecuencias relativas del conjunto de entrenamiento. Estas son las estimaciones de máxima verosimilitud de las probabilidades. Una clase a priori se puede calcular asumiendo clases equiprobables (es decir, a priori = 1/ (número de clases)), o mediante el cálculo de una estimación de la probabilidad de clase del conjunto de entrenamiento (es decir, el a priori de una clase dada = (número de muestras en la clase) / (número total de muestras)). Para la estimación de los parámetros de la distribución de una característica, se debe asumir una distribución o generar modelos de estadística no paramétrica de las características del conjunto de entrenamiento.

2.2. Implementación

De acuerdo con Guzmán Eduardo et al. (2018), en 1994 David JC Mackay desarrolló un algoritmo para la determinación de una red neuronal bayesiana implementada en lenguaje C que se puede consultar en la página del autor:
http://www.inference.phy.cam.ac.uk/mackay
y que corresponde con los pasos descritos en la figura 2.

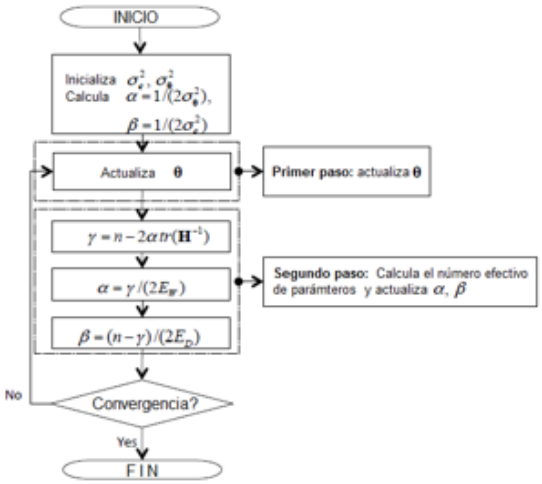

Figura 2. Diagrama de flujo para el algoritmo de la RNB. Guzmán Eduardo (2018).

Tomando como referencia el algoritmo de la figura 2 se procedió a la implementación del mismo en lenguaje Pytorch.

Se realizó la implementación de una red neuronal Bayesiana con dos capas ocultas y probabilidad a priori con distribución normal en todas sus activaciones, así como la desviación estándar parametrizada. La codificación de la red se hace mediante el uso de las clases definidas en “pyro” donde PyroSample sirve para declarar variables aleatorias como son los pesos y los sesgos. Por su parte, PyroParam sirve para declarar parámetros deterministas que se desean optimizar, como son los parámetros de la distribución a priori. Finalmente, PyroModule permite levantar (lift) una capa como Bayesiana, esto es, que sus parámetros sean variables aleatorias y no variables

Humanidades, Tecnología y Ciencia, del Instituto Politécnico Nacional
Certificado de Reserva de Derechos al Uso Exclusivo del Título: 04-2010-0326124-14000-203 ISSN: 2007-1957

deterministas. En cada uno, como los datos viven en un espacio bidimensional cuenta con dos unidades de entrada y num_hidden unidades de salida. Estas características definen como se deben expandir los pesos y los sesgos. Ver fragmento de código en figura 3.

```
class BayesianMLPClassifier(PyroModule):
    def __init__(self, num_hidden=10, prior_std=1.):
        super().__init__()
        prior = Normal(0, prior_std)
        self.layer1 = PyroModule[torch.nn.Linear](2, num_hidden)
        self.layer1.weight = PyroSample(prior.expand([num_hidden, 2]).to_event(2))
        self.layer1.bias = PyroSample(prior.expand([num_hidden]).to_event(1))
```

Figura 3. Fragmento de código capa de entrada. Elaboración propia.

En la capa de salida se tienen tres posibles clases por lo que se requiere expandir los pesos y el sesgo. Se levanta los torch.nn.Linear con PyroModule indicando una capa totalmente conectada Bayesiana. La capa linear tiene atributos que son los pesos y el sesgo. Se establece la probabilidad a priori en los pesos y en los sesgos con PyroSample. La función de activación es tangente hiperbólica. Ver fragmento de código en figura 4.

```
self.layer3 = PyroModule[torch.nn.Linear](num_hidden, 3)
self.layer3.weight = PyroSample(prior.expand([3, num_hidden]).to_event(2))
self.layer3.bias = PyroSample(prior.expand([3]).to_event(1))

self.activation = torch.nn.Tanh()
```

Figura 4. Fragmento de código capa de salida. Elaboración propia.

En la sección de forward se operan los cálculos de la capa oculta y de la función de activación (z), posteriormente son aplicados a la segunda capa (f). Mediante el uso de "pyro.plate" se crea una placa para el mini batch, se implementa la verosimilitud y considerando que se tienen tres clases con valores 0, 1 y 2 (000, 001 y 010) se requiere entonces una variable categórica que espera como valor el *logp*. El resultado de la capa de salida de la red corresponde con la media de la distribución normal, para ello se emplea "pyro.deterministic", es lo que constituye la observación sobre la variable y. Ver figura 5.

```
def forward(self, x, y=None):
    h = self.activation(self.layer1(x))
    #h = self.activation(self.layer2(h))
    f = self.layer3(h).squeeze(1)
    with pyro.plate("data", size=x.shape[0]):
        logp = pyro.deterministic("logp", f, event_dim=1)
        obs = pyro.sample("obs", Categorical(logits=logp), obs=y) # Multiclass
        #obs = pyro.sample("obs", dist.Bernoulli(logits=p), obs=y) # Binary
    return f
```

Figura 5. Fragmento de código etapa forward. Elaboración propia.

2.3. Entrenamiento del modelo

Una vez creado el modelo, se requiere de una guía (aproximación a la probabilidad a posteriori). La guía se puede realizar de forma manual o automática utilizando "pyro.infer.autoguide". Se propuso una guía diagonal normal que asume la no correlación entre los parámetros de la BNN. Posteriormente, se crea un objeto SVI que cuenta con su modelo, guía, optimizador (Adam) y función de costo (TraceMeanField_ELBO). Con esto, es posible evaluar las probabilidades a posteriori de los parámetros así como el posteriori predictivo empleando "pyro.infer.Predictive". Después de tres mil épocas se actualizará el modelo, cada diez épocas se actualiza el posteriori predictivo para tener constancia de lo que va prediciendo la red. Ver figura 6.

```
pyro.enable_validation(True)
pyro.set_rng_seed(123)
pyro.clear_param_store()
model = BayesianMLPClassifier(num_hidden=100, prior_std=10.)

from pyro.infer.autoguide import AutoDiagonalNormal
guide = AutoDiagonalNormal(model, init_scale=1e-1)

svi = pyro.infer.SVI(model, guide,
                     optim=pyro.optim.ClippedAdam({'lr':1e-2}),
                     loss=pyro.infer.TraceMeanField_ELBO())

epoch_loss = np.zeros(shape=(3000,))
for k in tqdm(range(len(epoch_loss))):
epoch_loss[k] = svi.step(x_train, y_train)
if k % 100 == 0:
predictive = pyro.infer.Predictive(model, guide=guide, num_samples=10)
samples = predictive(torch.from_numpy(np.c_[xx.ravel(), yy.ravel()].astype('float32')))
update_plot(k, samples)
```

Figura 6. Fragmento de código entrenamiento de la BNN. Elaboración propia.

Se genera la probabilidad a posteriori predictivo con lo cual se pueden calcular estadísticas tal como la moda o la entropía, de forma que se puede observar el comportamiento de la variabilidad de las predicciones.

Humanidades, Tecnología y Ciencia, del Instituto Politécnico Nacional
Certificado de Reserva de Derechos al Uso Exclusivo del Título: 04-2010-0326124-14000-203 ISSN: 2007-1957

3. Pruebas y resultados

Las redes bayesianas tienen la capacidad de predecir la incertidumbre de un modelo, de modo que se planteó durante la fase de prueba que los puntos de entrada se ordenaran en función de la incertidumbre en la predicción. Posteriormente, se descartaron iterativamente las predicciones en función de su incertidumbre y se evaluó la precisión del nuevo modelo. La predicción de todos los datos, independientemente del grado de certeza arrojó una precisión de validación de aproximadamente el 93%. Se logró observar que al marcar solo el 5% de los datos, se obtenía un aumento en la precisión del modelo por arriba de un 2%. Durante la fase de entrenamiento se tomaron 100 muestras categóricas del posterior predictivo obteniéndose una clasificación en cada una de ellas, con esta información es posible obtener la moda, esto es, la clase que más se repite. En la figura 7a se ilustran las tres clases clasificadas: "Grave" en color rojo, "Moderado" en color naranja y "Sano" en color gris. Se observa que la escala está normalizada y que la clasificación presenta un comportamiento satisfactorio luego de las tres mil épocas. Adicionalmente en figura 7b, se muestra la entropía asociada e indica el grado de variabilidad en las predicciones. Se puede observar que el color es claro donde todos los modelos realizan la misma predicción, en tanto que, donde el color es más obscuro corresponde donde los modelos muestran mayor diferencia, esto es, mayor incertidumbre. Finalmente, en la figura 7c se muestra el ELBO o descenso del error que es calculado como neg log de la verosimilitud, y que sería equivalente a la entropía cruzada de una RNN.

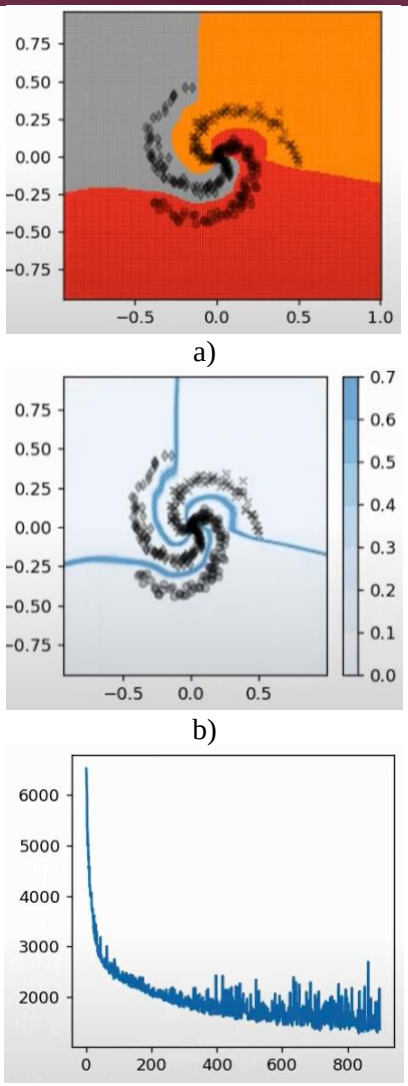

Figura 7. Se tomaron 100 muestras del posterior predictivo y se muestran: a) La moda, b) La entropía. y c) El ELBO. Elaboración propia.

4. Conclusiones

Las redes bayesianas resultaron ser un método importante no exclusivo en el sentido de que ofrecen un análisis cualitativo de los atributos y valores que se ven involucrados en el problema, sino también porque son capaces de dar cuenta de los aspectos cuantitativos de

esos atributos. Se pudo observar que la complejidad del clasificador bayesiano logró reducirse de forma significativa en espacio y tiempo de cálculo, sobre todo, cuando se consideran cada atributo condicionalmente independiente de los demás atributos. Adicionalmente, se puede mencionar que se logró interpretar adecuadamente la información de la base de datos proporcionada por los laboratorios Montecristo de Chalco Estado de México, a quienes se agradece su colaboración amplia y diligente. Se tomaron las informaciones de mil pacientes, para el entrenamiento de la red neuronal; de los cuales 700 fueron utilizados para la fase de entrenamiento de la red y 300 para la fase de prueba. Con base en los resultados obtenidos se puede mencionar que se logró una precisión superior al 90%.

Referencias

Aguilera Méndez Asdrúbal (2017). Esteatosis hepática no alcohólica: una enfermedad silente. Revista médica IMSS. 2018; 56(6):544-9.

Ghahramani, Z., & Beal, M. (2001). Propagation algorithms for variational bayesian learning. NIPS, (13), 507-513.

Graves Alex. (2014). Supervised sequence labelling with Recurrent Neural Network, Arxiv Preprint Arxiv: 1308.0850v5.

Guzmán Eduardo et al. (2018). Artificial Neural Networks: A Bayesian approach using parallel computing. Revista Colombiana de Estadística. Volume 41, Issue 2, pp. 173 to 189.

Ian Goodfellow, Yoshua Bengio y Aaron Courville. (2016). Deep learning book in preparation for MIT Press. En: URL http://www.deeplearningbook.org.

Wainwright, M., & Jordan, M. (2008). Graphical models, exponential families, and variational inference. Foundations and Trends in Machine Learning, 1(1), 1-305.