Proyecto 3: Redes Neuronales: Multilayer Perceptron

1st Carlos Villanueva Alcarraz
Maestria Computer Science
UTEC
Lima, Peru
carlos.villanueva.a@utec.edu.pe

2nd Renzo Alegre
Maestria Computer Science
UTEC
Lima, Peru
renzo.alegre@utec.edu.pe

3rd Sheila Stefany Ccahua Chavez
Maestria Computer Science
UTEC
Lima, Peru
sheila.ccahua@utec.edu.pe

I. Introducción

El objetivo de este informe es presentar la implementación de una red neuronal Multilayer Perceptron (MLP) para la clasificación de tumores malignos y benignos utilizando datos de la Universidad de Wisconsin el cual tiene etiquetas de B=Benigno y M=maligno. El proyecto se centra en entrenar la red MLP en un conjunto de datos que contiene información de 569 pacientes, identificando la presencia de tumores malignos o benignos a partir de características obtenidas de imágenes mamarias. El repositorio del proyecto se encuentra en el siguiente link: https://github.com/sheilaccahua202231011/Proyecto$_3.git$

II. Trabajos Relacionados

En el ámbito médico, el Multilayer Perceptron (MLP) ha demostrado ser una herramienta valiosa para el diagnóstico y la predicción de enfermedades, particularmente en el campo de la oncología y hematología. Desai y Shah (2020) exploraron el uso del MLP en la detección y diagnóstico del cáncer de mama, resaltando su capacidad para realizar clasificaciones binarias eficientes, una herramienta esencial en la identificación de células cancerosas como benignas o malignas [4]. Esta aplicación del MLP es significativa dada la importancia del diagnóstico temprano y preciso en el tratamiento del cáncer de mama.

Mojarad et al. (2010) llevaron a cabo un estudio en el que utilizaron el MLP para analizar un conjunto de biomarcadores en tumores de mama. Este enfoque subraya la habilidad del MLP para manejar relaciones no lineales y complejas entre marcadores, crucial para la predicción precisa de la progresión del cáncer de mama. El estudio demostró cómo el MLP puede ser eficaz en la interpretación de datos biomédicos complejos y en la predicción de resultados clínicos [5].

Por otro lado, Aeinfar et al. (2009) aplicaron el MLP para diagnosticar y predecir diversos trastornos sanguíneos y cáncer, utilizando resultados de análisis de sangre. Esta investigación destacó la eficiencia del MLP en la clasificación de patrones complejos y en la toma de decisiones médicas basadas en datos, proporcionando así una herramienta potencialmente poderosa para el diagnóstico clínico y la personalización del tratamiento [6].

Estas investigaciones ilustran el potencial del MLP para transformar el enfoque hacia el diagnóstico y tratamiento de enfermedades, aprovechando su habilidad para analizar y aprender de grandes conjuntos de datos clínicos, lo cual es crucial en la medicina moderna.

III. Explicación de la Arquitectura y Funcionamiento del MLP

La arquitectura del MLP consta de una capa de entrada, una o más capas ocultas y una capa de salida. La cantidad de neuronas en la capa de entrada es igual al número de características del conjunto de datos. Los hiperparámetros configurables incluyen el número de capas ocultas, el número de neuronas en cada capa oculta, el número de neuronas en la capa de salida y el tipo de función de activación.

El proceso de entrenamiento implica la propagación hacia adelante (forward propagation) de los datos a través de la red, seguido de la retropropagación del error (backpropagation) para ajustar los pesos de la red y minimizar la función de pérdida considerando que la función de pérdida utilizada es la Entropía Cruzada Binaria (Binary Cross-Entropy loss), que es comúnmente utilizada en problemas de clasificación binaria debido a que nuestra data implica solo dos posibilidades. Se ha implementado la posibilidad de elegir entre distintas funciones de activación, incluyendo Sigmoide, Tangente Hiperbólica (Tanh) y Rectified Linear Unit (RELU).

Ademas, para el diseño del MLP segun la base de datos dados por el presente proyecto tendremos en cuenta las siguientes consideraciones:

A. Número de Neuronas y Capas Ocultas:

- **Número de Neuronas en Capa de Entrada:** Debería ser igual al número de características o dimensiones en tus datos de entrada. Para la dataset presentada viene a tener 30 caracteristicas por cada paciente por lo que tendremos 30 neuronas en la capa de entrada.
- **Número de Neuronas en Capa de Salida:** Dependiendo del tipo de problema (clasificación binaria, clasificación multiclase, regresión, etc.), el número de neuronas en la capa de salida variará. Como en nuesta salida el problema abarca de identificar si el paciente

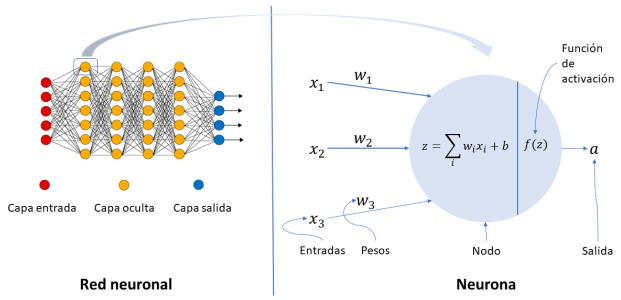


Fig. 1. Arquitectura del MLP

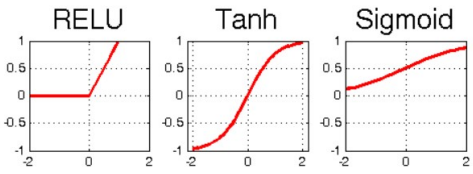


Fig. 2. Funciones de activacion

presenta un tumor benigno o maligno, reduciremos el problema a una clasficacion binaria.

- **Número de Neuronas en Capas Ocultas:**
  No hay reglas estrictas. Puedes empezar con una cantidad moderada y ajustar según sea necesario.
  Aumentar el número de neuronas puede aumentar la capacidad de la red para aprender patrones complejos, pero también puede llevar a un mayor riesgo de sobreajuste. Por eso se realizara un analisis exhaustivo con la experimentacion.
- **Número de Capas Ocultas:**
  Una sola capa oculta puede ser suficiente para problemas simples, mientras que problemas más complejos pueden beneficiarse de múltiples capas.
  La capacidad de generalización de la red a menudo mejora con más capas, pero también puede aumentar la complejidad computacional y el riesgo de sobreajuste. Por eso se realizara un analisis exhaustivo con la experimentacion.

B. Funciones de Activación:

- **Capa de Salida:**
  Para problemas de clasificación binaria, una función de activación sigmoide suele ser útil. Sin embargo, se utilizaran y compararan con las funciones de activacion de Tangente Hiperbólica (Tanh) y Rectified Linear Unit (RELU).
- **Capas Ocultas:**
  Se va a experimentar y comparar las funciones como Tanh, Sigmoid y RELU .

Para medir el rendimiento de nuestra red MLP, nos basaremos en:

C. Métricas para Ajuste:

- Error de Entrenamiento y Validación:
- Precisión (Accuracy):
- F1-Score, Precisión y Sensibilidad:
- Curvas ROC (Receiver Operating Characteristic) y AUC (Área bajo la curva):

IV. Red de Perceptrón Multicapa con Backpropagation y Binary Cross-Entropy Loss

A. Propagación hacia Adelante (Forward)

En la propagación hacia adelante, calculamos las activaciones y salidas en cada capa:

$$z^{(l)} = W^{(l)} a^{(l-1)} + b^{(l)}$$

$$a^{(l)} = \sigma(z^{(l)})$$

Donde:

- l es el índice de la capa.
- $W^{(l)}$ es la matriz de pesos.
- $a^{(l-1)}$ es la salida de la capa anterior.
- $b^{(l)}$ es el vector de sesgo.
- σ es la función de activación (por ejemplo, la función sigmoide).

B. Función de Pérdida Binary Cross-Entropy

La función de pérdida Binary Cross-Entropy mide la discrepancia entre las predicciones y las etiquetas reales:

$$L(y, \hat{y}) = -\frac{1}{N} \sum_{i=1}^{N} [y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)]$$

Donde:

- N es el número de ejemplos.
- y_i es la etiqueta verdadera.
- $\hat{y}_i$ es la salida predicha por la red.

C. Retropropagación (Backpropagation)

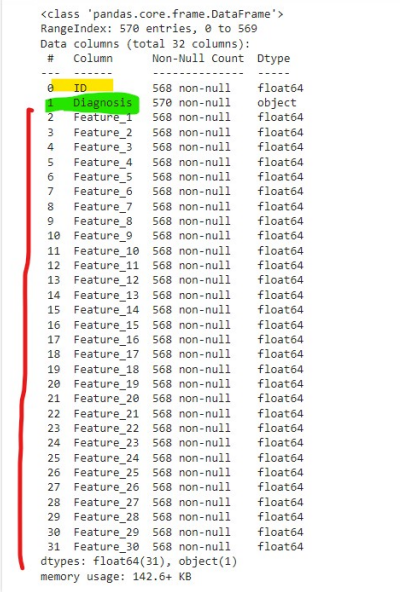
Durante la retropropagación, actualizamos los pesos y sesgos para minimizar la función de pérdida. Las actualizaciones se realizan mediante el descenso del gradiente:

$$W^{(l)} = W^{(l)} - \alpha \frac{\partial L}{\partial W^{(l)}}$$

$$b^{(l)} = b^{(l)} - \alpha \frac{\partial L}{\partial b^{(l)}}$$

Donde:

- α es la tasa de aprendizaje.
- $\frac{\partial L}{\partial W^{(l)}}$ y $\frac{\partial L}{\partial b^{(l)}}$ son las derivadas parciales de la función de pérdida respecto a los pesos y sesgos de la capa l, respectivamente.



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 570 entries, 0 to 569
Data columns (total 32 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   ID          568 non-null    float64
 1   Diagnosis   570 non-null    object
 2   Feature_1   568 non-null    float64
 3   Feature_2   568 non-null    float64
 4   Feature_3   568 non-null    float64
 5   Feature_4   568 non-null    float64
 6   Feature_5   568 non-null    float64
 7   Feature_6   568 non-null    float64
 8   Feature_7   568 non-null    float64
 9   Feature_8   568 non-null    float64
 10  Feature_9   568 non-null    float64
 11  Feature_10  568 non-null    float64
 12  Feature_11  568 non-null    float64
 13  Feature_12  568 non-null    float64
 14  Feature_13  568 non-null    float64
 15  Feature_14  568 non-null    float64
 16  Feature_15  568 non-null    float64
 17  Feature_16  568 non-null    float64
 18  Feature_17  568 non-null    float64
 19  Feature_18  568 non-null    float64
 20  Feature_19  568 non-null    float64
 21  Feature_20  568 non-null    float64
 22  Feature_21  568 non-null    float64
 23  Feature_22  568 non-null    float64
 24  Feature_23  568 non-null    float64
 25  Feature_24  568 non-null    float64
 26  Feature_25  568 non-null    float64
 27  Feature_26  568 non-null    float64
 28  Feature_27  568 non-null    float64
 29  Feature_28  568 non-null    float64
 30  Feature_29  568 non-null    float64
 31  Feature_30  568 non-null    float64
dtypes: float64(31), object(1)
memory usage: 142.6+ KB
```

Fig. 3. Datainfo de pacientes con tumor

V. Técnicas de Regularización

Para mejorar la capacidad de generalización y prevenir el sobreajuste en la red neuronal Multilayer Perceptron (MLP), se han implementado dos técnicas de regularización: la penalización en la actualización de pesos y el dropout.

A. Regularización de Penalización en la Actualización de Pesos

La regularización de penalización en la actualización de pesos se introduce para controlar la magnitud de los pesos en la red. La función de pérdida modificada con términos de penalización $L_{\text{penalizada}}$ se define como:

$$L_{\text{penalizada}} = L + \lambda \sum_{i=1}^{N} \|W_i\|^2$$

donde L es la función de pérdida original, λ es el parámetro de penalización y W_i son los pesos de la red. La adición de $\lambda \sum_{i=1}^{N} \|W_i\|^2$ penaliza los pesos más grandes durante la optimización, lo que ayuda a prevenir el sobreajuste.

Durante el proceso de retropropagación, las derivadas parciales de esta penalización se suman a las derivadas de la función de pérdida original. El término λ controla la fuerza de la penalización.

B. Dropout

El dropout es una técnica de regularización que implica desactivar aleatoriamente un porcentaje de las neuronas durante el entrenamiento. Esto evita que las neuronas se vuelvan dependientes de la presencia de otras neuronas específicas y mejora la generalización del modelo.

Durante el entrenamiento, cada neurona se desactiva con una probabilidad p. La salida de la neurona se multiplica por $1/(1-p)$ durante el entrenamiento para compensar la desactivación. La implementación del dropout se realiza típicamente en las capas ocultas de la red.

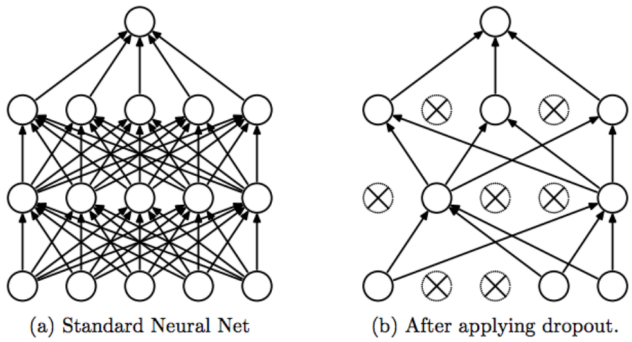


Fig. 4. Aplicación de Dropout

La probabilidad p es un hiperparámetro configurable que controla la proporción de neuronas desactivadas. Un valor común es $p = 0.5$, pero se puede ajustar según la complejidad del modelo y la cantidad de datos disponibles.

Se realizaron experimentos variando los valores de λ y p para evaluar el impacto en métricas de rendimiento como el error de entrenamiento y validación, la precisión y el F1-score. La selección de los valores óptimos se basó en maximizar la precisión y el F1-score mientras se minimiza el sobreajuste.

Los resultados experimentales indican que la combinación adecuada de regularización de penalización y dropout puede mejorar significativamente el rendimiento del modelo, permitiendo una mejor generalización en la clasificación de tumores benignos y malignos.

VI. Experimentos

A. Carga y Análisis del Dataset

La carga del conjunto de datos reveló que consta de 569 observaciones, cada una con 30 variables. La primera columna contiene el ID del paciente, la segunda indica el diagnóstico (M/B), y las 30 columnas subsiguientes son características médicas derivadas de imágenes. Como se observa en la figura 5. Ademas, debido a que el diagnostico de si el tumor es benigno o maligno se clasifican con B o M respectivamente, para el presente proyecto y que pase a ser utilizado por la red MLP cambiaremos el etiquetado donde:

- **Tumor Benigno=0**
- **Tumor Maligno=1**

En la figura 6 se muestra que 357 personas presentan un tumor benigno (0) y 212 presentan tumor maligno (1)

B. Separación del Conjunto de Datos y Proceso de Entrenamiento

Con el objetivo de evaluar el rendimiento del modelo, se dividió el conjunto de datos en dos partes: el 70% destinado entre entrenamiento y validacion (50% y 25 % respectivamente) y el 30% reservado para las pruebas. El split del 70% se da porque se quiere analizar si el modelo se encuentra

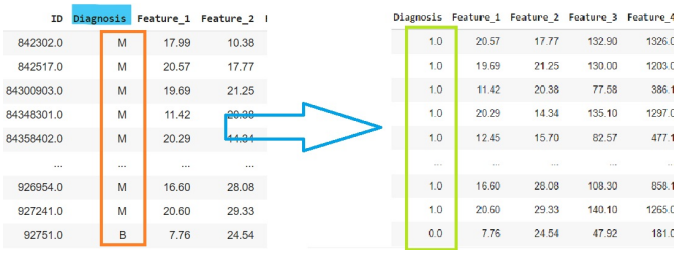


Fig. 5. Cambio de etiqueta

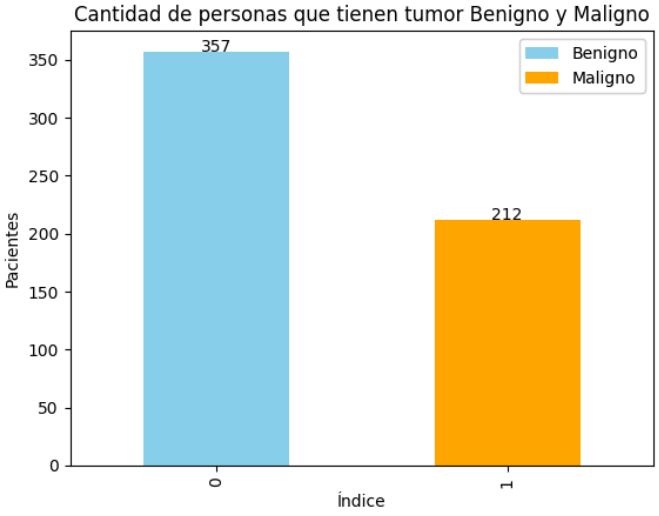


Fig. 6. Cantidad de personas que tienen tumor Benigno y Maligno.

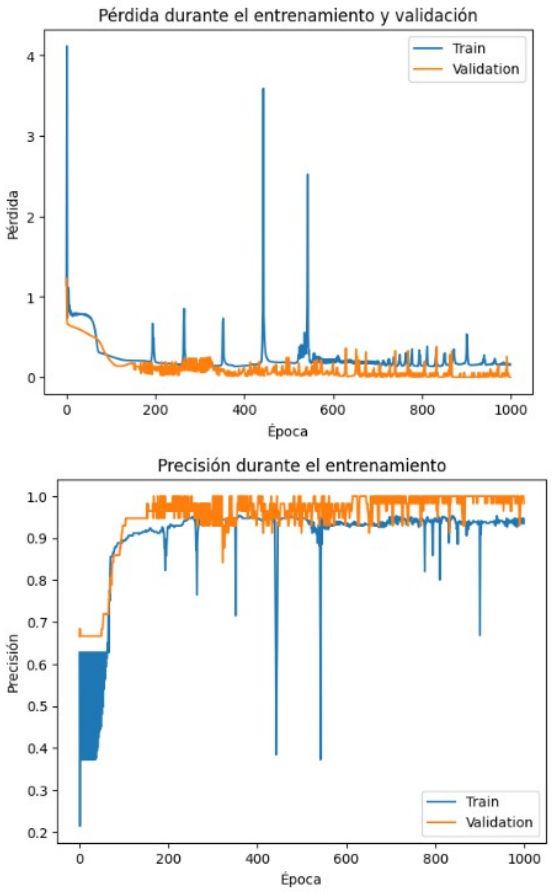


Fig. 7. Pérdida y Precisión durante el entrenamiento y validación.

con overfitting o underfitting como se pasara a mostrar en los diferentes modelos implementados.

1) MODELO 1: El modelo 1 consiste en una red de 6 capas con el siguiente esquema:

- 1 Capa de entrada con 30 neuronas
- 4 capas ocultas con 10 neuronas cada una
- 1 capa de salida con 1 neurona

Como se observa en la figura 7, no viene a representar un modelo correcto de clasificaion de MLP a pesar de que tiene un valor elevado de precision tal como se ve en la figura 8 que tiene una precision de 91.23%. Descartado este modelo, no se hara intentos de cambiar los hiperparametros de regularizacion y reduciremos la complejidad disminuyendo la cantidad de capas tal como se pasara a explicar el modelo 2.

2) MODELO 2: El modelo 2 consiste en una red de 4 capas con el siguiente esquema:

- 1 Capa de entrada con 30 neuronas
- 2 capas ocultas con 10 neuronas cada una
- 1 capa de salida con 1 neurona
- la funcion de activacion de todas las capas son sigmoid

Modificaremos lo que es el learning rate que viene a ser la tasa de aprendizaje un hiperparámetro crucial en el entrenamiento de redes neuronales. Asignamos un valor learning rate=0.001. Segun la figura 9 se observa que la convergencia es lenta con una posibilidad de convergencia a mínimos locales.

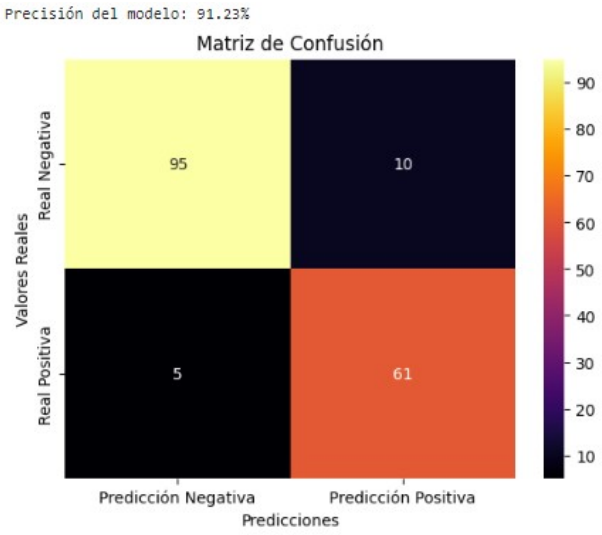


Fig. 8. Matriz de Confusión.

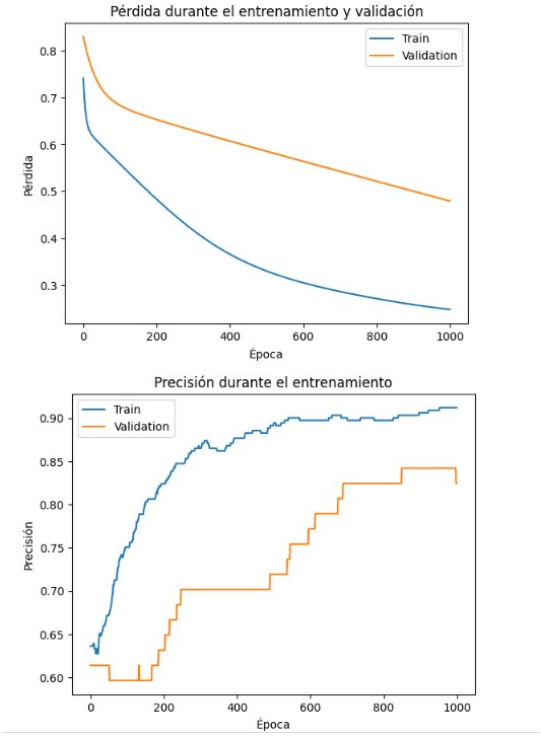


Fig. 9. Modelo 2 Learning rate 0.001

La precision tiene una diferenciacion marcada entre el train y el validation.

Con un learning rate= 0.01 sigue demorando en coonverger, ademas presenta mayor error en la etapa de train figura 10.

Con un learning rate= 0.1 se observa una convergencia mas rapida a las 100 epocas aproximadamente, ademas el modelo aprende ya que a travez de las epocas el error que estos presentan van disminuyendo hasta llegar a cierto punto figura 11.

Un learning rate= 0.64 nos muestra una funcion de perdida oscilatoria inestable figura 12 Concluyendo asi que un learning rate para el modelo 2 llevaria un valor de 0.1

3) MODELO 3: El modelo 3 consiste en una red de 4 capas con el siguiente esquema:

- 1 Capa de entrada con 30 neuronas
- 2 capas ocultas con 10 neuronas cada una
- 1 capa de salida con 1 neurona
- la funcion de activacion de todas las capas son tanh

Se observa que utilizando la misma tasa de aprendizaje la señal de learning rate= 0.1 empieza a oscilar figura 13, por lo que para este modelo debemos realizar una busqueda de este valor. Realizando una isnpeccion el valor adeucado usando en esta funcion de activacion de tanh es utilizando un Learning rate= 0.01. Es notorio ver que tiene una convergencia rapida de aprendizaje, segun muestra la figura 14

4) MODELO 4: El modelo 4 consiste en una red de 4 capas con el siguiente esquema:

- 1 Capa de entrada con 30 neuronas
- 2 capas ocultas con 10 neuronas cada una

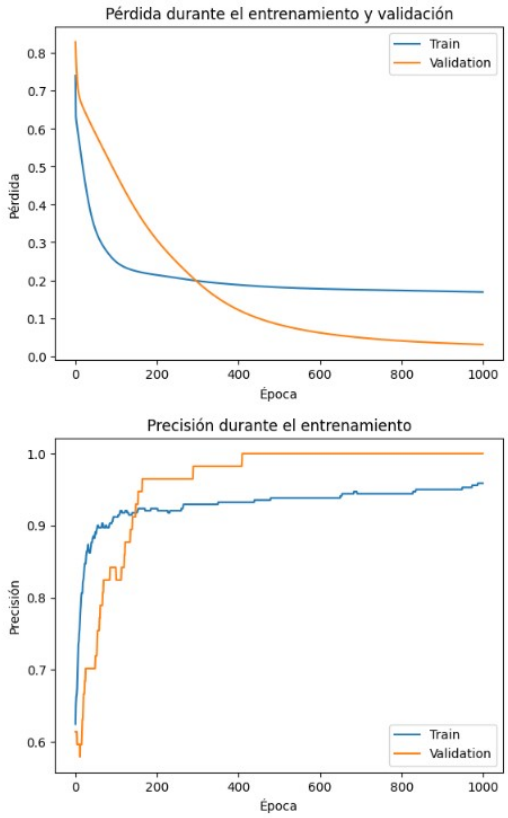


Fig. 10. Modelo 2 Learning rate 0.01

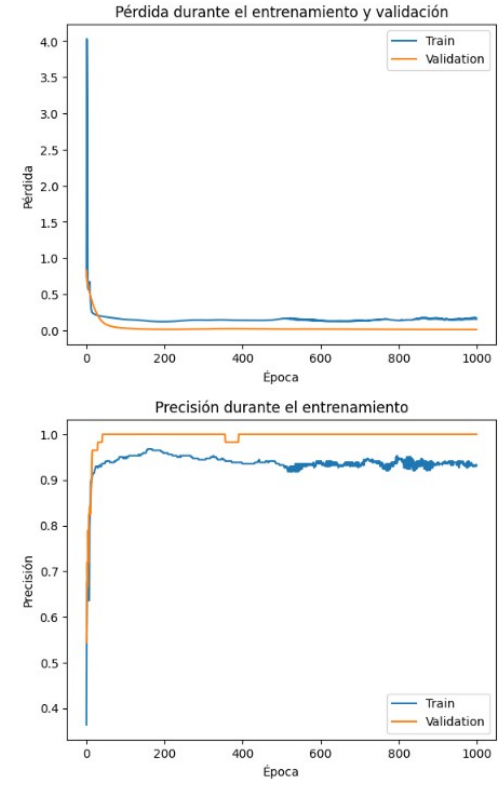


Fig. 11. Modelo 2 Learning rate 0.1

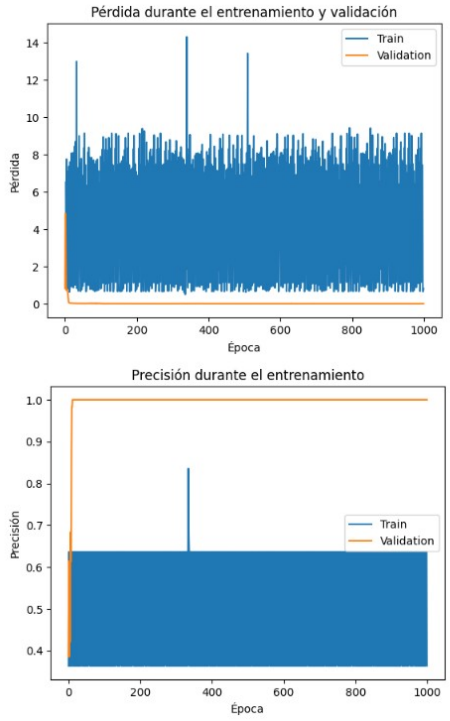


Fig. 12. Modelo 2 Learning rate 0.64

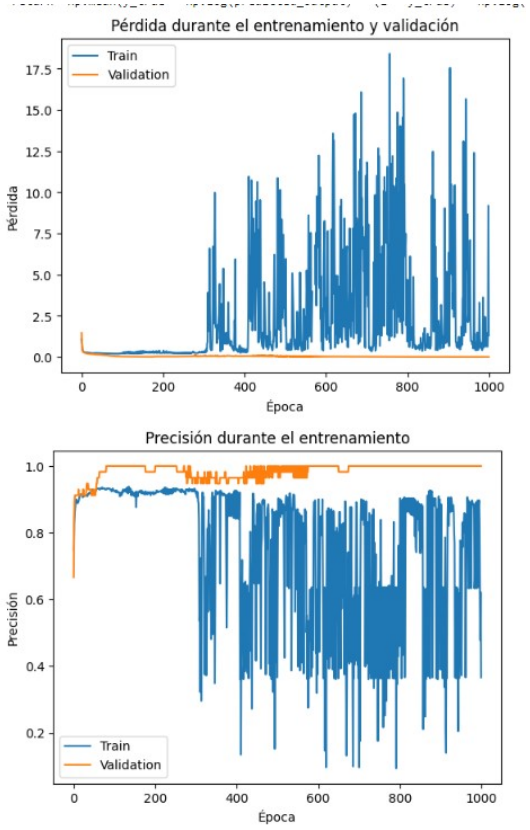


Fig. 13. Modelo 3 Learning rate 0.1

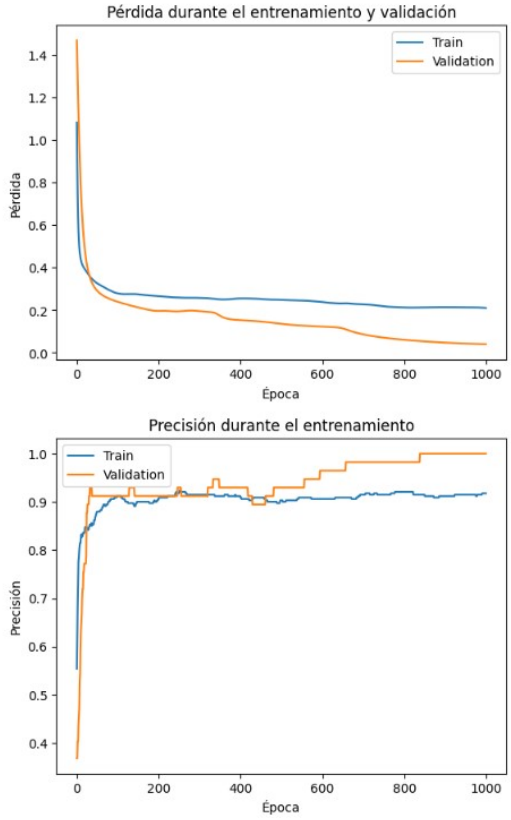


Fig. 14. Modelo 3 Learning rate 0.01

- 1 capa de salida con 1 neurona
- la funcion de activacion de todas las capas son RELU

	Accuracy
Sigmoid	0.918
Tanh	0.929
Relu	0.847

5) Resultados de testing entre Sigmoid, Tanh y Relu: En la tabla 1 podemos ver la diferencia de accuracy , mostrando que Tanh viene a tener un buen rendimiento al momento de testear la data.

Mostramos tambien las matriz de confusion con las diferentes funciones de activacion.

VII. Concluciones

- Se observó que la Configuración 2 con dos capas ocultas (10 neuronas cada una) proporcionó un equilibrio entre complejidad del modelo y capacidad de generalización.
- La función de activación tanh (Modelo 3) y Sigmoid (Modelo 2) demostraron ser más efectivas que la sigmoid (Modelo 4) en términos de rendimiento y mitigación del problema de desvanecimiento del gradiente.
- Ajustar la tasa de aprendizaje puede influir en la velocidad con la que el modelo converge. Una tasa de aprendizaje demasiado alta puede hacer que el modelo diverja, mientras que una tasa demasiado baja puede ralentizar la convergencia

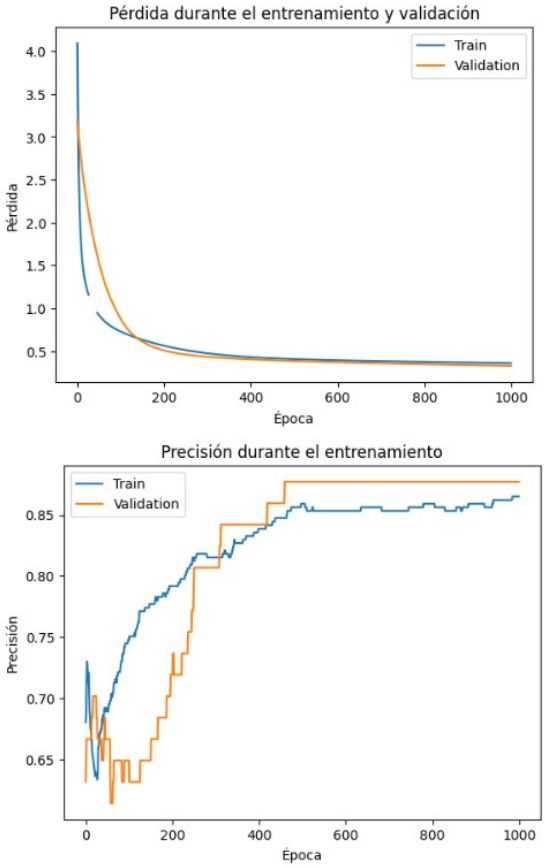


Fig. 15. Modelo 4 Learning rate 0.0001

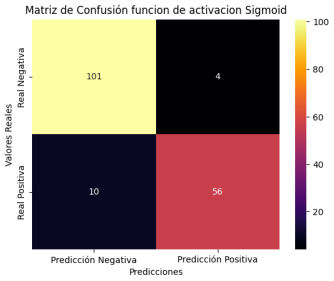


Fig. 16. Matriz de confusion funcion de activacion Sigmoid

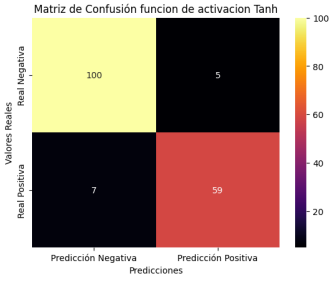


Fig. 17. Matriz de confusion funcion de activacion Tanh

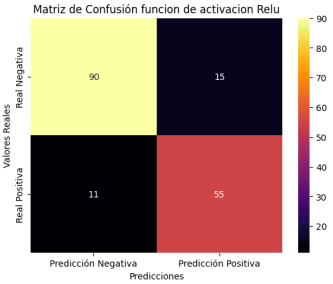


Fig. 18. Matriz de confusion funcion de activacion Relu

- ReLU y tanh mostró una convergencia más rápida durante el entrenamiento
- Es crucial monitorear las métricas de rendimiento en conjuntos de entrenamiento y prueba para evitar sobreajuste o subajuste.
- La elección adecuada de capas y neuronas depende de la complejidad del conjunto de datos y la tarea de clasificación. En este caso, dos capas ocultas resultaron ser una elección equilibrada a diferencia del Modelo 1.

REFERENCES

[1] Taud, H., Mas, J.F. (2018). Multilayer Perceptron (MLP). En M.T. Camacho Olmedo, Geomatic Approaches for Modeling Land Change Scenarios (pp. 451-455). Springer International Publishing. https://doi.org/10.1007/978-3-319-60801-3-27.

[2] Mohammadi, J., Ataei, M., Khaloo Kakaie, R., Mikaeil, R., Shaffiee Haghshenas, S. (2018). Prediction of the Production Rate of Chain Saw Machine using the Multilayer Perceptron (MLP) Neural Network. Civil Engineering Journal, 4(7). https://doi.org/10.28991/cej-0309196.

[3] Mia, M. M. A., Biswas, S. K., Chowdhury Urmi, M., Siddique, A. (2015). An Algorithm For Training Multilayer Perceptron (MLP) For Image Reconstruction Using Neural Network Without Overfitting. International Journal of Scientific Technology Research, 4(02).

[4] Desai, M., Shah, M. (2020). An anatomization on Breast Cancer Detection and Diagnosis employing Multi-layer Perceptron Neural Network (MLP) and Convolutional Neural Network (CNN). Clinical eHealth. https://doi.org/10.1016/j.ceh.2020.11.002.

[5] Mojarad, S. A., Dlay, S. S., Woo, W. L., Sherbet, G. V. (2010). Breast Cancer Prediction and Cross Validation Using Multilayer Perceptron Neural Networks.

[6] Aeinfar, V., Mazdarani, H., Deregeh, F., Hayati, M., Payandelr, M. (2009). Multilayer Perceptron Neural Network with Supervised Training Method for Diagnosis and Predicting Blood Disorder and Cancer.