DEPARTAMENTO
DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA

# Trabajo Práctico I

## Redes Neuronales Artificiales

Redes Neuronales
Primer Cuatrimestre de 2022


| Integrante | LU | Correo electrónico |
|---|---|---|
| Lucas Iván Kruger | 799/19 | Lucaskruger10@gmail.com |
| Sebastián Cantini Budden | 576/19 | sebascantini@gmail.com |
| Juan Cruz Barcos | 463/20 | juancruzbarcos@hotmail.com |





Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires
Ciudad Universitaria - (Pabellón I/Planta Baja)
Intendente Güiraldes 2160 - C1428EGA
Ciudad Autónoma de Buenos Aires - Rep. Argentina
Tel/Fax: (54 11) 4576-3359
http://www.fcen.uba.ar

# 1. Introducción

En el presente informe vamos a estudiar la implementación de un perceptrón multicapa en Python utilizando la libreria NumPy, la cual está altamente optimizada para la multiplicación de matrices. Por lo tanto, nos habilita entrenar la red neuronal más rápido. Esta implementación de bajo nivel de un perceptrón multicapa está adaptada al pseudocódigo visto en clase, que nos permite entender mejor los mecanismos detrás de esta. Utilizaremos dos datasets, uno que nos permitirá entrenar la red para clasificar tumores en benigno o maligno. Mientras que el otro nos permitirá determinar los requerimientos de carga energética para la calefacción y refrigeración de edificios en función de ciertas características de los mismos.

Variamos parámetros como la cantidad de capas, la cantidad de neuronas o el learning rate, con el fin de aprender sus impactos e implicaciones y así tenerlas en cuenta para futuras investigaciones y proyectos.

# 2. Implementación

Optamos por crear una clase modelo en Python, la cual guarda las matrices de peso, la cantidad de neuronas por capa, las funciones a usar en cada capa y el learning rate necesario para realizar un entrenamiento y predicción. Usamos la función train y predict enviando como parámetros los datos necesarios para operar con nuestro modelo.

Este formato fue elegido por su practicidad ya que podremos conservar el modelo de una forma práctica y declarativa para un mejor entendimiento del proceso.

# 3. Experimentación

## 3.1. Diagnostico de cancer de mamas

Este primer conjunto de datos contiene los resultados de un examen diagnóstico de cáncer de mamas. Como mencionamos anteriormente, buscamos implementar una red neuronal que pueda clasificar los tumores en dos categorías; benignos y malignos.

Comenzamos buscando la mejor configuración para nuestro modelo. Cada diagnostico dentro del dataset contiene diez características, y al solo tener que clasificar en una u otra categoría, solo vamos a obtener un único valor de salida. Por ende, nuestra capa interna y externa serán diez y uno respectivamente para toda configuración. Utilizamos un learning rate constante de 0.2 para las configuraciones a medir.

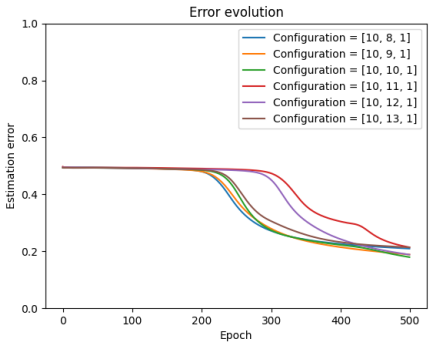


Figura 1: Resultados de distintas configuraciones con una capa interna.

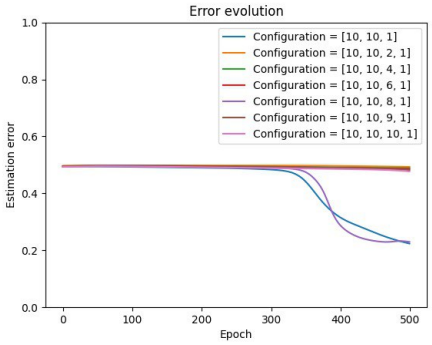


Figura 2: Resultados de distintas configuraciones con dos capas internas. Incluye la configuración [10, 10, 1].

Particionamos este experimento en dos para simplificar los conjuntos de resultados, por un lado, tenemos las configuraciones con una única capa interna. Mientras por el otro, tenemos los resultados con dos. Viendo la *figura 1* podemos notar como luego de diez capaz internas empieza a aparecer una pérdida de velocidad de aprendizaje. Por lo cual, resulta lógico que hay un cierto de límite de cuantas neuronas nos conviene utilizar para la capa interna. Se puede ver como luego de 500 épocas, todos los modelos generan un error similar. En otras palabras, más neuronas nos lleva a necesitar más épocas de entrenamiento sin ninguna ventaja notable. Optamos por quedarnos con la configuración de [10, 10, 1] ya que esta fue la que nos brindó mejores resultados.

Para el experimento de configuraciones de dos capas internas, incluimos la configuración seleccionada anteriormente para ver como compara con las nuevas configuraciones y así poder elegir la mejor. La *figura 2* nos brinda resultados interesantes. Por un lado, casi todas las configuraciones nuevas no tienen mejora en las primeras 500 épocas. Tiene sentido dado que, al tener más capas y una gran cantidad de neuronas, estas tardarían más iteraciones en corregirse. Sin embargo, la configuración mas pesada ([10, 10, 10, 1]) tuvo un aprendizaje similar a la configuración de una capa seleccionada anteriormente, la cual sorprendentemente tardo más épocas en reducir error que en la *figura 1*.

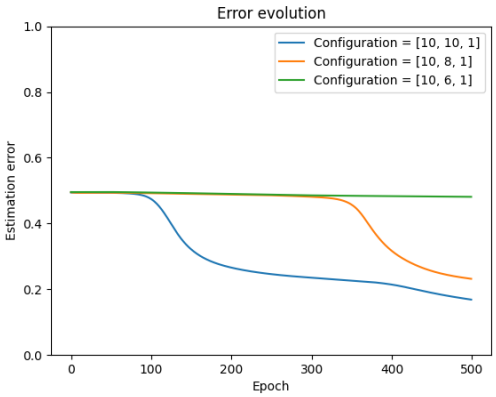


Figura 3: Mejores configuraciones según los experimentos anteriores.

Como vimos en la *figura 2* la configuración de [10, 10, 1] necesito una cantidad de épocas adicionales, por lo que volvimos a correr el experimento con las mejores configuraciones para ver si se mantenía esta observación. Mientras existen las inconsistencias en los resultados, eso viene de que, al generar los pesos de las matrices, se utilizan valores al azar, por lo que tiene sentido que la velocidad de aprendizaje sea dispareja entre un entrenamiento y otro. Por suerte, la mejor configuración sigue siendo [10, 10, 1] por lo cual esta será la que utilizaremos.

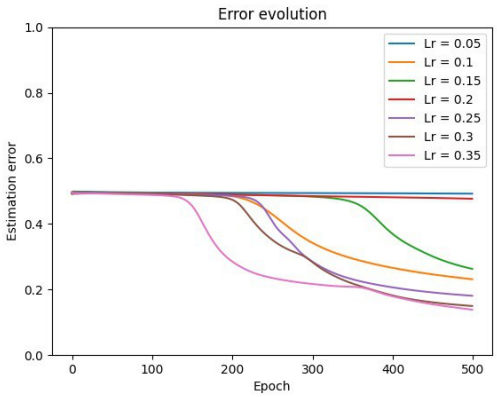


Figura 4: Resultados de uso de distintos learning rates en el modelo con configuración de [10, 10, 1].

Buscando los distintos learning rates creímos que uno menor nos daría mejores resultados, aunque tardaría más en converger. Hay que notar que cuanto menor sea el learning rate, hay una mayor probabilidad de atorarse en mínimos locales. Por otro lado, mientras más grande sea, más difícil resulta mantenerlo en un mínimo local o absoluto. Lógicamente el más rápido en mostrar progreso es el learning rate más alto, sin embargo, no creímos que daría los mejores resultados. Viendo el progreso de $lr = 0{,}35$ notamos que alrededor de la época 350, llega a un mínimo local, esto se ve en la perpendicularidad entre la línea y el eje de error. Pero al ser lo suficiente mente grande, pudo salir y mejorar el entrenamiento. Se ve una situación similar con $lr = 0{,}30$ alrededor de la época 300. Por otro lado, $lr = 0{,}05$ y $lr = 0{,}20$ no mostraron progreso. Esto se puede deber a que se encuentran en un mínimo local y el learning rate no es lo suficientemente alto como para salir.

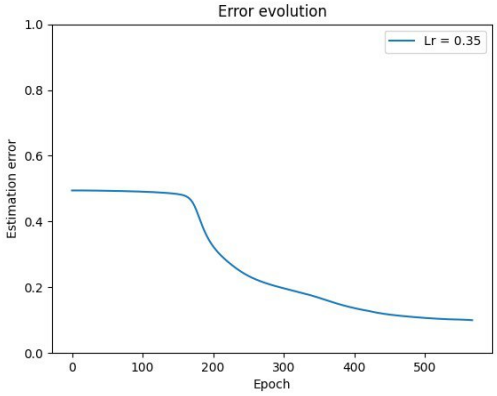


Figura 5: Resultados de convergencia de configuración [10, 10, 1] y learning rate 0.35.

Finalmente, hicimos un último experimento con este conjunto de datos para ver cuanto tardaría en converger nuestro modelo con el modelo elegido y el mejor learning rate encontrado. Consideramos que el modelo converge cuando el error es menor a 0.1. Este modelo tardo 568 épocas en converger y sigue un patrón similar a los gráficos anteriores, empezando a aprender alrededor de la época 200 y estando muy cerca de la meta alrededor de 500 épocas.

## 3.2. Eficiencia energética

El segundo conjunto de datos contiene características de distintos edificios y buscamos, a partir de estos determinar los requerimientos de carga energética para la calefacción y refrigeración de los mismos. Hay un par de diferencias importantes a notar en relación al set de datos anterior. Primero y principal

estamos calculando carga energética, esto implica que el problema no es uno de clasificación como el anterior. La segunda diferencia importante es que buscamos la carga energética para la calefacción por un lado y la refrigeración por el otro. Por lo que nuestra red, requiere dos salidas.

Similar al conjunto de datos anterior comenzaremos buscando la mejor configuración. Empezando con una capa con ocho neuronas (dado que tenemos ocho características por edificio) y finalizando con una capa con dos neuronas. El error para este conjunto de datos es medido como el promedio de los errores de cada salida. Utilizaremos un learning rate de 0.3 para estos primeros experimentos, una vez elegida la configuración, pasaremos a buscar el mejor learning rate para esa configuración.

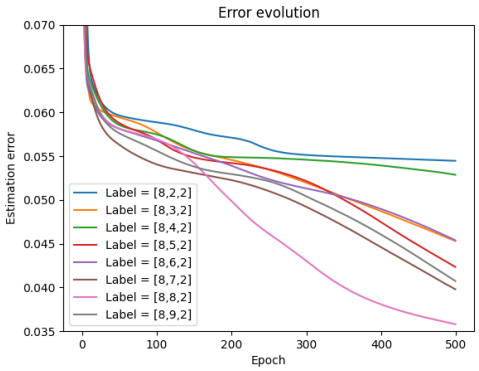


Figura 6: Resultados de distintas configuraciones con una capa interna.

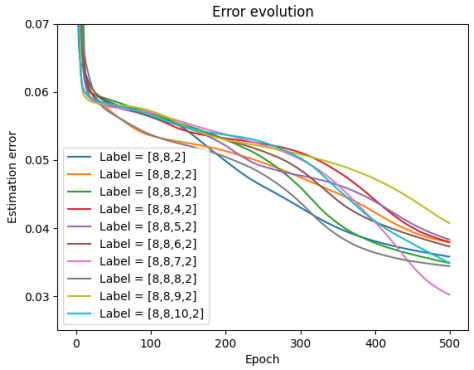


Figura 7: Resultados de distintas configuraciones con dos capas internas. Incluye la configuración [8, 8, 2].

Similar a los resultados obtenidos de *figura 1*, la *figura 6* muestra que la configuración con mejores resultados es la que tiene la misma cantidad de neuronas en la capa interna como en la capa de entrada. Se ve una mejora cuantas más neuronas hasta las ocho neuronas. Con este set de datos esto se ve más claro el efecto de cada neurona. La complejidad de la salida esperada requiere más neuronas para calcular más precisamente.

Viendo la *figura 7* notamos una mejora significante con mínimos menores a los de la *figura 6*. La complejidad de lo esperado es mayor a los que buscábamos con el primer conjunto de datos y tiene más salidas, por lo que se beneficia de esa capa adicional. La configuración de una capa interna elegida logra vencer a algunas de las configuraciones de dos capas que en su mayoría tienen menos de 7 neuronas en su segunda capa. Similar a lo visto con la primera capa, se encontró que la mejor cantidad de neuronas para las capas internas fue similar a la cantidad de neuronas a la capa de entrada.

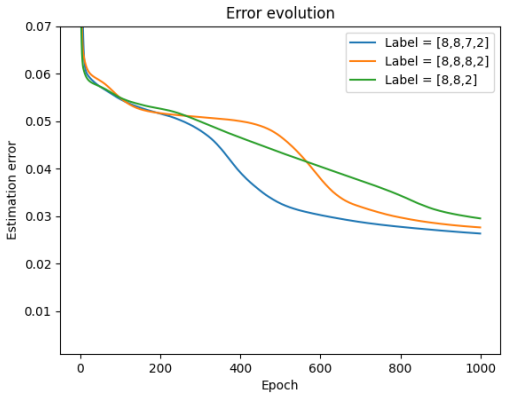


Figura 8: Mejores configuraciones según los gráficos anteriores.

Por las mismas razones que con el conjunto de datos anterior, decidimos volver a entrenar mejores configuraciones, particularmente agregamos [8, 8, 2] para ver si un nuevo entrenamiento brindaba nuevas conclusiones. Sin embargo, ambas configuraciones con dos capas internas lograron superar a esa configuración. La configuración [8, 8, 7, 2] fue consistentemente la mejor de todas las probadas, por ende, es la que utilizaremos.

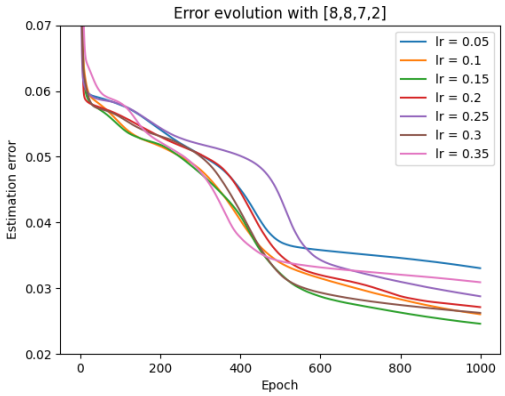


Figura 9: Resultados de uso de distintos learning rates en el modelo con configuración de [8, 8, 7 2].

La búsqueda del learning rate fue similar a la que hicimos para el primer conjunto de datos. Sin embargo, los resultados fueron distintos, pero con similitudes. Primero notamos que con todos los learning rates se converge con una velocidad similar. Nuevamente el $lr = 0{,}05$ se es el más lento, sin embargo, es seguido por $lr = 0{,}35$, el cual había sido nuestro mejor learning rate para el conjunto de datos anterior. Creemos que, al incrementar la cantidad de capas internas, es mejor tener un menor learning rate ya que este afecta a todas las capas. Es decir que si se elige un learning rate muy grande todas serán impactadas generando más error. Utilizando nuestra configuración de dos capas internas notamos que el mejor learning rate es $lr = 0{,}15$.

# 4. Conclusión

Comparando los resultados de las búsquedas de las configuraciones de ambos conjuntos de datos podemos concluir que la estructura está relacionada con la complejidad de lo que el modelo intenta predecir o clasificar. Por un lado, la clasificación de los tumores cancerígenos es una clasificación binaria, puede ser benigno o maligno, por lo que tener muchas capas es innecesario. Cuando buscamos predecir la eficiencia energética, no buscamos clasificar en una cantidad fija de grupos. Los valores se mueven en un rango y buscamos calcular el valor correcto dentro de ese rango, lo cual es más complejo que solo una clasificación binaria. Es fácil ver como esta se beneficia de una segunda capa, en especial cuando tiene menos características que el primer conjunto de datos. Las capas adicionales ayudan a distinguir distintos patrones que ayudan a la capa final a predecir el valor correcto.

El learning rate recomendado depende de la cantidad de capas y la complejidad del espacio de solución. Nuestro primer conjunto solo podemos obtener benigno o maligno, por lo que un learning rate alto nos benefició bastante. Nuestro segundo conjunto buscaba un valor dentro de un rango y tenía más capas internas, por ende, era necesario un progreso más lento y controlado.