



DEPARTAMENTO  
DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA

# Redes Neuronales

## Trabajo práctico 1

### Parser

#### *Resumen*

#### *Entrenamiento de Redes Neuronales*

Integrante	LU	Correo electrónico
Negri, Franco	893/13	franconegri2004@hotmail.com
?, ?	?/?	?@?.com

Palabras claves:

TP

## Índice

<b>1. Introduccion</b>	<b>3</b>
<b>2. Desarrollo</b>	<b>3</b>
2.1. Preprocesamiento De Los Datos . . . . .	3
2.2. Implementacion Del algoritmo . . . . .	3
2.3. Experimentación sobre datos de Diagnostico de Cancer . . . . .	3
2.3.1. Convergencia del algoritmo . . . . .	3
2.3.2. Performance Vs Learning Rate . . . . .	4
2.3.3. Performance Vs Cantidad De Neuronas . . . . .	5
2.3.4. Performance Vs Cantidad De iteraciones . . . . .	5
2.4. Eficiencia energética . . . . .	6
2.4.1. Convergencia del algoritmo . . . . .	6
2.4.2. Performance Vs Learning Rate . . . . .	6
2.4.3. Cantidad De neuronas Vs Performance . . . . .	7
2.4.4. Epocas Vs Performance . . . . .	7
<b>3. Conclusiones</b>	<b>8</b>

## 1. Introduccion

En este trabajo se intentara utilizar tecnicas de redes neuronales profundas con la intención de resolver dos problemas diferentes:

En el primer problema, contaremos con características de imágenes de células y por otro lado con el diagnóstico final de si resultaron ser cáncer o no cáncer.

Para el segundo problema intentaremos predecir la carga de calefacción y la carga de refrigeración de un edificio a partir de ocho características dadas de un edificio.

Para ambos problemas utilizaremos una red neuronal con retro-propagación del error a la que se entrenara con instancias de estos datos para ver si son generalizables. Además se intentaran buscar los parámetros adecuados (cantidad de neuronas, cantidad de capas, learning rate, momentum) para mejorar la performance de la misma.

## 2. Desarrollo

### 2.1. Preprocesamiento De Los Datos

Con el objetivo de darle mayor estabilidad matemática a los datos y quitar algo del ruido que puedan tener, realizamos un preprocesamiento de los mismos. El mismo consistió en quitar los outliers dentro del set de datos y luego normalizarlos con media 0 y varianza 1.

### 2.2. Implementacion Del algoritmo

El algoritmo que implementamos consistió en una red neuronal profunda con retropropagación del error. Este, descrito de manera informal consiste en introducir una instancia del problema, calcular la salida, compararla con la salida esperada y retropropagar el error, con el objetivo de modificar las matrices de pesos y minimizar la diferencia. El sistema implementado se dice ser *online* ya que los pesos de las matrices son ajustados luego de retropropagar el error de cada instancia en vez de hacerlo al final de la época.

A fin de tener una medida de como va “aprendiendo” la red neuronal, luego de presentarle a la red todas las instancias una vez, calculamos la norma de la diferencia entre las respuestas obtenidas y las esperadas. Consideraremos esta nuestra “norma del error” la tomaremos como una métrica adecuada para saber cuan buenos resultados devuelve nuestra red.

Como optimización adicional se agregó un término de momentum. La idea tras esto consiste en darle a cada peso de la matriz una “inercia” que le permita continuar avanzando en la dirección en la que avanzó en la iteración anterior. El objetivo consiste en disminuir las oscilaciones con cada pequeño cambio en la matriz.

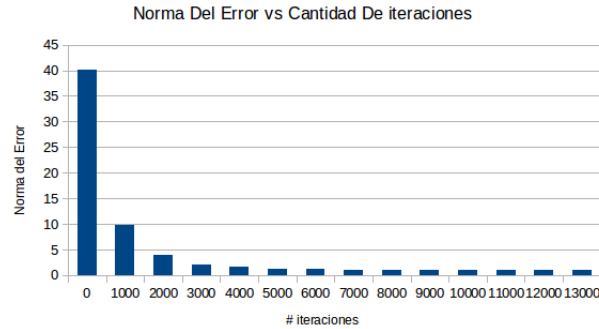
### 2.3. Experimentación sobre datos de Diagnostico de Cancer

#### 2.3.1. Convergencia del algoritmo

Para este problema utilizaremos una neurona en la última capa con función de activación logística. Para ser consistentes con esto, las salidas esperadas adoptarán los valores 1 si resulto ser cáncer maligno, y 0 si resulto ser benigno. La experimentación consistirá en buscar los parámetros óptimos que nos den la mayor tasa de aciertos para nuestro data set.

Como primera instancia comprobaremos que la red converge efectivamente a la solución esperada. Para eso tomamos un learning rate de 0,01, 7 neuronas en la capa oculta y con 1300 épocas

graficamos la norma del error cada 1000 iteraciones:



El grafico muestra que, efectivamente, nuestro algoritmo minimiza la diferencia entre las soluciones obtenidas y las deseadas, minimizando así tambien la norma del error.

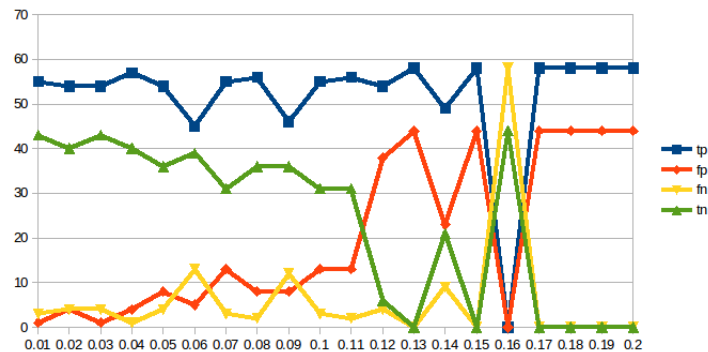
En las secciones subsiguientes, realizaremos diferentes experimentaciones sobre los hiperparámetros del algoritmo para intentar aproximar a la solución ideal.

### 2.3.2. Performance Vs Learning Rate

Para evitar el sobreajuste las experimentaciones todas las experimentaciones desde este punto se realizarán con la siguiente metodología. Se dividirá el set de datos de entrenamiento provisto por la catedra en dos sets de datos distintos. Uno se utilizará para entrenar la red, mientras que el otro se medirán que tan buenos fueron los resultados obtenidos. Con esto se espera reducir el overfittning que la red pueda generar y comprobar de manera mas acertada que tan buenos resultados dará la red sobre datos reales del problema.

Ademas, para visualizar los resultados mas claramente utilizaremos una matriz de confución, que nos permitirá discernir entre falsos positivos (fp), falsos negativos(fn) y instancias clasificadas correctamente (tp y tn).

En esta sección queremos experimentar es como se comporta el algoritmo al variar el learning rate. Para ello, dejamos constantes las cantidad de epocas de entrenamiento en 10000 y variamos el learning desde 0,01 hasta 0,2 aumentando de a 0,01



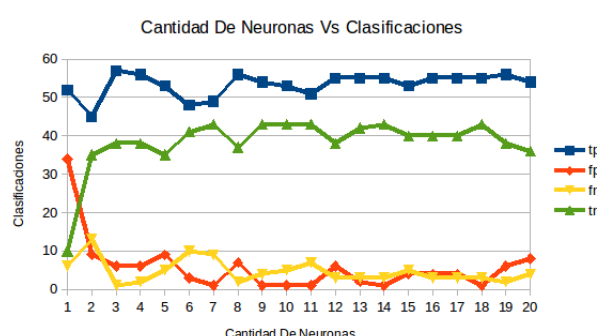
El grafico muestra varios resultados interesantes. En el rango  $[0,01,0,11]$  puede verse que la red arroja buenos resultados, obteniendo en general un 90 % de clasificaciones correctas. Pasado ese rango puede verse como los falsos positivos y los falsos negativos sufren un incremento muy significativo, llegando al final a casi un 50 % de clasificaciones incorrectas. Consideramos que para estos casos la red neuronal divergió, posiblemente debido a que multiplicar el gradiente por un valor muy grande lleva a "pasarnos" del minimo y por lo tanto reduciendo la precisión del algoritmo.

A partir de esta experimentación, para los siguientes experimentos decidimos tomar un learning rate igual a 0,02. Consideramos que uno mayor puede producir que el algoritmo diverga, mientras que uno mas pequeño hará que tarde demasiado en alcanzar la solución final.

### 2.3.3. Performance Vs Cantidad De Neuronas

Otro experimento interesante para realizar es ver como cambia la calidad de los resultados dependiendo de la cantidad de neuronas en la red. Para probar esto mantenemos la cantidad de iteraciones y el learning rate fijos en 1000 y 0,02 respectivamente y variamos la cantidad de neuronas desde 1 a 20.

Los resultados obtenidos se grafican en el siguiente grafico:



Como puede observarse, una cantidad de neuronas muy pequeña reduce considerablemente la calidad de los resultados. Esto puede deberse a

En este caso decidimos utilizar una cantidad de neuronas igual a 3.

### 2.3.4. Performance Vs Cantidad De iteraciones

Como siguiente paso buscaremos analizar el comportamiento de la red para distintas cantidades de iteraciones, viendo si existe algun cambio significativo en este sentido. Para ello dejamos fijo el learning rate fijo en 0,01 ya que consideramos que este nos da los resultados mas aceptables y variamos la cantidad de iteraciones.

ej1/test\_iter.png

Para una cantidad menor a 600 iteraciones puede observarse que la red neuronal tiene una performance mala obteniendo un porcentaje de falsos negativos y falsos positivos cercano al 50 %. Ya con 700 iteraciones en adelante la performance del algoritmo mejora drasticamente, rondando el porcentaje de falsos negativos y falsos positivos al rededor de 10 % y 20 %. Para 2000 y 2100 iteraciones obtenemos un 7 % de falsos negativos.

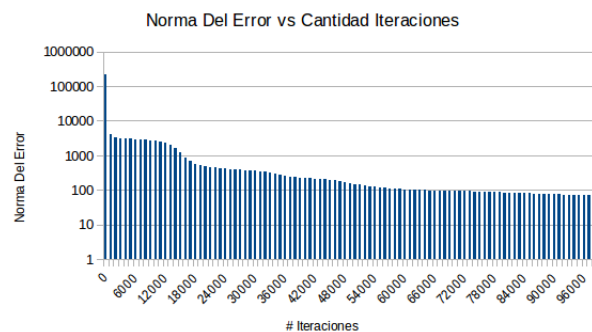
Decidimos que la mejor cantidad de iteraciones en este caso será de 900. El motivo de esta elección reside en lo antes expuesto, una cantidad menor produce que el algoritmo todavía no converga a la solución ideal, mientras que una cantidad mayor, si bien parece aumentar la precisión de la solución, tambien podría deberse al overfitting de los datos.

## 2.4. Eficiencia energética

Este problema presenta ciertas diferencias con respecto al anterior. En particular en este caso queremos hacer una regresión lineal sobre los datos. Por lo tanto, elegimos una función de activación lineal en la ultima capa. Ademas para cada conjunto de atributos se deberán volver dos resultados diferentes, por lo que la ultima capa tendrá dos neuronas de salida.

### 2.4.1. Convergencia del algoritmo

De igual manera que para el caso del cancer de mama, lo primero que queremos determinar es que el algoritmo comberge a la solución deseada. Por lo que nuevamente tomamos un lerning rate de 0,01, 30 neuronas en la capa oculta y con 100000 epocas graficamos la norma del error cada 1000 iteraciones:

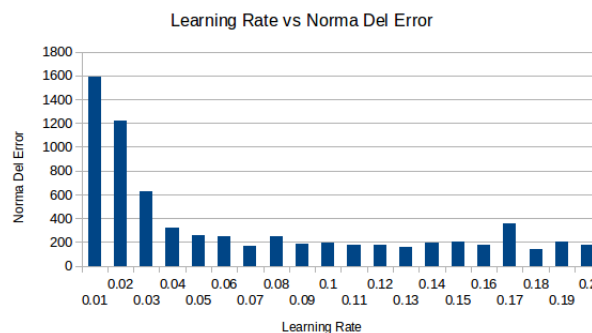


A simple vista se puede notar que en este caso el algoritmo converge de manera mucho mas lenta (notar que el grafico se encuentra en escala logaritmica).

Podemos observar que el algoritmo efectivamente converge a la solución deseada. Nuestro siguiente objetivo será ajustar los parametros de manera adecuada para intentar que la solución devuelta por el algoritmo sea lo mas aproximada a la solución real.

### 2.4.2. Performance Vs Learning Rate

En esta sección esperamos determinar un learning rate adecuado para nuestro algoritmo. A priori pareciera que 0,01 resulta muy bajo para este caso, resultando en una convergencia muy lenta. Para este experimento dejamos la cantidad de epocas fija en 10000, la cantidad de neuronas de la capa oculta en 30 y variamos el learning rate desde 0,01 a 0,2 con un step de 0,01. Los resultados obtenidos fueron:

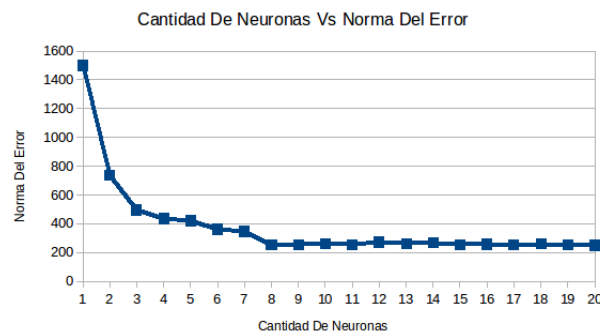


Como puede verse, en este caso un learning rate mayor aumenta considerablemente la convergencia del algoritmo.

Por este motivo, elegimos como nuestro learning rate optimo el de 0,05. Consideramos que un learning rate excesivo, si bien puede reducir la norma del error a niveles mayores, puede producir que el algoritmo no generalice las soluciones y simplemente memorice las instancias de entrenamiento.

### 2.4.3. Cantidad De neuronas Vs Performance

Al igual que en el caso del cancer, queremos ver como afecta la cantidad de neuronas a los resultados. Para ello nos basamos en el experimento anterior para obtener un learning rate que consideramos aceptable, dejamos la cantidad de iteraciones en 10000 y variamos la cantidad de neuronas desde 1 a 20.

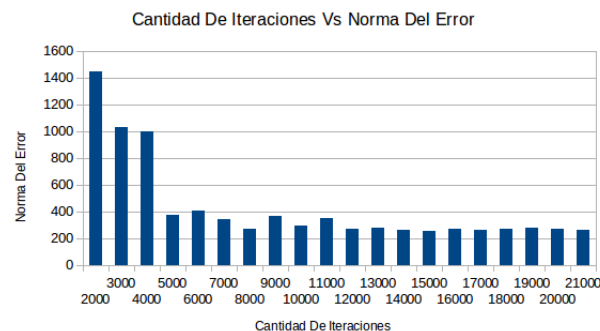


En el grafico puede observarse que para una cantidad muy pequeña de neuronas, los resultados resultan mucho peores. Aumentando la cantidad de neuronas a 3 ya la norma del error mejora sustancialmente, mientras que para una cantidad mayor de neuronas el algoritmo parece alcanzar una meceta.

Por este motivo consideramos que para este caso la cantidad optima de neuronas es 8.

### 2.4.4. Epocas Vs Performance

Como experimento final para esta sección buscaremos encontrar cual es la cantidad optima de iteraciones para este algoritmo. Para ello utilizaremos los parametros que consideramos mejores en la experimentación anterior (cantidad de neuronas = 8 y learning rate = 0.02) y variamos la cantidad de epocas desde 2000 hasta 21000:



Para 2000 a 4000 epocas, es correcto asumir que el algoritmo todavía no había convergido a una solución, encontrándose la norma del error en el orden de los 1000. Desde las 5000 iteraciones en adelante puede verse que el error cae repentinamente y se estabiliza en aproximadamente 200.

Por este motivo consideramos tomar para este caso una cantidad de epocas igual a 7000.

### 3. Conclusiones

El tema esta copado, pero no nos dio el tiempo. =(