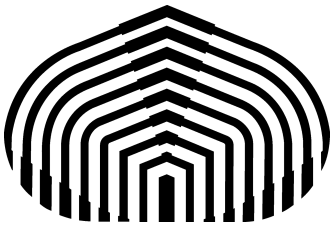


UNIVERSIDAD SIMÓN BOLÍVAR
DEPARTAMENTO DE COMPUTACIÓN Y TECNOLOGÍA DE LA INFORMACIÓN
CI-5438 INTELIGENCIA ARTIFICIAL II

Daniel Marín 10-10419
Dayana Rodrigues 10-10615
Luis Carlos Díaz 11-10293

Proyecto 2: Redes Neuronales

1. Resúmen.

La solución de las tareas propuestas para el proyecto fueron desarrolladas en dos partes, primero se implementó el algoritmo *backpropagation* para una red multicapa *feedforward*, con la finalidad de clasificar datos en dos tipos de problemas.

El primer problema pretende lograr la clasificación de puntos en un plano pertenecientes a dos regiones diferentes, la primera región corresponde al cuadrado cuya diagonal está dada por el segmento que une los puntos (0,0) y (20,20) menos el área ocupada por la segunda región, que corresponde a una circunferencia centrada en el punto (10,10) y radio 6, por lo que su ecuación canónica resulta $(x-10)^2 + (y-10)^2 = 36$.

Los conjuntos de entrenamiento para este problema son tres, de 500, 1000 y 2000 patrones compuestos por un punto (x,y) en el plano descrito y la región del mismo al cual pertenece el punto, adicionalmente se generaron tres nuevos conjuntos de entrenamiento con la misma cantidad de patrones que los conjuntos anteriores.

Como conjunto de prueba se tomaron los puntos de un barrido completo de la región cuadrada, evaluando las configuraciones en base a errores de entrenamiento, de prueba, falsos positivos y negativos.

El segundo problema consiste en la construcción de dos clasificadores para los datos *Iris Data Set*, uno de los clasificadores separa las *Iris Setosa* del resto, mientras que el segundo clasificador separa los datos en cada una de las tres clases.

Los datos de entrenamiento consisten de un conjunto con 4 datos con descripciones físicas de la planta: longitud del sépalo, ancho del sépalo, longitud del pétalo y ancho del

pétalo, así como la clase a la que corresponde el patrón: Iris Setosa, Iris Versicolor e Iris Virginica.

Los datos de prueba para este segundo problema se tomaron de diversos porcentajes de los datos del conjunto de entrenamiento descrito anteriormente.

2. Detalles de implementación.

Todo el código que forma parte de la solución fue escrito en Python, haciendo uso de las librerías *numpy* para la manipulación de la data y *matplotlib* para la generación de las gráficas que ilustran las soluciones. El código de la solución está estructurado en 5 archivos donde:

- *mlp.py*: contiene la implementción de la red neuronal junto con las funciones de entrenamiento y prueba.

- *new_data.py*: contiene las funciones que generan nuevos datos de entrenamiento y prueba.

- *ejercicio2.py*: contiene la función que muestra al usuario las opciones de la pregunta 2 y ejecuta la solución.

- ejercicio3.py: contiene la funcion que muestra al usuario las opciones de la pregunta 3 y ejecuta la solución.

- *main.py:* archivo que contiene la función principal, que muestra al usuario un menú de opciones que le permite seleccionar la solución que desee.

3. Resultados.

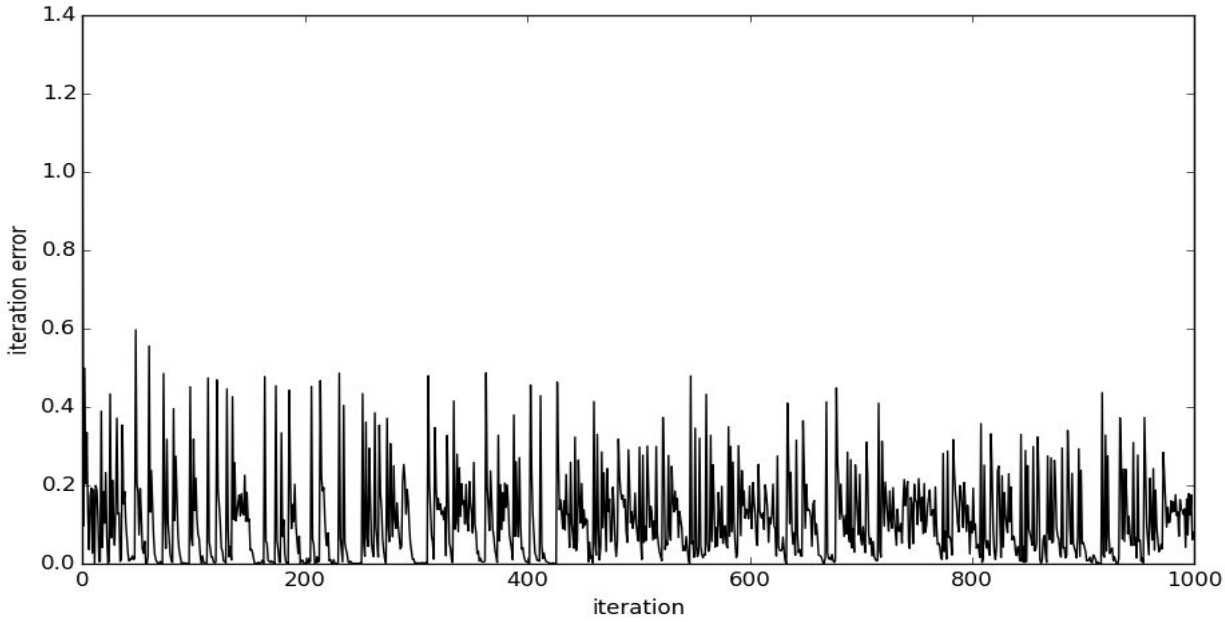
Para los conjuntos de datos del ejercicio 2 se realizaron entrenamientos y pruebas de la red neuronal para cada conjunto de entrenamiento y valores de 2 a 10 neuronas en la capa oculta, obteniendo los siguientes errores de entrenamiento y prueba para cada instancia:

datos x neuronas	500 dados	1000 dados	2000 dados	500 generados	1000 generados	2000 generados
2	train: 0.06835514 test: 0.182	train: 0.07841099 test: 0.239	train: 0.07806713 test: 0.21	train: 0.07909489 test: 0.246	train: 0.07636037 test: 0.287	train: 0.07996561 test: 0.25
3	train: 0.06845053 test: 0.196	train: 0.07941727 test: 0.25	train: 0.06335671 test: 0.09	train: 0.05963233 test 0.104	train: 0.077927 test: 0.087	train: 0.05442968 test: 0.076
4	train: 0.07637611 test: 0.232	train: 0.07766545 test: 0.26	train: 0.0632124 test: 0.147	train: 0.07903865 test: 0.228	train: 0.07968683 test: 0.258	train: 0.07874217 test: 0.241
5	train: 0.03954998 test: 0.05	train: 0.07850608 test: 0.233	train: 0.07107345 test: 0.0845	train: 0.04751229 test: 0.098	train: 0.07719909 test: 0.203	train: 0.0799903 test: 0.243
6	train: 0.06964549 test: 0.198	train: 0.07885537 test: 0.224	train: 0.0440621 test: 0.066	train: 0.07924236 test: 0.246	train: 0.07778211 test: 0.227	train: 0.07942024 test: 0.252
7	train: 0.06930367 test: 0.19	train: 0.07812041 test: 0.237	train: 0.07758452 test: 0.216	train: 0.07946772 test: 0.244	train: 0.07630214 test: 0.048	train: 0.08018964 test: 0.249
8	train: 0.07684767 test: 0.232	train: 0.04150136 test: 0.073	train: 0.04509002 test: 0.0725	train: 0.04451604 test: 0.056	train: 0.04405487 test: 0.051	train: 0.07320401 test: 0.072
9	train: 0.06835268 test: 0.184	train: 0.05504065 test: 0.071	train: 0.07255584 test: 0.056	train: 0.0487155 test: 0.084	train: 0.04523482 test: 0.076	train: 0.0806617 test: 0.2345
10	train: 0.04858649 test: 0.066	train: 0.04221487 test: 0.065	train: 0.06989472 test: 0.114	train: 0.07914574 test: 0.232	train: 0.07809894 test: 0.203	train: 0.07925124 test: 0.2585

Como se observa en la tabla anterior, los mejores resultado se obtienen para las instancias con 500 y 2000 patrones, obteniendo errores aceptablemente pequeños para

diferentes números de neuronas en la capa oculta, sin embargo, la instancia con mejor resultados corresponde a los datos de 2000 patrones dados y una red de 6 neuronas en la capa oculta, por lo que esta configuración fue usada para realizar la clasificación de los puntos correspondientes a un barrido completo de la región estudiada, obteniendo como resultados
error de prueba = 0.0736202333105.

Como el mejor conjunto de prueba es el conjunto de datos del archivo *datos_P2_EM2017_N2000.txt*, se graficó el error de iteración para esta instancia:



En la gŕafica se observa como el error cometido en cada iteración se mantiene en valores relativamentes bajo, a pesar de los picos de error obtenidos durante las iteraciones el error tiene siempre a disminuir, ya que incluso estos picos son cada vez menos altos conforme avanzan las iteraciones.

Para el conjunto de datos del ejercicio 3 (Iris Dataset) se realizaron dos clasificadores. Un clasificador binario que separa las Iris-setosa del resto y un clasificador que separa cada una por su tipo. Para la codificación del clasificador binario se usó una variable dummy (0/1). Para la codificación del clasificador de tipo se usó una codificación 1-C (una variable dummy por categoría). Los resultados se muestran a continuación:

Tabla de Resultados para Iris Data Set (Clasificador Binario)

n/ %	50%	60%	70%	80%	90%
4	Training Time: 20.037 Error: 0.000 Test: 75/75	Training Time: 23.768 Error: 0.000 Test: 60/60	Training Time: 26.401 Error: 0.000 Test: 45/45	Training Time: 29.881 Error: 0.000 Test: 30/30	Training Time: 33.749 Error: 0.000 Test: 15/15
5	Training Time: 25.629 Error: 0.000 Test: 75/75	Training Time: 32.754 Error: 0.000 Test: 60/60	Training Time: 38.408 Error: 0.000 Test: 45/45	Training Time: 38.390 Error: 0.000 Test: 30/30	Training Time: 40.352 Error: 0.000 Test: 15/15
6	Training Time: 26.444 Error: 0.000 Test: 75/75	Training Time: 31.811 Error: 0.000 Test: 60/60	Training Time: 37.100 Error: 0.000 Test: 45/45	Training Time: 42.341 Error: 0.000 Test: 30/30	Training Time: 47.451 Error: 0.000 Test: 15/15
7	Training Time: 30.495 Error: 0.000 Test: 75/75	Training Time: 36.574 Error: 0.000 Test: 60/60	Training Time: 42.743 Error: 0.000 Test: 45/45	Training Time: 48.721 Error: 0.000 Test: 30/30	Training Time: 54.702 Error: 0.000 Test: 15/15
8	Training Time: 34.482 Error: 0.000 Test: 75/75	Training Time: 41.230 Error: 0.000 Test: 60/60	Training Time: 48.058 Error: 0.000 Test: 45/45	Training Time: 55.270 Error: 0.000 Test: 30/30	Training Time: 61.904 Error: 0.000 Test: 15/15
9	Training Time: 38.307 Error: 0.000 Test: 75/75	Training Time: 46.106 Error: 0.000 Test: 60/60	Training Time: 53.809 Error: 0.000 Test: 45/45	Training Time: 61.346 Error: 0.000 Test: 30/30	Training Time: 68.927 Error: 0.000 Test: 15/15
10	Training Time: 42.530 Error: 0.000 Test: 75/75	Training Time: 51.120 Error: 0.000 Test: 60/60	Training Time: 59.809 Error: 0.000 Test: 45/45	Training Time: 68.177 Error: 0.000 Test: 30/30	Training Time: 76.500 Error: 0.000 Test: 15/15

Tabla de Resultados para Iris Data Set (Clasificador de tipo)

n/ %	50%	60%	70%	80%	90%

4	Training Time: 24.914 Error: 0.000 Test: 72/75	Training Time: 29.879 Error: 0.000 Test: 59/60	Training Time: 34.812 Error: 0.000 Test: 44/45	Training Time: 39.875 Error: 0.000 Test: 30/30	Training Time: 44.676 Error: 0.000 Test: 14/15
5	Training Time: 30.315 Error: 0.000 Test: 72/75	Training Time: 36.312 Error: 0.000 Test: 59/60	Training Time: 42.597 Error: 0.000 Test: 44/45	Training Time: 48.579 Error: 0.000 Test: 29/30	Training Time: 54.874 Error: 0.000 Test: 14/15
6	Training Time: 35.951 Error: 0.000 Test: 72/75	Training Time: 43.323 Error: 0.000 Test: 60/60	Training Time: 50.273 Error: 0.000 Test: 44/45	Training Time: 57.493 Error: 0.000 Test: 29/30	Training Time: 64.676 Error: 0.000 Test: 14/15
7	Training Time: 41.433 Error: 0.000 Test: 72/75	Training Time: 49.650 Error: 0.000 Test: 59/60	Training Time: 58.128 Error: 0.000 Test: 44/45	Training Time: 66.316 Error: 0.000 Test: 29/30	Training Time: 74.534 Error: 0.000 Test: 14/15
8	Training Time: 46.996 Error: 0.000 Test: 72/75	Training Time: 56.286 Error: 0.000 Test: 60/60	Training Time: 65.761 Error: 0.000 Test: 44/45	Training Time: 75.165 Error: 0.000 Test: 29/30	Training Time: 84.400 Error: 0.000 Test: 14/15
9	Training Time: 52.601 Error: 0.000 Test: 72/75	Training Time: 63.068 Error: 0.000 Test: 60/60	Training Time: 73.527 Error: 0.000 Test: 44/45	Training Time: 84.000 Error: 0.000 Test: 29/30	Training Time: 94.325 Error: 0.000 Test: 14/15
10	Training Time: 57.841 Error: 0.000 Test: 72/75	Training Time: 69.191 Error: 0.000 Test: 60/60	Training Time: 81.139 Error: 0.000 Test: 44/45	Training Time: 92.751 Error: 0.000 Test: 30/30	Training Time: 104.056 Error: 0.000 Test: 14/15

Para todos los casos probados con el clasificador binario la red predijo los resultados correctamente el 100% de las veces, con un error de entrenamiento menor a 10^{-5}.

Para los casos probados con el clasificador de tipo las predicciones no fueron exactas (peores resultados 14/15 $\sim$ 93.33% de efectividad), aunque el error de entrenamiento se mantuvo menor a 10^{-5}. Esta imprecisión en la efectividad puede deberse a datos ambiguos o poco comunes con los demás en alguna de las muestras (entrenamiento o prueba).

4. Conclusiones.

En los resultados de los experimentos realizados podemos contemplar que:

- No hay manera sencilla de saber el número óptimo de neuronas en la capa oculta para maximizar el desempeño de la red. Una gran cantidad de neuronas no asegura mejores resultados. Como el tiempo de entrenamiento incrementa con la cantidad de neuronas lo ideal sería escoger el mínimo número de neuronas que provean el mejor resultado. El inconveniente es que el método para hallar este mínimo es probando.

- En general mientras más grande es la muestra de entrenamiento, más precisa es la red. Podemos ver que en algunos casos, más datos de entrenamiento conllevan a un incremento en el error.

Al momento de entrenar una red neural hay que estudiar factores como: cuantas neuronas habrá en la capa oculta (si habrá solo una o más capas ocultas), la cantidad (tamaño de la muestra) y selección de los datos de entrenamiento y la codificación de sus rasgos. Estos determinarán el desempeño de la red.