Please submit **all the source files** together with **a short report** of your work and an XML with your best trained network.

All the files should be sent **in a single compressed archive**.

In your report you should include a brief description of the data set, with appropriate references to the literature. In addition, please describe any special features of the set (e.g. missing values).

You must submit a script file named dataset.py with all the steps, including a graphical display of the error versus the number of epochs, the computation of the confusion matrix, and saving the net weights to an XML file.

You should try with different number of neurons in the hidden layer and different values for the parameters of the algorithm (learning rate, momentum). Then, you should compare the performance of each network with respect to the error, and the number of epochs until convergence.

El conjunto de datos que se ha utilizado a la hora de hacer las pruebas para esta sesión fue la de “Haberman's Survival Data Set“.

Este conjunto de datos contiene casos de estudio llevado a cabo sobre la supervivencia de los pacientes que se habían sometido a cirugía para el cáncer de mama. Este estudio se realizó entre 1958 y 1970 en el Hospital de la Universidad de Chicago Billings.

Información sobre la muestra:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Características del conjunto:** | Multivariable | **Cantidad de instancias:** | 306 | **Área:** | Vida |
| **Características del atributo:** | Entero | **Número de atributos:** | 3 | **Fecha Donación** | 1999-03-04 |
| **Tareas asociadas:** | Clasificación | **Faltan valores?** | No | **Número de visitas en la Web[**[**1**](http://archive.ics.uci.edu/ml/datasets/Haberman%27s+Survival)**]:** | 44989 |

Información sobre los Atributos:

1. Edad del paciente en el momento de la operación (numérico)

2. Año en el que el paciente se hizo la operación (año - 1900, numérico)

3. Número de ganglios axilares positivos detectados (numérico)

4. Estado de supervivencia (clase atributo)

- 1 = el paciente sobrevivió 5 años o más

- 2 = el paciente murió dentro de 5 años

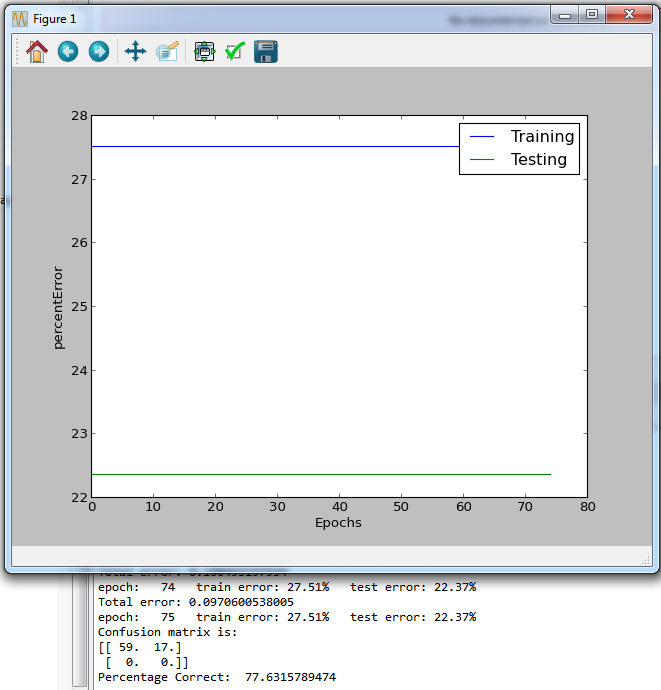
A la hora de realizar el script, el estado de supervivencia de los pacientes se modificó de la siguiente manera:

- 0 = el paciente sobrevivió 5 años o más

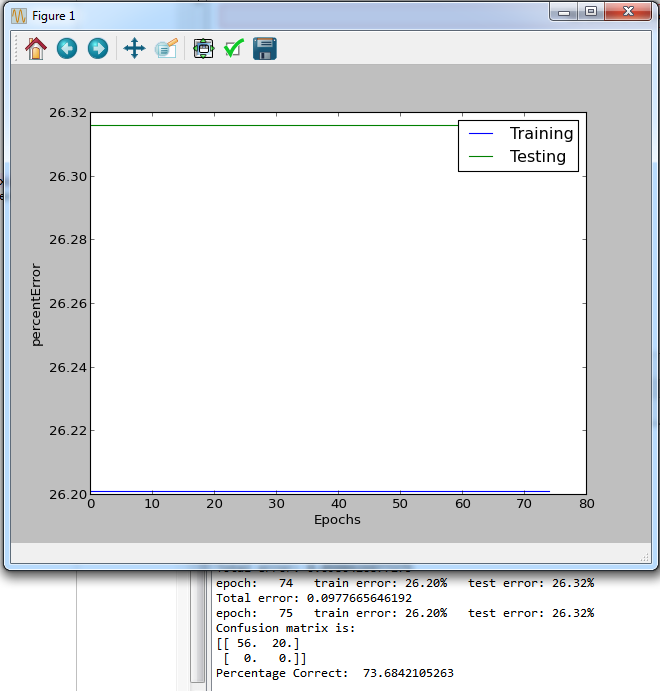
- 1 = el paciente murió dentro de 5 años

La primera versión que se hizo del script fue sin normalizar los valores que se recibían de entrada y los resultados fueron unos muy extraños, porque parecía que las neuronas no entrenaban en ningún momento.

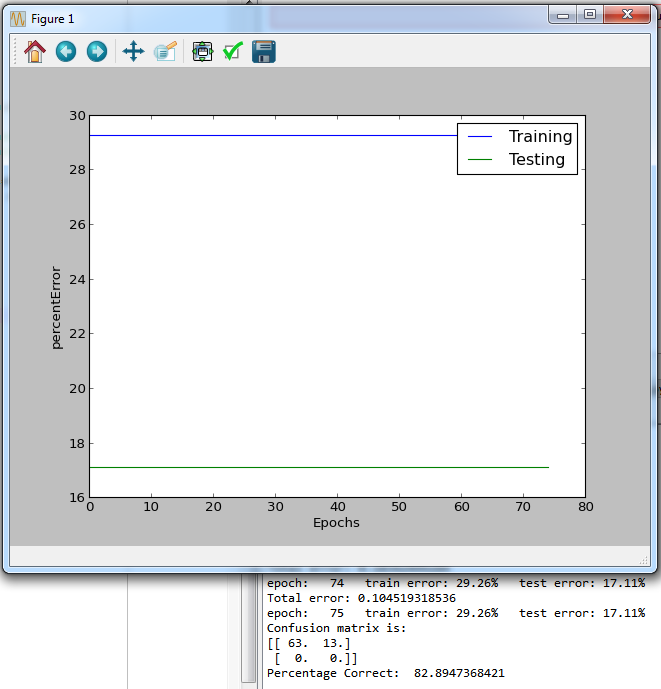
Se intentó cambiar el momentum, el número de hiden o el número de epochs pero resultado fue siempre el mismo. El script clasificada todos los valores de entrada en las de tipo 0 (que el paciente sobrevivió 5 años o más), aunque a veces tenía unas ligeras variaciones como se puede observar en el quinto gráfico.



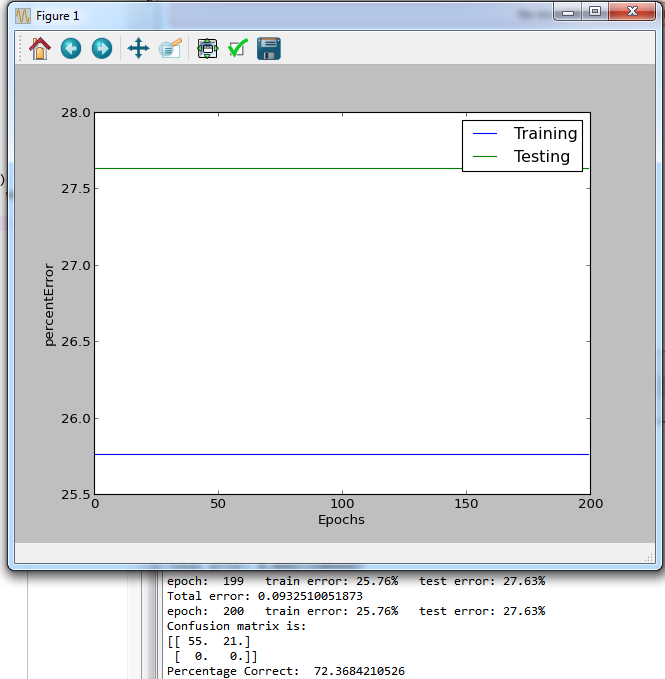
75 epoch, 10 hiden, momentum 0,1 y división de muestra 0.25



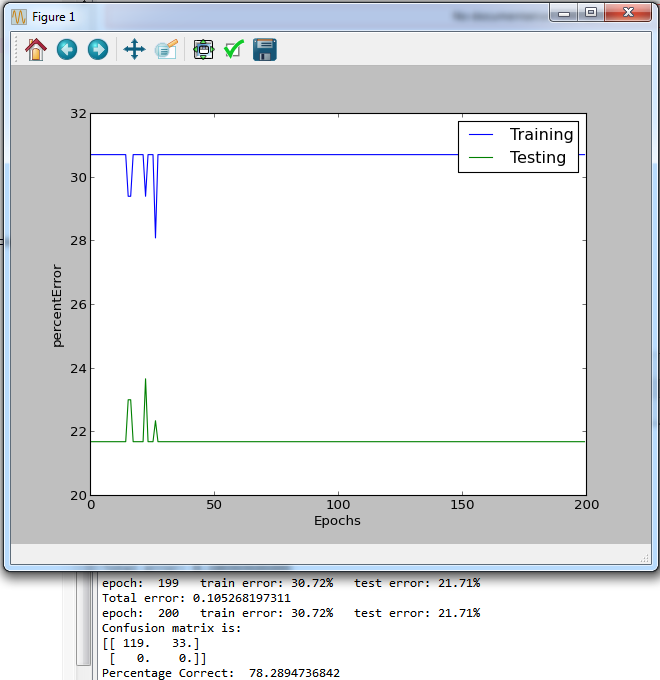
75 epoch, 5 hiden, momentum 0,1 y división de muestra 0.25



75 epoch, 5 hiden, momentum 0,5 y división de muestra 0.25



200 epoch, 5 hiden, momentum 0,5 y división de muestra 0.25



200 epoch, 5 hiden, momentum 0,5 y división de muestra 0.50

El motivo de ese resultado es porque el error que se tenía de la muestra era muy grande y por lo tanto realizamos una segunda versión del programa, donde utilizamos la normalización para mejorar los resultados.

Este es el proceso de normalización utilizado en la práctica:

f = file('C:\Users\Myhay\Desktop\PracticaSI\Practica2\_Network\haberman.data', 'r') # leemos el fichero

r = csv.reader(f)

list\_data = list(r)

f.close()

# sacar la media

total\_A = 0

total\_B = 0

total\_C = 0

for i in range(len(list\_data) - 1):

row = list\_data[i]

total\_A += float(row[0])

total\_B += float(row[1])

total\_C += float(row[2])

mediaA = total\_A/len(list\_data)

mediaB = total\_B/len(list\_data)

mediaC = total\_C/len(list\_data)

print "Media\_A", mediaA, "Media\_B", mediaB, "Media\_C", mediaC

# sacar la desviación

total\_desviacion\_A = 0

total\_desviacion\_B = 0

total\_desviacion\_C = 0

for i in range(len(list\_data) - 1):

row = list\_data[i]

total\_desviacion\_A += (float(row[0]) - mediaA)\*\*2

total\_desviacion\_B += (float(row[1]) - mediaB)\*\*2

total\_desviacion\_C += (float(row[2]) - mediaC)\*\*2

print "Media\_desviacion\_A", total\_desviacion\_A, "Media\_desviacion\_B", total\_desviacion\_B, "Media\_desviacion\_C", total\_desviacion\_C

standard\_derivation\_A = math.sqrt(total\_desviacion\_A/len(list\_data))

standard\_derivation\_B = math.sqrt(total\_desviacion\_B/len(list\_data))

standard\_derivation\_C = math.sqrt(total\_desviacion\_C/len(list\_data))

print standard\_derivation\_A, standard\_derivation\_B, standard\_derivation\_C

for i in range(len(list\_data) - 1):

row = list\_data[i]

a = (float(row[0]) - mediaA)/standard\_derivation\_A

b = (float(row[1]) - mediaB)/standard\_derivation\_B

c = (float(row[2]) - mediaC)/standard\_derivation\_C

e = row[3]

if(e == '1'):

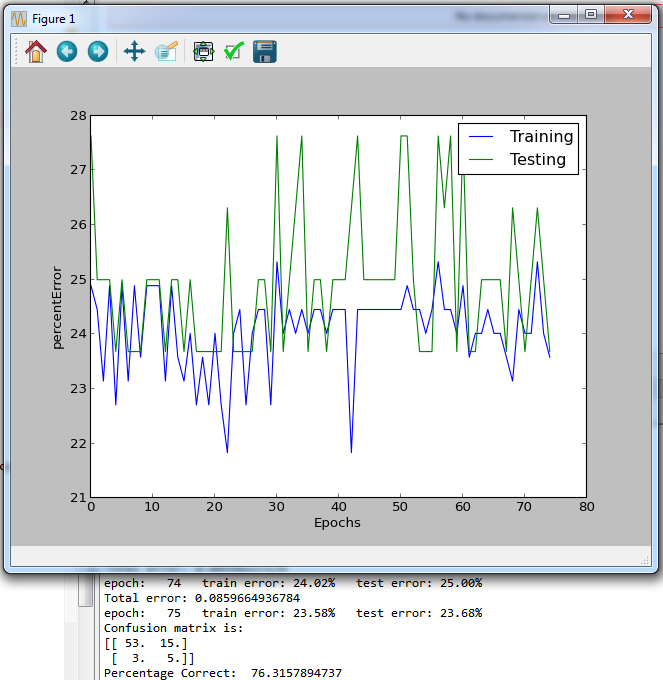
e = 0

elif(e == '2' ):

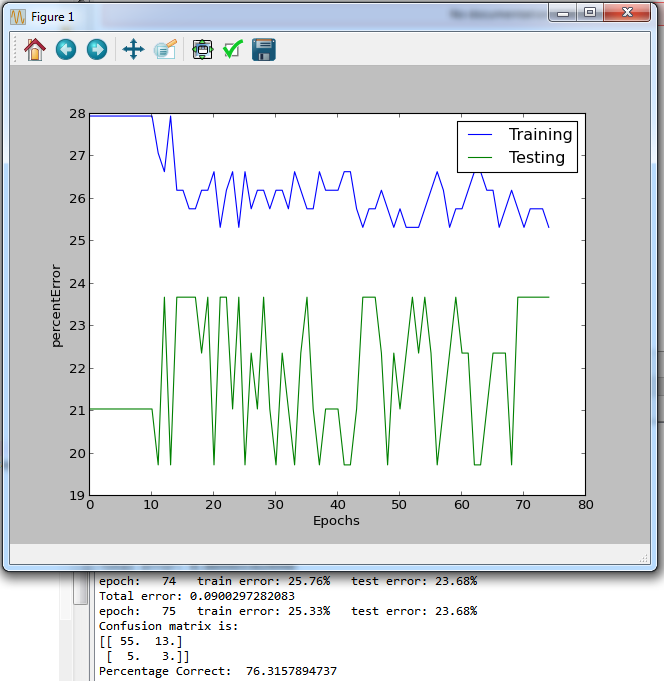
e = 1

dataBase.addSample((a, b, c), (e,)) #añadimos la muestra

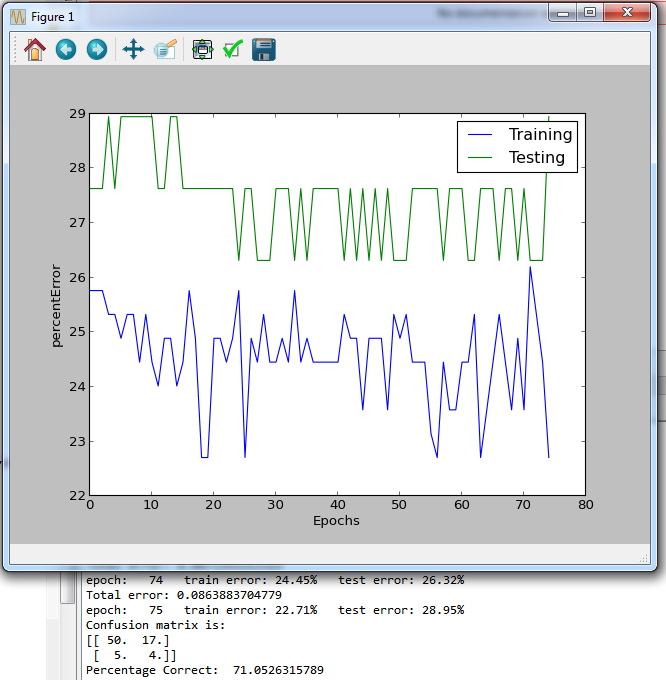
Después de normalizar la muestra de datos, los resultados obtenidos fueron los siguientes:



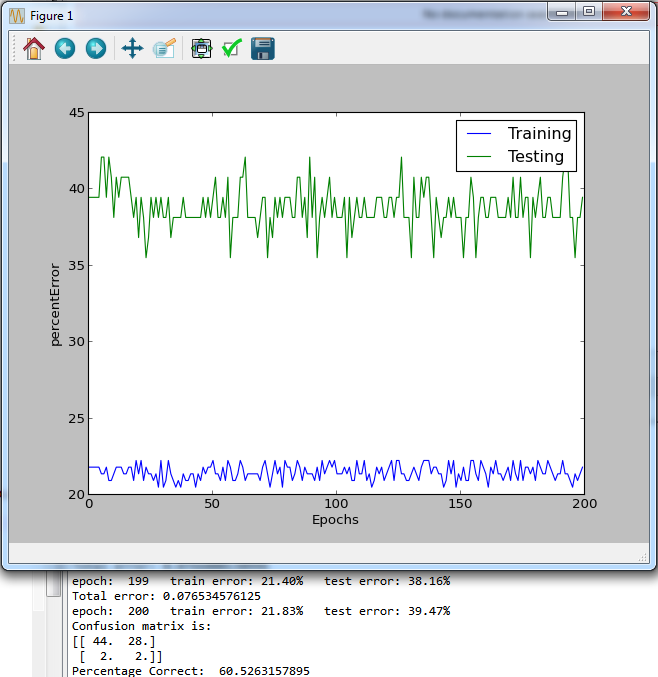
75 epoch, 10 hiden, momentum 0,1 y división de muestra 0.25



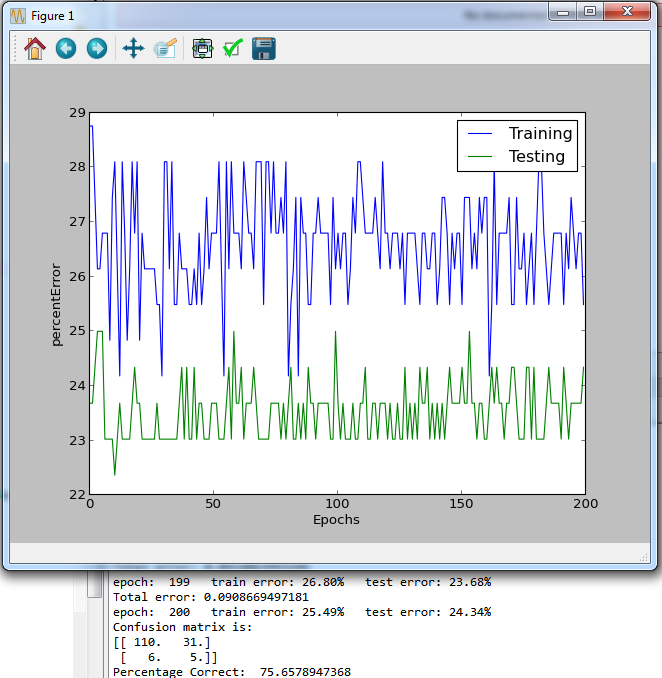
75 epoch, 5 hiden, momentum 0,1 y división de muestra 0.25



75 epoch, 5 hiden, momentum 0,5 y division de muestra 0.25



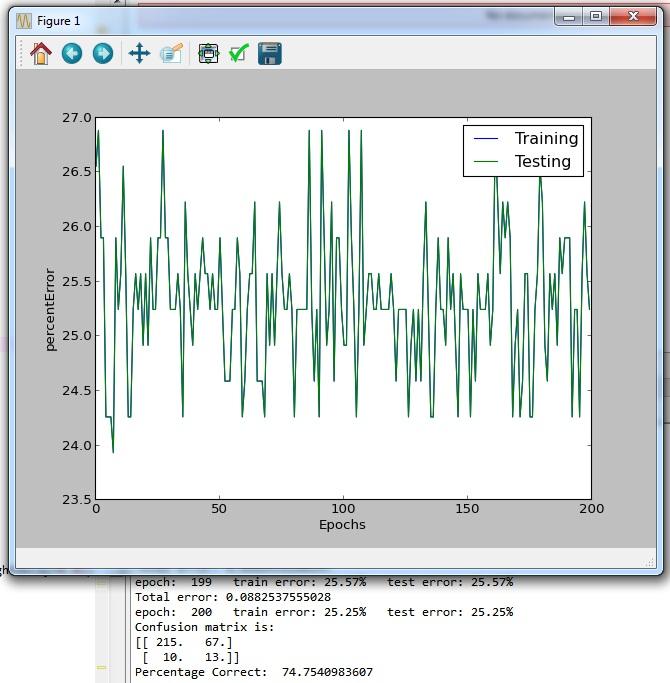
200 epoch, 5 hiden, momentum 0,5 y división de muestra 0.25



200 epoch, 5 hiden, momentum 0,5 y división de muestra 0.50

Esta vez, las neuronas se estaban entrenando y no clasificaba la muestra sólo del primer tipo, pero aunque esta clasificando bien una gran cantidad de elementos de la muestra, sigue teniendo un alto porcentaje de error.

Como prueba definitiva hemos utilizado toda la base de datos para realizar el entrenamiento y prueba de las neuronas.



El resultado fue el esperado, dado que al entrenar las neuronas y realizar los tests con los mismos datos, el gráfico tiene que superponerse y la matriz de confusión no tiene que variar mucho con respecto a los valores obtenidos de las pruebas anteriores.

Conclusión:

Disminuir el error de este problema no es fácil y hemos decidido que las pruebas realizadas sobre la muestra de datos es más que suficiente y que no vale la pena seguir intentando mejorar el resultado por falta de conocimientos.

El valor más alto obtenido fue en la versión del programa número uno y prueba numero tres, donde conseguimos un 82% de acierto pero no se puede tener en consideración.

David Dag Mora Zapata

Mihaita Alexandru Lupoiu