REDES DE NEURONAS ARTIFICIALES

Práctica 2 Parte 1

Repositorio de GitHub:
https://github.com/Cocytus72/Redes-de-Neuronas-P2

15/12/2023

Lucas Gallego Bravo - 100429005
E-mail: 100429005@alumnos.uc3m.es

Prof. Ángel Carrasco Fernández

Índice:

1. Introducción:

Para esta primera parte de la segunda práctica de la asignatura, se nos pide utilizar el perceptrón multicapa en una tarea de clasificación.

Estamos haciendo uso de un dataset clasificación de trayectorias de diferentes individuos en diferentes medios de transporte, y el objetivo es clasificar posteriormente en base a los datos, en que vehículo dicho individuo va, hay 5 posibilidades de medio de transporte: andando, bici, bus, conduciendo y metro.

Primero se ha tenido que hacer un preproceso de los datos, del cual se hablará más adelante, tras este preproceso, se ha hecho uso de One-Hot Encoding en la variable de salida, de esta forma facilitamos al perceptrón la clasificación.

2. Preproceso de los datos:

En el preproceso de los datos de esta parte de la práctica, lo primero que se ha hecho es analizar el dataset. Lo que se ha visto tras su análisis es que había algunas columnas en el mismo que no servían a la hora de hacer la clasificación, por lo que se ha decidido eliminarlas. Estas columnas son:

- std_delta_times, std_hours, std_distances, std_velocities, std_accelerations, std_headings, std_heading_changes, std_heading_change_rates, std_stops, std_turnings, track_id, user_id

Todas las columnas que empiezan por std han sido eliminadas ya que se han considerado repetitivas al tener el valor de la columna original, por otro lado las columnas de id se han eliminado ya que para una clasificación de este tipo, datos como el id del usuario no son relevantes.

Una vez se han borrado estas columnas de nuestro dataset, hemos normalizado los datos de todas las columnas restantes excepto de la columna de target. A la hora de hacer la normalización se ha hecho lo mismo que en la entrega anterior. Primero fuera del bucle hemos obtenido los máximos y mínimos de todas las columnas de nuestro dataset. Una vez tenemos dichos máximos y mínimos, tras una serie de operaciones matemáticas obtenemos el valor normalizado para ese dato de nuestro dataset, por lo que posteriormente los sustituimos.

Tras normalizar los datos, se ha dividido el dataset en test y train, haciendo que el conjunto de train sea 2/3 del dataset tras su normalización. Por lo que el test se queda con el 1/3 restante.

Posteriormente, se realiza One-Hot Encoding a nuestra columna target tanto en el conjunto de test como en el de entrenamiento. Haciendo de esta forma que se pase de una clasificación numérica, a una clasificación binaria.

3. Perceptrón multicapa:

En esta parte del documento, vamos a analizar el perceptrón multicapa creado para esta parte de la práctica.

Como ya se ha indicado anteriormente, la tarea que se nos pide es una de clasificación, pero hay algunos valores del perceptrón que se nos ha pedido que no toquemos, siendo estos que el número de neuronas de la capa de salida sea igual al número de clases del problema, que la activación de la capa de salida sea del tipo softmax, que el optimizador del perceptrón sea el SGD. Sin embargo, hay otras cosas con las que se ha preferido no experimentar en esta práctica, como bien puede ser batch_size o momento, variables las cuales se ha preferido no modificar su valor en ningún momento.

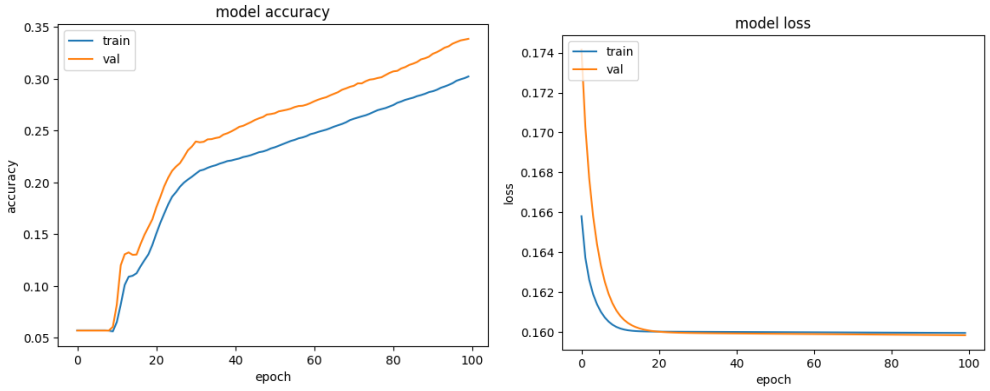
El perceptrón multicapa proporcionado calcula al principio los pesos de manera automática dependiendo del número de apariciones de cada clase en el dataset proporcionado. El criterio de parada utilizado en todo momento ha sido Checkpoint, ya que se ha considerado mejor que Earlystopping ya que las cantidades de datos y número de épocas no eran muy elevados.

3.1 Experimentación realizada y evolución de los errores:

Para poder seleccionar el mejor modelo, primero debemos experimentar, con los hiperparametros de nuestro perceptrón multicapa con el fin de encontrar el modelo que mejor se ajuste a nuestro problema.

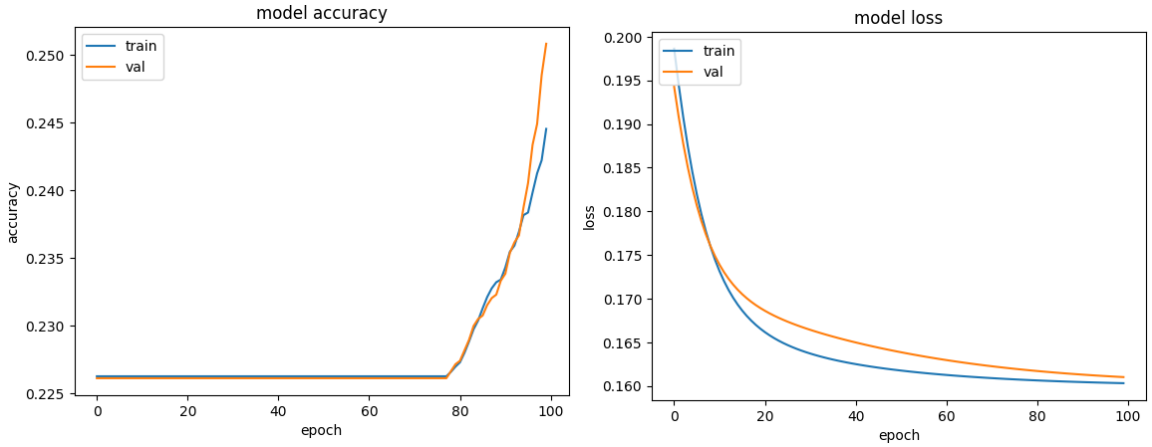
Para evaluar al mejor modelo, vamos a seleccionar aquellos que obtengan un valor de loss en validación menor. En este apartado se va a mostrar la evolución de accuracy y loss de los conjuntos de train y validación de los modelos que se han considerado más interesantes para esta parte.

Debido a esto, el primer modelo que se quiere enseñar es el primero que se ha realizado, ya que es el que ha comenzado todo este proceso:



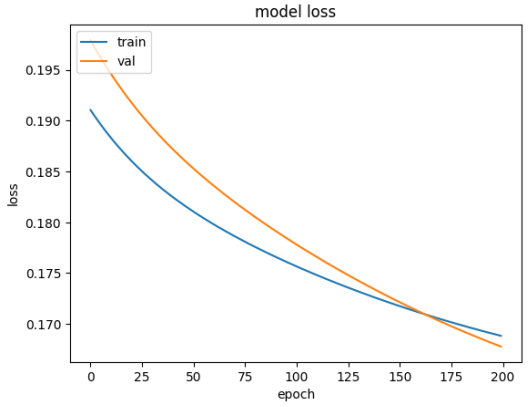
Como se puede observar, estos primeros resultados ya resultan bastante buenos. El modelo sigue aprendiendo y mejorando con cada época, lo cual es una buena señal

El siguiente modelo que se quiere mostrar es el segundo que se ha creado, ya que la evolución de accuracy ha resultado bastante interesante:



Tal y como se puede observar, de momento la evolución de loss en ambos modelos se mantiene similar.

Finalmente, lo último que se enseñará es el loss del que posteriormente se considera el mejor modelo, el modelo 8:



3.2 Todos los experimentos realizados:

A continuación se muestran en una tabla los datos de todos los experimentos realizados con el Perceptrón multicapa en esta primera parte de la práctica. Cabe destacar que a la hora de aumentar el tamaño de las capas ocultas (siempre entre 1 y 3). Para todos los MSE de train y validación, se usará el mejor dato de ese modelo (criterio de parada utilizado).

Primero se va a mostrar una tabla la cual describe todos los modelos con los que se ha experimentado (hiperparametros) los cuales van a ser enumerados por orden de ejecución en el notebook. Posteriormente se hará una segunda tabla la cual muestre los resultados obtenidos de cada modelo indicando siempre a qué modelo pertenece. El hiperparametro batch_size se ha mantenido en 32 para todas las pruebas.

Nombre del Modelo	Tasa de aprendizaje	Nº Iteraciones	Nº Capas ocultas	Nº neuronas de las capas ocultas
Modelo 1	0,002	100	1	50
Modelo 2	0,0002	100	1	50
Modelo 3	0,0002	100	2	50, 50
Modelo 4	0,0003	100	2	50, 50
Modelo 5	0,0003	100	2	25, 25
Modelo 6	0,0002	200	2	25, 25
Modelo 7	0.00002	200	2	50, 50
Modelo 8	0.00003	200	2	50, 50
Modelo 9	0.00003	200	2	75, 75
Modelo 10	0.00005	200	2	75, 75

Una vez se han definido todos los modelos con los que se ha experimentado, podemos mostrar los resultados obtenidos con ellos, como se ha mencionado anteriormente, solo se mostraran los valores de los resultados obtenidos para los mejores modelos con cada modelo (mejor peso). Cabe destacar que al el Loss de cada subconjunto es el MSE. La columna Epoch indica en qué ciclo se consiguió el mejor modelo, se empieza a contar desde el 1.

Nombre Modelo	Epoch	Train Loss	Val Loss	Train Acc	Val Acc
Modelo 1	100	0.1600	0.1598	0.3022	0.3386
Modelo 2	100	0.1603	0.1610	0.2445	0.2508
Modelo 3	100	0.1611	0.1630	0.1860	0.1860
Modelo 4	7	0.1734	0.1561	0.4798	0.3800
Modelo 5	100	0.1600	0.1604	0.2263	0.2261
Modelo 6	1	0.1675	0.1558	0.3798	0.3800
Modelo 7	200	0.1662	0.1666	0.1861	0.1666
Modelo 8	200	0.1629	0.1625	0.3798	0.3800
Modelo 9	200	0.1622	0.1616	0.3604	0.3579
Modelo 10	117	0.1614	0.1595	0.1505	0.1505

Una vez tenemos los resultados de todos los modelos con los que hemos experimentado, podemos seleccionar cuales de ellos son los mejores y probar con ellos el balanceo. Para este balanceo vamos a cambiar a mano los pesos, para dar más importancia a aquellas clases que se están viendo afectadas en mayor medida por la clasificación. Dicho esto, el modelo elegido como mejor es:

- Modelo 8

Ya que a pesar de no ser el que da un valor de val_loss más bajo, es el modelo que sigue mejorando hasta la última época y que tiene uno de los valores de accuracy más altos.

3.2 Balanceo en los mejores modelos:

A la hora de balancear los pesos de nuestro perceptrón se ha optado por normalizar los pesos dividiendo por el peso mínimo. De esta forma los pesos están más equilibrados y obtenemos los siguientes resultados:

Epoch	Train Loss	Test Loss	Train Acc	Test Acc	Test Recall	Test Precisión	Test F1 Score
100	0.1647	0.1550	0.2263	0.2224	0.2224	0.0495	0.0809

Matriz de confusión:

```
[[   0    0 3517    0    0]
 [   0    0 1925    0    0]
 [   0    0 2161    0    0]
 [   0    0 1478    0    0]
 [   0    0  636    0    0]]
```

Al igual que antes, la columna epoch indica en qué época el modelo dejó de mejorar el loss de train.

A la hora de hacer la predicción sobre el conjunto de test, se han usado los mejores pesos del modelo para así mejorar el resultado de estas predicciones.

4. Conclusión:

A modo de conclusión me gustaría decir que al haber realizado la práctica yo solo, me ha resultado bastante complicado el hecho de dividirse el trabajo y hacer otras asignaturas de la universidad.

Me hubiera gustado realizar algún que otro test más ya que creo que el mejor modelo con el balanceo de los pesos hecho hubiera dado un mejor resultado con otro número de épocas u razón de aprendizaje, ya que al usar train sin la división de validación, el tamaño de los datos aumenta. Sin embargo considero que la predicción de test ha dado buen resultado.

Cabe destacar que como no sabía el tipo de fichero en el que se querían que se guardaran las predicciones, he hecho tanto un csv como un txt, por eso he entregado ambos.