INTRODUCCIÓN A LOS MODELOS COMPUTACIONALES

Práctica 3: Redes neuronales de funciones de base radial

4º Curso-Grado en Ingeniería Informática

UCO-Escuela Politécnica Superior de Córdoba

Manuel Casas Castro - 31875931R

i72cascm@uco.es

ÍNDICE

1.- Pasos a realizar para entrenar el modelo RBF **3**

2. Experimentos y análisis de los resultados **4**

2.1 Descripción de las bases de datos utilizadas: 4

2.2 Descripción de los valores de los parámetros considerados: 5

2.3 Resultados obtenidos y análisis: 6

-Configuración 1: 6

-Configuración 2: 8

-Configuración 3: 13

3. Bibliografía **16**

https://scikit-learn.org/stable/index.html **16**

1.- Pasos a realizar para entrenar el modelo RBF

Durante la realización de esta práctica, se implementará una red neuronal compuesta por nodos RBF, es decir, una red neuronal de base radial donde las neuronas tienen funciones locales a diferencia de las anteriores prácticas donde se utilizaban perceptrones multicapa.

Estas neuronas dependen de la distancia que existe entre el vector entrada a la función y un vector almacenado en la misma identificado como “centro”. Cada vez que a una de estas neuronas se le muestra un nuevo patrón, esta se encarga de calcular la distancia que existe hasta el centro y la neurona será activada en función de si dicha distancia es menor o mayor a un parámetro llamado radio.

Pasos de entrenamiento:

1.- Lectura de datos a entrenar, para este caso utilizaremos ficheros de datos de formato “.csv”, los leeremos y transformaremos los datos en las matrices correspondientes de entrenamiento y test para los valores de entrada y salida de nuestro programa.

2.- Inicializamos los centros y mediante el algoritmo de K Means, realizaremos Clustering para calcular la posición de cada uno de los centros inicializados con anterioridad.

3.- Cálculo de los radios para cada neurona mediante la distancia media que existe (diámetro) dividido entre dos (radio).

4.- Obtención de los pesos de capa de salida mediante regresión logística o clasificación.

5.- Comparar los resultados obtenidos junto a las predicciones.

2. Experimentos y análisis de los resultados

En este apartado, se realizarán los distintos experimentos con las bases de datos aportadas y los análisis de los resultados obtenidos mediante las distintas ejecuciones.

2.1 Descripción de las bases de datos utilizadas:

- **Función seno:** Compuesta por 120 patrones de entrenamiento y 41 patrones de test. Obtenida mediante la agregación de ruido aleatorio a la función seno.

- **Base de datos quake:** Compuesta por 1633 patrones de entrenamiento y 546 patrones de test. El objetivo de esta base de datos es la de averiguar la fuerza de un terremoto (escala Richter).

- **Base de datos de Parkinson:** Compuesta por 4406 patrones de entrenamiento y 1469 patrones de test. Sus valores de entrada son una serie de datos clínicos relacionados con la enfermedad de Parkinson y datos de medidas biométricas de la voz. Sus salidas son el valor motor y yoyal del UPDRS.

- **Base de datos divorce:** Compuesta por 127 patrones de entrenamiento y 43 de test con los datos de encuestas que pretenden predecir si se producirá un divorcio o no en una pareja. Las respuestas son de escala 0 a 4.

- **Base de datos noMNIST:** Compuesta por 900 patrones de entrada y 300 patrones de salida. Está formada por un conjunto de letras de la “a” a la “f” escritas en diferentes tipografías ajustadas a una rejilla de 28 x 28 píxeles.

2.2 Descripción de los valores de los parámetros considerados:

- **-t, --train_file:** Parámetro con el nombre del fichero con los datos de entrenamiento.

- -**T, --test_file:** Parámetro con el nombre del fichero con los datos de test.

- **-c, --classification:** Parámetro booleano que indica si el problema es de clasificación o no.

- **-r, --ratio_rbf:** Parámetro con la razón en tanto por uno de neuronas RBF con respecto al total de patrones en entrenamiento. Si no es especificado su valor es 0,1.

- **-l, --l2:** Booleano que indica si la regularización de L2 en lugar de la L1. Si no se especifica se utiliza L1.

- **-e, --eta:** Indica el valor del parámetro eta. Por defecto utiliza eta = 1e-2.

- **-o, --outputs:** Parámetro con el número de columnas de salida que tiene el conjunto de datos.

- **-h, --help:** Muestra ayuda del programa.

2.3 Resultados obtenidos y análisis:

-Configuración 1:

Para esta primera configuración de experimentos, consideraremos un número de neuronas en capa oculta (n1) igual al 5%, 15%, 25% y 50% del número de patrones de la base de datos. Utilizaremos regularización L1 y $\eta = 10^{-5}$.

-Función seno:

RBF	Training MSE	Test MSE
5%	0.013817 +- 0.000189	0.022182 +- 0.000245
15%	0.011216 +- 0.000068	0.367074 +- 0.079658
25%	0.010370 +- 0.000025	1.904300 +- 0.396859
50%	0.010357 +- 0.000002	2.063271 +- 0.401769

-Base de datos quake:

RBF	Training MSE	Test MSE
5%	0.028397 +- 0.000056	0.028377 +- 0.000262
15%	0.025373 +- 0.000101	0.040941 +- 0.002148
25%	0.022196 +- 0.000078	0.948248 +- 0.298864
50%	0.018450 +- 0.000077	158.170121 +- 58.579061

-Base de datos Parkinson:

RBF	Training MSE	Test MSE
5%	0.001670 +- 0.000067	0.002037 +- 0.000055
15%	0.000751 +- 0.000010	0.001291 +- 0.000087
25%	0.000413 +- 0.000016	0.001052 +- 0.000057
50%	0.000134 +- 0.000002	0.001282 +- 0.000140

-Base de datos divorce:

RBF	Training MSE	Test MSE	CCR train	CCR test
5%	0.000000 +- 0.000000	0.000000 +- 0.000000	100.00% +- 0.00%	100.00% +- 0.00%
15%	0.000001 +- 0.000000	0.000008 +- 0.000008	100.00% +- 0.00%	100.00% +- 0.00%
25%	0.000001 +- 0.000001	0.004479 +- 0.005148	100.00% +- 0.00%	99.07% +- 1.14%
50%	0.000001 +- 0.000000	0.004115 +- 0.004201	100.00% +- 0.00%	99.53% +- 0.93%

-Base de datos noMNIST:

RBF	Training MSE	Test MSE	CCR train	CCR test
5%	0.029815 +- 0.001447	0.029202 +- 0.001339	88.09% +- 0.54%	88.67% +- 0.87%
15%	0.000254 +- 0.000132	0.043202 +- 0.004372	100.00% +- 0.00%	86.33% +- 1.43%
25%	0.000034 +- 0.000006	0.039804 +- 0.004230	100.00% +- 0.00%	86.73% +- 1.48%
50%	0.000018 +- 0.000002	0.035192 +- 0.001981	100.00% +- 0.00%	87.93% +- 0.49%

Para este conjunto de ejecuciones, podemos observar que de manera generalizada (la base de datos de parkinson actúa de manera diferente) a más número de neuronas entrenamos en la red, peores son los resultados obtenidos (mayor error cuadrático medio y menor acierto ccr en clasificación).

Esto se debe al sobreentrenamiento causado por establecer un número elevado de neuronas RBF en la capa oculta respecto al total de patrones que contiene la base de datos.

En el caso de la base de datos Parkinson, el comportamiento es completamente contrario al resto de base de datos. Esto se debe a que es una base de datos con un número considerablemente mayor de patrones (con patrones muy diferentes entre sí) que el resto y por tanto, escoger un número mayor de neuronas provoca un mejor entrenamiento sin llegar al sobreentrenamiento.

-Configuración 2:

Para esta serie de ejecuciones, utilizaremos solamente aquellas base de datos empleadas para la clasificación, escogiendo en primer lugar el número de neuronas de capa oculta que mejores resultados produjo en la configuración 1.

Las ejecuciones tendrán como variante el valor de eta ($\eta = 1$, $\eta = 0{,}1$, $\eta = 0{,}01$, $\eta = 0{,}001$, ..., $\eta = 10^{-10}$) y el tipo de regularización empleado (L1 y L2).

-Base de datos divorce:

Para L1:

η	Training MSE	Test MSE	CCR train	CCR test
1	0.019916 +- 0.000734	0.021780 +- 0.000447	97.64% +- 0.00%	97.67% +- 0.00%
1e-1	0.005612 +- 0.001197	0.014107 +- 0.001736	99.21% +- 0.00%	97.67% +- 0.00%
1e-2	0.000214 +- 0.000110	0.001031 +- 0.001534	100.00% +- 0.00%	100.00% +- 0.00%
1e-3	0.000009 +- 0.000007	0.000069 +- 0.000124	100.00% +- 0.00%	100.00% +- 0.00%
1e-4	0.000000 +- 0.000000	0.000001 +- 0.000001	100.00% +- 0.00%	100.00% +- 0.00%
1e-5	0.000000 +- 0.000000	0.000000 +- 0.000000	100.00% +- 0.00%	100.00% +- 0.00%
1e-6	0.000000 +- 0.000000	0.000000 +- 0.000000	100.00% +- 0.00%	100.00% +- 0.00%
1e-7	0.000000 +- 0.000000	0.000000 +- 0.000001	100.00% +- 0.00%	100.00% +- 0.00%
1e-8	0.000000 +- 0.000000	0.000000 +- 0.000001	100.00% +- 0.00%	100.00% +- 0.00%
1e-9	0.000000 +- 0.000000	0.000000 +- 0.000001	100.00% +- 0.00%	100.00% +- 0.00%
1e-10	0.000000 +- 0.000000	0.000000 +- 0.000001	100.00% +- 0.00%	100.00% +- 0.00%

Para L2:

η	Training MSE	Test MSE	CCR train	CCR test
1	0.021194 +- 0.000810	0.023025 +- 0.000376	97.64% +- 0.00%	97.67% +- 0.00%
10-1	0.015424 +- 0.001866	0.020498 +- 0.000534	97.80% +- 0.31%	97.67% +- 0.00%
10-2	0.007780 +- 0.001360	0.017003 +- 0.001241	98.58% +- 0.31%	97.67% +- 0.00%
10-3	0.001432 +- 0.000591	0.005938 +- 0.001706	99.84% +- 0.31%	98.60% +- 1.14%
10-4	0.000086 +- 0.000049	0.000845 +- 0.001182	100.00% +- 0.00%	100.00% +- 0.00%
10-5	0.000003 +- 0.000002	0.000204 +- 0.000382	100.00% +- 0.00%	100.00% +- 0.00%
10-6	0.000000 +- 0.000000	0.000050 +- 0.000100	100.00% +- 0.00%	100.00% +- 0.00%
10-7	0.000000 +- 0.000000	0.000050 +- 0.000100	100.00% +- 0.00%	100.00% +- 0.00%
10-8	0.000000 +- 0.000000	0.000049 +- 0.000098	100.00% +- 0.00%	100.00% +- 0.00%
10-9	0.000000 +- 0.000000	0.000049 +- 0.000098	100.00% +- 0.00%	100.00% +- 0.00%
10-10	0.000000 +- 0.000000	0.000049 +- 0.000098	100.00% +- 0.00%	100.00% +- 0.00%

-Base de datos noMNIST:

Para L1:

η	Training MSE	Test MSE	CCR train	CCR test
1	0.045744 +- 0.000146	0.043087 +- 0.000493	84.78% +- 0.35%	88.00% +- 0.30%
10-1	0.022507 +- 0.000445	0.025228 +- 0.000246	91.89% +- 0.23%	91.20% +- 0.78%
10-2	0.008199 +- 0.000554	0.030453 +- 0.000879	97.87% +- 0.24%	88.87% +- 0.96%
10-3	0.000990 +- 0.000334	0.041107 +- 0.004654	99.98% +- 0.04%	86.60% +- 1.73%
10-4	0.000313 +- 0.000183	0.043002 +- 0.004671	100.00% +- 0.00%	86.27% +- 1.83%
10-5	0.000254 +- 0.000132	0.043202 +- 0.004372	100.00% +- 0.00%	86.33% +- 1.43%
10-6	0.000243 +- 0.000141	0.043547 +- 0.004522	100.00% +- 0.00%	86.13% +- 1.67%
10-7	0.000245 +- 0.000139	0.043565 +- 0.004508	100.00% +- 0.00%	86.07% +- 1.60%
10-8	0.000245 +- 0.000139	0.043565 +- 0.004508	100.00% +- 0.00%	86.07% +- 1.60%
10-9	0.000245 +- 0.000139	0.043565 +- 0.004508	100.00% +- 0.00%	86.07% +- 1.60%
10-10	0.000245 +- 0.000139	0.043565 +- 0.004508	100.00% +- 0.00%	86.07% +- 1.60%

Para L2:

η	Training MSE	Test MSE	CCR train	CCR test
1	0.049194 +- 0.000220	0.047693 +- 0.000373	84.27% +- 0.23%	87.20% +- 0.45%
10-1	0.030787 +- 0.000451	0.030122 +- 0.000340	89.16% +- 0.40%	91.07% +- 0.57%
10-2	0.020509 +- 0.000440	0.024916 +- 0.000340	92.60% +- 0.27%	91.33% +- 0.42%
10-3	0.011880 +- 0.000326	0.027499 +- 0.000753	96.58% +- 0.28%	89.00% +- 0.42%
10-4	0.004395 +- 0.000332	0.035574 +- 0.002080	99.31% +- 0.18%	87.53% +- 1.31%
10-5	0.000777 +- 0.000256	0.042644 +- 0.005024	100.00% +- 0.00%	86.33% +- 1.53%
10-6	0.000188 +- 0.000205	0.046317 +- 0.005313	100.00% +- 0.00%	85.33% +- 1.58%
10-7	0.000129 +- 0.000168	0.046971 +- 0.004922	100.00% +- 0.00%	85.27% +- 1.57%
10-8	0.000105 +- 0.000163	0.046347 +- 0.004991	100.00% +- 0.00%	85.40% +- 1.51%
10-9	0.000151 +- 0.000197	0.047346 +- 0.005141	100.00% +- 0.00%	85.33% +- 1.56%
10-10	0.000136 +- 0.000175	0.046923 +- 0.005009	100.00% +- 0.00%	85.40% +- 1.44%

Para todas las ejecuciones realizadas, podemos observar que sin tener en cuenta el tipo de regularización empleada, a mayor tasa de aprendizaje, peores resultados está obteniendo el modelo. Esto podría deberse de igual manera a la configuración 1 que el modelo se está sobreentrenando y por tanto, estamos obteniendo peores resultados.

-Configuración 3:

Para esta serie de ejecuciones, vamos a comparar los resultados obtenidos con anterioridad en todas las bases de datos (escogeremos la ejecución con el número de neuronas en capa oculta que mejor resultado otorgó la configuración 1) utilizando el nuevo parámetro para la función de K-Means: init='k-means++'.

```
if (classification):

    centr = init_centroids_classification(train_inputs, train_outputs, num_rbf)

    kmeans = KMeans(n_clusters = num_rbf, init='k-means++', n_init = 1, max_iter = 500).fit(train_inputs, train_outputs)

else:
    kmeans = KMeans(n_clusters = num_rbf, init = 'k-means++', n_init = 1, max_iter = 500).fit(train_inputs,train_outputs)
```

-Función seno:
RBF (0.05%)

KMeans	Train MSE	Test MSE
Random	0.013817 +- 0.000189	0.022182 +- 0.000245
K-means++	0.022301 +- 0.000292	0.022301 +- 0.000292

-Base de datos quake:
RBF(0.05%)

KMeans	Train MSE	Test MSE
Random	0.028397 +- 0.000056	0.028377 +- 0.000262
K-means++	0.028221 +- 0.000047	0.028686 +- 0.000103

-Base de datos Parkinson:
RBF(0.05%)

KMeans	Train MSE	Test MSE
Random	0.028397 +- 0.000056	0.028377 +- 0.000262
K-means++	0.001617 +- 0.000068	0.001899 +- 0.000073

-Base de datos divorce:
RBF(0.05%)

KMeans	Train MSE	Test MSE	CCR train	CCR test
Random	0.000000 +- 0.000000	0.000000 +- 0.000000	100.00% +- 0.00%	100.00% +- 0.00%
K-means++	0.001617 +- 0.000068	0.001899 +- 0.000073	100.00% +- 0.00%	100.00% +- 0.00%

-Base de datos noMNIST:
RBF(0.05%)

KMeans	Train MSE	Test MSE	CCR train	CCR test
Random	0.029815 +- 0.001447	0.029202 +- 0.001339	88.09% +- 0.54%	88.67% +- 0.87%
K-means++	0.029078 +- 0.000597	0.029127 +- 0.000643	87.93% +- 0.31%	88.93% +- 0.83%

Podemos observar que existen ejecuciones que mejoran los resultados obtenidos como puede ser la base de datos de Parkinson, otras que mantienen resultados como la base de datos quake y otras que empeoran su resultado como la base de datos de función seno.

k-means + + trata de seleccionar de manera más inteligente los centros iniciales de los clusters para acelerar la convergencia de los resultados.

3. Bibliografía

---

https://scikit-learn.org/stable/index.html

https://stackoverflow.com/questions/3518778/how-do-i-read-csv-data-into-a-record-array-in-numpy

https://www.aprendemachinelearning.com/k-means-en-python-paso-a-paso/

https://www.it-swarm-es.com/es/python/es-sklearn.metrics.mean-squared-error-cuanto-mas-grande-mejor-negado/836560115/

https://docs.scipy.org/doc/scipy/reference/generated/scipy.spatial.distance.squareform.html