Redes Neuronales Multicapa II

Modelos de la computación (Aprendizaje supervisado)

Francisco Fernández Navarro

Departamento de Lenguajes y Ciencias de la Computación
Área: Ciencias de la Computación e Inteligencia Artificial

Índice de contenidos

1. Perceptrón multicapa: Modelo probabilístico y regla de clasificación
   - Regla de clasificación
   - Modelo probabilístico
2. Aplicaciones del Perceptrón Multicapa
   - Introducción
   - Casos de estudio
3. Redes de funciones de base radial
   - Introducción
   - Modelo funcional
   - Problema de optimización
   - Estimación de parámetros
4. Validación de modelos
   - Problemas estimación hiper-parámetros
   - Estimación mediante validación cruzada

Regla de clasificación

- El vector de salida para el patrón $\mathbf{x}_n$ del modelo podría definirse como $\hat{\mathbf{y}}(\mathbf{x}_n) = (y_{n1}, y_{n2}, \ldots, y_{nJ}) \in \mathbb{R}^J$. Tal y como podemos observar el vector de salida da como resultado la salida del patrón $\mathbf{x}_n$ en formato continuo, es decir, $\hat{\mathbf{y}}(\mathbf{x}_n) \in \mathbb{R}^J$.
- En el problema de clasificación, se requieren salidas en formato discreto, es decir, $\{0, 1\}^J$.
- Para lograr esto, se realiza una discretización de la salida continua de la red, creando un vector $\hat{\mathbf{o}}(\mathbf{x}_n) \in \{0, 1\}^J$ cuya componente j-ésima es igual a 1 si la dimensión correspondiente en $\hat{\mathbf{y}}(\mathbf{x}_n)$ es la máxima, y cero en caso contrario.

Índice de contenidos

1. Perceptrón multicapa: Modelo probabilístico y regla de clasificación
   - Regla de clasificación
   - Modelo probabilístico
2. Aplicaciones del Perceptrón Multicapa
   - Introducción
   - Casos de estudio
3. Redes de funciones de base radial
   - Introducción
   - Modelo funcional
   - Problema de optimización
   - Estimación de parámetros
4. Validación de modelos
   - Problemas estimación hiper-parámetros
   - Estimación mediante validación cruzada

Modelo probabilístico

- El problema de usar una función sigmoidal en la capa de salida de una red neuronal es que esta función está diseñada para producir salidas en el rango de $[0, 1]$ para cada neurona, lo que significa que cada neurona de salida genera una probabilidad independiente entre 0 y 1.
- Adecuado para problemas de clasificación binaria, donde se desea determinar si un ejemplo pertenece a una de dos clases mutuamente excluyentes.
- En un problema multiclase la función sigmoidal no garantiza que las salidas de todas las neuronas de la capa de salida sumen 1.

Modelo probabilístico

Para abordar este problema en problemas de clasificación multiclase, se utiliza la función softmax en la capa de salida. La función softmax toma como entrada un vector de números reales y produce un vector de probabilidades normalizadas que suman 1. Matemáticamente, la función softmax se define como:

$$\mathrm{P}(\mathcal{C}_j, \mathbf{x}_n) = \frac{\exp(\hat{y}_{nj})}{\sum_{j=1}^{J} \exp(\hat{y}_{nj})},$$

donde $\mathrm{P}(\mathcal{C}_j \mathbf{x}_n)$ es la probabilidad de pertenecencia a la clase $\mathcal{C}_j$ del patrón n-ésimo.

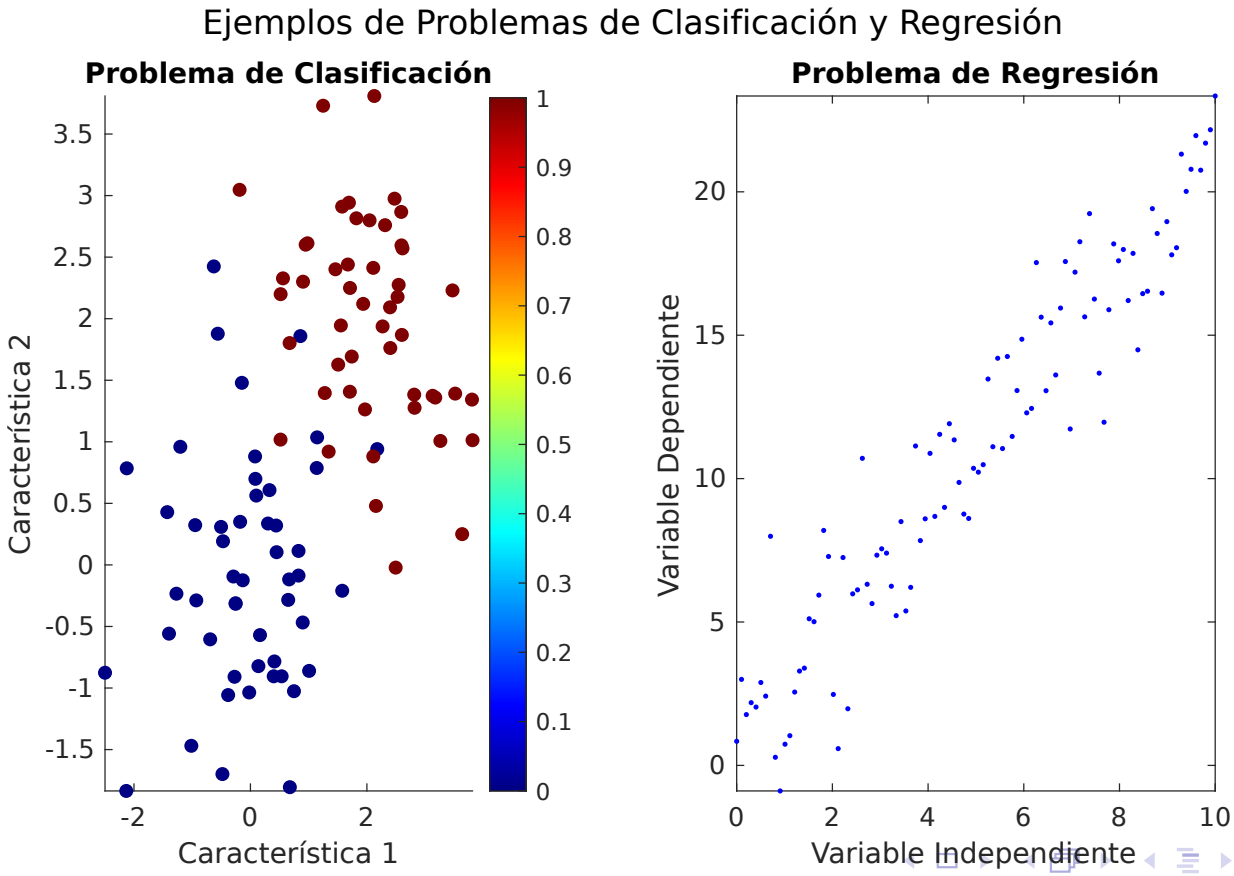
En resumen, mientras que la función sigmoidal es adecuada para problemas de clasificación binaria, la función softmax es preferible para problemas de clasificación multiclase.

Índice de contenidos

1. Perceptrón multicapa: Modelo probabilístico y regla de clasificación
   - Regla de clasificación
   - Modelo probabilístico
2. Aplicaciones del Perceptrón Multicapa
   - Introducción
   - Casos de estudio
3. Redes de funciones de base radial
   - Introducción
   - Modelo funcional
   - Problema de optimización
   - Estimación de parámetros
4. Validación de modelos
   - Problemas estimación hiper-parámetros
   - Estimación mediante validación cruzada

Introducción

- El perceptrón multicapa es un **aproximador universal**.
  - Puede calcular cualquier función booleana.
  - Puede aproximar cualquier función continua.
- **Clasificación** de patrones (2 ó más clases)
- **Regresión**



Índice de contenidos

1. Perceptrón multicapa: Modelo probabilístico y regla de clasificación
   - Regla de clasificación
   - Modelo probabilístico
2. Aplicaciones del Perceptrón Multicapa
   - Introducción
   - Casos de estudio
3. Redes de funciones de base radial
   - Introducción
   - Modelo funcional
   - Problema de optimización
   - Estimación de parámetros
4. Validación de modelos
   - Problemas estimación hiper-parámetros
   - Estimación mediante validación cruzada

Predicción de la deuda soberana

Definición problema

Problema de clasificación ordinal para estimar la calidad de la deuda soberana de los países de la Eurozona (A+++, A++, . . . , C). Esta calificación de deuda es emitida por agencias de calificación crediticia como Standard & Poor's (S&P) y Moody's. Las variables utilizadas para estimar estas calificaciones incluyen la Deuda Pública, PIB, inflación, entre otros.

Addressing the EU sovereign ratings using an ordinal regression approach

Francisco Fernández-Navarro, Pilar Campoy-Muñoz, Mónica de la Paz-Marín, César Hervás-Martínez *Member, IEEE,*, Xin Yao, *Fellow, IEEE,*

***Abstract*—The current European debt crisis has drawn considerable attention to credit rating agencies' news about sovereign ratings. From a technical point of view, credit rating constitutes a typical ordinal regression problem because credit rating agencies generally present a scale of risk composed several categories. This fact motivated the use of an ordinal regression approach for**

amount of capital to cover their credit risks (see, inter alia, [2], [3]).

These CRAs have been in the spotlight during the ongoing European sovereign debt crisis. This crisis has been a theater of sovereign credit rating downgrades, a widening of sovereign

Microbiología predictiva

Definición problema

Determinar si el Staphylococcus aureus crecerá o no en un entorno dado en función de tres características ambientales: pH, temperatura y actividad del agua (aw).

Applied Soft Computing 11 (2011) 3012–3020

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Evolutionary q-Gaussian Radial Basis Function Neural Network to determine the microbial growth/no growth interface of *Staphylococcus aureus*

Francisco Fernández-Navarro[a,*], César Hervás-Martínez[a], M. Cruz-Ramírez[a], Pedro Antonio Gutiérrez[a], Antonio Valero[b]

[a] Department of Computer Science and Numerical Analysis, University of Córdoba, Campus de Rabanales, Albert Einstein Building, 3rd Floor, 14071 Córdoba, Spain
[b] Department of Food Science and Technology, University of Córdoba, Campus de Rabanales, Darwin Building, 14014 Córdoba, Spain

Predicción crisis en banco

Definición problema

Determinar si un banco está en riesgo de entrar en crisis financiera o no en función de diversas variables financieras y económicas. Algunas variables que podrían ayudar en este problema incluyen el ratio de capitalización, la liquidez, o el tamaño del banco, entre otras.

Omega 38 (2010) 333–344

Contents lists available at ScienceDirect

Omega

journal homepage: www.elsevier.com/locate/omega

Hybridizing logistic regression with product unit and RBF networks for accurate detection and prediction of banking crises

P.A. Gutiérrez [a], M.J. Segovia-Vargas [b], S. Salcedo-Sanz [c,*], C. Hervás-Martínez [a], A. Sanchis [d], J.A. Portilla-Figueras [c], F. Fernández-Navarro [a]

[a] Department of Computer Science and Numerical Analysis, Universidad de Córdoba, Córdoba, Spain
[b] Department of Financial Economics and Accounting I, Universidad Complutense de Madrid, Madrid, Spain
[c] Department of Signal Theory and Communications, Universidad de Alcalá, 28871 Alcalá de Henares, Madrid, Spain
[d] Bank of Spain, Madrid, Spain

Índice de contenidos

1. Perceptrón multicapa: Modelo probabilístico y regla de clasificación
   - Regla de clasificación
   - Modelo probabilístico
2. Aplicaciones del Perceptrón Multicapa
   - Introducción
   - Casos de estudio
3. Redes de funciones de base radial
   - Introducción
   - Modelo funcional
   - Problema de optimización
   - Estimación de parámetros
4. Validación de modelos
   - Problemas estimación hiper-parámetros
   - Estimación mediante validación cruzada

Introducción a Redes RBF

Motivación Principal

Las Redes de Funciones de Base Radial (RBF) son un tipo de red neuronal artificial que se diferencia de las redes MLP (Perceptrón Multicapa) en su arquitectura y funcionamiento.

Neuronas de Capa Oculta en Puntos Específicos

- En una red RBF, la capa oculta tiene neuronas que se colocan en puntos específicos del espacio de entrada. Estos puntos se llaman "centros".
- Cada neurona de la capa oculta se asocia con un centro y se activa cuando los datos de entrada están cerca de ese centro en el espacio de entrada.

Introducción a Redes RBF

Función de activación

La función de activación utilizada en las neuronas de la capa oculta de una RBF suele ser la función gaussiana.

Proceso de Evaluación

- Cuando se presenta un dato de entrada a la red, cada neurona de la capa oculta calcula su activación basada en la distancia entre el dato de entrada y su centro correspondiente utilizando la función gaussiana.
- Estas activaciones representan cuán cerca está el dato de entrada de cada centro.

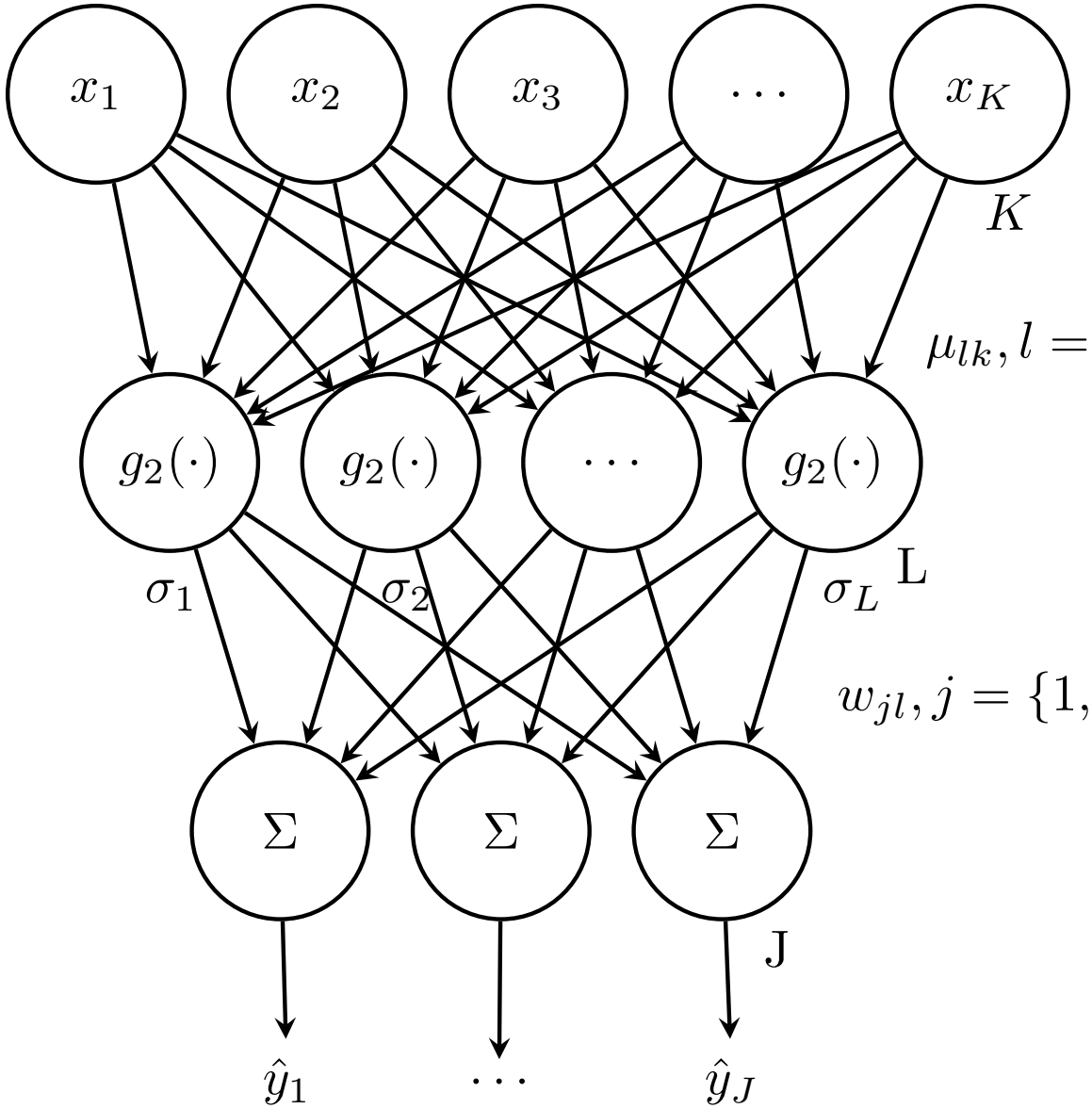
Aprendizaje

- El aprendizaje en una red RBF implica ajustar los centros y las anchuras de las funciones gaussianas para que la red pueda representar eficazmente la relación entre los datos de entrada y las salidas deseadas.
- Se pueden utilizar técnicas como el algoritmo k-means para inicializar los centros y métodos de optimización para ajustar los parámetros.

Índice de contenidos

1. Perceptrón multicapa: Modelo probabilístico y regla de clasificación
   - Regla de clasificación
   - Modelo probabilístico
2. Aplicaciones del Perceptrón Multicapa
   - Introducción
   - Casos de estudio
3. Redes de funciones de base radial
   - Introducción
   - Modelo funcional
   - Problema de optimización
   - Estimación de parámetros
4. Validación de modelos
   - Problemas estimación hiper-parámetros
   - Estimación mediante validación cruzada

El modelo no tiene sesgo en capa oculta. Por simplicidad, tampoco lo consideraremos en capa de salida (donde sí que podría tenerlo).



$\mu_{lk}, l = \{1, \ldots, L\}, k = \{1, \ldots, K\}$

$w_{jl}, j = \{1, \ldots, J\}, l = \{1, \ldots, L\}$

Nomenclatura Redes RBF

Parámetros

En Redes RBF tenemos que estimar los valores de dos matrices y un vector:

- $\mathbf{w} = (\mathbf{w}_1\ \mathbf{w}_2\ \ldots \mathbf{w}_J) \in \mathbb{R}^{L \times J}$: Matriz con los valores de los pesos sinápticos de las neuronas que conectan capa de salida con capa oculta. $\mathbf{w}_j \in \mathbb{R}^L$ son las conexiones de la neurona de salida j con la capa oculta, $\forall j \in \{1, \ldots, J\}$.
- $\boldsymbol{\mu} = (\boldsymbol{\mu}'_1, \boldsymbol{\mu}'_2, \ldots, \boldsymbol{\mu}'_L) \in \mathbb{R}^{L \times K}$: Matriz con los centroides de las neuronas en capa oculta. $\boldsymbol{\mu}_l \in \mathbb{R}^K$ es el centroide de la neurona l-ésima, $\forall l \in \{1, \ldots, L\}$.
- $\boldsymbol{\sigma} = (\sigma_1, \sigma_2, \ldots, \sigma_L) \in \mathbb{R}^L$: Vector con los radios de las neuronas de capa oculta. Determinan el ancho de activación de la neurona. $\sigma_l \in \mathbb{R}$ es el radio (ancho) de la neurona l-ésima, $\forall l \in \{1, \ldots, L\}$.

Modelo funciónal

Modelo funcional

La salida de la red RBF para un patrón n-ésimo, $\mathbf{x}_n \in \mathbb{R}^K$ es:

$$\hat{y}_{nj} = \sum_{l=1}^{L} w_{lj} s_{ln} = \sum_{l=1}^{L} w_{lj} \mathrm{g}_2 \left(\mathbf{x}_n ; \boldsymbol{\mu}_l, \sigma_l \right) . \tag{1}$$

Definición de variables

- $\hat{y}_{nj}$: Estimación de salida de la clase j para el patrón n.
- s_{ln}: Salida de la neurona en capa oculta l para el patrón n.
- $\mathrm{g}_2(\mathbf{x}_n; \boldsymbol{\mu}_l, \sigma_l)$: Función de transferencia de capa oculta.

Importante

Observar que, por simplicidad, no hemos incluido función de transferencia en la capa de salida (aunque se podría haber añadido).

Funciones de transferencia redes RBF

$$\text{Función Gaussiana:} \quad \mathrm{g}_2(\mathbf{x}_n; \boldsymbol{\mu}_l, \sigma_l) = \exp\left(-\frac{u_l^2}{\sigma_l^2}\right)$$

$$\text{Cuádrica Inversa de Hardy:} \quad \mathrm{g}_2(\mathbf{x}_n; \boldsymbol{\mu}_l, \sigma_l, \beta_l) = (\sigma_l^2 + u_l^2)^{\beta_l}$$

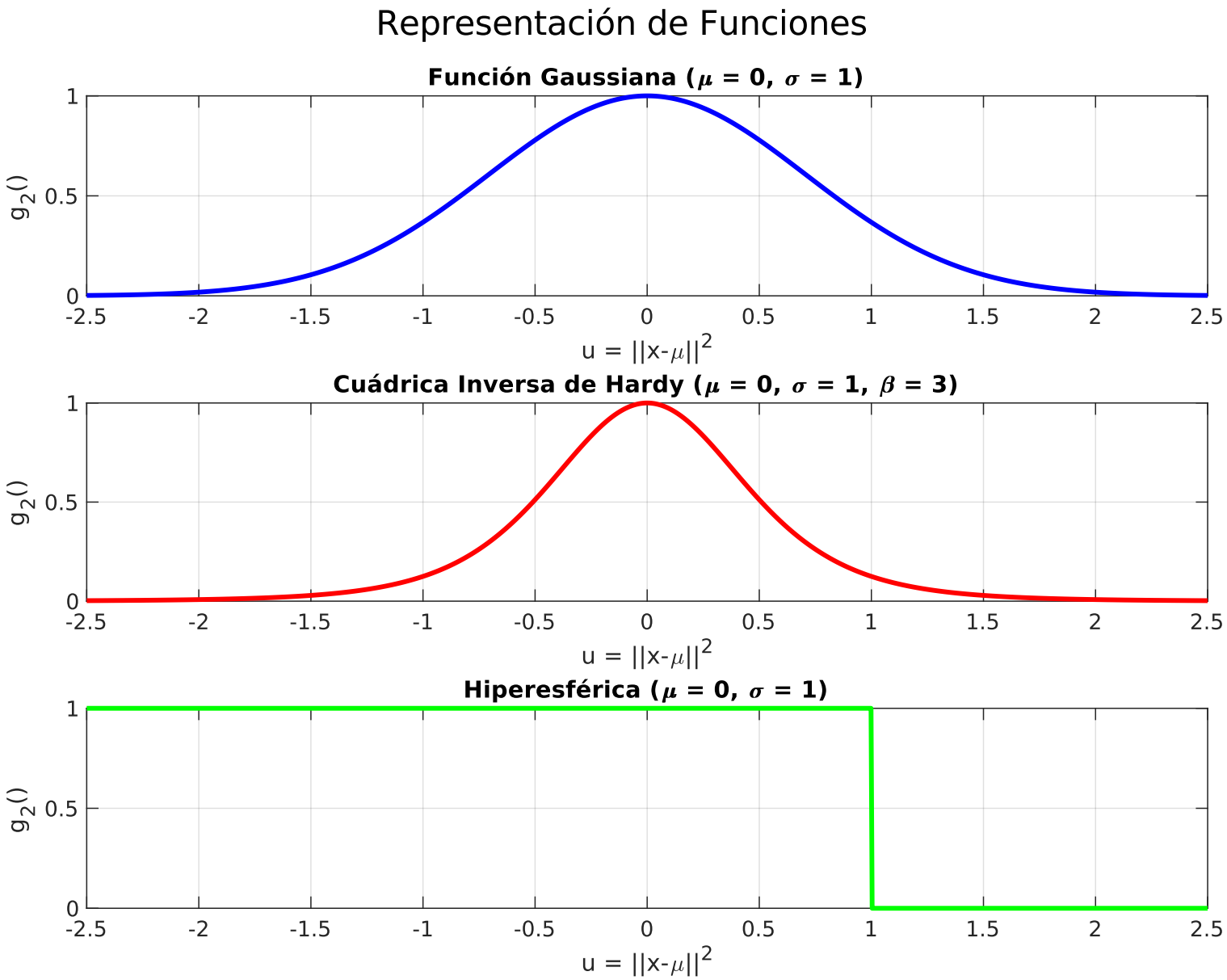
$$\text{Hiperesférica:} \quad \mathrm{g}_2(\mathbf{x}_n; \boldsymbol{\mu}_l, \sigma_l) = \begin{cases} 1 & \text{si } u_l \leq \sigma_l \\ 0 & \text{en caso contrario} \end{cases}$$

$$\text{con } u = ||\mathbf{x}_n - \boldsymbol{\mu}_l||^2.$$

Importante

Nosotros nos centraremos en la gaussiana (por simplicidad y porque es la más empleada en el entorno académico).

Funciones de transferencia redes RBF



Índice de contenidos

1. Perceptrón multicapa: Modelo probabilístico y regla de clasificación
   - Regla de clasificación
   - Modelo probabilístico
2. Aplicaciones del Perceptrón Multicapa
   - Introducción
   - Casos de estudio
3. Redes de funciones de base radial
   - Introducción
   - Modelo funcional
   - Problema de optimización
   - Estimación de parámetros
4. Validación de modelos
   - Problemas estimación hiper-parámetros
   - Estimación mediante validación cruzada

Problema de optimización

Problema de optimización

Los parámetros del modelo RBF se estiman a través del siguiente problema de optimización:

$$\min_{\mathbf{w}\in\mathbb{R}^{L\times J},\boldsymbol{\mu}\in\mathbb{R}^{L\times K},\boldsymbol{\sigma}\in\mathbb{R}^{L}} \quad E(\mathbf{w},\boldsymbol{\mu},\boldsymbol{\sigma}) = \frac{1}{2}\sum_{j=1}^{J}\sum_{n=1}^{N}(y_{nj}-\hat{y}_{nj})^2.$$

Índice de contenidos

1. Perceptrón multicapa: Modelo probabilístico y regla de clasificación
   - Regla de clasificación
   - Modelo probabilístico
2. Aplicaciones del Perceptrón Multicapa
   - Introducción
   - Casos de estudio
3. Redes de funciones de base radial
   - Introducción
   - Modelo funcional
   - Problema de optimización
   - Estimación de parámetros
4. Validación de modelos
   - Problemas estimación hiper-parámetros
   - Estimación mediante validación cruzada

Nomenclatura

El procedimiento será iterativo ($i = 1, \ldots, I$) y las modificaciones se realizarán de patrón a patrón ($1 \leq n \leq N$), por lo que será necesario definir:

- $\hat{y}_{nj}(i)$: Estimación de salida de la clase j para el patrón n, en la iteración i.
- $w_{lj}(i, n)$: Peso sináptico neurona salida j y neurona oculta l, en la iteración i (patrón n).
- $\mu_{lk}(i, n)$: Centroide que conecta la neurona oculta l y entrada k (iteración i, patrón n).
- $\sigma_l(i, n)$: Radio de la neurona oculta l (iteración i, patrón n).

Pesos de capa oculta a capa de salida

Estimación $w_{lj}, j = \{1, \ldots, J\}, l = \{1, \ldots, L\}$

La regla de modificación de los pesos sinápticos de la capa de salida será:

$$w_{lj}(i,n) = w_{lj}(i,n-1) + \Delta w_{lj}(i,n), \tag{2}$$

donde

$$\Delta w_{lj}(i,n) = -\eta \frac{\partial E}{\partial w_{lj}(i,n)} = \eta(y_{nj} - \hat{y}_{nj}(i)) \exp\left(-\frac{||\mathbf{x}_n - \boldsymbol{\mu}_l(i,n)||^2}{(\sigma_l(i,n))^2}\right), \tag{3}$$

donde $\boldsymbol{\mu}_l(i,n) = (\mu_{l1}(i,n), \mu_{l2}(i,n), \ldots, \mu_{lK}(i,n)) \in \mathbb{R}^K$.

Pesos de capa oculta a capa de entrada

Estimación $\mu_{lk}, k = \{1, \ldots, K\}, l = \{1, \ldots, L\}$

La regla de modificación de los pesos sinápticos de la capa de oculta (centroides) será:

$$\mu_{lk}(i,n) = \mu_{lk}(i,n-1) + \Delta\mu_{lk}(i,n), \tag{4}$$

donde

$$\begin{aligned}
\Delta\mu_{lk}(i,n) &= -\eta \frac{\partial E}{\partial \mu_{lk}(i,n)} \\
&= \eta(y_{nj} - \hat{y}_{nj}(i)) \sum_{j=1}^{J} w_{lj}(i,n) \exp\left(-\frac{||\mathbf{x}_n - \boldsymbol{\mu}_l(i,n)||^2}{(\sigma_l(i,n))^2}\right) \\
&\quad \frac{2(x_{nk} - \mu_{lk}(i,n))}{(\sigma_l(i,n))^2}.
\end{aligned}$$

Pesos de capa oculta a capa de entrada

Estimación $\sigma_l, l = \{1, \ldots, L\}$

La regla de modificación de los pesos sinápticos de la capa de oculta (radios) será:

$$\sigma_l(i,n) = \sigma_l(i,n-1) + \Delta\sigma_l(i,n), \tag{5}$$

donde

$$\begin{aligned}
\Delta\sigma_l(i,n) &= -\eta \frac{\partial E}{\partial \sigma_l(i,n)} \\
&= \eta(y_{nj} - \hat{y}_{nj}(i)) \sum_{j=1}^{J} w_{lj}(i,n) \exp\left(-\frac{||\mathbf{x}_n - \boldsymbol{\mu}_l(i,n)||^2}{(\sigma_l(i,n))^2}\right) \\
&\quad \frac{2||\mathbf{x}_n - \boldsymbol{\mu}_l(i,n)||^2}{(\sigma_l(i,n))^3}.
\end{aligned}$$

Índice de contenidos

1. Perceptrón multicapa: Modelo probabilístico y regla de clasificación
   - Regla de clasificación
   - Modelo probabilístico
2. Aplicaciones del Perceptrón Multicapa
   - Introducción
   - Casos de estudio
3. Redes de funciones de base radial
   - Introducción
   - Modelo funcional
   - Problema de optimización
   - Estimación de parámetros
4. Validación de modelos
   - Problemas estimación hiper-parámetros
   - Estimación mediante validación cruzada

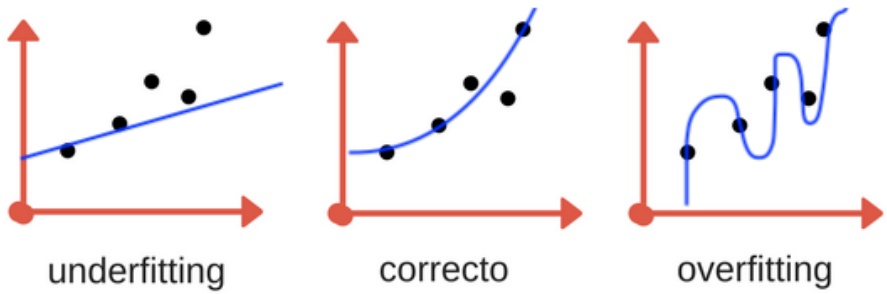
Problemas estimación

Definición hiper-parámetro

Parámetro que no se aprende automáticamente del conjunto de datos durante el proceso de entrenamiento del modelo, sino que debe configurarse antes de iniciar el entrenamiento. Suelen controlar la complejidad del modelo. En el caso de redes neuronales, L.

Posibles problemas

- Subajuste (**underfitting**). Falta de precisión en la predicción. Deberíamos aumentar el valor de L.
- Superajuste (**overfitting**). Falta de capacidad de generalización. Deberíamos reducir el valor de L.



Índice de contenidos

1. Perceptrón multicapa: Modelo probabilístico y regla de clasificación
   - Regla de clasificación
   - Modelo probabilístico
2. Aplicaciones del Perceptrón Multicapa
   - Introducción
   - Casos de estudio
3. Redes de funciones de base radial
   - Introducción
   - Modelo funcional
   - Problema de optimización
   - Estimación de parámetros
4. Validación de modelos
   - Problemas estimación hiper-parámetros
   - Estimación mediante validación cruzada

Validación cruzada

1. División de los datos

Divide tu conjunto de datos en tres conjuntos distintos: entrenamiento, validación y test. Ejemplo: 70% para entrenamiento, 15% para validación y 15% para test.

2. Selección de hiperparámetros

Elije un conjunto de valores para el hiperparámetro a optimizar. Ejemplo: $L \in \{5, 10, 15, 20\}$.

3. Bucle de entrenamiento y validación

Iterar a través de los diferentes valores del hiperparámetro y para cada valor:

- Entrena tu modelo en el conjunto de entrenamiento utilizando el valor específico del hiperparámetro.
- Evalúa el rendimiento del modelo en el conjunto de validación.

Validación cruzada

4. Selección del mejor hiperparámetro

Después de completar el bucle de entrenamiento y validación para todos los valores del hiperparámetro, selecciona el valor del hiperparámetro que dio el mejor rendimiento en el conjunto de validación.

5. Evaluación final en el conjunto de prueba

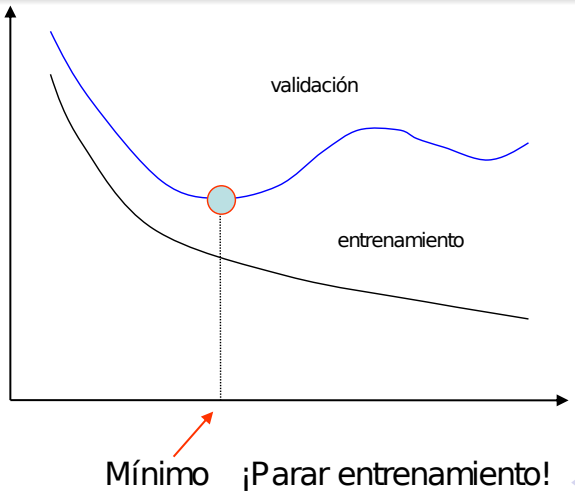
Una vez que hayas seleccionado el valor óptimo del hiperparámetro en función de los resultados de validación, evalúa el modelo final con ese valor en el conjunto de prueba (entrenamiento + validación).

Validación cruzada

Objetivo final

Que el clasificador consiga un error de generalización pequeño (error en el conjunto de test).

El error de entrenamiento de un clasificador decrece monótonamente durante la fase de entrenamiento, mientras que el error sobre el conjunto de validación decrece hasta un punto a partir del cual crece (superajuste). El proceso de entrenamiento debe finalizar cuando se alcance el primer mínimo de la función del error de validación.



¡Gracias por vuestra atención!