

Modelos de Redes Neuronales para Regresión Ordinal basados en Técnicas de Descenso por Gradiente

Autor:

Raúl Pérula Martínez

Directores:

Dr. Pedro Antonio Gutiérrez Peña

Dr. César Hervás Martínez

Universidad de Córdoba
Escuela Politécnica Superior

01 de septiembre de 2011

- 1 Introducción
- 2 Antecedentes
- 3 Análisis del Sistema
- 4 Diseño
- 5 La Aplicación
- 6 Resultados
- 7 Conclusiones
- 8 Bibliografía

- 1 Introducción
- 2 Antecedentes
- 3 Análisis del Sistema
- 4 Diseño
- 5 La Aplicación
- 6 Resultados
- 7 Conclusiones
- 8 Bibliografía

Modelado de sistemas

Definición

- Consiste en establecer una relación funcional entre las variables que intervienen en un fenómeno de estudio. Dos de los problemas más comunes en el campo del modelado son los de *regresión* y *clasificación*.

Regresión ordinal

- Los algoritmos de regresión ordinal son un método matemático que modela la relación entre una variable dependiente Y en escala ordinal y las variables independientes X. Permite determinar si un objeto o patrón tiene un grado de una determinada cualidad mayor o menor que otro objeto, utilizando para ello una escala ordinal que nos indica la posición relativa.
- Son numerosas las áreas de investigación que abordan problemas de clasificación y regresión, lo cual causa que exista una gran cantidad de algoritmos asociados a estas metodologías, y que se propongan cada año en las revistas técnicas especializadas.

Definición del Problema

- Desarrollar la implementación software del algoritmo teórico ORNNet realizado por el grupo de investigación AYRNA para resolver problemas de regresión ordinal mediante redes neuronales haciendo la implementación e integración para el toolbox de Matlab *nnet*.
- Realizar una comparativa de los resultados obtenidos al aplicar los algoritmos de clasificación nominal y ordinal utilizados.
- Desarrollar una interfaz gráfica usable que muestre el funcionamiento de dicho algoritmo y proporcione una visión clara del método implementado.

La interfaz gráfica podrá realizar:

- 1 Carga de un conjunto de datos.
- 2 Configuración de la red neuronal ordinal.
- 3 Entrenamiento y simulación de la red neuronal ordinal.
- 4 Exportación de los resultados.

Objetivos

- Desarrollar un algoritmo de regresión ordinal basado en redes neuronales, mediante la utilización de una función de ranking $f(x)$ formada por la suma ponderada de un conjunto de funciones de base de tipo sigmoide. En concreto los objetivos son:
 - 1 Implementar el algoritmo iRPROP+ (el cual es una mejora del algoritmo RPROP) para el toolbox de Matlab *nnet*.
 - 2 Desarrollar el algoritmo de regresión ordinal basado en Redes Neuronales realizado de forma teórica por el grupo de investigación AYRNA.
 - 3 Modificar el modelo funcional estándar de redes neuronales para clasificación considerando una red neuronal de una sola neurona en capa de salida y aplicar una transformación de la salida que aproxime un valor de probabilidad de pertenencia a cada una de las clases.
 - 4 Implementar una interfaz gráfica que facilite la utilización del algoritmo implementado.

Factores Estratégicos

- La aplicación desarrollada será multiplataforma de manera que pueda ser utilizada en otros Sistemas Operativos.
- Se utilizará el lenguaje de programación proporcionado por el software Matlab, debido a su alta capacidad de cómputo, su toolbox para el manejo e implementación de redes neuronales y a la posibilidad de realizar interfaces gráficas.
- Se utilizará el lenguaje \LaTeX , para la realización de la documentación, gracias a su capacidad para generar documentación formal de manera intuitiva y sencilla, destacando su buena presentación.
- Para la realización de diagramas *UML* se usará la herramienta *Dia*.

- 1 Introducción
- 2 Antecedentes**
- 3 Análisis del Sistema
- 4 Diseño
- 5 La Aplicación
- 6 Resultados
- 7 Conclusiones
- 8 Bibliografía

Redes neuronales

- Una de las alternativas más utilizadas en los últimos años para la resolución de *problemas de clasificación no lineal* ha sido la aplicación de redes neuronales artificiales.
- Ésta técnica de modelado es enormemente flexible y en general suele producir buenos resultados.
- Una red neuronal posee una **capa de entrada** (por la que se introducen los valores de las variables independientes o variables de entrada), una o más **capas ocultas** (que realizan la transformación no lineal de dichos valores) y una **capa de salida** (de la que se pueden obtener los valores predichos de la variable dependiente de salida).

Algoritmo iRPROP+ (RPROP mejorado)

- Es una variante del algoritmo RPROP.
- Añade un paso de vuelta atrás al algoritmo que permite evitar los óptimos locales.
- Cuando el cambio en un parámetro (w_{ij}) de la red produzca un aumento del valor de la función de error (E), el algoritmo vuelve al estado anterior a producirse dicho cambio.

$$\Delta w_{ij}(t) = \begin{cases} \alpha^+ \cdot \Delta w_{ij}(t-1), & \text{si } \frac{\partial E}{\partial w_{ij}}(t-1) \cdot \frac{\partial E}{\partial w_{ij}}(t) > 0 \\ \Delta w_{ij}(t-1) - \Delta w_{ij}(t-2), & \text{si } \frac{\partial E}{\partial w_{ij}}(t-1) \cdot \frac{\partial E}{\partial w_{ij}}(t) < 0 \\ & \text{y } E(t) > E(t-1) \\ \Delta w_{ij}(t-1), & \text{si } \frac{\partial E}{\partial w_{ij}}(t-1) \cdot \frac{\partial E}{\partial w_{ij}}(t) = 0 \end{cases}$$

Medidas de rendimiento

Matriz de confusión

- Contiene información acerca de las clasificaciones reales y las predichas realizadas por un sistema de clasificación.
- Es una herramienta de visualización usada normalmente en aprendizaje supervisado.
- Uno de los beneficios de usar una matriz de confusión es, que es muy fácil ver si el sistema distingue entre las clases y poder construir a partir de ella medidas de “bondad” del clasificador.

Medidas de rendimiento

Confusion Matrix

	1	2	3	
1	69 43.9%	0 0.0%	0 0.0%	100% 0.0%
2	2 1.3%	13 8.3%	4 2.5%	68.4% 31.6%
3	1 0.6%	0 0.0%	68 43.3%	98.6% 1.4%
	95.8% 4.2%	100% 0.0%	94.4% 5.6%	95.5% 4.5%
	1	2	3	
	Target Class			

Medidas de rendimiento

Ratio Correctamente Clasificado (CCR)

- Cuando hay más de dos clases en el modelo.
- Se obtiene a partir de la matriz de confusión.
- Es la suma del número de predicciones correctas dividido por el número total de elementos de la matriz.

$$CCR = \frac{\text{nº de bien clasificados}}{\text{nº total de patrones}}$$

Medidas de rendimiento

Error Absoluto Medio (MAE)

- Es una cantidad usada para medir como de buenos son los pronósticos o predicciones de los resultados. Es necesario asignar etiquetas consecutivas a cada una de las J clases de la clasificación ordinal.

$$MAE = \frac{1}{J} \sum_{i=1}^J |f_i - y_i| = \sum_{i=1}^J |e_i|$$

- El error absoluto medio es una media de los errores absolutos, $e_i = f_i - y_i$, donde f_i es la predicción e y_i es el valor verdadero.

- 1 Introducción
- 2 Antecedentes
- 3 Análisis del Sistema**
- 4 Diseño
- 5 La Aplicación
- 6 Resultados
- 7 Conclusiones
- 8 Bibliografía

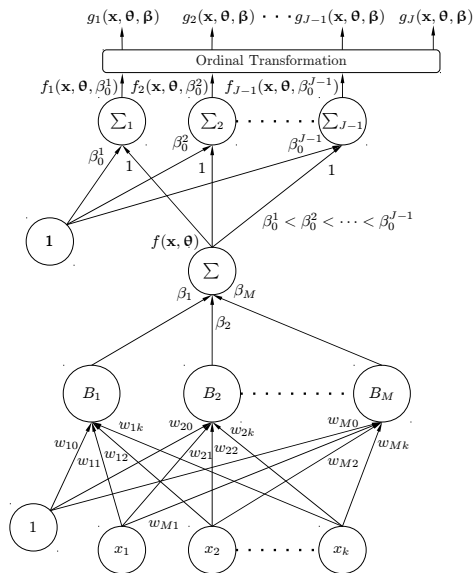
Algoritmo de Regresión Ordinal ORNNet

La idea de este algoritmo es proyectar los patrones a una línea recta y clasificar según los distintos umbrales de la última capa.

Modelo de red

- El modelo de red neuronal artificial ordinal se basa en:
 - Una capa de entrada.
 - Dos capas ocultas.
 - Una capa de salida.

Algoritmo de Regresión Ordinal ORNNet

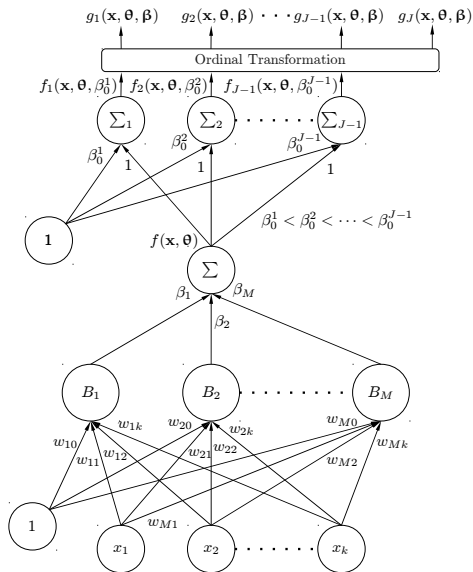


Algoritmo de Regresión Ordinal ORNNet

Capa de entrada

- La capa de entrada tomará los valores de las variables de entrenamiento de entrada (x_1, x_2, \dots, x_n) que al pasar a la siguiente capa generarán nuevos valores (B_1, B_2, \dots, B_M) haciendo uso de funciones de base sigmoides, utilizando los valores de los pesos (w_1, w_2, \dots, w_i) en las M funciones de base.
- Para obtener los valores de la primera capa oculta se utilizará una función de transferencia (sigmoide), por ejemplo, una *logsig* o *tansig*.

Algoritmo de Regresión Ordinal ORNNet

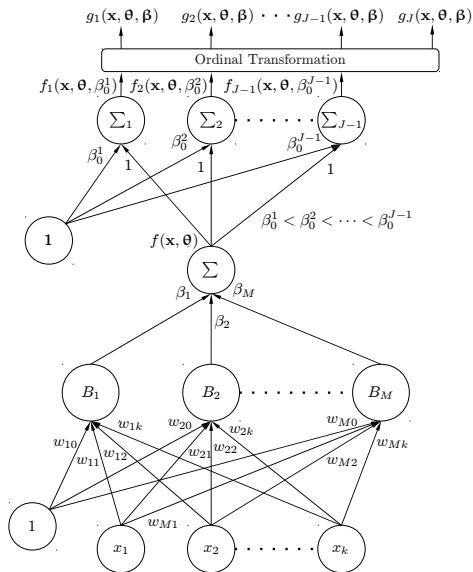


Algoritmo de Regresión Ordinal ORNNet

Capas ocultas

- La primera capa oculta obtendrá los nuevos valores de las funciones base a partir de los datos de entrada.
- La segunda capa oculta, tendrá solo una neurona asociada a una combinación lineal sin sesgo de las funciones de base.
- Esta última capa utilizará la función *purelin* como función de transferencia fija.

Algoritmo de Regresión Ordinal ORNNet

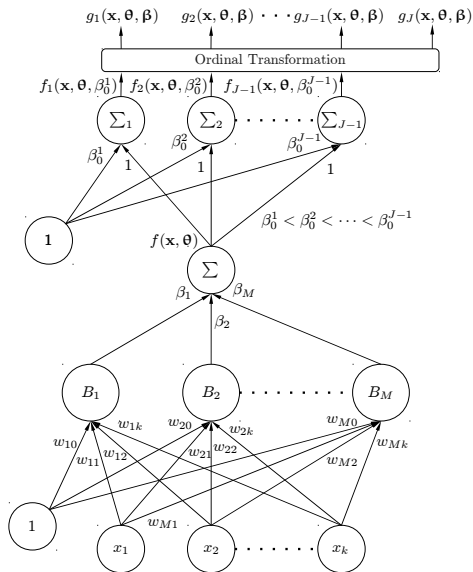


Algoritmo de Regresión Ordinal ORNNet

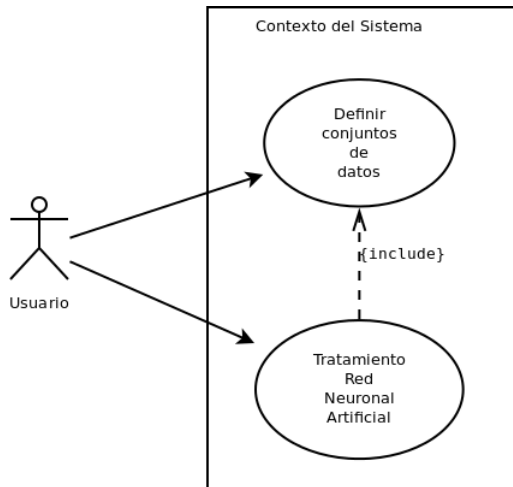
Capa de salida

- Tendrá el mismo número de neuronas que número de clases menos una.
- El valor de los pesos se encuentra fijo a 1.
- Utilizará la función de transferencia *logsig*.
- Las condición que existe en la capa de salida es que el valor de los *sesgos* o *umbrales* tiene que mantener el orden, es decir, $\beta_0^1 < \beta_0^2 < \dots < \beta_0^{J-1}$.

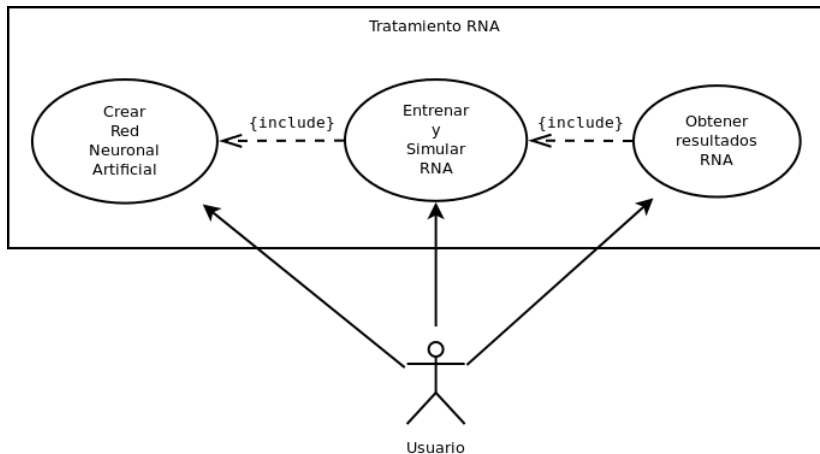
Algoritmo de Regresión Ordinal ORNNet



Casos de Uso I



Casos de Uso II



Análisis de la Interfaz

- Estructura atractiva que resulte familiar al usuario.
- Manejo fácil e intuitivo.
- Acceso a opciones de configuración.
- Posibilidad de realizar vuelta atrás para modificar los datos y poder obtener nuevos resultados.
- Los resultados obtenidos deben mostrarse de manera lo más parecida posible a como son mostrados en el toolbox *nnet* de Matlab.

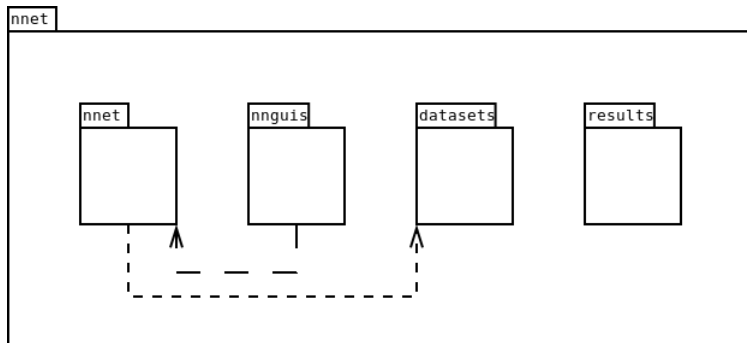
- 1 Introducción
- 2 Antecedentes
- 3 Análisis del Sistema
- 4 Diseño**
- 5 La Aplicación
- 6 Resultados
- 7 Conclusiones
- 8 Bibliografía

Tecnologías

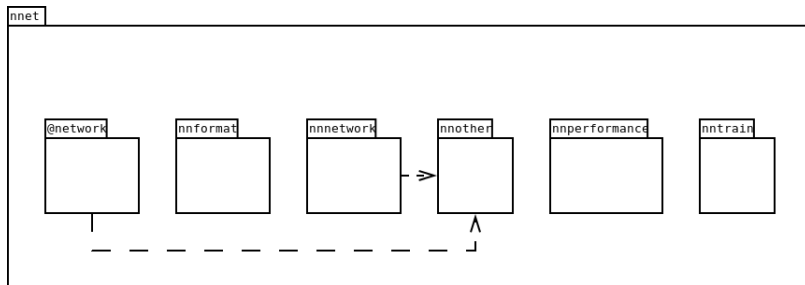
Matlab

- Incluye operaciones vectoriales y matriciales que son fundamentales para resolver los problemas científicos y de ingeniería.
- Ofrece la posibilidad de programar y desarrollar algoritmos más rápidamente que con los lenguajes tradicionales porque ya no hay que realizar tareas administrativas de bajo nivel.
- Es un software multiplataforma y eso facilita que cualquiera pueda tanto hacer desarrollos, como ejecutar programas ya hechos, en su propio sistema operativo.
- Su toolbox *nnet* (Neural Network Toolbox) proporciona herramientas para el diseño, implementación, visualización y simulación de redes neuronales artificiales.
- Proporciona la posibilidad de crear Interfaces Gráficas de Usuario (GUI).

Arquitectura del Sistema I

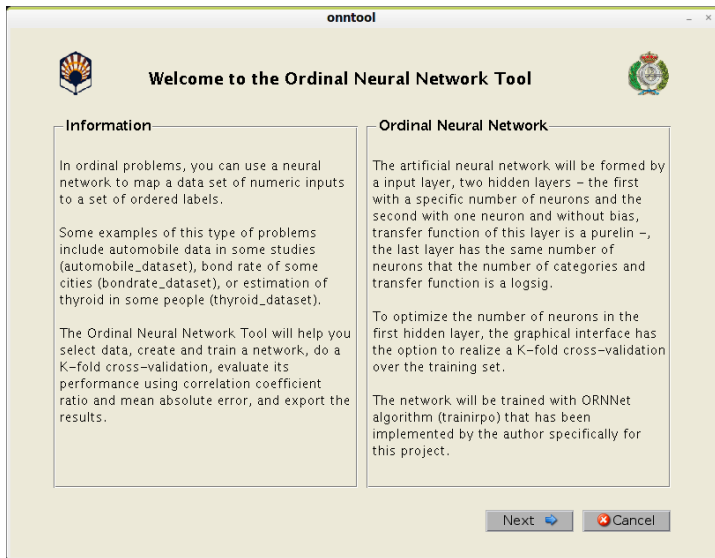


Arquitectura del Sistema II




- 1 Introducción
- 2 Antecedentes
- 3 Análisis del Sistema
- 4 Diseño
- 5 La Aplicación**
- 6 Resultados
- 7 Conclusiones
- 8 Bibliografía

Pantalla Principal







Pantalla de Carga de Datos

onntool



 **Select Data**



Train Data

Input data:  

Target data:  




Test Data

Input data:  

Target data:  


Information

Select popup menu if it has content, else push open button to select a dataset since a file.

 Back  Next  Cancel

Pantalla de Configuración de la RNA ordinal

onntool



Network Configuration

Parameters

K-fold?

k:

10

K-fold

If you prefer to manually configure the neural net, specify number of neurons and transfer function of the hidden layer:

Neurons:

20

Transfer function:

tansig

Information

Ordinal Neural Network Configuration:

Now you have to configure the architecture of the neural network. If you want to perform a K-fold cross-validation to adjust the number of hidden neurons, you must select the option "Yes" in the popup menu. A K-fold cross-validation will be performed over the training set, and the number of neurons with the lowest MAE error will be selected as the optimum one.

If you don't want to realize a K-fold, you must select the option "No". You will have to specify manually the number of neurons. Finally, the transfer function (tansig or logsig) for the hidden layer has to be chosen in both cases.


Back

Next

Cancel

Pantalla de Entrenamiento


onntool



Train Network

Train

Train an ordinal neural network with the ORNNNet algorithm:



Test results

	Dataset	CCR	MAE	NH

Diagrama


Back

Next

Cancel

Pantalla de Exportación de Datos

onntool




Save Results

Save Results

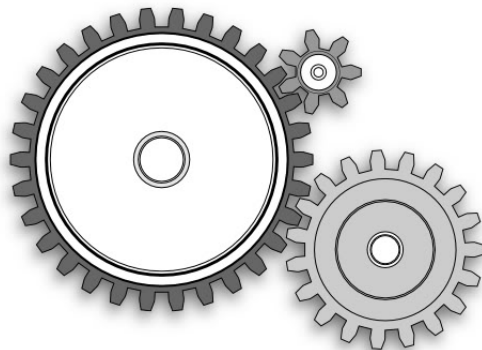
Save network to MATLAB network object named:

Save outputs to MATLAB matrix named:

Save results to MATLAB object named:

 Save Results

Ejemplo de ejecución



- 1 Introducción
- 2 Antecedentes
- 3 Análisis del Sistema
- 4 Diseño
- 5 La Aplicación
- 6 Resultados**
- 7 Conclusiones
- 8 Bibliografía

Resultados de las pruebas

Los resultados obtenidos por el algoritmo ordinal ORNNet son mejores que los del algoritmo nominal, puesto que:

- En 12 de los conjuntos de datos el *CCR* es mayor que los resultados para el algoritmo nominal.
- En 15 conjuntos de datos es menor el valor del *MAE* para el ordinal.
- En 11 conjuntos de datos la desviación típica es menor para el *CCR* utilizando la clasificación ordinal frente a la nominal.
- Y en 14 ocasiones también es menor la desviación típica para el *MAE* utilizando la clasificación ordinal.

Resultados de las pruebas

dataset	Nominal				Ordinal			
	CCRMean	CCRSD	MAEMean	MAESD	CCRMean	CCRSD	MAEMean	MAESD
ERA	0.249333	0.026623	1.360267	0.150086	0.267600	0.025415	1.240133	0.081750
ESL	0.694536	0.028794	0.329235	0.031200	0.714481	0.041744	0.302732	0.044685
LEV	0.598933	0.033564	0.431067	0.037665	0.607067	0.040936	0.419333	0.043490
SWD	0.561733	0.031839	0.467867	0.037294	0.567733	0.027270	0.451467	0.033855
automobile	0.600000	0.089548	0.580128	0.144536	0.641667	0.085480	0.443590	0.117023
balance	0.928662	0.026469	0.083864	0.034446	0.963907	0.018041	0.039066	0.019843
bondrate	0.560000	0.077509	0.606667	0.124229	0.568889	0.083475	0.584444	0.093765
car	0.977701	0.010900	0.025617	0.012634	0.968827	0.010955	0.031327	0.011039
contact-lenses	0.661111	0.202869	0.500000	0.327536	0.650000	0.153815	0.427778	0.184055
depression	0.653283	0.192474	0.509091	0.316757	0.647475	0.149711	0.430808	0.179999
eucalyptus	0.599638	0.036704	0.476993	0.055290	0.650906	0.035615	0.385145	0.032663
newthyroid	0.956790	0.032014	0.043210	0.032014	0.964198	0.024284	0.035802	0.024284
pasture	0.722222	0.119421	0.311111	0.138101	0.648148	0.127468	0.351852	0.127468
squash-stored	0.656410	0.139567	0.366667	0.154750	0.661538	0.115340	0.353846	0.125507
squash-unstored	0.669231	0.131151	0.330769	0.131151	0.653846	0.156150	0.348718	0.160001
tae	0.414912	0.085952	0.764912	0.145938	0.441228	0.072047	0.648246	0.068233
thyroid	0.944630	0.008216	0.101574	0.015156	0.929426	0.006682	0.118352	0.013353
winequality-red	0.578250	0.016856	0.459833	0.020128	0.580833	0.020754	0.449917	0.022993
winequality-white	0.543728	0.010689	0.515374	0.009745	0.540789	0.012448	0.508408	0.012904

- 1 Introducción
- 2 Antecedentes
- 3 Análisis del Sistema
- 4 Diseño
- 5 La Aplicación
- 6 Resultados
- 7 Conclusiones**
- 8 Bibliografía

Conclusiones de Autor

- Se ha desarrollado la implementación del algoritmo propuesto de forma teórica ORNNet para regresión ordinal basado en redes neuronales artificiales.
- Se ha realizado una comparativa de los resultados obtenidos por el clasificador nominal y el ordinal, siendo el ordinal más preciso y estable.
- Se ha implementado una interfaz amigable y usable para cualquier tipo de usuario.
- Se han modularizado todos los ficheros y funciones para que tenga una compatibilidad e integración completa con el toolbox *nnet* de Matlab.

Futuras Mejoras

- Realizar un estudio de los nuevos algoritmos basados en regresión ordinal que aparezcan y, a partir de estos, implementar un nuevo algoritmo que mejorase la efectividad de los resultados obtenidos.
- Realizar la adaptación al toolbox *nnet* de Matlab del resto de algoritmos o desarrollos que el grupo de investigación AYRNA tiene de proyectos o estudios anteriores basados en redes neuronales artificiales.
- Mejoras varias en la interfaz:
 - 1 Añadir la funcionalidad para la realización del método nominal además del ordinal.
 - 2 Dotar de un aspecto más moderno a los elementos de la interfaz.
 - 3 Solucionar alguna limitación actual, como mostrar los resultados de cada entrenamiento añadiéndolos en la tabla de resultados.
 - 4 Soporte para más formatos de archivos de entrada de los conjuntos de datos.

- 1 Introducción
- 2 Antecedentes
- 3 Análisis del Sistema
- 4 Diseño
- 5 La Aplicación
- 6 Resultados
- 7 Conclusiones
- 8 Bibliografía**

Bibliografía I



W. Chu and S.S. Keerthi.

"New Approaches to Support Vector Ordinal Regression".
Proc. 22nd Int. Conf. Machine Learning (ICML '05), pp.
145-152. 2005.



L. Lin and H.-T. Lin.

"Ordinal Regression by Extended Binary Classification".
Advances in Neural Inform. Processing Systems, vol. 19, pp.
865-872, MIT Press. 2007.



McCullagh.

"P. Regression models for ordinal data".
Journal of the Royal Statistical Society, Series B
(Methodological), 42, 109–142. 1980.

Bibliografía II



Shashua and A. Levin.

"Ranking with Large Margin Principle: Two Approaches".
Advances in Neural Inform. Processing Systems, vol. 15,
961-968, MIT Press. 2003.



Riedmiller, Martin.

"RPROP - Descripción and Implmentation Details".
University of Karlsruhe. Technical Report. 1994.



Igel, Christian; Hüsken, Michael.

"Empirical evaluation of the improved RPROP learning
algorithms".
Institut für Neuroinformatik, Ruhr-Universität Bochum.
Germany. 2001.