Universidad de Córdoba
Escuela Politécnica Superior de Córdoba
Departamento de informática y análisis numérico

Anteproyecto

Grado en Ingeniería Informática
Mención en Computación

Algoritmos de entrenamiento de redes neuronales artificiales para la biblioteca Orca-Python

Autor: Adrián López Ortiz

Directores:
Pedro Antonio Gutiérrez Peña
David Guijo Rubio

Firma del autor y los directores del Trabajo Fin de Grado:

- **Autores:**

Fdo: Adrián López Ortiz

- **Directores:**

Firmado por GUTIERREZ PEÑA PEDRO ANTONIO - 45735457A el día 20/11/2020 con un certificado emitido por AC FNMT Usuarios

Fdo: Pedro Antonio Gutiérrez Peña

Firmado por GUIJO RUBIO DAVID - 45886352H el día 21/11/2020 con un certificado emitido por AC FNMT Usuarios

Fdo: David Guijo Rubio

Índice

1. Datos del proyecto

Título: Algoritmos de entrenamiento de redes neuronales artificiales para la biblioteca Orca-Python.

Autor:

- **Nombre:** Adrián López Ortiz
- **Email:** p42loora@uco.es
- **Titulación:** Grado en Ingeniería Informática
- **Mención:** Computación

Directores del proyecto:

- Pedro Antonio Gutiérrez Peña
- David Guijo Rubio

2. Introducción

Actualmente, los sistemas informáticos, procesan y recopilan una cantidad desmesurada de datos, que a priori son superfluos, pero que si son tratados y estudiados desde cierto punto de vista, pueden generar nueva información de interés. Esta extracción de información, gracias al tratamiento de datos, se denomina **minería de datos**. Más concretamente, el proyecto se basa en el campo del **aprendizaje automático**.

Aprendizaje supervisado (o *supervised learning*, en inglés), es un campo de las ciencias de la computación, que genera modelos matemáticos capaces de predecir información. El aprendizaje automático puede dividirse en dos sub-categorías.

- El **aprendizaje supervisado**,(*supervised learning*) en el cuál los patrones del conjunto de datos a estudiar cuentan con una etiqueta que proporciona un experto de la materia a la que pertenezcan los datos a tratar.
- **Aprendizaje no supervisado** (o *unsupervised learning*, en inglés), en el cuál los patrones del conjunto de datos no están supervisados, es decir, no están etiquetados.

El proyecto se adhiere a la primera de estas dos categorías. Concretamente, el proyecto tratará problemas de clasificación o regresión ordinal, siendo ambas denominaciones válidas para referirse a problemas en los que hay más de dos clases (**problemas multiclase**), y además estas muestran un determinado orden natural y el principal objetivo es aprender a clasificar nuevos datos en una de dichas categorías.

Esto implica que si estamos clasificando una serie de patrones en categorías ordinales, la magnitud de los errores depende de la distancia (en la escala ordinal), entre la categoría predicha y la real. Por ejemplo, si intentamos predecir la categoría de un determinado equipo de fútbol, no sería igual de grave confundir un equipo de 1ª división con uno de 2ªA que confundirlo con uno de 2ªB, ya que los categorías guardan un orden natural, 1ª>2ªA>2ªB.

Una **Red Neuronal Artificial** (RNA), es un modelo computacional basado en un conjunto de unidades simples (emulaciones matemáticas del modelo de respuesta de una neurona biológica), interconectadas de forma análoga a las neuronas del cerebro humano.

Una RNA es un modelo no lineal, el cual puede acumular conocimiento en sus coeficientes y estructura a través de un proceso de aprendizaje, **la retropropagación del error** (*Back propagation*, en inglés). Después del proceso de aprendizaje, una RNA es capaz de aproximar una función continua.

Dado el reciente interés por la clasificación ordinal [1], se cree necesario el desarrollo de herramientas software que faciliten la aplicación de métodos de clasificación ordinal a problemas de otras áreas de conocimiento, generalmente menos familiarizadas con este campo.

En este sentido, una de las herramientas existentes es el *framework* **ORCA** (*Ordinal Regression and Classification Algorithms*) [2], desarrollado por el grupo de investigación AYRNA [3]. Este *framework* implementa un gran número de algoritmos para la resolución de problemas de clasificación ordinal, además de diferentes métricas de rendimiento, paralelismo y ejecuciones de experimentos automáticas. El *framework* ORCA está implementado en *MATLAB/Octave*.

Se utilizará ORCA como base para el desarrollo del proyecto, proporcionando la misma funcionalidad pero en un lenguaje y entorno mucho más flexible y actual, como es *Python* [4] frente al uso de *MATLAB* u *Octave*, que son lenguajes menos extendidos y con menor proyección. El uso de un lenguaje como *Python*, permitiría mayor facilidad de difusión y uso del *framework* dentro de la comunidad científica.

Este proyecto se une a una serie de desarrollos en una biblioteca que incluye una funcionalidad similar a ORCA pero en lenguaje de programación *Python* (Orca-Python)[5], para la cual aún no se han incorporado RNAs.

3. Objetivos

Desarrollo de algoritmos de entrenamiento de redes neuronales artificiales para la biblioteca software en *Python* [4], (**Orca-Python**), basándonos en el *framework* ORCA [2] del grupo de investigación AYRNA [3], que se encuentra programado en *MATLAB/Octave* [6].

De esta forma, este proyecto extenderá la biblioteca Orca-Python [5], incluyendo métodos nuevos para el entrenamiento de redes neuronales en clasificación ordinal.

En concreto, se acometerá el desarrollo de los siguientes métodos de entrenamiento de redes neuronales (ORCA incluye implementaciones de ambos métodos):

- *Neural Network based on Proportional Odd Model*, NNPOM [6,9]: Red neuronal basada en modelo de posibilidades proporcionales (en inglés, *Proportional Odds Model*, POM), que implementa una red neuronal para la regresión ordinal.El modelo está compuesto por una capa oculta y una capa de salida, ésta última solo contiene una neurona de salida con tantos umbrales como número de clases menos uno. En esta neurona de la capa de salida, se aplica el modelo estándar POM, con la finalidad de proporcionar salidas probabilísticas. El modelo estándar de POM se aplica en esta neurona para proporcionar salidas probabilísticas.

- *Neural Network with Ordered Partitions*, NNOP [10]: Red neuronal con particiones ordenadas (NNOP), que implementa otro modelo alternativo de red neuronal para regresión ordinal. Este modelo considera el esquema de codificación de *OrderedPartitions* [1] para las etiquetas y una regla para las decisiones basadas en el primer nodo cuya salida es superior a un umbral predefinido (T = 0.5). El modelo tiene una capa oculta y una capa de salida con tantas neuronas como el número de clases menos una.

4. Antecedentes

4.1. *Python* para las ciencias de la computación

Python [4] es un lenguaje de programación interpretado y de propósito general. Ha ido ganando mucha popularidad en la comunidad científica gracias a su facilidad de aprendizaje, legibilidad y capacidad de ampliación mediante diversas bibliotecas. Además, cuenta una licencia de código abierto denominada *Python Software Foundation License* [7].

Entre la gran variedad de bibliotecas con las que es posible ampliar el lenguaje, se encuentran muchas dedicadas a facilitar tareas en el ámbito de las ciencias de la computación, incluido el aprendizaje automático como las que se indican a continuación y que se han visto durante el grado.

4.1.1. *NumPy*

NumPy [8] es una biblioteca que proporciona objetos para almacenar matrices n-dimensionales y diversos métodos para trabajar con las mismas, de una forma sencilla y eficiente. Cuenta con una licencia de código abierto BSD [9].

4.1.2. *Pandas*

Pandas [10] es una biblioteca que proporciona las estructuras y métodos necesarios para la manipulación y el análisis de datos. Cuenta con una licencia de código abierto BSD.

4.1.3. *Matplotlib*

Matplotlib [11] es una bibliotecas de representación de datos que permite generar gráficos de alta calidad y que posee integración con la librería *NumPy*, que le proporciona los datos a representar. Cuenta con una licencia de código abierto *Python Software Foundation License*.

4.1.4. *Scikit-Learn*

Scikit-Learn [12] es una biblioteca software programada sobre *Numpy* [8], *Matplotlib* [11] y *SciPy* [13]. Proporciona distintos objetos y métodos para la realización de problemas de clasificación, regresión, *clustering* y preprocesamiento. Su uso es sencillo y cuenta con una buena documentación y una licencia de código abierto BSD [9].

4.2. ORCA

ORCA [2], como ya se ha comentado, es un *framework* desarrollado por el grupo de investigación AYRNA [3], que pertenece al Departamento de Informática y Análisis Numérico de la Universidad de Córdoba.

Esta herramienta implementa una gran cantidad de algoritmos de clasificación ordinal y de métricas de rendimiento. Está desarrollada en el lenguaje de programación *MATLAB*,[6]. Entre los algoritmos implementados se encuentran NNPOM y NNOP, que son algoritmos estocásticos, es decir, producen resultados diferentes para dos ejecuciones diferentes y que explicamos a continuación.

4.2.1. *Neural Network based on Proportional Odd Model*, NNPOM

Neural Network based on Proportional Odd Model, cuyas siglas, y por las que empezaremos a denominar al método en adelante son **NNPOM**. Implementa un modelo de red neuronal para la regresión ordinal. El modelo tiene una capa oculta y una capa de salida con solo una neurona pero tantos umbrales, como el número de clases menos uno. El modelo estándar de POM se aplica en esta neurona para proporcionar salidas probabilísticas.

4.2.2. *Neural Network with Ordered Partitions*, NNOP

Neural Network with Ordered Partitions, cuyas siglas y por las que empezaremos a denominar al método **NNOP**. Este modelo considera el esquema de codificación de Particiones Ordenadas (u *Ordered Partitions*, en inglés) para las etiquetas y una regla para las decisiones basadas en el primer nodo cuya salida es superior a un umbral predefinido (T = 0.5). El modelo tiene

una capa oculta y una capa de salida con tantas neuronas como el número de clases menos una. Pertenece al grupo de algoritmos denominado modelos de umbral [1].

4.3. Orca-Python

Orca-Python [5] es una adaptación y ampliación del *framework* ORCA. Fue desarrollado en el Trabajo Fin de Grado de Iván Bonaque Muñoz.

Esta herramienta implementa algunos de los algoritmos de clasificación ordinal y métricas de rendimiento que posee ORCA. Está desarrollada en el lenguaje Python [4] y permite utilizar los algoritmos implementados en la biblioteca Scikit-Learn [12].

5. Fases de desarrollo

5.1. Estudio y análisis del problema

Se estudiará el problema desde el punto de vista teórico, comprendiendo el funcionamiento de los métodos a adaptar, para más adelante poder comprobar que los resultados obtenidos sean correctos.

También se necesita un aprendizaje profundo del lenguaje de programación Python [4], el cual se usará para la implementación de los métodos, así cómo de las bibliotecas mencionadas en la sección 4.1 y la comprensión del sistema de módulos del lenguaje Python [4] para poder enlazar los algoritmos adaptados a la biblioteca Orca-Python [5] de forma óptima. De forma eventual, se necesitará un nivel básico de comprensión del lenguaje de la biblioteca predecesora ORCA, es decir, el lenguaje propio del entorno *MATLAB* [6].

5.2. Diseño

Cuando los requisitos del proyecto queden correctamente analizados y estudiados, se procederá a su diseño y posterior implementación. El diseño garantizará cumplir los objetivos y requisitos del proyecto de la forma mas eficiente posible.

Se analizará el diseño de las estructuras de datos que se utilizarán para realizar la adaptación de *MATLAB* [6], a Python [4], asegurando su consistencia y eficiencia.

5.3. Codificación

Se utilizará el lenguaje de programación Python, junto a las diferentes bibliotecas mencionadas en la sección 4.1.

5.4. Pruebas

El producto final se someterá a un conjunto de pruebas para comprobar su correcto funcionamiento y verosimilitud del resultado obtenido por los algoritmos adaptados.

5.5. Documentación

Es necesario que el producto final del proyecto sea entendible y pueda ser utilizado por terceras partes no involucradas en el desarrollo del proyecto, dedicando el mínimo esfuerzo posible en su comprensión. Para lograr esto, se redactará un manual técnico en el cual se explicará los distintos elementos que componen el proyecto, y un manual de usuario donde se recogerá la funcionalidad y el uso correcto del producto.

Además, el código fuente se encontrará comentado para mejorar su comprensión.

6. Recursos

A continuación se describen los recursos humanos y materiales implicados en el desarrollo de este trabajo fin de grado.

6.1. Recursos humanos

Autor: Adrián López Ortiz.
Alumno de 4º curso del Grado en Ingeniería Informática

Director: Pedro Antonio Gutiérrez Peña.
Profesor Titular de la Universidad de Córdoba del Dpto. de Informática y Análisis Numérico y miembro investigador del grupo AYRNA.

Director: David Guijo Rubio
Contratado predoctoral por el programa Formación al Profesorado Universitario (FPU) del Ministerio de Educación, Cultura y Deporte de España y miembro investigador del grupo AYRNA.

6.2. Recursos materiales

A continuación se numeran los recursos software y hardware que se utilizarán en el desarrollo de este trabajo fin de grado.

6.2.1. Recursos software

- Linux versión proporcionada por la UCO.
- Python como lenguaje de programación
- LaTeX, para la realización de la documentación.

6.2.2. Recursos hardware

Ordenador proporcionado por la UCO, usado por el proyecista, consistente en un ordenador de sobremesa con las siguientes características:

- Procesador Intel (R) Core (TM) i3-2100 2-core a 3.10 GHZ.
- Memoria RAM de 8GB DDR3 a 1300 MHz.
- Tarjeta Gráfica Intel HD Graphics 2000.
- Disco duro HDD 500 GB 7200 rpm.

7. Planificación temporal

La planificación a seguir a lo largo de las fases del proyecto (Ver sección 5) se especifican en la siguiente tabla:

Fase de desarrollo	**Año 2021**				**Horas estimadas**
	Mes 1	**Mes 2**	**Mes 3**	**Mes 4**	
Estudio y análisis del problema	30	10	-	-	40
Diseño	20	30	10	-	60
Implementación	-	20	35	30	85
Pruebas	-	10	15	20	45
Documentación	10	10	20	30	70
Total	60	80	80	80	300

Tabla 7.1: Especificación del reparto de la carga de trabajo del proyecto.

Referencias

[1] Pedro Antonio Gutiérrez, María Pérez Ortiz, Javier Sánchez Monedero, Francisco Fernández Navarro y Cesar Hervás Martínez. «Ordinal Regression Methods: Survey and Experimental Study». En: *IEEE Transactions on Knowledge and Data Engineering* 28.1 (jul. de 2015), págs. 127-146. DOI: 10.1109/TKDE.2015.2457911.

[2] Javier Sánchez Monedero, Pedro Antonio Gutiérrez y María Pérez Ortiz. «ORCA: A Matlab/Octave Toolbox for Ordinal Regression». En: *Journal of Machine Learning Research, 20* (2019), págs. 1-4.

[3] *Departamento de Informática y Análisis Numérico de la Universidad de Córdoba. Aprendizaje y redes neuronales artificiales*. URL: https://www.uco.es/grupos/ayrna/index.php/es. [Online. Última consulta: 02-12-2019].

[4] G. van Rossum. *Python tutorial. Technical Report CS-R9526, Centrum voor Wiskunde en Informatica (CWI)*. Mayo de 1995.

[5] Iván Bonaque Muñoz, Pedro Antonio Gutiérrez Peña y Javier Sánchez Monedero. *Trabajo de Fin de Grado. Framework en Python para problemas de clasificación ordinal*. 2019.

[6] MATLAB. *version 9.0 (R2016a)*. The MathWorks Inc., Natick, Massachusetts, 2016.

[7] *History and License*. URL: https://docs.python.org/3/license.html. [Online. Última consulta: 02-12-2019].

[8] *Sitio oficial de NumPy*. URL: https://numpy.org/. [Online. Última consulta: 02-12-2019].

[9] *Texto de licencia BSD original*. URL: http://www.xfree86.org/3.3.6/COPYRIGHT2.html#6. [Online. Última consulta: 02-12-2019].

[10] *Sitio oficial de Pandas*. URL: https://pandas.pydata.org/. [Online. Última consulta: 02-12-2019].

[11] *Sitio oficial de Matplotlib*. URL: https://matplotlib.org/. [Online. Última consulta: 02-12-2019].

[12] *Sitio oficial de Scikit-Learn*. URL: https://scikit-learn.org/stable/. [Online. Última consulta: 02-12-2019].

[13] *Sitio oficial de SciPy*. URL: https://www.scipy.org/. [Online. Última consulta: 02-12-2019].