Python para ciencia e ingeniería: de los métodos básicos a las aplicaciones de Machine Learning

Jorge Delgado, Carlos Medrano

jorgedel@unizar.es ctmedra@unizar.es

Departamento de Matemática Aplicada
Universidad Zaragoza

Departamento de Ingeniería Electrónica y Comunicaciones
Universidad Zaragoza

3 de Octubre de 2025 – Zaragoza

Outline

Presentación

Jorge Delgado Gracia

- Profesor Titular del Departamento de Matemática Aplicada
- Universidad de Zaragoza

Carlos Tomás Medrano Sánchez

- Profesor Titular del Departamento de Ingeniera Electrónica y Comunicaciones
- Universidad de Zaragoza

Experiencia personal (CM)

- Previo 1998: Fortran, Pascal, Mathematica; Mac y estaciones Unix
- 1998-2005 (aprox): C, Matlab; Windows? Linux?
- 2005-2009 (aprox): Búsqueda de alternativas: Octave, Scilab, Python?
- Finalmente: Python, C o C++ si es necesario; R para algunos temas estadísticos; Linux y Mac (menor medida)

Experiencia personal (JD)

- Previo 2000: Fortran, C; Mac y Windows
- 2000-2009 (aprox): C y Mathematica; Linux y Windows
- 2010-2017 (aprox): Matlab y Mathematica; Linux y Windows
- 2018-ahora: Matlab, Python, Mathematica; Mac

¿Dónde se usa Python?

- Ingeniería y Ciencia aplicada
  - Simulación numérica (matemáticas, física, química, biología computacional): Numpy, Scipy, statsmodels, Biopython
  - Optimización e investigación operativa (logística, planificación): Pyomo, OR-Tools
  - Procesamiento de señales e imágenes: OpenCV, scikit-image.
- Ciencia de datos y estadística
  - Análisis de datos: Numpy, Pandas
  - Visualización: Matplotlib, Seaborn, Plotly
  - Machine Learning e IA: Scikit-learn, TensorFlow, PyTorch
- Educación
- Electrónica, Robótica e IoT
  - Control de robots: ROS
  - Programación de microcontroladores (Raspberry Pi, MicroPython)
  - Kicad

Python: interés creciente

- JD, CM: Cursos (Escuela de doctorado, grupo G9 universidad): hay estudiantes de Geología, Medicina, Veterinaria, Psicología, Economía, Contabilidad, Arquitectura !!!
- JD, CM: Curso en dpto. Economía aplicada
- JD, CM: Curso para banca
- Curso para personal técnico y de administración
- Experto universitario en Astrofísica (EUPT) (incluye Python)
- Numerosos TFG, TFM basados en Python
- ...

Numpy

Numpy: web

- Numpy: creado en 2005. Última versión 2.3.
- Cuenta con soporte de instituciones/empresas
- Numpy: librería fundamental en la que se apoyan el resto: arrays y operaciones en arrays.
- Código de base en C (rápido).
- Transformada discreta de Fourier, Polinomios, Álgebra lineal, ...

Scipy

Scipy:web

- Scipy: desde 2001, ultima versión 1.16.1.
- Desarrollo inicial de Travis Oliphant, Pearu Peterson, Eric Jones
- Cuenta con soporte de instituciones/empresas
- Optimización, integración y resolución de ecuaciones diferenciales, procesado de señal, estadística, ...
- Código de base en C, C++, Fortran

Pandas

Página web oficial: `https://pandas.pydata.org/`

- Comienza su desarrollo en 2008 (Wes McKinney en AQR Capital)
  - Última versión: 2.3.2
- En 2009 se convierte en software de código abierto
- En 2012 se publica la primera edición de Python for Data Analysis
- Licencia BSD
- Pandas = Python + Cython + C, sobre NumPy.

Pandas incluye las siguientes funciones:

- Manipulación y análisis de datos
- Objetos Series y DataFrame
- Exportación e importación de datos desde ficheros y web
- Manejo de datos perdidos

Pandas

Objetos:

- `Series`
  - Un vector pero con clave (`index`)
- `DataFrame`
  - Una matriz pero con claves (`index` y `columns`)

Matplotlib

Página web oficial: https://matplotlib.org/

Matplotlib nos permite realizar gráficas y figuras de alta calidad

- Se distribuyó por primera vez en 2003
  - Última versión 3.10
- Escrita originalmente por John D. Hunter
  - Responsables Michael Droettboom
- Distribuido bajo una licencia de estilo BSD

Incluye las siguiente funciones:

- Dibujar imágenes, gráficos de contorno, gráficos de dispersión, gráficos lineales,...
- Exportar a una gran variedad de formatos de ficheros.
- Funciona en scripts de Python, en el shell de IPython y en cuadernos (Notebooks) Jupyter como este

Matplotlib

- Matplotlib tiene una gran cantidad de funciones y opciones (difíciles de recordar)
- Para casi cualquier figura que tengas que hacer hay un ejemplo del que puedes copiar el código
  - La galería de ejemplos es una gran fuente de ejemplos `https://matplotlib.org/stable/gallery/index.html`
- Para tareas cotidianas pueden ser preferiblen librerías basadas en Matplotlib
  - Pandas
  - seaborn

Scikit-learn

`https://scikit-learn.org/stable/`Scikit-learn: web

- Iniciado en 2007. En 2010 personas del INRIA (Francia) toman el liderazgo (Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort and Vincent Michel)
- Primera versión: 02/2010. Ahora: 1.7.2.
- Cuenta con soporte financiero de instituciones/empresas
- Librería de Machine Learning "clásico"
- Clasificación, regresión, clustering
- Preprocesado, reducción de la dimensionalidad, selección de modelos
- Muy bien documentada
- Uniforme (fácil de intercambiar un modelo con otro)
- Puedes aprender Machine Learning con ella

Conceptos ML - Scikit-learn

- Clasificación vs regresión
- Aprendizaje supervisado vs no supervisado
- Preprocesado: escalado, selección de características, reducción de la dimensionalidad (ejemplo: estandarizado)
- Entrenamiento y test: división entre conjuntos
- Validación cruzada: cross-validation
- Evaluación (métricas): ejemplo accuracy
- Objetos más complejos: cross-validation+search, pipelines

Clasificador: SVM - Scikit-learn

- Máquinas de Vectores Soporte (SVM)
- Problemas no lineales.
- Se basa en el concepto de margen-problema dual
- Truco del kernel: varios tipos de kernel
- Kernel típico: $exp(-\gamma|x_i - x_j|^2)$
- Complejidad conceptual alta ... pero fácil de usar en scikit-learn

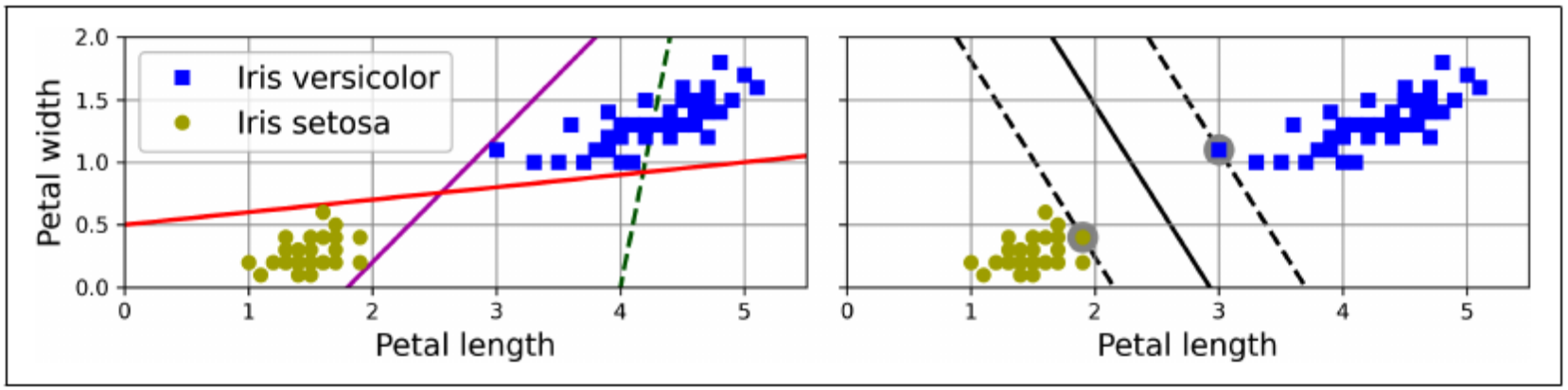

Figure 5-1. Large margin classification

Regresión

Minimizar la función de pérdida dada por la raíz del error cuadrático medio

$$RMSE(\mathbf{X}, \mathbf{y}, h_\theta) = \sqrt{\frac{1}{m}\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2}$$

o equivalentemente

$$MSE(\mathbf{X}, \mathbf{y}, h_\theta) = \frac{1}{m}\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$$

Regresión lineal y polinómica

- Regresión lineal

$$RMSE(\mathbf{X}, \mathbf{y}, \theta) = \sqrt{\frac{1}{m}\sum_{i=1}^{m}\left(\theta \cdot x^{(i)} - y^{(i)}\right)^2}$$

o

$$MSE(\mathbf{X}, \mathbf{y}, \theta) = \frac{1}{m}\sum_{i=1}^{m}\left(\theta \cdot x^{(i)} - y^{(i)}\right)^2$$

- Regresión polinómica:
  - Grado n, una sola característica $\hat{y} = \theta_0 + \theta_1 \underbrace{x}_{=x_1} + \cdots + \theta_n \underbrace{x^n}_{=x_n}$
  - Grado 2, dos características:

$$\hat{y} = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 \underbrace{x_1^2}_{=x_3} + \theta_4 \underbrace{x_1 x_2}_{=x_4} + \theta_5 \underbrace{x_2^2}_{=x_5}$$

Regresión Ridge

La regularización de Tikhonov es una versión regularizada de la regresión lineal. Su función de coste es:

$$J(\theta) = MSE(\theta) + \frac{\alpha}{m}\sum_{i=1}^{n}\theta_i^2 = \frac{1}{m}\sum_{i=1}^{m}\left(\theta^T x^{(i)} - y^{(i)}\right)^2 + \frac{\alpha}{m}\sum_{i=1}^{n}\theta_i^2$$
$$= \frac{1}{m}\|y - X\theta\|_2^2 + \frac{\alpha}{m}\|\theta\|_2^2$$

- Se ajustan los datos
- Se mantienen los pesos del modelo tan pequeños como sea posible
- El término de regularización sólo durante la fase de entrenamiento
- α hiperparámetro que controla cuánto se regulariza el modelo
- θ_0 no se regulariza.
- Escalar los datos (`StandardScaler`) antes de la regresión

Regresión Lasso

Lasso: "Least Absolute Shrinkage and Selection Operator". El término de regularización añadido a la función de coste usa la norma l_1:

$$J(\theta) = MSE(\theta) + 2\alpha \sum_{i=1}^{n} |\theta_i| = MSE(\theta) + \|\theta_i\|_1.$$

La regresión Lasso tiende a eliminar los pesos de las características menos importantes (se fijan a cero):

- Hace automáticamente una selección de características y nos da un modelo disperso con unos pocos pesos) no nulos

Regresión SVM

Máquinas de vector soporte:

- Clasificación
  - Lineal
  - No lineal
    - Añadir características polinómicas
    - Usar Kernel: polinomio, RBF Gaussiano
- Regresión

Truco para usar SVM en tareas de regresión

En lugar de ajustar la calle más ancha posible entre dos clases mientras se limitan las violaciones de ese margen, SVM para regresión intenta ajustar tantas instancias como sea posible en la calle mientras se limitan las violaciones de ese margen

La anchura de la calle se controla mediante el hiperparámetro ϵ

Clases de Scikit-Learn: `LinearSVR` y `SVR` usando kernel

Árboles de decisión para regresión

- Un árbol de regresión es muy similar a uno de clasificación. Principal diferencia: en cada hoja en vez de predecir una clase se predice un valor
- Classification and Regression Tree (CART) algorithm
- Función de coste:

$$J(k, t_k) = \frac{m_{izq}}{m} MSE_{izq} + \frac{m_{dch}}{m} MSE_{dch}$$

donde

$$MSE_{hoja} = \frac{\sum_{i \in hoja} (\hat{y}_{hoja} - y^{(i)})^2}{m_{hoja}}, \quad \hat{y}_{hoja} = \frac{\sum_{i \in hoja} y^{(i)}}{m_{hoja}}$$

- Tendencia a sobreajustar
- Clase `DecisionTreeRegressor`

Boosting

Boosting se refiere a cualquier método de conjunto que combina varios predictores débiles en un predictor fuerte

- Se entrenan los predictores secuencialmente, cada uno intentando corregir su predecesor

Métodos Boosting más populares:

- AdaBoost
- Gradient Boosting
- Extreme gradient boosting
  `https://xgboost.readthedocs.io/en/stable/`

Deep-Learning: Keras

Keras:web

- Keras presenta una API para Deep Learning
- Varios backends: TensorFlow, Jax, PyTorch
- Pretende hacer más fácil el desarrollo

Redes Neuronales: Una capa oculta

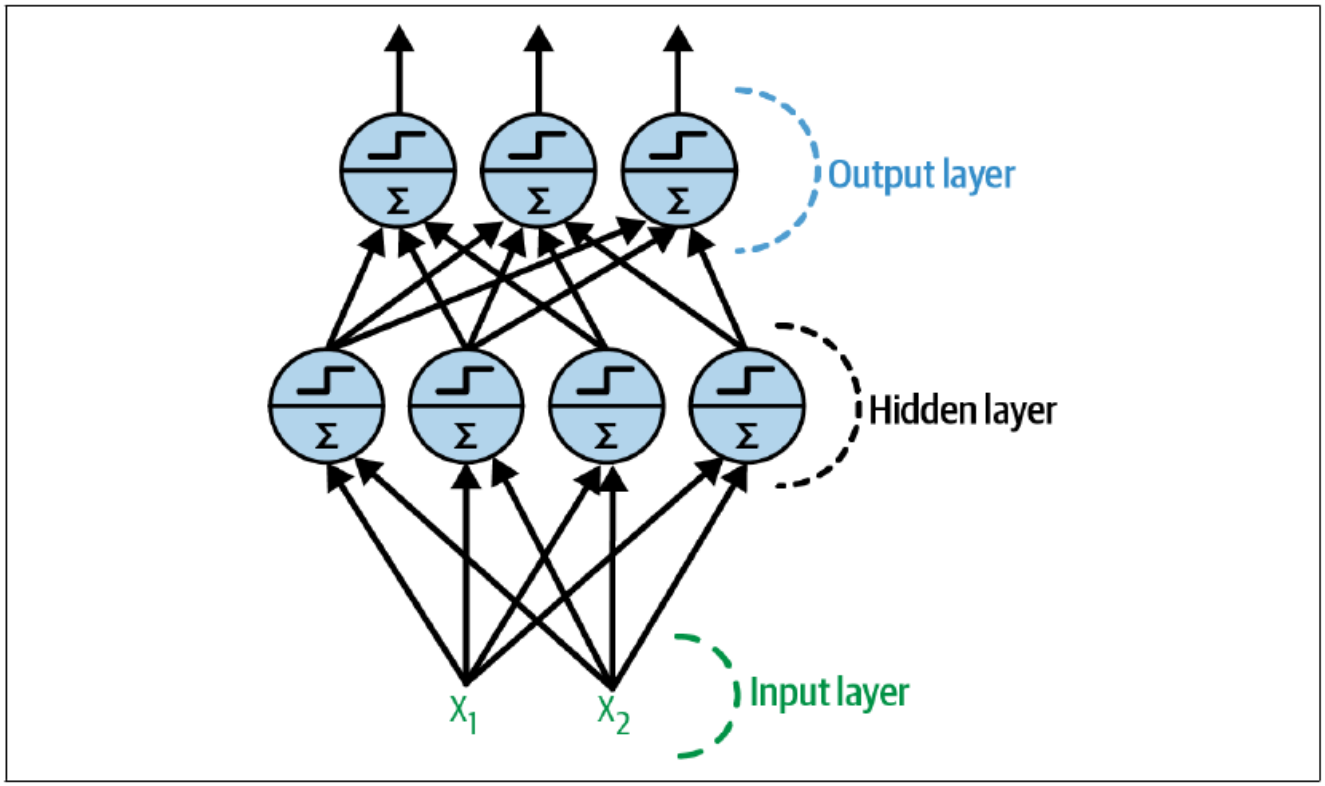

Figure 10-7. Architecture of a multilayer perceptron with two inputs, one hidden layer of four neurons, and three output neurons

Deep: CNN convolutional NN

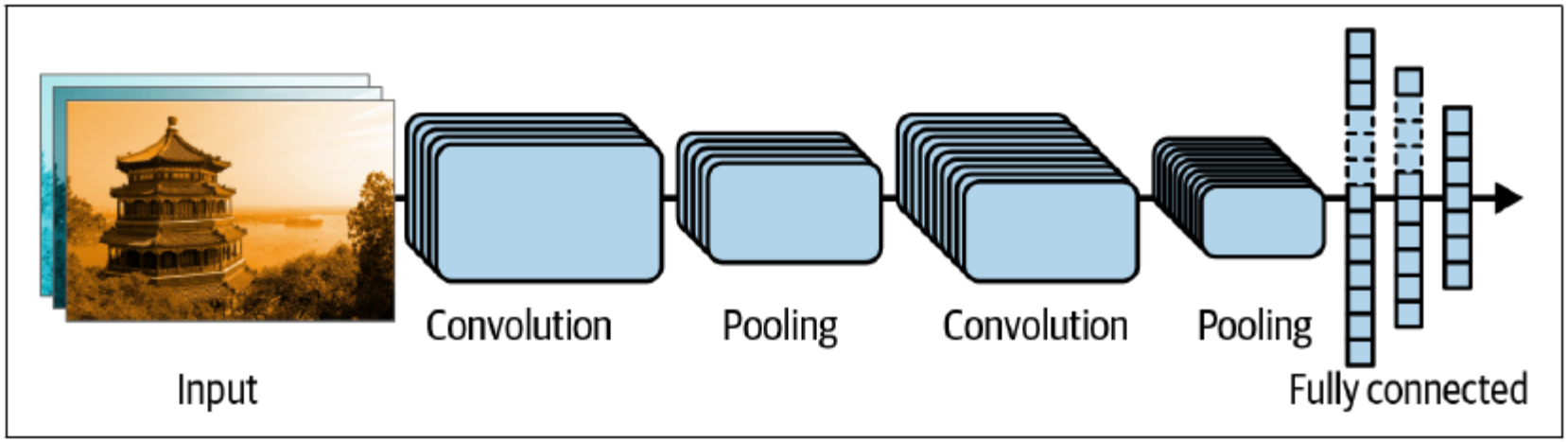

Figure 14-12. Typical CNN architecture

NLP

- NLP: Procesamiento de lenguage natural: IA para procesar lenguaje
- LLM (large Language Models): tienen gran cantidad de parámetros (hasta cientos de miles de millones de parámetros)
- Aparecen conceptos como embedding, encoding (representar una frase por ejemplo), decoding (generar texto), attention mechanism, transformer

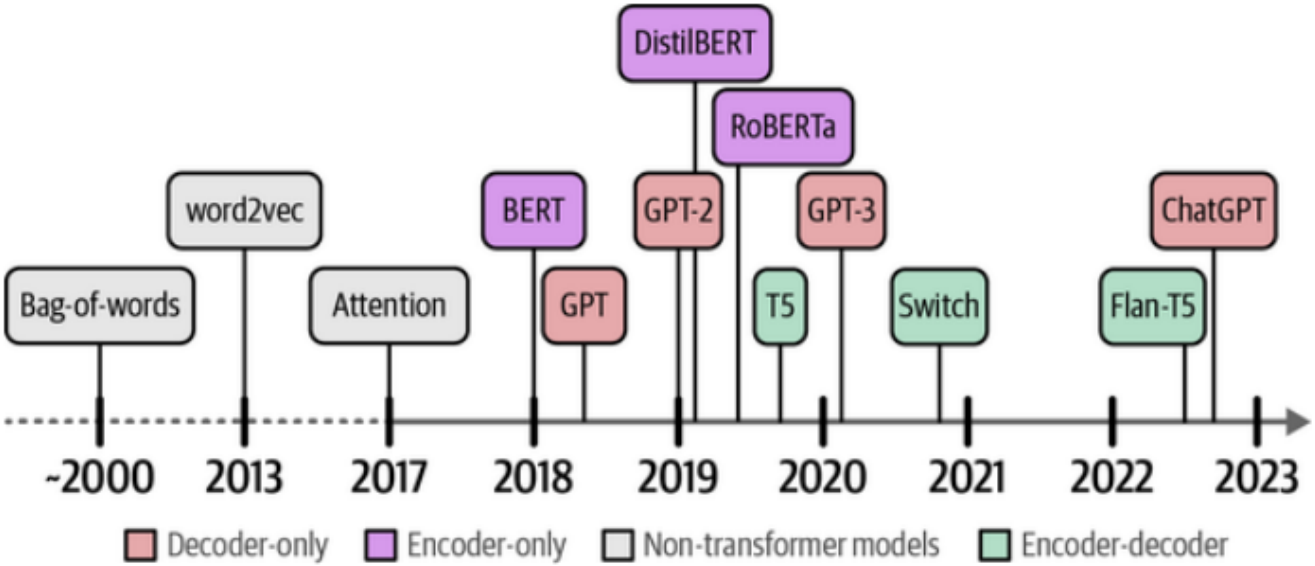

Figure 1-1. A peek into the history of Language AI.

API de Gemini (Google)

- Automatización de las "conversación"
- Salida estructurada: structured output with json
- Multimodal: trabajo con varios tipos de documentos: image understanding
- Por supuesto: limitado en procesado, más modelos y procesado/tiempo de pago.

Modelos Open Source

- Ejecución en local (datos sensibles)
- Adaptación a un problema
- OJO incluso para ejecutar el modelo se necesita hardware adecuado (GPU)
- Hugging Face: plataforma de colaboración en IA.