Técnicas Selectas de Machine Learning y Aplicaciones

UNIVERSIDAD
COMPLUTENSE
MADRID

Juan Manuel Hurtado Restrepo
Facultad de Ciencias Matemáticas
Universidad Complutense de Madrid

Trabajo de Fin de Grado presentado para optar al
Doble Grado en Matemáticas y Física

Julio de 2025

Resumen

El objetivo central de este trabajo es explorar los fundamentos de algunos temas tradicionalmente inscritos en el campo de *machine learning* o aprendizaje automático; concretamente, nos hemos propuesto abordar de forma general los problemas de clasificación y regresión, para luego ahondar en selectas técnicas que ofrecen respuestas a las célebres preguntas que éstos plantean.

Muchas de las propustas de solución escogidas han captado la atención de buena parte de la comunidad científica en los últimos años, gracias al excelente desempeño que han exihibido y al futuro prometedor que se les augura. Los algoritmos referenciados comprenden, pero no se limitan a: *regresión lineal*, *regularizaciones de la regresión lineal*, *regresión logística*, *regresión de procesos Gaussianos*, *clasificador naive Bayes*, *linear discriminant analysis*, *support vector machine*, *decision trees* y *clasificador k-nearest neighbors*.

Si bien los métodos antes mencionados pertenecen a la familia de *aprendizaje supervizado*, incluimos también la técnica *k-means clustering* de aprendizaje no supervizado que destaca por los méritos demostrados en numerosos casos de aplicación.

Para finalizar la empresa que nos hemos propuesto llevar a cabo, pondremos a prueba tres algoritmos de regresión y tres de clasificación, usando dos bases de datos distintas y cuidadosamente identificadas, una para cada problema. Concluiremos exponiendo y comentando los resultados de estos análisis de forma comparativa.

Abstract

The main goal of this work is to explore the foundations of some topics traditionally included in the field of machine learning. Specifically, we propose a general approach to classification and regression problems, and then delve into selected techniques that offer answers to the famous questions they pose.

Many of the chosen solutions have captured the attention of a large part of the scientific community in recent years, thanks to their excellent performance and promising future. The algorithms mentioned include, among others: *linear regression*, *linear regression regularizations*, *logistic regression*, *Gaussian process regression*, *artificial neural networks*, *naive Bayes classifier*, *linear discriminant analysis*, *support vector machines*, *decision trees*, and *k-nearest neighbors* classifier. While the aforementioned methods belong to the supervised learning family, we also include the unsupervised learning technique, *k-means clustering*, which stands out for its proven advantages in numerous application cases.

To conclude our proposed task, we will test three regression algorithms and three classification algorithms, using two different and carefully identified databases, one for each problem. We will close by showing and discussing the results of these analyses in a comparatively way.

Índice general

Capítulo 1

Introducción

En la última década hemos sido testigos del auge explosivo de los campos de la inteligencia artificial y el aprendizaje automático, y si bien existen razones válidas para este crecimiento, la popularidad que éstos han ganado también ha contribuido a desvirtuar los términos y sobre todo a generar confusión respecto a sus alcances. Es por ello que consideramos importante intentar arrojar luz sobre materias [32], [46].

La inteligencia artificial (IA) nació en los años cincuenta del siglo XX, cuando un grupo de pioneros de la naciente ciencia informática empezó a preguntarse si era posible hacer que los ordenadores "pensaran". Una definición concisa de la IA podría ser la siguiente: esfuerzo por automatizar tareas intelectuales normalmente realizadas por humanos.

Durante mucho tiempo, los expertos creyeron posible diseñar una inteligencia artificial de nivel humano elaborando manualmente un conjunto suficientemente amplio de reglas explícitas para manipular el conocimiento; este enfoque conocido como IA simbólica, fue el paradigma dominante hasta finales de los ochenta. Si bien la IA simbólica demostró ser adecuada para resolver problemas lógicos bien definidos, como jugar al ajedrez, resultó muy difícil determinar reglas explícitas para resolver problemas más complejos, como la clasificación de imágenes, el reconocimiento de voz y la traducción de idiomas. A partir de los años noventa, el enfoque del aprendizaje automático adquirió protagonismo, desplazando a la IA simbólica de la primera línea de investigación. En nuestros días, la IA se ha transformado en un basto campo que abarca áreas como el aprendizaje automático, el reconocimiento de patrones y el aprendizaje profundo [10], [26], [36].

El término *machine learning* (ML) o en español aprendizaje automático, fue acuñado en 1959 por Arthur L. Samuel en el contexto de la resolución de juegos de damas usando máquinas. El aprendizaje automático surge motivado por la pregunta: si en lugar de que los programadores creen manualmente las reglas de procesamiento de datos, ¿podría un ordenador aprenderlas automáticamente al analizar los datos? Esta cuestión abre las puertas a un nuevo paradigma informático, en el que se introducen los datos y las respuestas esperadas a partir de ellos, a continuación la máquina generan las reglas y estas se aplican a nuevos datos que producen respuestas originales.

Las raíces del ML se hunden profundamente en un amplio y fértil abanico de disciplinas que incluye entre otras, a la estadśtica, la probabilidad, el procesamiento de señales, las teorías de la información y decisión, la ingeniería electrónica y por supuesto las ciencias de la computación. Posiblemente lo más llamativo del aprendizaje automático sea el variado rango de aplicaciones, que va desde el procesamiento de imágenes y del lenguaje natural hasta la predicción de complejos fenómenos astrofísicos [1], [29].

El aprendizaje automático tradicionalmente se divide en tres categorías que corresponden a enfoques de aprendizaje distintos, aplicados según sea la naturaleza de los datos o información disponible. Si bien los grandes grupos son *aprendizaje supervisado*, *aprendizaje no supervisado* y *aprendizaje por refuerzo*, en este documento nos centramos especialmente en el primero y trataremos de forma marginal el segundo [28], [37], [19].

1.0.1. Aprendizaje supervisado

El aprendizaje supervisado se basa en un conjunto de datos, de los que se conocen los correspondientes resultados esperados o de salida. El objetivo es entonces entrenar un sistema de aprendizaje automático con la información anterior, para predecir los resultados cuando se presenten nuevas entradas [24].

1.0.2. Aprendizaje no supervisado

En el enfoque de aprendizaje no supervisado, no se dispone de datos de salida como referencia, un caso típico de aplicación lo encontramos en el problema de agrupación. Todo el proceso de modelado se lleva a cabo sobre un conjunto de datos formado sólo por entradas al sistema, no se tiene información sobre las categorías de esos datos. Por tanto, el sistema tiene que ser capaz de reconocer patrones para poder etiquetar nuevas entradas [24].

1.1. Problema de regresión

Si bien existe evidencia de que Isaac Newton empleó un método de características similares en un estudio sobre los equinoccios, el término regresión fue introducido de forma precisa por primera vez por Francis Galton, quien en 1886 formuló la llamada *ley de la regresión*. Galton infirió esta ley examinando una muestra $\{(X_i, Y_i)\}$ de 205 familias, donde X_i representaba la desviación de la altura promedio de los padres de la i-ésima familia, mientras que Y_i era la desviación del promedio de altura de la descendencia correspondiente. Denotando por $E[Y \mid X]$ la desviación media de la altura de la descendencia cuyos padres tienen una desviación de la altura de X, la ley de regresión de Galton se escribe como:

$$E[Y \mid X] = \frac{2}{3}X. \tag{1.1}$$

La expresión anterior corresponde claramente a una línea recta de intercepto cero, a partir de este resultado la idea era conseguir vincular el valor de la variable de *entrada* X con el valor de la variable de *salida* Y. Tras Galton, científicos como George U. Yule, Karl Pearson y Ronald A. Fisher, basándose sobre todo en el método de mínimos cuadrados de Gauss y Legendre, contribuyeron significativamente al desarrollo de lo que hoy se conoce como *análisis de regresión*.

El objetivo último del análisis de regresión es resolver el *problema de la regresión*, el cuál consiste en predecir una variable de salida Y de valor real a partir de una variable de entrada X, dado un conjunto de N observaciones $D = \{(x_1, y_1), \ldots, (x_N, y_N)\}$ denominado de entrenamiento. La idea es emplear la información de dicho conjunto para "entrenar" un predictor o función de regresión con la intención de adivinar el valor de Y para el correspondiente de X. Sin embargo, el carácter experimental de la mayoría de aplicaciones con mucha frecuencia trae aparejado falta de información o la presencia de ruido, hechos que dificultan conocer las distribuciones de

las variables. De aquí que a la generalización del modelo de Galton dada por $E[Y \mid X] = f(X, \mathbf{\Omega})$, donde f es una función de la variable de entrada X y un vector de parámetros desconocidos $\mathbf{\Omega}$ cuyos valores es necesario determinar a partir de los resultados de las observaciones, requiera de la adición de un término que exprese la incertidumbre de los escenarios de información incompleta o distorcionada, para conseguir una predicción satisfactoria de la variable Y,

$$Y = f(X, \mathbf{\Omega}) + \varepsilon. \tag{1.2}$$

Si bien la tarea del análisis de regresión ha trascendido de simplemente derivar una relación lineal a partir de datos, hasta convertirse en un enorme campo de estudio que involucra a multitud de áreas entre los que se encuentran la estadística y el aprendizaje automático, las tareas fundamentales de éste siguen siendo tres: 1) la elección de un modelo de regresión, lo que implica supuestos sobre la dependencia de la función de regresión f respecto de X y $\mathbf{\Omega}$; 2) una estimación de los parámetros $\mathbf{\Omega}$ en el modelo seleccionado; y 3) la comprobación de las hipótesis estadísticas sobre el resultado de la regresión. [4], [45]

1.2. Problema de clasificación

En términos generales el *problema de clasificación*, consiste de cuatro elementos principales: una colección de N objetos O_i con $i \in \{1, \ldots, N\}$, cada uno representado por un vector real de dimensión d, $\boldsymbol{x}_i \in \mathbb{R}^d$; un conjunto $\{1, \ldots, M\}$ de clases o categorías; una variable Y etiquetadora que toma valores en dicho conjunto; y una función clasificadora f que mapea vectores del espacio $\mathbb{R}^d$ al conjunto de etiquetas de clase $\{1, \ldots, M\}$.

A partir de estas piezas, la idea es realizar hipótesis que nos permitan aproximar la probabilidad conjunta $p_{\boldsymbol{X}Y}$, siendo $\boldsymbol{X}$ la variable aleatoria que engloba todos los posibles valores que caracterizan los objetos que queremos clasificar, y con ello determinar el criterio de clasificación de f.

Si bien generalmente consideramos todo $\mathbb{R}^d$ como espacio de entrada en el que se define f, puede ocurrir que dicho espacio sea menor en ciertas situaciones como cuando algunas de las entradas del vector aleatorio $\boldsymbol{X}$ toman valores en subconjuntos propios de $\mathbb{R}$. En la práctica, el tamaño del conjunto de prueba suele aproximarse al del conjunto de entrenamiento integrado por los N puntos de $\mathbb{R}^d$.

De una forma u otra, un clasificador divide el espacio en tantos conjuntos como clases haya. La frontera entre estos subconjuntos de $\mathbb{R}^d$ se conocen como límites de clasificación o decisión de f.

Un operador que resulta de gran interés en tanto que nos permite tener una medida del grado de exactitud de un clasificador y por extensión de una técnica de clasificación, es el corchete de Iverson cuya intención original era la de generalizar la delta de Kronecker (atribuido al matemático canadiense Kenneth E. Iverson). Dada una proposición lógica, el corchete retorna 1 si ésta se satisface ó 0 si no:

$$[Q] = \begin{cases} 1, \text{ si } Q \text{ es verdadero} \\ 0, \text{ en otro caso} \end{cases}. \tag{1.3}$$

La medida estándar que nos permite determinar la fracción de objetos mal clasificados al aplicar un determinado clasificador f, emplea el corchete de Iverson y se define como:

$$\varepsilon \quad = \quad \int_{\mathbb{R}^d} \sum_{y \in \{1,\ldots,M\}} [f(x) \neq y]\, p_{\boldsymbol{X}\, Y}(x, y)\, dx. \tag{1.4}$$

Esta medida se conoce con diversos nombres, entre los más populares se destacan: *tasa de error*, *error de clasificación* y *probabilidad de clasificación errónea*. No obstante, también es frecuente encontrar en la literatura especializada el términos *precisión*, que no es otra cosa que $1 - \varepsilon$. Cuanto menor sea ε para un clasificador en particular, mejor diremos que es ya que tendrá un rendimiento superior según las expectativas.

El error de clasificación cumple una segunda función, y es que también actúa como instrumento para jerarquizar y decidir que clasificador es mejor. Este hecho sin embargo, deja la puerta abierta a un escenario en el que no existe un único clasificador óptimo para un problema concreto [4], [47].

1.3. Fundamentos de estimación de parámetros

En esta sección trataremos las generalidades de dos de los métodos más populares para la estimación de parámetros. Ampliamente utilizados en áreas de la estadística y el machine learning, nos serán de utilidad para justificar los resultados de las técnicas que estudiaremos en los capítulos siguientes.

1.3.1. Método de máxima verosimilitud

Desarrollado por los connotados matemáticos británicos Francis Y. Edgeworth y Ronald A. Fisher, el *método de máxima verosimilitud* o MLE por sus siglas en inglés, es un método general, empleado para construir estimadores de parámetros desconocidos.

Sea una muestra aleatoria $X_1, \ldots, X_n$, proveniente de población X con distribución de probabilidad $f_{\boldsymbol{\omega}}$ que depende de un parámetro desconocido $\boldsymbol{\omega} \in \Omega \subseteq \mathbb{R}^m$; partiendo de una realización de esta muestra $\boldsymbol{x} = (x_1, \ldots, x_n)$, la tarea consiste en estimar $\boldsymbol{\omega}$. Denominamos *función de verosimilitud* $L(\boldsymbol{\omega})$, a la distribución conjunta de la muestra evaluada en $\boldsymbol{x}$,

$$L(\boldsymbol{\omega}) \quad = \quad f_{\boldsymbol{\omega}}(x_1, \ldots, x_n). \tag{1.5}$$

El objetivo de la estimación de máxima verosimilitud es encontrar los valores de los parámetros del modelo que maximizan la función de verosimilitud en el espacio de parámetros Ω, esto es:

$$L(\hat{\boldsymbol{\omega}}) \quad = \quad \max_{\boldsymbol{\omega} \in \Omega} L(\boldsymbol{\omega}). \tag{1.6}$$

La cantidad $\hat{\boldsymbol{\omega}}$ se denomina justamente *estimador de máxima verosimilitud*. En la práctica, es recomendable trabajar con el logaritmo de la función de verosimilitud, también conocido como *función de log-verosimilitud*,

$$l(\boldsymbol{\omega}) \quad = \quad \log L(\boldsymbol{\omega}). \tag{1.7}$$

Puesto que el logaritmo es una función monótona, el máximo de $l(\boldsymbol{\omega})$ se presenta para el mismo valor de $\boldsymbol{\omega}$ que maximiza $L(\boldsymbol{\omega})$. Si $l(\boldsymbol{\omega})$ es una función derivable respecto de $\boldsymbol{\omega} = (\omega_1, \ldots, \omega_m)$ en el interior del espacio

paramétrico Ω, la condición suficiente para determinar el estimador de máxima verosimilitud es resolver las conocidas *ecuaciones de verosimilitud*,

$$\frac{\partial}{\partial \omega_k} \log f_{\boldsymbol{\omega}}(\boldsymbol{x}) = 0, \ \forall k \in \{1, \ldots, m\}. \tag{1.8}$$

Para comprobar que el estimador $\hat{\boldsymbol{\omega}}$, deducido de las ecuaciones de verosimilitud, corresponde realmente a un máximo, la matriz hessiana evaluada en éste $H(\hat{\boldsymbol{\omega}})$ debe ser semidefinida negativa. Y por supuesto, la complejidad de todos los cálculos anteriormente indicados se simplificaría notablemente, si la distribución conjunta del vector aleatorio $\boldsymbol{X}$ factoriza como producto de las distribuciones marginales, $f_{\boldsymbol{\omega}}(\boldsymbol{x}) = \prod_{l=1}^{n} f_{\boldsymbol{\omega}}(x_l)$; esto nos permite rrescribir las ecuaciones de verosimilitud como: [35], [33]

$$\frac{\partial}{\partial \omega_k} \sum_{l=1}^{n} \log f_{\boldsymbol{\omega}}(x_l) = 0, \ \forall k \in \{1, \ldots, m\}. \tag{1.9}$$

1.3.2. Método de mínimos cuadrados

Derivado de forma independientemente por Carl F. Gauss y Adrien M. Legendre, el *principio* o *método de mínimos cuadrados* busca estimar los parámetros $\boldsymbol{\Omega} = (\omega_1, \ldots, \omega_m)$ del modelo de regresión, apalancándose para ello en la suma de los errores al cuadrado, $E_c = \sum_k \varepsilon_k^2$. El objetivo es minimizar esta suma cuadrada de las diferencias entre las observaciones de salida y el valor que arroja el predicor al evaluarlo en el dato de entrada,

$$\text{mín} \sum_{k=1}^{n} \varepsilon_k^2 \quad \textit{donde}, \quad \varepsilon_k = y_k - f(x_k, \boldsymbol{\Omega}). \tag{1.10}$$

Si bien a priori la idea resulta fácil de entender, ésta puede ser analizada desde diferentes perspectivas. Concentrándonos sólo en el caso bidimensional tenemos que: si consideramos la distancia vertical entre las observaciones y los valores del predictor, el método se conoce como *regresión directa* o *mínimos cuadrados ordinarios* (OLS); si tenemos en cuenta la distancia en dirección horizontal, hablamos del método de *regresión inversa*; o si alternativamente, elegimos la distancia perpendicular entre las observaciones y las predicciones, entonces estamos tratando con el método de *regresión ortogonal* o *regresión del eje mayor* [35].

Los diferentes enfoques, arrojan distintas estimaciones de los parámetros $\omega_1, \ldots, \omega_m$, las cuáles presentan diferentes propiedades estadísticas. El más popular de todos es la regresión directa.

El uso E_c, en lugar de la suma de los valores absolutos de los errores, $E_a = \sum_k |\varepsilon_k|$, se justifica en tanto que esto permite tratar los residuos como una cantidad continua diferenciable, lo que conduce a una forma analítica mucho más sencilla para los parámetros de ajuste. Sin embargo, al utilizar los cuadrados de los desplazamientos, los puntos atípicos pueden tener un efecto desproporcionado en el ajuste, algo que podría no ser deseable según el problema en concreto.

Un detalle que no puede pasar desapercibido es que en el principio de mínimos cuadrados no se requiere hacer ninguna suposición sobre la distribución de probabilidad de los errores, ε_k, para al obtener las estimaciones de los parámetros. No obstante, para realizar inferencias estadísticas, se asume que los errores ε_k son variables

aleatorias de distribución normal idéntica ($E[\varepsilon_k] = 0$, $Var(\varepsilon_k) = \sigma^2$) e independientes ($Cov(\varepsilon_k, \varepsilon_l) = 0 \ \ \forall k \neq l$, con $k, l \in \{1, \ldots, n\}$).

El procedimiento general, se basa en calcular las derivadas parciales de E_c, respecto de cada parámetro ω_l, igualar éstas a cero y así generar un conjunto de ecuaciones,

$$\frac{\partial E_c}{\partial \omega_l} = -2 \sum_{k=1}^{n} [y_k - f(x_k, \mathbf{\Omega})] \frac{\partial f(x_k, \mathbf{\Omega})}{\partial \omega_l} = 0, \ \forall l = 1, \ldots, m. \tag{1.11}$$

En la versión más conocida de este procedimiento, se asume que la función f es lineal respecto de los parámetros, pudiendo expresarse como combinación lineal de unas funciones $\{\phi_j\}_{j=1}^{m}$ que llamamos funciones base,

$$f(x_k, \mathbf{\Omega}) = \sum_{j=1}^{m} \omega_j \, \phi_j(x_k), \ \forall k = 1, \ldots, n. \tag{1.12}$$

Este caso se conoce como *mínimos cuadrados lineales* o LLS por sus siglas en inglés. Las ecuaciones que se obtienen al sustituir f, se conocen como *ecuaciones normales de Gauss*,

$$\sum_{k=1}^{n} \left(\sum_{j=1}^{m} \omega_j \, \phi_j(x_k) \right) \phi_l(x_k) = \sum_{k=1}^{n} y_k \, \phi_l(x_k), \ \forall l = 1, \ldots, m. \tag{1.13}$$

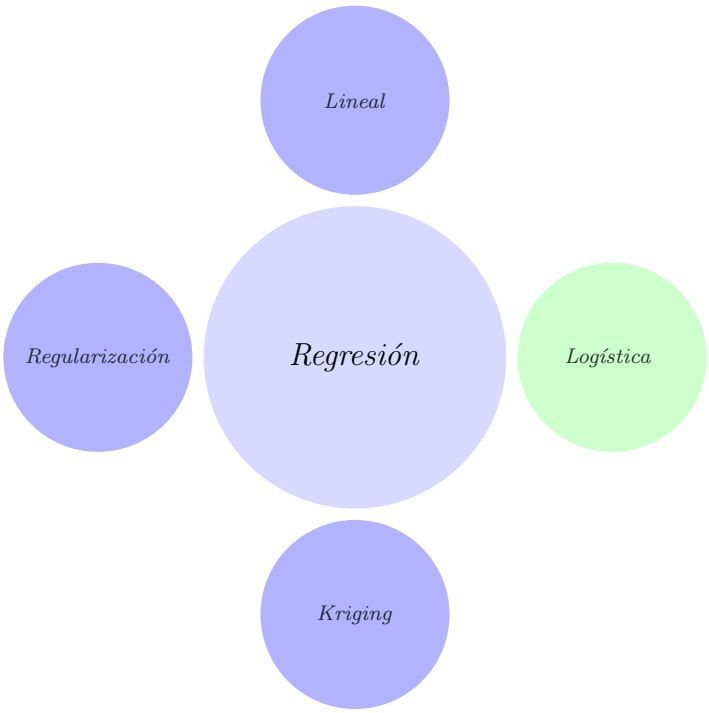
Las expresiones anteriores, condensadas en su forma matricial y sabiendo que usamos la notación $\langle \phi_j, \phi_l \rangle := \phi_j(x_k)\phi_l(x_k)$, tienen el siguiente aspecto:

$$\begin{pmatrix} \langle \phi_1, \phi_1 \rangle & \cdots & \langle \phi_1, \phi_m \rangle \\ \vdots & \ddots & \vdots \\ \langle \phi_m, \phi_1 \rangle & \cdots & \langle \phi_m, \phi_m \rangle \end{pmatrix} \begin{pmatrix} \omega_1 \\ \vdots \\ \omega_m \end{pmatrix} = \begin{pmatrix} \sum_{k=1}^{n} y_k \, \phi_1(x_k) \\ \vdots \\ \sum_{k=1}^{n} y_k \, \phi_m(x_k) \end{pmatrix}. \tag{1.14}$$

En el ajuste por *mínimos cuadrados no lineales* o NLLS por sus siglas en inglés, puede aplicarse de forma iterativa el LLS a una forma linealizada de la función f hasta alcanzaar condiciones de convergencia, típicamente se usa la expansión polinómica de Taylor. Sin embargo, a menudo también es posible linealizar la función desde el principio y seguir empleando métodos lineales para determinar los parámetros de ajuste sin recurrir a procedimientos iterativos. Existen más variantes del método de mínimos cuadrados como el *total least squares*, *generalized least squares* (GLS) o los métodos con poderaciones que buscan subsanar algunas las limitaciones del planteamiento original, como cuando no se cumplen las hipótesis de homocedasticidad, independencia estocástica o se tienen datos demasiado dispersos [35].

Capítulo 2

Técnicas de Regresión



Una definición simple pero redonda de *modelo*, es la de *descripción de un estado o proceso*. Una característica peculiar de los modelos bajo esta óptica, es que a diferencia de lo que sucede con las hipótesis y teorías, éstos no son verdaderos o falsos; su validación se hace a través de la solidez de sus resultados sometidos a comparaciones en el colectivo y con la realidad.

Dentro del universo de los modelos estadísticos, los modelos de regresión son posiblemente los más conocidos. Este conjunto de importancia capital, a su vez se bifurca en dos tipos: los lineales y los no lineales. Donde entendemos que el término lineal engloba tanto los casos de *simple linear regression* (SLR) donde hay una única variable regresora, como aquellos en los que hay dos o más variables independientes o lo que en la literatura suele denominarse *multiple linear regression* (MLR).

El modelo de regresión general, supone que existe una relación funcional f entre la variable respuesta Y y la variable predictora X; dicha relación está mediada por un conjunto de parámetros $\mathbf{\Omega} \equiv (\omega_i)_{i\in I}$. El cuarto elemento indispensable que cierra la estructura del modelo, es el error ε o discrepancia entre los valores respuesta y la predicción [35], [14],

$$\begin{cases} Y & \sim & f(X,\mathbf{\Omega}) \\ \varepsilon & = & Y - f(X,\mathbf{\Omega}) \end{cases} . \tag{2.1}$$

2.1. Lineal

El *modelo de regresión lineal* es posiblemente una de las piedras angulares de la estadística como disciplina moderna, tiene al mismo tiempo un impacto enorme en múltiples campos que van desde las ciencias sociales y humanidades hasta las ciencias más puras como la física y la química. En su forma más simple adopta la siguiente apariencia:

$$Y \quad = \quad \omega_0 + \omega_1 X + \varepsilon. \tag{2.2}$$

Si observamos antentamente, en la expresión anterior (2.2) la función predictora incluye una sola variable aleatoria X, además de los parámetros ω_0 y ω_1. No obstante, esta versión que geométricamente puede interpretarse como una recta, de ahí que muchos autores suelan referirse a la *recta de regresión*, puede extenderse con cierta naturalidad a un conjunto de n variables predictoras y $n+1$ parámetros, geométricamente interpretables como un hiperplano y puestas en el modelo al completo como:

$$Y \quad = \quad \omega_0 + \omega_1 X_1 + \cdots + \omega_n X_n + \varepsilon = \omega_0 + \sum_{k=1}^{n} \omega_k X_k + \varepsilon. \tag{2.3}$$

Escalando todavía más el modelo de regresión lineal, llegamos hasta la forma matricial que algunos autores juzgan más conveniente renombrar *modelo de regresión multilineal*. El hecho distintivo de esta versión, consiste en que la variable respuesta $\boldsymbol{Y}$ ahora es un vector de n componentes al igual que el error $\boldsymbol{E}$. Por su parte, los parámetros $\boldsymbol{\Omega}$ se organizan en una matriz de tamaño $n \times (m+1)$, siendo $m \geq 1$. Lo anterior, podemos resumirlo en la siguiente expresión,

$$\begin{aligned} \boldsymbol{Y} \quad &= \quad \boldsymbol{\Omega}^T \boldsymbol{X} + \boldsymbol{E} \\ \textit{donde} \quad &\begin{cases} \boldsymbol{Y}^T, \boldsymbol{E}^T, \boldsymbol{X}^T \in \mathbb{R}^n \\ \boldsymbol{\Omega} \in M_{n\times(m+1)}(\mathbb{R}) \end{cases} . \end{aligned} \tag{2.4}$$

Una reformulación sencilla que podemos hacer respecto del modelo de regresión lineal, expuesto en la expresión 2.3 y que en el futuro nos será de gran utilidad; es la siguiente:

$$\begin{aligned} Y \quad &= \quad \boldsymbol{X}^T \boldsymbol{\Omega} + \varepsilon \\ \textit{donde} \quad &\begin{cases} \boldsymbol{X}^T = (X_0, X_1, \ldots, X_n), \ \textit{con} \ \ X_0 = 1 \\ \boldsymbol{\Omega}^T = (\omega_0, \omega_1, \ldots, \omega_n) \end{cases} . \end{aligned} \tag{2.5}$$

Hasta ahora hemos contemplado el caso en el que la función predictora $f(X, \boldsymbol{\Omega})$ es lineal en ambas variables. Sin embargo, esta dependencia lineal tanto de X y como de ω, si bien le brinda una atractiva inocencia, al mismo tiempo limita notablemente el alcance. Una primera modificación que podemos hacer al modelo de regresión lineal con la intención de generalizarlo, es eliminar la relación lineal con la variable predictora $\boldsymbol{X}$, pero conservando ésta en los parámetros $\boldsymbol{\Omega}$. Esta pequeña modificación nos permite escribir lo siguiente:

$$Y = \sum_{k=0}^{n} \omega_k \, \phi_k(X_1, \ldots, X_p) + \varepsilon = \omega_0 + \sum_{k=1}^{n} \omega_k \, \phi_k(X_1, \ldots, X_p) + \varepsilon = \boldsymbol{\Phi}^T(\boldsymbol{X}) \cdot \boldsymbol{\Omega} + \varepsilon \tag{2.6}$$

$$\textit{donde} \quad \begin{cases} \boldsymbol{X} = (X_1, \ldots, X_p) \\ \boldsymbol{\Phi}^T = (\phi_0, \ldots, \phi_n) \;\; \textit{con} \;\; \phi_0(\boldsymbol{X}) = 1 \\ \boldsymbol{\Omega}^T = (\omega_0, \ldots, \omega_n) \end{cases} .$$

Las funciones $\{\phi_k\}_{k=0}^n$ se denominan *funciones básicas*, mientras ω_0 es el *parámetro de sesgo*. Tras saborear la propuesta de modelo de regresión lineal anterior, resulta tentador preguntarse que sucede si continuamos relajando las condiciones exigidas a la función predictora $f(X, \boldsymbol{\Omega})$, es decir si ahora permitimos que sea no lineal en los parámetros o directamente que no sea lineal en ninguna componente. La respuesta corta a estas cuestiones es que al aceptar estos cambios, realmente estamos adentrándonos en el reino de los *modelos de regresión no lineales*, cuyo objetivo se aparta desgraciadamente de nuestra materia de estudio. Aún así y solo con el ánimo de ilustrar, podemos presentar un par de ejemplos de esta familia [39],

$$Y = \omega_0 + X_1^{\omega_1} + \sinh(\omega_2) \, X_2^2 + \varepsilon \quad , \quad Y = \omega_0 \, e^{\omega_1 X_1} + \omega_2 \, X_2^{-3} + \varepsilon. \tag{2.7}$$

Retomando la discusión sobre las funciones básicas, existen numerosos ejemplos del tipo que suelen emplearse, sin embargo aquí citaremos sólo algunas de las que juzgamos más destacadas, a saber: polinómicas $\phi_k(z) = z^k$, sigmoides $\phi_k(z) = \frac{1}{1+\exp\{-(z-\mu_k)/s\}}$ y gaussianas $\phi_k(z) = \exp\left\{-\frac{(z-\mu_k)^2}{2\,s^2}\right\}$.

El modelo descrito en 2.6, no es en absoluto definitvo, en tanto que es admite al menos una generalización más. Este hecho que nos impulsa a plantear una versión adicional en la que tanto la variable respuesta $\boldsymbol{Y}$ como el error $\boldsymbol{E}$ son vectores como en 2.4, pero esta vez usando la formulación heredada de las funciones básicas $\boldsymbol{\Phi}$,

$$\boldsymbol{Y} = \boldsymbol{\Omega}^T \boldsymbol{\Phi}(\boldsymbol{X}) + \boldsymbol{E} \tag{2.8}$$

$$\textit{donde} \quad \begin{cases} \boldsymbol{Y}^T, \boldsymbol{E}^T \in \mathbb{R}^n \\ \boldsymbol{\Omega} \in M_{(m+1)\times n}(\mathbb{R}) \\ \boldsymbol{\Phi}^T \in \mathbb{R}^{m+1} \;\; \textit{con} \;\; \phi_0(\boldsymbol{X}) = 1 \\ \boldsymbol{X} \in \mathbb{R}^p \end{cases} .$$

Tras plantear los distintos modelos de regresión lineal, un punto que resulta crucial tratar en este estudio, son las hipótesis centrales sobre las que se apoyan estos modelos. Y es que además de la *relación de dependencia lineal*, se precisan otras premisas de partida, como son: [40]

- *Homocedasticidad:* Los errores o discrepacias ε_k tienen todos las misma varianza, $Var(\varepsilon_k) = \sigma^2 < \infty, \; \forall k$.
- *Homogeneidad:* El valor esperado de los errores ε_k es cero, $E(\varepsilon_k) = 0$. Para cada valor de $\boldsymbol{X}$, la discrepancia tomará distintos valores de forma aleatoria, pero no tomará sistemáticamente valores positivos o negativos, sino que tomará algunos valores mayores que cero y otros menores que cero, de tal forma que el valor esperado será cero.

- *Normalidad:* Los errores presentan una distribución normal, $\varepsilon \sim N(0, \sigma^2)$.

- *Independencia estocástica:* Las observaciones son independientes, de este hecho se deriva que los errores no están correlacionados, $Cov(\varepsilon_k, \varepsilon_l) = 0$ para todo k y l. Esto quiere decir que el valor de una discrepancia para cualquier observación muestral no está influenciado por los valores de las discrepancias correspondientes a otras observaciones.

Adicionalmente, en los casos en los que se dispone de más de una variable predictora, resulta indispensable contar con una hipótesis más que buscar garantizar que ninguna variable pueda ser explicada por las demás, es decir que no se presente un escenario de dependencia lineal.

- *Independencia lineal.* Las variables explicativas $\{X_1, \ldots, X_n\}$ son linealmente independientes, es decir que ninguna puede ser combinación lineal de las otras.

Después abordar las premisas centrales en torno a los cuales orbita la lógica subyacente al modelo de regresión lineal, procedemos a exponer la estimación del vector de parámetros para la versión vista en 2.6. Dicha estimación se obtiene usando el método de máxima verosimilitud y considerando una muestra de entrenamiento $D = \{(\boldsymbol{x}_k, y_k)\}_{k=1}^N \subseteq \mathbb{R}^p \times \mathbb{R}$,

$$\begin{aligned} \hat{\boldsymbol{\Omega}} &= \left(\boldsymbol{\Psi}^T \boldsymbol{\Psi}\right)^{-1} \boldsymbol{\Psi}^T \boldsymbol{y} \\ \textit{donde} \qquad & \boldsymbol{y}^T = (y_1, \ldots, y_N). \end{aligned} \tag{2.9}$$

La matriz $\boldsymbol{\Psi} = (\psi_{lk})$, conocida como *matriz de densidad* o *matriz de regresión*, se define a través de sus entradas como $\psi_{lk} = \phi_k(\boldsymbol{x}_l)$ con $k \in \{0, \ldots, n\}$ y $l \in \{1, \ldots, N\}$; o más explícitamente:

$$\boldsymbol{\Psi} = \begin{pmatrix} \phi_0(\boldsymbol{x}_1) & \ldots & \phi_n(\boldsymbol{x}_1) \\ \vdots & \ddots & \vdots \\ \phi_0(\boldsymbol{x}_N) & \ldots & \phi_n(\boldsymbol{x}_N) \end{pmatrix}. \tag{2.10}$$

La expresión para la estimación de $\boldsymbol{\Omega}$ dada en 2.9, puede reformularse empleando la *pseudoinversa de Moore-Penrose* definida como:

$$\boldsymbol{\Psi}^\dagger := \left(\boldsymbol{\Psi}^T \boldsymbol{\Psi}\right)^{-1} \boldsymbol{\Psi}^T. \tag{2.11}$$

Esto da lugar a una expresión más compacta y en muchos aspectos cómoda, $\hat{\boldsymbol{\Omega}} = \boldsymbol{\Psi}^\dagger \boldsymbol{y}$. Si bien existen distintos indicadores clave que nos permiten conocer la bondad del ajuste, uno de los más intuitivos es el de *precisión*, que no es otra cosa el inverso de la varianza,

$$\hat{\beta}^{-1} = \frac{1}{N} \sum_{k=1}^{N} \left[y_k - \hat{\boldsymbol{\Omega}}^T \boldsymbol{\Phi}(\boldsymbol{x}_k) \right]^2. \tag{2.12}$$

El análisis de la varianza o ANOVA por sus siglas en inglés (analysis of variance), nos proporciona otras conocidas métricas que dan cuenta de la bondad de los ajustes. Ejemplo de ello es que de la descomposición de la

variabilidad total del modelo $VT = \sum_{k=1}^{N}(y_k - \overline{y})^2$, como suma de los términos $VNE = \sum_{k=1}^{N}(y_k - \hat{y}_k)^2$ o *suma de los residuos cuadrados* (denominada por otros autores como *variabilidad no explicativa*) y $VE = \sum_{k=1}^{N}(\hat{y}_k - \overline{y})^2$ o *variabilidad explicativa*, $VT = VE + VNE$, se deriva el célebre *coeficiente de determinación* que es el cuadrado del también famoso *coeficiente de correlación* R [14],

$$R^2 = \frac{\sum_{k=1}^{N}(\hat{y}_k - \overline{y})^2}{\sum_{k=1}^{N}(y_k - \overline{y})^2} \tag{2.13}$$

$$\textit{donde} \quad \begin{cases} \overline{y} = \frac{1}{N}\sum_{k=1}^{N} y_k \\ \hat{y}_k = \hat{\mathbf{\Omega}}^T \mathbf{\Phi}(\boldsymbol{x}_k) \end{cases} .$$

Para concluir la sección, notamos que la solución para el modelo más general que planteamos en 2.8, se asemeja al resultado que hemos expuesto en 2.9; de hecho un procedimiento habitual es considerar n problemas de regresión desacoplados, en lugar del sistema matricial completo. Dado el conjunto de entrenamiento, $D = \{(\boldsymbol{x}_k, \boldsymbol{y}_k)\}_{k=1}^{N} \subseteq \mathbb{R}^p \times \mathbb{R}^n$, la estimación de la matriz de parámetros que nos proporciona el método de máxima verosimilitud para la versión matricial es la siguiente:

$$\hat{\mathbf{\Omega}} = \left(\mathbf{\Psi}^T \mathbf{\Psi}\right)^{-1} \mathbf{\Psi}^T \mathbf{Y} \tag{2.14}$$

$$\textit{donde} \quad \mathbf{Y} = \begin{pmatrix} — & \boldsymbol{y}_1^T & — \\ & \vdots & \\ — & \boldsymbol{y}_k^T & — \\ & \vdots & \\ — & \boldsymbol{y}_N^T & — \end{pmatrix} \in M_{N\times n}(\mathbb{R}), \quad \boldsymbol{y}_k = \begin{pmatrix} y_{k,1} \\ \vdots \\ y_{k,n} \end{pmatrix} .$$

Para este caso la matriz pseudoinversa de Moore-Penrose es de tamaño $(m+1) \times N$, $\mathbf{\Psi}^{\dagger} \in M_{(m+1)\times N}(\mathbb{R})$, y como antes $\Psi_{lk} = \phi_k(\boldsymbol{x}_l)$ para $k \in \{0, \ldots, m\}$ y $l \in \{1, \ldots, N\}$. Para escenarios como este, el análisis de la bondad del ajuste conviene hacerlo de forma individualizada a los n problemas desacoplados, es decir que basata con remitirnos a los resultados inmediatamente anteriores [18], [2], [49], [14].

2.2. Regularización

Los modelos de *regularización*, están basados en la construcción que proporciona el modelo de regresión lineal. Como ya hemos visto, el procedimiento habitual para ajustar este modelo es emplear el método de mínimos cuadrados o el máxima verosimilitud, es decir, optimizar la suma del cuadrado de los residuos,

$$E_c = \sum_{k=1}^{N-1}\left(y_k - \omega_0 - \sum_{l=1}^{m} x_{kl}\,\omega_l\right)^2 . \tag{2.15}$$

Este procedimiento arroja resultados razonables, con poco sesgo cuando el tamaño del conjunto de datos $N-1$ es mayor que el de parámetros m, $N \gg m$. Las dificultades surgen cuando el cardinal del conjunto de parámetros (m) crece o cuando las correlaciones de las variables predictoras es alta; entonces la estimación del modelo tendrá mucha varianza y habrá un sobreajuste. La solución pasa por forzar a que el modelo tenga menos complejidad para así reducir su varianza.

Una forma de conseguirlo es mediante la regularización o *shrinkage* de la estimación de los parámetros, que consiste en considerar todas las variables predictoras, pero forzando a que algunos de los parámetros se estimen mediante valores muy próximos a cero, o directamente con ceros. Estas técnicas provocarán un pequeño aumento en el sesgo, pero a cambio una notable reducción en la varianza y una interpretación más sencilla del modelo como resultado. Si bien existen númerosas técnicas de regularización, hemos decidido destacar tres de las más recurrentes en aplicaciones: *ridge*, *LASSO* y *elastic net* [15].

2.2.1. Ridge

La técnica de regularización *ridge*, también conocida como regularización de Tikhonov, en honor a Andrey N. Tikhonov se emplea para estimar los coeficientes de modelos de regresión lineal múltiple en escenarios donde las variables independientes están altamente correlacionadas.

El planteamiento de este método está basado en un trabajo de Hoerl y Kennard de principios de los setenta, que emplea la norma L_2 o $\|\cdot\|_2^2$ ($\|\cdot\|_2^2 : \mathbb{R}^m \to \mathbb{R}$, $\boldsymbol{z} \mapsto \|\boldsymbol{z}\|_2^2 = \sum_{l=1}^m |z_l|^2$) para pesar las entradas del vector de parámetros. Para λ un hiperparámetro, la solución al problema de optimización subyacente tiene el siguiente aspecto:

$$\hat{\boldsymbol{\Omega}}^{ridge} = \underset{\boldsymbol{\Omega}}{\operatorname{argmin}} \left\{ \sum_{k=1}^{n-1} \left(y_k - \omega_0 - \sum_{l=1}^{m} x_{kl}\omega_l \right)^2 + \lambda \sum_{l=1}^{m} |\omega_l|^2 \right\}. \tag{2.16}$$

De forma equivalente, para $t > 0$ el planteamiento formal del problema es:

$$\begin{aligned} \underset{\boldsymbol{\Omega}}{\text{mín}} \quad & \sum_{k=1}^{n-1} \left(y_k - \omega_0 - \sum_{l=1}^{m} x_{kl}\omega_l \right)^2 \\ \text{sujeto a:} \quad & \sum_{l=1}^{m} |\omega_l|^2 \ \leq \ t. \end{aligned} \tag{2.17}$$

La solución de esta técnica de regularización del modelo de regresión, puede ser facilmente calculada empleando cualquiera de las estrategias de optimización, tipicamente el método de máxima verosimilitud o el de mínimos cuadrados [15], [18]:

$$\hat{\boldsymbol{\Omega}}^{ridge} = \left(\mathbf{X}^T\mathbf{X} + \lambda \boldsymbol{I} \right)^{-1} \mathbf{X}^T \boldsymbol{y}. \tag{2.18}$$

2.2.2. LASSO

La técnica *least absolute shrinkage and selection operator* mejor conocida por sus siglas LASSO, fue introducido por Robert Tibshirani al final de la década de los noventa. Está basado en el uso de una penalización que emplea la norma L_1 o $\|\cdot\|_1^1$ ($\|\cdot\|_1^1 : \mathbb{R}^m \to \mathbb{R}$, $\boldsymbol{z} \mapsto \|\boldsymbol{z}\|_1^1 = \sum_{l=1}^m |z_l|$), para pesar las entradas del vector de parámetros. Para λ un hiperparámetro, la solución al problema de optimización subyacente sería:

$$\hat{\boldsymbol{\Omega}}^{lasso} \quad = \quad \underset{\boldsymbol{\Omega}}{\operatorname{argmin}} \left\{ \sum_{k=1}^{n-1} \left(y_k - \omega_0 - \sum_{l=1}^{m} x_{kl}\, \omega_l \right)^2 + \lambda \sum_{l=1}^{m} |\omega_l| \right\}. \tag{2.19}$$

Con $t > 0$, el problema subyacente en cuestión formalmente planteado es:

$$\begin{aligned} \underset{\boldsymbol{\Omega}}{\text{mín}} \quad & \sum_{k=1}^{n-1} \left(y_k - \omega_0 - \sum_{l=1}^{m} x_{kl}\, \omega_l \right)^2 \\ \text{sujeto a:} \quad & \sum_{l=1}^{m} |\omega_l| \ \leq \ t. \end{aligned} \tag{2.20}$$

La *regresión de ángulo mínimo* o *least-angle regression* (LARS o en ocasiones LAR) es una técnica relativamente nueva, ideada por Bradley Efron y sus colaboradores en los primeros años de la década de los 2000. Está estrechamente relacionada con la regresión LASSO, a tal punto que proporciona un algoritmo extremadamente eficiente para calcular su solución.

LARS utiliza una estrategia similar a la *forward stepwise regression*, añadiendo secuencialmente una variable al conjunto activo. En el primer paso, se identifica la variable más correlacionada con la respuesta, y en lugar de ajustar esta variable se mueve el coeficiente de ésta de forma continua hacia su valor de mínimos cuadrados, lo que provoca que su correlación con el residuo disminuya en valor absoluto. En cuanto otra variable alcanza la correlación con el residuo, el proceso se detiene. La segunda variable se une entonces al conjunto activo, y sus coeficientes se mueven juntos de forma que sus correlaciones se mantienen estables y decrecientes [13].

De la técnica LASSO existen numerosas variantes, como la *Fused LASSO* o el *Group LASSO* que buscan hacer visible una dependencia temporal de las variables o flexibilizar la selección de éstas [15], [18].

2.2.3. Elastic net

La técnica *elastic net*, combina linealmente las penalizaciones L_1 y L_2 de los métodos de LASSO y ridge. Este hecho, hace que este método regularización suela ser más preciso que ambos métodos por separado. Para el hiperparámetro $0 \leq \alpha \leq 1$, la solución al problema de optimización subyacente sería:

$$\hat{\boldsymbol{\Omega}}^{e.n.} \quad = \quad \underset{\boldsymbol{\Omega}}{\operatorname{argmin}} \left\{ \sum_{k=1}^{n-1} \left(y_k - \omega_0 - \sum_{l=1}^{m} x_{kl}\, \omega_l \right)^2 + \lambda \sum_{l=1}^{m} \left(\alpha\, |\omega_l|^2 + (1-\alpha)\, |\omega_l| \right) \right\}. \tag{2.21}$$

De manera equivalentemente, para $t > 0$ el planteamiento formal del problema es:

$$\begin{aligned} \underset{\boldsymbol{\Omega}}{\text{mín}} \quad & \sum_{k=1}^{n-1} \left(y_k - \omega_0 - \sum_{l=1}^{m} x_{kl}\, \omega_l \right)^2 \\ \text{sujeto a:} \quad & \sum_{l=1}^{m} \left(\alpha\, |\omega_l|^2 + (1-\alpha)\, |\omega_l| \right) \ \leq \ t. \end{aligned} \tag{2.22}$$

Esta generalización deja la puerta abierta a otras en las que la norma considerada sea $\|\cdot\|_p^p$ con $p \in \mathbb{N} \cup \{\infty\}$ ($(\|\cdot\|_p^p : \mathbb{R}^m \to \mathbb{R},\ \boldsymbol{z} \mapsto \|\boldsymbol{z}\|_p^p = \sum_{l=1}^{m} |z_l|^p)$), y por tanto la solución al problema de optimización asociado sería del estilo de: [15], [18]

$$\hat{\boldsymbol{\Omega}} \quad = \quad \underset{\boldsymbol{\Omega}}{\operatorname{argmin}} \left\{ \sum_{k=1}^{n-1} \left(y_k - \omega_0 - \sum_{l=1}^{m} x_{kl}\, \omega_l \right)^2 + \lambda \sum_{l=1}^{m} |\omega_l|^p \right\}. \tag{2.23}$$

2.3. Logística

Al contrario de lo que parece sugerir su nombre, la regresión logística es mayoritariamente usada para abordar el problema de clasificación. Esta técnica ha cosechando numerosos éxitos sobre todo en el campo de la bioestadística donde las respuestas binarias ocurren con bastante frecuencia; por ejemplo, los pacientes que sobreviven o mueren, o los padecen o no una enfermedad. La idea central detrás del algoritmo está motivada por el deseo de modelar las distribuciones de probabilidad a posteriori de las clases mediante funciones lineales de $\boldsymbol{X}$.

Suponiendo que la variable respuesta Y toma valores en el conjunto finito de clases $\{1, \ldots, m\}$, el modelo tiene la siguiente forma:

$$\begin{aligned} \log \left\{ \frac{P(Y = 1 \mid \boldsymbol{X} = \boldsymbol{x})}{P(Y = m \mid \boldsymbol{X} = \boldsymbol{x})} \right\} &= \omega_{1,0} + \boldsymbol{\omega}_1^T \boldsymbol{x} \\ &\vdots \\ \log \left\{ \frac{P(Y = m-1 \mid \boldsymbol{X} = \boldsymbol{x})}{P(Y = m \mid \boldsymbol{X} = \boldsymbol{x})} \right\} &= \omega_{m-1,0} + \boldsymbol{\omega}_{m-1}^T \boldsymbol{x}. \end{aligned} \tag{2.24}$$

La elección de la última clase como denominador es completamente arbitraria, su uso tan sólo responde a la comodidad que brinda en la notación. Aplicando la exponencial y operando las ecuaciones anteriores de forma que se respete $\sum_{k=1}^{m} P(Y = k \mid \boldsymbol{X} = \boldsymbol{x}) = 1$, tenemos el siguiente resultado:

$$\begin{aligned} P(Y = k \mid \boldsymbol{X} = \boldsymbol{x}) &= \frac{\exp\{\omega_{k,0} + \boldsymbol{\omega}_k^T \boldsymbol{x}\}}{1 + \sum_{l=1}^{m-1} \omega_{l,0} + \boldsymbol{\omega}_l^T \boldsymbol{x}}, \; k \in \{1, \ldots, m-1\} \\ P(Y = m \mid \boldsymbol{X} = \boldsymbol{x}) &= \frac{1}{1 + \sum_{l=1}^{m-1} \omega_{l,0} + \boldsymbol{\omega}_l^T \boldsymbol{x}}, \; k \in \{1, \ldots, m-1\}. \end{aligned} \tag{2.25}$$

El modelo de regresión logística suele ajustarse empleando el método de máxima verosimilitud, aprovechandose así del hecho de que es conocida la forma explícita de $P(Y \mid X)$. Si coleccionamos los parámetros del modelo en $\Omega = \{\omega_{1,0}, \boldsymbol{\omega}_1^T, \ldots, \omega_{m-1,0}, \boldsymbol{\omega}_{m-1}^T\}$, denotamos las probabilidades condicionadas como $P(Y = k \mid \boldsymbol{X} = \boldsymbol{x}) = p_k(\boldsymbol{x}; \Omega)$ y recordamos que las clases son disjuntas, la función de log-verosimilitud para un conjunto de N observaciones es como sigue:

$$l(\Omega) \quad = \quad \sum_{i=1}^{N} \log p_{g_i}(\boldsymbol{x}_i; \Omega). \tag{2.26}$$

Si nos limitamos al caso donde existen sólo dos clases, escenario que con frecuencia se presenta en múltiples aplicaciones, entonces notando por $p_1(\boldsymbol{x}; \boldsymbol{\Omega}) = p(\boldsymbol{x}; \boldsymbol{\Omega})$ y $p_0(\boldsymbol{x}; \boldsymbol{\Omega}) = 1 - p(\boldsymbol{x}; \boldsymbol{\Omega})$ las probabilidades y siendo los resultados de clase posibles $y_i \in \{0, 1\}$, la función de log-verosimilitud se reduce a:

$$\begin{aligned} l(\boldsymbol{\Omega}) &= \sum_{i=1}^{N} \left(y_i \boldsymbol{\Omega}^T \boldsymbol{x}_i - \log \left\{ 1 + e^{\boldsymbol{\Omega}^T \boldsymbol{x}_i} \right\} \right) \\ \textit{donde} \qquad & \boldsymbol{\Omega}^T = (\omega_{1,0}, \boldsymbol{\omega}_1^T), \; \boldsymbol{x}_i^T \in \mathbb{R}^{n+1} \; \forall i. \end{aligned} \tag{2.27}$$

Incluso en estas circusntancias, las ecuaciones a las que nos conduce el método de máxima verosimilitud son no lineales, por tanto se hace necesario emplear un algoritmo auxiliar que nos permita obtener una solución computable. En esta situación suele ser de ayuda el algoritmo de *Newton-Raphson*, cuya aplicación nos lleva al siguiente sistema matricial:

$$\begin{aligned} \boldsymbol{\Omega}^{new} &= (\mathbf{X}^T \boldsymbol{W} \mathbf{X})^{-1} \mathbf{X}^T \boldsymbol{W} \boldsymbol{z} \qquad (2.28) \\ \textit{donde} \quad & \boldsymbol{z} = \mathbf{X}\boldsymbol{\Omega}^{old} + \boldsymbol{W}^{-1}(\boldsymbol{y} - \boldsymbol{p}), \\ \mathbf{X} = \begin{pmatrix} — & \boldsymbol{x}_1^T & — \\ & \vdots & \\ — & \boldsymbol{x}_N^T & — \end{pmatrix} \quad , \quad & \boldsymbol{W} = \begin{pmatrix} p(\boldsymbol{x}_1;\boldsymbol{\Omega}^{old})(1-p(\boldsymbol{x}_1;\boldsymbol{\Omega}^{old})) & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & p(\boldsymbol{x}_N;\boldsymbol{\Omega}^{old})(1-p(\boldsymbol{x}_N \end{pmatrix}, \\ \boldsymbol{y} = \begin{pmatrix} y_1 \\ \vdots \\ y_N \end{pmatrix} \quad , \quad & \boldsymbol{p} = \begin{pmatrix} p(\boldsymbol{x}_1;\boldsymbol{\Omega}^{old}) \\ \vdots \\ p(\boldsymbol{x}_N;\boldsymbol{\Omega}^{old}) \end{pmatrix}. \end{aligned}$$

Estas ecuaciones se resuelven de forma iterativa, siendo $\boldsymbol{\Omega}^{new}$ la actualización y $\boldsymbol{\Omega}^{old}$ el valor del paso anterior del vector de parámetros; debemos tener presente además que en cada iteración cambian los valores de $\boldsymbol{p}$, $\boldsymbol{W}$ y $\boldsymbol{z}$. Este procedimiento se conoce como *iteratively reweighted least squares* (IRLS), y es solución del *problema de mínimos cuadrados ponderados*, dada por:

$$\hat{\boldsymbol{\Omega}}^{new} = \underset{\boldsymbol{\Omega}}{\operatorname{argmin}} \left\{ (\boldsymbol{z} - \mathbf{X}\boldsymbol{\Omega})^T \boldsymbol{W} (\boldsymbol{z} - \mathbf{X}\boldsymbol{\Omega}) \right\}. \qquad (2.29)$$

Si bien suele tomarse como valor inicial $\boldsymbol{\Omega}^{old} = 0$ para el procedimiento iterativo, esto no garantiza la convergencia. Cuando el número de clases es $m \geq 3$, también es posible aplicar el algoritmo IRLS, aunque ya no haya garantía de que la matriz de ponderación $\boldsymbol{W}$ sea diagonal [18], [30], [42], [15].

2.4. Kriging

El método *kriging* también conocido como *regresión de procesos Gaussianos*, fue desarrollado inicialmente por el ingeniero de minas sudafricano Danie G. Krige en un intento por pefilar las distribución del metal en un yacimiento de oro. Posteriormente, el matemático francés Georges F. Matheron formalizó la técnica de Krige, inaugurando así el término kriging en la literatura.

En líneas generales, podemos afirmar que la técnica se basa en el principio de que los puntos próximos en el espacio tienden a tener valores más parecidos que los puntos más distantes. Asume además que los datos recogidos de una determinada muestra se encuentran correlacionados en el espacio.

2.4.1. Introducción a los métodos kernel

Muchos modelos paramétricos no lineales, como las redes neuronales, usan durante la fase de aprendizaje un conjunto de datos de entrenamiento para obtener una estimación puntual del vector de parámetros o para determinar una distribución a posteriori basada en este vector. Posteriormente, los datos de entrenamiento

se descartan y las predicciones para las nuevas entradas se basan exclusivamente en el vector de parámetros aprendido.

Sin embargo, exite un tipo de técnicas de aprendizaje automático denominadas basadas en memoria o *memory-based* en el que los datos de entrenamiento o un subconjunto suyo, se conservan y usan durante la fase de predicción. Un ejemplo de esto, es el modelo de funciones de probabilidad de Parzen que comprende una combinación lineal de funciones *kernel* cada una de las cuales está centrada en un dato del conjunto de entrenamiento.

Para los modelos que contemplan aplicaciones $\boldsymbol{\phi} : \mathcal{X} \to \mathcal{V}, \boldsymbol{x} \mapsto \boldsymbol{\phi}(\boldsymbol{x})$, con $\mathcal{X}$ el espacio de entrada y $\mathcal{V}$ un espacio con producto interior, las funciones kernel pueden definirse por medio de la siguiente relación:

$$\begin{aligned} k(\boldsymbol{x}, \boldsymbol{x}') &= \langle \boldsymbol{\phi}(\boldsymbol{x}), \boldsymbol{\phi}(\boldsymbol{x}') \rangle = \boldsymbol{\phi}(\boldsymbol{x})^T \boldsymbol{\phi}(\boldsymbol{x}') = \sum_{i=1}^{m} \phi_i(\boldsymbol{x}) \phi_i(\boldsymbol{x}') \\ \textit{donde} \qquad & \boldsymbol{\phi}^T = (\phi_1, \ldots, \phi_m). \end{aligned} \tag{2.30}$$

De la definición anterior, es facil ver que las funciones kernel ($k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$) satisfacen la propiedad simetría $k(\boldsymbol{x}, \boldsymbol{x}') = k(\boldsymbol{x}', \boldsymbol{x})$. Además de las condiciones anteriores, es conveniente exigir que las funciones kernel sean no negativas, $k(\boldsymbol{x}, \boldsymbol{x}') \geq 0$. El concepto de kernel fue introducido en el campo del aprendizaje automático por Mark A. Aizerman en los años sesenta, pero tendrían que trancurrir treinta años más para que Bernhard E. Boser lo rescatara para dar fundamentos a la técnica de *support vector machines* (SVM). Desde entonces el interés por las funciones kernel ha crecido de forma espectacular, lográndose importantes avances tanto a nivel teórico como de las aplicaciones.

Existe una rica variedad de funciones kernel, empleadas en los más diversos contextos; sin embargo, el kernel más simple posible se deriva de considerar la aplicación identidad $\boldsymbol{\phi}(\boldsymbol{x}) = \boldsymbol{x}$, que nos conduce a:

$$k(\boldsymbol{x}, \boldsymbol{x}') = \boldsymbol{\phi}(\boldsymbol{x})^T \boldsymbol{\phi}(\boldsymbol{x}') = \boldsymbol{x}^T \boldsymbol{x}'. \tag{2.31}$$

Por norma general, dadas las funciones kernel válidas $k_1(\boldsymbol{x}, \boldsymbol{x}')$ y $k_2(\boldsymbol{x}, \boldsymbol{x}')$, las funciones $ck_1(\boldsymbol{x}, \boldsymbol{x}')$, $q(k_1(\boldsymbol{x}, \boldsymbol{x}'))$, $\exp\{k_1(\boldsymbol{x}, \boldsymbol{x}')\}$, $k_1(\boldsymbol{\phi}(\boldsymbol{x}), \boldsymbol{\phi}(\boldsymbol{x}'))$, $f(\boldsymbol{x}) k_1(\boldsymbol{x}, \boldsymbol{x}') f(\boldsymbol{x}')$, $\boldsymbol{x} A \boldsymbol{x}'$, $k_1(\boldsymbol{x}, \boldsymbol{x}') + k_2(\boldsymbol{x}, \boldsymbol{x}')$ y $k_1(\boldsymbol{x}, \boldsymbol{x}') \cdot k_2(\boldsymbol{x}, \boldsymbol{x}')$ son también funciones kernel válidas para $c \geq 0$ una constante, $q(\cdot)$ una función polinómica de coeficientes no negativos, $f(\cdot)$ una función arbitraria y A una matriz simétrica semidefinida positiva. Un tipo muy especial de funciones kernel, son los denominados *kerneles estacionarios* que invariantes bajo traslaciones dependen sólo de la diferencia de los argumentos de entrada, $k(\boldsymbol{x}, \boldsymbol{x}') = k(\boldsymbol{x} - \boldsymbol{x}')$. También son de gran interés los *kerneles homogéneos* que dependen de la distancia entre los argumentos de entrada, $k(\boldsymbol{x}, \boldsymbol{x}') = k(\|\boldsymbol{x} - \boldsymbol{x}'\|)$.

Un tipo de función kernel frecuentemente usada y que para la sección que nos ocupa tiene un atractivo especial, es el llamado *kernel gaussiano* cuya apariencia y nombre recuerdan a la aún más famosa función de densidad,

$$k(\boldsymbol{x}, \boldsymbol{x}') = \exp\{-\|\boldsymbol{x} - \boldsymbol{x}'\|^2 / 2\sigma^2\}. \tag{2.32}$$

Si desarrollamos el cuadrado de la norma que figura en el argumento de la exponencial anterior, tenemos lo siguiente:

$$\|\boldsymbol{x} - \boldsymbol{x'}\|^2 \quad = \quad \boldsymbol{x}^T\boldsymbol{x} - 2\boldsymbol{x}^T\boldsymbol{x'} + (\boldsymbol{x'})^T\boldsymbol{x'}. \tag{2.33}$$

Este resultado nos permite reescribir el kernel gaussiano en función del kernel lineal mónico de término independiente cero $k_0(\boldsymbol{x}, \boldsymbol{x'}) = \boldsymbol{x}^T\boldsymbol{x'}$, de la siguiente manera:

$$k(\boldsymbol{x}, \boldsymbol{x'}) \quad = \quad \exp\{-k_0(\boldsymbol{x}, \boldsymbol{x})/2\sigma^2\} \exp\{k_0(\boldsymbol{x}, \boldsymbol{x'})/\sigma^2\} \exp\{-k_0(\boldsymbol{x'}, \boldsymbol{x'})/2\sigma^2\}. \tag{2.34}$$

Con el ánimo de generalizar, si ahora partimos de una función kernel no lineal de base $\kappa(\boldsymbol{x}, \boldsymbol{x'})$, entonces el kernel gaussiano adoptará la siguiente forma:

$$k(\boldsymbol{x}, \boldsymbol{x'}) \quad = \quad \exp\left\{-\frac{1}{2\sigma^2}\left(\kappa(\boldsymbol{x}, \boldsymbol{x}) - 2\kappa(\boldsymbol{x}, \boldsymbol{x'}) + \kappa(\boldsymbol{x'}, \boldsymbol{x'})\right)\right\}. \tag{2.35}$$

Una estrategia eficaz para la construcción de kerneles, parte de utilizar un modelo generativo probabilístico $p(\boldsymbol{x})$ en un entorno discriminativo; este enfoque permite combinar las bondades de los modelos generativos y discriminativos. Así podemos definir un kernel como:

$$k(\boldsymbol{x}, \boldsymbol{x'}) \quad = \quad p(\boldsymbol{x})p(\boldsymbol{x'}). \tag{2.36}$$

Podemos extender el resultado anterior empleando el teorema de la probabilidad total y por tanto haciendo la suma de todos los productos posibles para una variable latente i de referencia,

$$k(\boldsymbol{x}, \boldsymbol{x'}) \quad = \quad \sum_i p(\boldsymbol{x} \mid i)\, p(\boldsymbol{x'} \mid i)\, p(i). \tag{2.37}$$

Si dos inputs $\boldsymbol{x}$ y $\boldsymbol{x'}$ dan un valor alto de la función kernel $k(\boldsymbol{x}, \boldsymbol{x'})$, entonces de las entradas se dice que son similares para un rango de componentes i de probabilidad significativa.

Ahora aplicando el límite al sumatorio anterior, conseguimos una función kernel determinada por una integral y con una variable latente continua denotada por $\boldsymbol{z}$,

$$k(\boldsymbol{x}, \boldsymbol{x'}) \quad = \quad \int_{\mathcal{X}} p(\boldsymbol{x} \mid \boldsymbol{z})\, p(\boldsymbol{x'} \mid \boldsymbol{z})\, p(\boldsymbol{z})\, d\boldsymbol{z}. \tag{2.38}$$

Una técnica alternativa a la construcción previa con modelos generativos usados como discriminativos, consiste en emplear los llamados kerneles de Fisher. Para conseguir este tipo de funciones kernel, partimos de un modelo paramétrico de distribución de los datos $p(\boldsymbol{x} \mid \boldsymbol{\theta})$, donde $\boldsymbol{\theta}$ representa el vector de parámetros. Teniendo presente el objetivo último que es medir el grado de similitud de los inputs $\boldsymbol{x}$ y $\boldsymbol{x'}$, determinamos a continuación la *función escrutadora* de Fisher $\boldsymbol{g}(\boldsymbol{\theta}, \boldsymbol{x}) = \nabla_{\boldsymbol{\theta}} \ln p(\boldsymbol{x} \mid \boldsymbol{\theta})$ y con ésta la matriz de información de Fisher $F = E_{\boldsymbol{x}}[\boldsymbol{g}(\boldsymbol{\theta}, \boldsymbol{x})\boldsymbol{g}(\boldsymbol{\theta}, \boldsymbol{x})^T]$. Con todos estos elementos, calculamos finalmente el kernel como:

$$k(\boldsymbol{x}, \boldsymbol{x'}) \quad = \quad \boldsymbol{g}(\boldsymbol{\theta}, \boldsymbol{x})^T F^{-1} \boldsymbol{g}(\boldsymbol{\theta}, \boldsymbol{x'}). \tag{2.39}$$

La presencia de la matriz de información de Fisher confiere al kernel una propiedad de invarianza bajo reparametrizaciones del modelo de densidad incluso si estas son no lineales ($\boldsymbol{\theta} \to \psi(\boldsymbol{\theta})$). En algunas aplicaciones resulta más apropiado emplear el kernel no invariante $k(\boldsymbol{x}, \boldsymbol{x'}) = \boldsymbol{g}(\boldsymbol{\theta}, \boldsymbol{x})^T \boldsymbol{g}(\boldsymbol{\theta}, \boldsymbol{x'})$ que omite esta matriz de información.

Finalmente, en muchos casos prácticos no es factible calcular la esperanza de la matriz de información de Fisher, por lo que se hace necesario realizar aproximaciones dentro de las cuales se destaca por su sencillez el promedio muestral: [2]

$$F \approx \frac{1}{N} \sum_{k=1}^{N} \boldsymbol{g}(\boldsymbol{\theta}, \boldsymbol{x}_n) \boldsymbol{g}(\boldsymbol{\theta}, \boldsymbol{x}_n)^T. \tag{2.40}$$

2.4.2. Regresión de procesos gaussianos

El objetivo último del enfoque de los *procesos gaussianos*, consiste en modelar la probabilidad a posteriori de la variable objetivo Y para un nuevo vector de entrada $\boldsymbol{X}$ y a partir de un conjunto de datos de entrenamiento dado $D = \{(\boldsymbol{x}_k, y_k)\}_{k=1}^N \subseteq \mathbb{R}^p \times \mathbb{R}$.

En el modelo $f(\boldsymbol{X}, \boldsymbol{\Omega}) = \boldsymbol{\Phi}^T(\boldsymbol{X}) \cdot \boldsymbol{\Omega}$ introducido en 2.6 y ampliamente discustido en la sección de regresión lineal, si incorporamos ahora una nueva hipótesis a partir de la cuál suponemos que la distribución del vector de parámetros es una gaussiana de media cero y varianza dependiente de la precisión α,

$$p(\boldsymbol{\Omega}) = N(0, \alpha^{-1} I_n). \tag{2.41}$$

Esta distribución inducirá otra sobre $f(\boldsymbol{X}, \boldsymbol{\Omega})$. Así dados el conjunto de datos D y un valor de $\boldsymbol{\Omega}$, la distribución conjunta correspondiente a las funciones de regresión, coleccionadas en el vector $\boldsymbol{f}^T = (f(\boldsymbol{x}_1, \boldsymbol{\Omega}), \ldots, f(\boldsymbol{x}_N, \boldsymbol{\Omega}))$, se caracterizará por los siguientes parámetros:

$$\begin{cases} E[\boldsymbol{f}] = \boldsymbol{\Psi} E[\boldsymbol{\Omega}] = 0 \\ Cov(\boldsymbol{f}) = E\left[\boldsymbol{f}\boldsymbol{f}^T\right] = \boldsymbol{\Psi} E\left[\boldsymbol{\Omega}\boldsymbol{\Omega}^T\right] \boldsymbol{\Psi}^T = \frac{1}{\alpha}\boldsymbol{\Psi}\boldsymbol{\Psi}^T \end{cases} \quad , \quad donde \quad \boldsymbol{\Psi} = (\phi_l(\boldsymbol{x}_k)). \tag{2.42}$$

Esta distribución igualmente gaussiana, a su vez implicará una distribución a posteriori predictiva $p(Y \mid f(\boldsymbol{X}, \boldsymbol{\Omega}))$ para cada nuevo valor del vector de entrada $\boldsymbol{X}$. En la expresión anterior, la covarianza realmente corresponde a la *matriz de Gram* $\boldsymbol{K}$, cuyas entradas se definen mediante la función kernel $k(\boldsymbol{x}, \boldsymbol{x'}) = \frac{1}{\alpha}\boldsymbol{\Phi}(\boldsymbol{x})^T \boldsymbol{\Phi}(\boldsymbol{x'})$, es decir:

$$K_{nm} = k(\boldsymbol{x}_n, \boldsymbol{x}_m) = \frac{1}{\alpha} \boldsymbol{\Phi}(\boldsymbol{x}_n)^T \boldsymbol{\Phi}(\boldsymbol{x}_m). \tag{2.43}$$

De forma general, un proceso gaussiano es un proceso estocástico $\{X_t \mid t \in T\}$ con $E[X_t] < \infty$ y en el $(X_{t_1}, \ldots, X_{t_n})$ tiene una distribución normal multivariante $\forall n \in \mathbb{N}$ y $\forall t_1, \ldots, t_n \in T$. En el contexto que nos ocupa, esto equivale a que $(f(\boldsymbol{x}_1, \boldsymbol{\Omega}), \ldots, f(\boldsymbol{x}_N, \boldsymbol{\Omega}))$ presenta una distribución de probabilidad conjunta de tipo gaussiana dado el conjunto de puntos $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$. Hablamos de *campo gaussiano*, cuando el vector de entrada $\boldsymbol{X}$ es bidimensional.

Un punto clave sobre los procesos estocásticos gaussianos es que la distribución conjunta sobre N variables se específica completamente mediante los estadísticos de segundo orden, es decir, la media y la covarianza. En la mayoría de las aplicaciones, no es posible disponer de información previa sobre la media, no obstante por simetría la consideramos igual a cero. La especificación del proceso gaussiano se completa, proporcionando la covarianza que viene dada por una función kernel.

En la perspectiva de los procesos gaussianos, prescindimos del procedimiento paramético anterior y, en su lugar definimos directamente una distribución de probabilidad a priori $p(Y \mid f(\boldsymbol{X}, \boldsymbol{\Omega}))$. También podemos definir directamente la función kernel que emplearemos, y es como no podría ser de otra forma el kernel gaussiano, $k(\boldsymbol{x}, \boldsymbol{x}') = \exp(-\theta|\boldsymbol{x} - \boldsymbol{x}'|)$.

Nuestro interés por la distribución conjunta de los valores y_k de entrenamiento condicionados a las predicciones correspondientes $f_k \equiv f(\boldsymbol{x}_k, \boldsymbol{\Omega}) = \boldsymbol{\Phi}^T(\boldsymbol{x}_k)\boldsymbol{\Omega}$, nos conduce entonces a una gaussiana isótropa que depende también de un hiperparámetro β, que representa la precisión del ruido,

$$\begin{array}{rcl} p(\boldsymbol{y} \mid \boldsymbol{f}) & = & N(\boldsymbol{y} \mid \boldsymbol{f}, \beta^{-1}\boldsymbol{I}_N) \\ \textit{donde} & & \left\{ \begin{array}{rcl} \boldsymbol{f}^T & = & (f_1, \ldots, f_N) \\ \boldsymbol{y}^T & = & (y_1, \ldots, y_N). \end{array} \right. \end{array} \tag{2.44}$$

En la expresión anterior I_N denota la matriz identidad de tamaño N. De la definición de proceso gaussiano, la distribución marginal de $\boldsymbol{f}$ es una gaussiana cuya media es cero y su covarianza está definida por una matriz de Gram $\boldsymbol{K}$, $p(\boldsymbol{f}) = N(0, \boldsymbol{K})$.

La función kernel que determina $\boldsymbol{K}$ se elige con la intención de expresar la propiedad de que si los puntos $\boldsymbol{x}_k$ y $\boldsymbol{x}_l$ se asemejan, entonces las correspondientes predicciones $f(\boldsymbol{x}_k, \boldsymbol{\Omega})$ y $f(\boldsymbol{x}_l, \boldsymbol{\Omega})$ estarán más estrechamente correlacionadas en contraste con puntos disímiles; por supuesto el concepto de similitud dependerá de la aplicación concreta.

Para hallar la distribución marginal $p(\boldsymbol{y})$, condicionada a los datos de entrada $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, necesitamos integrar sobre $\boldsymbol{f}$,

$$\begin{array}{rcl} p(\boldsymbol{y}) & = & \displaystyle\int p(\boldsymbol{y} \mid \boldsymbol{f}) p(\boldsymbol{f}) d\boldsymbol{f} = N(0, \boldsymbol{C}_N), \;\; C_N := (C_{kl}) \\ \textit{donde} & & C_{kl} \equiv Cov(\boldsymbol{x}_k, \boldsymbol{x}_l) = k(\boldsymbol{x}_k, \boldsymbol{x}_l) + \beta^{-1}\delta_{kl}, \;\; k, l \in \{1, \ldots, N\}. \end{array} \tag{2.45}$$

En la matriz de covarianza $\boldsymbol{C}_N$ de la expresión anterior, δ_{kl} es la delta de Dirac; ésta además refleja claramente las dos fuentes de aleatoriedad, a saber, la asociada con la elección de $f(\boldsymbol{X}, \boldsymbol{\Omega})$ y la asociada a ε.

Como hemos avanzado antes, nuestro objetivo es predecir la variable objetivo y_{N+1} para un nuevo vector de entrada x_{N+1}. Para hallar la distribución condicional $p(y_{N+1} \mid \boldsymbol{y})$, iniciamos por escribir la distribución conjunta $p(\boldsymbol{y}_{N+1})$, siendo $\boldsymbol{y}_{N+1}^T = (y_1, \ldots, y_N, y_{N+1})$ y aplicamos algunos resultados conocidos del cálculo de probabilidades, de modo que la distribución conjunta vendrá dada por:

$$\begin{array}{rcl} p(\boldsymbol{y}_{N+1}) & = & N(0, \boldsymbol{C}_{N+1}) \\ \textit{donde} & & \boldsymbol{C}_{N+1} = \begin{pmatrix} C_N & \boldsymbol{k} \\ \boldsymbol{k}^T & c \end{pmatrix} \in M_{(N+1)\times(N+1)}(\mathbb{R}), \\ \boldsymbol{k}^T = (k(\boldsymbol{x}_1, \boldsymbol{x}_{N+1}), \ldots, k(\boldsymbol{x}_N, \boldsymbol{x}_{N+1})) & , & c = k(\boldsymbol{x}_{N+1}, \boldsymbol{x}_{N+1}) + \beta^{-1}. \end{array} \tag{2.46}$$

Así pues, para una nueva observación $\boldsymbol{x}_{N+1}$, los parámetros que definen la distribución de la variable respuesta correspondiente y_{N+1}, $p(y_{N+1} \mid \boldsymbol{y})$, pueden estimarse como:

$$\begin{cases} m(\boldsymbol{x}_{N+1}) &= \boldsymbol{k}^T C_N^{-1} \boldsymbol{y} \\ \sigma^2(\boldsymbol{x}_{N+1}) &= c - \boldsymbol{k}^T C_N^{-1} \boldsymbol{k} \end{cases} . \tag{2.47}$$

Si observamos con detenimiento un deatlle del resultado anterior, vemos que tanto la media como la varianza de y_{N+1} dependen de la entrada $\boldsymbol{x}_{N+1}$, esto es un elemento diferenciador de la técnica de regresión de procesos gaussianos que más que outputs devuelve su distribución de probabilidad.

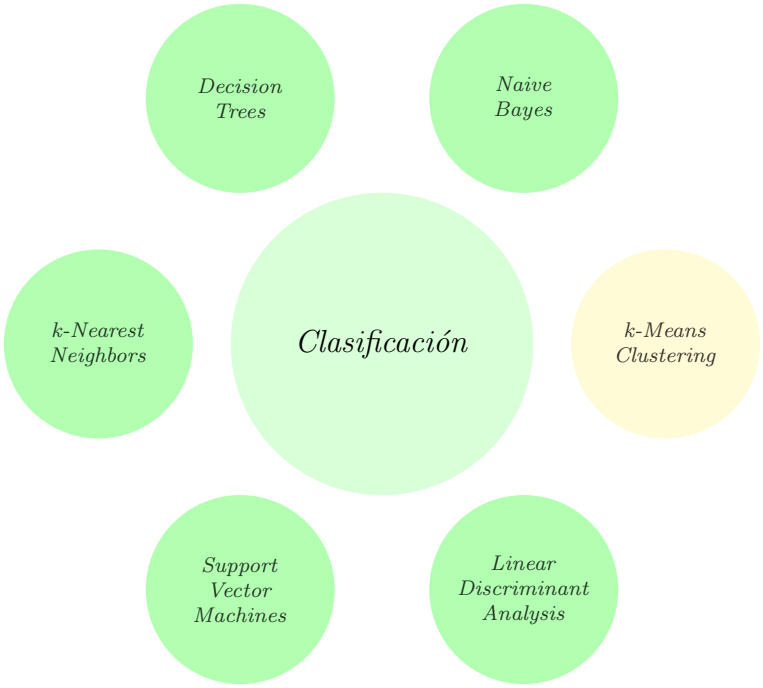
Desde el punto de vista computacional, utilizar procesos gaussianos implica invertir una matriz de tamaño $N \times N$, para la cual los métodos estándar requieren cálculos del órden $O(N^3)$. Si bien el proceso de inversión de la matriz debe realizarse una sóla vez para el conjunto de entrenamiento dado, la estimación de la complejidad no acaba allí, ya que para cada nuevo punto de prueba debe hacerse una multiplicación vector-matriz, cuyo coste es $O(N^2)$.

Hasta ahora sólo hemos considerado la regresión de procesos gaussianos para el caso de una sola variable objetivo, no obstante este formalismo puede extenderse con naturalidad a múltiples variables objetivo, en lo que se conoce como *co-kriging*. Otras extensiones populares de la técnica incluyen, el modelado de distribuciones en variedades de baja dimensión para aprendizaje no supervisado y la solución de ecuaciones diferenciales estocásticas.

Por último, si bien el modelo de regresión de procesos gaussianos realiza predicciones que se encuentran en todo el eje real, podemos realizar una adaptación de la técnica al problema clasificación, transformando la salida del proceso gaussiano mediante una función de activación no lineal adecuada [2], [48], [33], [43], [27], [38], [3].

Capítulo 3

Técnicas de Clasificación



Es frecuente encontrar referencias que presentan al problema de clasificación, como una variante del problema de regresión con la particularidad de que la naturaleza continua de la variable respuesta Y, pasa a ser discreta restringuiendo sus posibles valores a un conjunto finito.

La clasificación, emparentada con la agrupación en categorías o clústeres, son dos ejemplos del problema más general de *reconocimiento de patrones*, que se ocupa del descubrimiento de regularidades en los datos de entrada y en asignarles valores de salida. Uno de los primeros en incursionar en este tipo de técnicas fue Ronald A. Fisher, con su *función discriminante lineal* (FLD) que no es otra cosa que una regla binaria que asigna un grupo a cada observación.

Existen distintos enfoques que nos permiten construir clasificadores, sin embargo, los dos más empleados son el *generativo* y *discriminativo*. La lógica de la óptica generativa parte de estimar la distribución conjunta del modelo $P(X, Y)$ y luego calcular la porbabilidad condicional $P(Y \mid X)$ para clasificar. Mientras la perspectiva discriminativa basa la clasificación en el ajuste directo de la probabiliad condicional del modelo $P(Y \mid X)$.

En la práctica, se utilizan los diferentes enfoques según el problema específico, o se construyen sistemas híbridos que combinan las ventajas de éstos [29], [28], [23].

3.1. Naive Bayes

La técnica *Naive Bayes* forma parte de una familia más grande conocida como *Bayesian network* o redes bayesianas de la que podemos afirmar que su objetivo se centra en representar la distribución conjunta de un grupo de variables aleatorias. Aún en el caso más simple, donde las variables $X_1, \ldots, X_n$ toman valores binarios, la distribución conjunta requiere la especificación de $2^n - 1$ parámetros. Por otro lado, incluso para n pequeño, la representación explícita de la distribución conjunta es difícil manejar desde distintas perspectivas: computacionalmente es muy costosa de manipular y generalmente demasiado grande para almacenarse en la memoria; y estadísticamente si quisieramos construir la distribución a partir de los datos, requeriríamos que sus dimensiones fueran ridículamente bastas para estimar los parámetros de forma robusta. Es en este contexto, donde aproximaciones con hipótesis como la independencia de las variables adquieren relevancia en tanto que permiten representar la distribución de forma sencilla y compacta.

El modelo que plantea Naive Bayes, es justamente reconocido por la premisa de independencia (condicional) de la variables predictoras, de allí se deriva el calificativo de ingenuo o *naive*; pues si bien es una característica que le confiere una ventaja a nivel de interpretación, registringe también el alcance y la precisión en las aplicaciones.

Para introducir formalmente el método, partimos un resultado clásico del cálculo de probabilidades responsable parcial del nombre, hablamos del *teorema de Bayes*; según éste en un espacio de probabilidad $(\Omega, \mathfrak{S}, P)$, dados $B \subseteq \mathfrak{S}$ y $\{A_i\}_{i \in I}$ una partición del espacio muestral Ω, tenemos lo siguinete:

$$P(A_i \mid B) \quad = \quad \frac{P(A_i)\, P(B \mid A_i)}{\sum_{i \in I} P(A_i)\, P(B \mid A_i)}. \tag{3.1}$$

Este resultado trasladado al lenguaje de las variables aleatorias, se configura en la expresión que tenemos a continuación:

$$\begin{aligned} P(Y \mid X_1, \ldots, X_n) \quad &= \quad \frac{P(Y, X_1, \ldots, X_n)}{P(X_1, \ldots, X_n)} = \frac{P(\boldsymbol{X} \mid Y) P(Y)}{P(\boldsymbol{X})} \\ \textit{donde} \qquad & \boldsymbol{X}^T = (X_1, \ldots, X_n). \end{aligned} \tag{3.2}$$

Poniéndonos ya en el escenario en el que la variable respuesta Y toma valores en el conjunto $\{1, \ldots, m\}$, donde cada i representa una de las m clases diferentes; introducimos la hipótesis encargada de la mitad restante del nombre de la técnica. Por supuesto, nos referimos a la condición naive según la cuál las variables X_i son independientes, este hecho nos permite reescribir la probabilidad a posteriori de la siguiente manera:

$$\begin{aligned} P(Y \mid \boldsymbol{X}) \quad &= \quad \frac{P(Y)}{Z} \prod_{l=1}^{n} P(X_l \mid Y) \\ \textit{donde} \qquad & Z = P(\boldsymbol{X}) = \sum_{k=1}^{m} P(Y = k)\, P(\boldsymbol{X} \mid Y = k),\ k \in \{1, \ldots, m\}. \end{aligned} \tag{3.3}$$

En el problema de optimización que se presenta tras el último resultado, mientras la verosimilitud marginal $P(\boldsymbol{X})$ es una factor prescindible, la probabilidad a priori de clase $P(Y = k)$ puede calcularse empleando la regla de Laplace,

$$P(Y=k) \quad = \quad \frac{N_k}{N} \tag{3.4}$$
$$\textit{donde} \qquad \begin{cases} N_k := \textit{número de elementos de clase } k \textit{ en el conjunto de entrenamiento} \\ N := \textit{total de elementos en el conjunto de entrenamiento} \end{cases}.$$

El argumento de la técnica desemboca así en maximizar la probabilidad a posteriori, teniendo presente las reducciones de las consideraciones anteriores. El resultado será por tanto la clase que retorne el mayor valor del producto de la verosimilitud $P(\boldsymbol{X} \mid Y = y)$ y la probabilidad a priori $P(Y = y)$,

$$\hat{y} \quad = \quad \underset{y \in \{1,\ldots,m\}}{\operatorname{argmax}} \; P(Y=y) \prod_{l=1}^{n} P(X_l \mid Y = y). \tag{3.5}$$

La técnica concluye con la elección la elección de un modelo de verosimilitud, algo que por supuesto está sujeto a los casos casos concretos de aplicación.

Si bien son muchas las opciones disponibles entre las que se puede elegir la funcón de verosimiltud inmersa en el producto que buscamos hacer máximo, existen tres modelos que suelen imponerse, a saber:

1. **Multinomial (Multinomial NB):** en este tipo se asume que las variables de entrada provienen de distribuciones multinomiales. Esta variante resulta útil cuando se emplean datos discretos, y suele aplicarse para el procesamiento del lenguaje natural.

   Los parámetros característicos en este caso son $\boldsymbol{x}^T = (x_1, \ldots, x_n)$ y $\boldsymbol{p}^T = (p_1, \ldots, p_n)$, donde cada x_l cuenta el número de ocurrencias del evento l y p_l es su probabilidad; se satisface además que $\sum_{l=1}^{n} p_l = 1$;

$$P(\boldsymbol{X} = \boldsymbol{x} \mid Y = k) \quad = \quad \frac{(\sum_{l=1}^{n} x_l)!}{\prod_{l=1}^{n} x_l!} \prod_{l=1}^{n} p_{kl}^{x_l} \tag{3.6}$$
$$\textit{donde} \qquad p_{kl} = P(l \mid Y = k).$$

2. **Gaussiano (Gaussian NB):** como indica su nombre, se trata de una variante del clasificador naive Bayes, que utiliza distribuciones gaussianas. Este modelo se ajusta hallando la media μ_k y la varianza σ_k^2 de cada clase. Estos parámetros suelen estimarse segmentando el conjunto de entrenamiento en m clases, haciendo los respectivos cálculos:

$$P(\boldsymbol{X} = \boldsymbol{x} \mid Y = k) \quad = \quad \frac{1}{(2\pi)^{n/2} |\boldsymbol{\Sigma}|^{1/2}} \exp\left\{ -\frac{1}{2} (\boldsymbol{x} - \boldsymbol{\mu}_k)^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{x} - \boldsymbol{\mu}_k) \right\} \tag{3.7}$$
$$\textit{donde} \qquad \boldsymbol{\Sigma} = \begin{pmatrix} \sigma_1^2 & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & \sigma_m^2 \end{pmatrix}.$$

3. **Bernoulli (Bernoulli NB):** esta otra variante de la técnica de clasificación, se utiliza con variables booleanas, es decir, variables con dos valores, típcamente 0 y 1,

$$P(\boldsymbol{X} = \boldsymbol{x} \mid Y = k) \quad = \quad \prod_{l=1}^{n} p_{kl}^{x_l} (1 - p_{kl})^{(1 - x_l)} \tag{3.8}$$
$$\textit{donde} \qquad p_{kl} := \textit{probabilidad de que la clase } k \textit{ genere el término } x_l.$$

En el problema de la clasificación, una herramienta muy útil que permite visualizar el desempeño de un algoritmo es la *matriz de confusión*. Las columnas de esta matriz representan el número de predicciones de cada clase, mientras que las filas representan a las instancias en la clase real. La siguiente tabla corresponde a la matriz de confusión estandar para dos categorías etiquetadas como *positivo* y *negativo*.

Total de Casos $\boldsymbol{T = P + N}$		**Valor Predicho**	
		Positivo	**Negativo**
Valor Real	**Positivo**	*Verdadero positivo* (***VP***)	*Falso negativo* (***FN***)
	Negativo	*Falso positivo* (***FP***)	*Verdadero negativo* (***VN***)

Cuadro 3.1: Modelo de matriz de confusión binaria.

En la tabla anterior T es el número total de elementos clasificados, mientras que $P = VP + FN$ es el total de clasificados en la categoría positivo y $N = VN + FP$ el total de clasificados en la categroía negativo; siendo VP y VN son los totales de clasificados correctamente en positivo y negativo, y FN y FP los clasificados incorrectamente, respectivamente. Incluso disponiendo sólo de dos catergorías, es posible extraer númerosos indicadores que dan cuenta del rendimiento de un determinado algoritmo, no obstante elegimos destacar la *precisión*, uno de los más populares y en este constexto entendida como la suma del número de aciertos en cada categoría entre la suma de los respectivos totales, $(VP + VN)/(P + N)$. Las consideraciones que hemos hecho sobre la matriz de confusión anterior, son fácilmente extensibles para los casos en los que el número de clases es ≥ 3 [25], [30].

3.2. k-Nearest neighbors

El algoritmo *k-nearest neighbors* (k-NN) o en español de vecinos más próximos, se encuentra entre los más simples dentro del universo del aprendizaje automático. La idea central consiste en memorizar el conjunto de entrenamiento, para luego predecir la clase o valor de cualquier nueva entrada basándose en las etiquetas de sus vecinos más cercanos en dicho conjunto. De modo que la lógica de esta técnica parte de la suposición de que las características o posiciones utilizadas para describir los puntos del dominio son relevantes para sus etiquetados, hecho que hace probable que los puntos cercanos tengan la misma etiqueta.

Este es un método no paramétrico, utilizado tanto en problemas de clasificación como de regresión, el modelo en que se base almacena la información del conjunto de datos de entrenamiento, de modo que las predicciones se realizan directamente a partir de las instancias más cercanas en este espacio.

Partiendo de un conjunto de datos de entrenamiento, $D = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)\} \subseteq \mathbb{R}^n \times \mathcal{Y}$ donde $\mathcal{Y}$ es un conjunto finito para el problema de clasificación o bien es $\mathbb{R}$ para el de regresión; y dada una nueva observación $\boldsymbol{x} \in \mathbb{R}^n$, el algoritmo sigue los siguientes pasos:

1. Calculamos la distancia de $\boldsymbol{x}$ y cada punto $\boldsymbol{x}_i$.

2. Seleccionamos los k puntos del conjunto de entrenamiento más cercanos a $\boldsymbol{x}$ según las distancias calculadas. Este conjunto de estos vecinos y su correspondientes valores en $\mathcal{Y}$ lo denotamos por $D_k(\boldsymbol{x}) \subseteq D$.

3. Tratándose del problema de clasificación, asignamos a $\boldsymbol{x}$ la clase más frecuente o repetida entre sus k vecinos:

$$\hat{y} = \underset{c\in\mathcal{Y}}{\operatorname{argmax}} \sum_{(\boldsymbol{x}_i,y_i)\in D_k(\boldsymbol{x})} \chi_c(y_i) \tag{3.9}$$

$$\textit{donde} \qquad c \equiv \{c\}, \;\; \chi_A(z) = \begin{cases} 1, \; z \in A \\ 0, \; z \notin A \end{cases} \quad .$$

Para el problema de regresión, calculamos el valor promedio ponderado o simple de los valores de los vecinos cercanos:

$$\hat{y} = \frac{1}{k} \sum_{(\boldsymbol{x}_i,y_i)\in D_k(\boldsymbol{x})} y_i. \tag{3.10}$$

Un punto crucial de la técnica $k-$NN es la elección de la métrica o distancia, que da pie al concepto "cercanía". Y es que existen un rico abanico de funciones distancia a elegir, no obstante entrelas más usadas figuran:

1. La *distancia euclidiana* o d_2, de todas es la más utilizada. Dados dos puntos $\boldsymbol{r} = (r_1, \ldots, r_n)$ y $\boldsymbol{q} = (q_1, \ldots, q_n)$ de $\mathbb{R}^n$, esta métrica se define como:

$$d_2(\boldsymbol{r}, \boldsymbol{q}) = \sqrt{\sum_{i=1}^{n} |q_i - r_i|^2}. \tag{3.11}$$

2. La *distancia de Manhattan* o d_1 es la suma de las diferencias absolutas de sus coordenadas, es decir:

$$d_1(\boldsymbol{r}, \boldsymbol{q}) = \sum_{i=1}^{n} |q_i - r_i|. \tag{3.12}$$

3. La *distancia de Minkowski* o d_p, es una generalización de las distancias euclidiana y Manhattan,

$$d_p(\boldsymbol{r}, \boldsymbol{q}) = \left(\sum_{i=1}^{n} |q_i - r_i|^p \right)^{\frac{1}{p}}. \tag{3.13}$$

De donde podemos rederivar los casos particulares de:

- $p = 1$, es la distancia de Manhattan.
- $p = 2$, es la distancia euclidiana.
- $p \to \infty$, es la distancia de Chebyshev que no es otra cosa que la máxima diferencia entre cualquier coordenada,

$$d_\infty(\boldsymbol{r}, \boldsymbol{q}) = \max_i \{|q_i - r_i|\}. \tag{3.14}$$

La selección óptima del valor k a menudo se realiza mediante técnicas de validación cruzada, probando diferentes valores de k y seleccionando el que genera el mejor rendimiento. Una heurística común es elegir un k impar para problemas de clasificación binaria para evitar empates en la votación [18].

En la versión simple de $k-$NN, todos los k vecinos más cercanos contribuyen igualmente a la votación o al promedio. Sin embargo, se puede introducir una ponderación por distancia, donde determinados vecinos tienen una mayor influencia en la predicción que otros, según sea el caso de aplicación concreto [18], [42], [32], [41] [2].

3.3. Decision trees

Existen múltiples métodos basados en árboles que arrojan luz tanto sobre el problema de regresión como en el clasificación. Estos en general implican la segmentación del conjunto en secciones más simples para realizar la predicción dada una observación, tomando como referencia algunas medidas de tendencia central como la media o la moda. El nombre de árboles de decisión o en inglés *decision trees*, se deriva del hecho de que las reglas utilizadas para segmentar el espacio pueden dar lugar a una estructura tipo árbol.

Si bien los métodos basados en árboles son sencillos de implementar y fáciles de interpretar, no suelen competir con los mejores enfoques de aprendizaje supervisado, en términos de precisión de la predicción. Hay una serie de técnicas de refinamiento como el *bagging*, *random forests* y *boosting* que combinan un gran número de árboles, lo que a menudo puede resultar en mejoras significativas en la precisión de la predicción, a costa de cierta pérdida en la interpretación.

3.3.1. CART

Como hemos comentado antes, los árboles de decisión pueden aplicarse tanto a problemas de regresión como de clasificación. Autores notables como Leo Breiman desarrollaron el algoritmo *CART*, acrónimo en inglés de *Classification And Regression Trees*, que procede mediante la partición recursiva y binaria del espacio de entrada, para luego determinar una subpartición óptima que nos acerca a la predicción; por tanto, la construcción que propone esta técnica es un proceso en dos pasos.

La estructura gráfica del algoritmo CART es un árbol invertido, con la raíz en la cima, hojas o nodos terminales y ramas o nodos ramificados. Es justamente en los dos tipos de nodos existentes donde se pone de manifiesto el carácter binario; mientras los nodos ramificados se etiquetan con una condición, las hojas lo son con una clase o valor de la variable de respuesta.

Cuando se proporciona un árbol, es fácil usarlo para dar una predicción. De hecho, para determinar el valor de Y para una X dada, basta con recorrer el único camino desde la raíz hasta una hoja, respondiendo a la secuencia de preguntas dadas por las divisiones sucesivas [34].

Partiendo un conjunto de datos de entrenamiento, $D = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)\}, \subseteq \mathbb{R}^n \times \mathcal{Y}$, donde $\mathcal{Y}$ es o bien $\mathbb{R}$ para el problema de regresión o un subconjuto finito suyo para el problema de clasificación. El árbol se construye de forma recursiva, acatando los siguientes pasos.

1. Para cada nodo, consideramos todas las entradas $\boldsymbol{x}_k$ y todos los posibles puntos de corte s.

2. Dividimos el conjunto de datos en dos subconjuntos, a saber:

$$\begin{cases} D_1 & = \{(\boldsymbol{x}_k, y_i) \in D \mid x_k^j \leq s\} \\ D_2 & = \{(\boldsymbol{x}_k, y_i) \in D \mid x_k^j > s\} \end{cases}, \quad x_k^j, s \in \mathbb{R},\ j \in \{1, \ldots, n\}. \tag{3.15}$$

3. Calculamos la ganancia en pureza si se trata del problema de clasificación o la reducción del error en el caso de regresión. La impureza en la clasificación viene dada por el índice de Gini:

$$G(t) = 1 - \sum_{k=1}^{m} p_k^2 \tag{3.16}$$

$$\textit{donde} \qquad p_k = \frac{1}{|t|} \sum_{k=1}^{|t|} \chi_c(y_k).$$

p_k es la proporción de instancias de clase k en el nodo t. La ganancia de Gini al dividir un nodo es:

$$\Delta G = G(t) - \left(\frac{|D_1|}{|t|} G(D_1) + \frac{|D_2|}{|t|} G(D_2) \right). \tag{3.17}$$

Para la regresión, tenemos el error cuadrático medio:

$$MSE(t) = \frac{1}{|t|} \sum_{k=1}^{|t|} (y_k - \bar{y})^2 \tag{3.18}$$

$$\textit{donde} \qquad \bar{y} = \frac{1}{|t|} \sum_{k=1}^{|t|} y_k.$$

$\bar{y}$ es el promedio de las salidas en el nodo t. La ganancia en este caso, corresponde a:

$$\Delta\text{MSE} = \text{MSE}(t) - \left(\frac{|D_1|}{|t|} \text{MSE}(D_1) + \frac{|D_2|}{|t|} \text{MSE}(D_2) \right). \tag{3.19}$$

4. Eligimos la división que maximiza la mejora del criterio.

Para evitar el sobreajuste, el algoritmo CART aplica una técnica denominada *poda*, con la que se pretende eliminar ramas que aportan poca mejora, utilizando una medida de costo-complejidad:

$$R_\alpha(T) = R(T) + \alpha |T|. \tag{3.20}$$

Donde $R(T)$ es el error total del árbol, $|T|$ es el número de hojas y $\alpha \geq 0$ es un parámetro de regularización.

Para la fase de predicción, en el caso de clasificación un nuevo punto $\boldsymbol{x}$ se clasifica según la clase mayoritaria en la hoja donde cae. Mientras que para la regresión, la predicción es el promedio $\bar{y}$ de la hoja correspondiente [18], [15].

En esta subsección, hemos visto cómo construir un árbol de decisión para los problemas de regresión y clasificación. Si bien en muchas situaciones, estos árboles funcionan muy bien, existen propuestas alternativas que buscan obtener un mayor rendimiento con una arquitectura basada en estos árboles. Hablamos de un tipo de técnicas denominas *métodos de conjunto*, en el que se agregan los árboles y se ponderan sus resultados, ofreciendo así un rendimiento superior a costa de una mayor complejidad computacional y algorítmica.

En los métodos de conjunto, suele denominarse *aprendiz débil* a un solo árbol de decisión, que son las unidades de trabajo. Existen varios enfoques en la literatura que combinan con éxito múltiples aprendices débiles para crear un modelo sólido; la función de cada aprendiz débil del conjunto es captar ciertos aspectos de la información contenida en los datos utilizados para entrenarlo. El objetivo último de los métodos de conjunto es unir de forma óptima los aprendices débiles para lograr una mayor precisión y reducir así el sobreajuste [15].

3.3.2. Bagging

La palabra *bagging* es una contracción de *bootstrap aggregating*, nombre con el que también se conoce la técnica. La traducción más precisa de este vocablo sería "empaquetar", algo que casualmente se acerca a la esencia del método que puede resumirse en los siguientes pasos:

1. Dividir los datos de entrenamiento en un número predeterminado de conjuntos, con muestreo aleatorio con reemplazo, es decir que la misma muestra puede aparecer en múltiples conjuntos. Las muestras se denominan *bootstrap*.

2. Entrenar el árbol de decisión mediante el método CART o $ID3$ (Iterative Dichotomiser 3) [29] utilizando cada conjunto de datos. Cada árbol aprendido se denomina *aprendiz débil*.

3. Agregar todos los aprendices débiles promediando los resultados de cada aprendiz para el caso de regresión, y en el caso de clasificación votando. Este paso implica optimizar, de modo que se minimice el error de predicción.

4. El resultado tras agregar todos los aprendices débiles es el resultado final.

Una de las ventajas inherentes a este método, es la resiliencia con respecto a valores atípicos del conjunto de datos [24], [15], [5].

3.3.3. Random forest

La técnica *random forest* o *bosque aleatorio* va un paso más allá que el método bagging, haciendo que el agregado sea aún más resiliente ante variaciones importantes de lo datos. Incluso utilizando un conjunto de datos de entrada cuidadosamente diseñado, no todas las entradas tienen la misma influencia en el resultado; por no mencionar, que puede ocurrir que ciertos datos pueden tener interdependencias que pueden afectar su influencia en el resultado de forma contraproducente. La arquitectura del random forest mejora el rendimiento del modelo en estas situaciones, al particionar los datos de entrada para cada aprendiz débil. La consecuencia de esto es que cada aprendiz débil sólo ve una fracción de las muestras.

Como en la técnica bagging, las entradas también se muestrean aleatoriamente con reemplazo, en un proceso denominado *método de subespacio aleatorio* [21], [20], y es que cada aprendiz débil trabaja en un subespacio del conjunto de datos. La ventaja de esta estrategia de muestreo reside en que mejora la diversidad entre los árboles individuales y hace que el modelo sea más robusto y resiliente a datos ruidosos. Si bien el algoritmo original lo propuso Tin K. Ho, éste fue posteriormente modificado por Leo Breiman para fusionarlo con otros enfoques existentes [24], [16], [15], [18], [6], [7].

3.3.4. Boosting

La diferencia fundamental entre la técnica *boosting*, respecto del bagging y el random forest, reside en el entrenamiento secuencial de los árboles frente al entrenamiento paralelo. Mientras en los métodos bagging y random forest, los aprendices débiles son generados mediante muestreo aleatorio o subespacios aleatorios, y por su independencia, pueden entrenarse en paralelo; en la técnica boosting se emplea un enfoque diferente, entrenándose

un primer árbol a partir de una muestra aleatoria de datos, para el segundo árbol los datos utilizados dependen del resultado del entrenamiento del primero, y de esta manera continúa el entrenamiento con el tercer árbol y los sucesivos. Por lo tanto, en la lógica del boosting el entrenamiento de un árbol depende del entrenamiento de su predecesor inmediato en la secuencia contemplada.

Debido a que el computo no se realiza de forma paralela, el entrenamiento de los árboles es significativamente más lento que en los casos de bagging y random forest. Una vez se han entrenado todos los árboles, la salida de cada uno de ellos se combina con las ponderaciones adecuadas para generar el resultado final. Pese a la desventaja computacional que presenta la técnica, suele preferirse a otras debido al grado de precisión que suele exhibir [24], [15].

3.4. Support vector machine

En líneas generales podemos afirmar que la Máquina de vector soporte o en inglés *Support vector machine* (SVM), es una técnica que permiten clasificar datos mediante hiperplanos óptimos que maximizan la distancia entre clases. Esta estrategia de aprendizaje supervizado, inicialmente introducida por Vladimir N. Vapnik, Alexey Y. Chervonenkis y sus colaboradores, resulta muy potente a tal punto que en pocos años ha superado a la mayoría de los demás sistemas en una amplia variedad de aplicaciones.

Si bien el algoritmo SVM es capaz de manejar tareas de clasificación tanto lineales como no lineales, cuando los datos no son separables linealmente se utilizan funciones kernel para transformarlos a un espacio de mayor dimensión y permitir la separación lineal. La elección del tipo de función kernel dependerá de las características de los datos y de su propósito específico de uso.

Situándonos en el problema de clasificación binaria donde tenemos puntos de datos de dos clases diferentes, las SVM buscarán un hiperplano que divida las dos clases de manera que la distancia entre este hiperplano y los datos más cercanos de cada clase sea máxima. Estos puntos más cercanos se conocen como *vectores de soporte*, y la distancia a ellos *margen*. Esta forma de proceder es por supuesto extensible cuando contamos con un número de clases que es ≥ 3.

La importancia de un margen amplio radica en que éste hace que el clasificador sea más robusto a nuevas entradas que puedan estar cerca del límite de decisión.

Como hemos avanzado antes, podemos toparnos con uno de dos escenarios posibles, en los que bien o los datos pueden separarse linealmente o bien su complejidad. A continuación estudiamos por separado las generalidades de ambas opciones.

3.4.1. Clasificación linealmente separable

Nuevamente en el problema de clasificación binaria, sea el conjunto de entrenamiento $D = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_N, y_N)\}$, con $\boldsymbol{x}_k \in \mathbb{R}^n$ e $y_k \in \{-1, +1\}$. Un hiperplano en este contexto puede definirse por la siguiente ecuación:

$$\begin{aligned} \boldsymbol{\Omega}^T \boldsymbol{x} + b &= 0 \\ \textit{donde} \qquad & \begin{cases} \boldsymbol{\Omega} \in \mathbb{R}^n \\ b \in \mathbb{R} \end{cases} . \end{aligned} \tag{3.21}$$

El objetivo será por tanto encontrar el *vector de pesos* normal al hiperplano $\boldsymbol{\Omega}$ y el *término de sesgo* b, tales que el hiperplano separe correctamente las clases. Esto significa que para todos los puntos de la clase $+1$, $\boldsymbol{\Omega}^T \boldsymbol{x}_k + b > 0$, y para todos los puntos de la clase -1, $\boldsymbol{\Omega}^T \boldsymbol{x}_k + b < 0$. Estas condiciones pueden resumirse en una sola, como sigue:

$$y_k \left(\boldsymbol{\Omega}^T x_k + b\right) > 0, \ \ \forall k. \tag{3.22}$$

Partiendo de nuestro plano de decisión $H_0 : \boldsymbol{\Omega}^T \boldsymbol{x} + b = 0$, una forma práctica de definir el margen, es considerar dos hiperplanos paralelos a éste, que pasan por los vectores de soporte:

$$\begin{cases} H_1 : \boldsymbol{\Omega}^T \boldsymbol{x} + b = +1 \\ H_2 : \boldsymbol{\Omega}^T \boldsymbol{x} + b = -1 \end{cases} . \tag{3.23}$$

Mientra los datos pertenecientes a la clase $+1$ deben satisfacer $\boldsymbol{\Omega}^T \boldsymbol{x}_k + b \geq +1$, y los de la clase -1 satisfacen $\boldsymbol{\Omega}^T \boldsymbol{x}_k + b \leq -1$. Una vez más, combinando esto con las etiquetas de clase y_k, obtenemos la restricción:

$$y_k \left(\boldsymbol{\Omega}^T \boldsymbol{x}_k + b\right) \geq 1, \ \ \forall k. \tag{3.24}$$

La distancia entre los dos hiperplanos H_1 y H_2 da lugar al margen. Teniendo presente que la distancia entre H_1 y H_0 es $\frac{1}{||\boldsymbol{\Omega}||}$, y que la distancia entre H_2 y H_0 es $\frac{1}{||\boldsymbol{\Omega}||}$, el margen es igual a $\frac{2}{||\boldsymbol{\Omega}||}$.

Para maximizar este margen, necesitamos minimizar $||\boldsymbol{\Omega}||$, lo que es equivalente a minimizar $\frac{1}{2}||\boldsymbol{\Omega}||^2$. Luego la clasificación linealmente separable, se reduce al siguiente problema de optimización:

$$\begin{aligned} \text{mín :} \qquad & \tfrac{1}{2}\,||\boldsymbol{\Omega}||^2 \\ \text{sujeto a:} \quad & y_k \left(\boldsymbol{\Omega}^T \boldsymbol{x}_k + b\right) \geq 1 \quad \forall k = 1, \ldots, N. \end{aligned} \tag{3.25}$$

Este problema suele resolverse empleando los multiplicadores de Lagrange [2], [18].

3.4.2. Clasificación no linealmente separable

En la mayoría de los casos reales, los datos no son separables linealmente. Para manejar esto, los SVM introducen el concepto de *margen soft* mediante el uso de variables de holgura $\xi_k \geq 0$ para cada punto $\boldsymbol{x}_k$:

- Si $\xi_k = 0$, el punto está correctamente clasificado y fuera del margen.
- Si $0 < \xi_k < 1$, el punto está correctamente clasificado pero dentro del margen.
- Si $\xi_k \geq 1$, el punto está mal clasificado.

La restricción por tanto se modifica a:

$$y_k\,(\boldsymbol{\Omega}^T\boldsymbol{x}_k + b) \geq 1 - \xi_k \quad \forall k = 1, \ldots, N. \tag{3.26}$$

El objetivo ahora no sólo es maximizar el margen, sino también minimizar la suma de las violaciones del margen. Esto se logra añadiendo un término de penalización al problema de optimización:

$$\begin{aligned} \text{mín}: \quad & \tfrac{1}{2}||\boldsymbol{\Omega}||^2 + C\sum_{i=1}^{n}\xi_k \\ \text{Sujeto a:} \quad & y_k\,(\boldsymbol{\Omega}^T\boldsymbol{x}_k + b) \geq 1 - \xi_k \quad \forall k = 1, \ldots, N \\ & \xi_k \geq 0 \quad \forall i = 1, \ldots, N \\ & C > 0, \ (\textit{hiperparámetro de penalización}). \end{aligned} \tag{3.27}$$

Una de las características más llamativas de los SVM es su capacidad para manejar problemas de clasificación no linealmente separables usando las funciones kernel. La idea es transformar los datos del espacio original a un espacio de mayor dimensión donde puedan ser linealmente separables.

Matemáticamente, la solución al problema de optimización se puede dar en su forma dual. La forma dual involucra solo productos escalares entre los vectores de entrada: $\langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle = \boldsymbol{x}_i^T\boldsymbol{x}_j$.

Definiendo una aplicación $\boldsymbol{\Phi} : \mathbb{R}^n \to \mathbb{R}^d$ para $d > n$, que transforma los datos a un espacio de mayor dimensión, el hiperplano de decisión en este nuevo espacio sería:

$$\boldsymbol{\Omega}^T\boldsymbol{\Phi}(\boldsymbol{x}) + b = 0. \tag{3.28}$$

Determinar $\boldsymbol{\Phi}(\boldsymbol{x}_k)$ explícitamente puede ser complicado, además que hacer el producto escalar puede ser computacionalmente muy costoso o inviable si d es muy grande. Esto se resuelve introduciendo una función kernel, $k(\boldsymbol{x}_i, \boldsymbol{x}_j)$, que calcula el producto escalar en el espacio de alta dimensión sin necesidad de calcular explícitamente $\boldsymbol{\Phi}$. Esto eficienta el algoritmo, pues en el espacio de mayor dimensión un hiperplano lineal puede separar las clases [12], [11], [2], [18], [2].

3.5. Linear discriminant analysis

El análisis de discriminante lineal, conocido también como análisis discriminante normal (ADN), análisis de función discriminante (AFD) o más frecuentemente por las siglas en inglés LDA (*linear discriminant analysis*), está basado en el método del discriminante lineal de Fisher, desarrollado por Ronald A. Fisher en la década de 1930 y más tarde refinado por Calyampudi R. Rao. Esta técnica sigue un marco de modelo generativo, lo significa que se consideran las distribuciones de los datos en cada clase; la forma de proceder es identificar una combinación lineal de características de las distribuciones que permita generar una condición de separación de dos o más clases. También es habitual encontrar referencias al LDA como método de reducción de la dimensionalidad, pues los datos con dos o más dimensiones se proyectan a una sola dimensión, de manera que el proceso de clasificación

resulte más fácil.

Al igual que en otras técnicas de clasificación, el objetivo es determinar la probabilidad de clase a posteriori $P(Y \mid \boldsymbol{X})$ y a partir de allí deducir la clase con la probabilidad más alta para el conjunto de datos. Si partimos de una función de verosimilitud $f_k(\boldsymbol{x})$ que no es otra cosa que la densidad condicional de clase de $\boldsymbol{X}$ para $Y = k$, y una probabilidad a priori de clase π_k con $k \in \{1, \ldots, m\}$; entonces una simple aplicación del teorema Bayes nos conducirá a la siguiente relación:

$$P(Y = k \mid \boldsymbol{X} = \boldsymbol{x}) \quad = \quad \frac{f_k(\boldsymbol{x})\,\pi_k}{\sum_{l=1}^{m} f_l(\boldsymbol{x})\,\pi_l}, \;\; \boldsymbol{x}^T = (x_1, \ldots, x_n). \tag{3.29}$$

Lo que observamos a partir de la expresión anterior es que disponer de $f_k(\boldsymbol{x})$ es equivalente a contar con $P(Y \mid \boldsymbol{X})$. Concretamente, en la técnica que nos ocupa, así como en el análisis de discriminante cuadrático o *quadratic discriminant analysis* (QDA), emplean una verosimilitud gaussiana multivariante,

$$f_k(\boldsymbol{x}) \quad = \quad \frac{1}{(2\pi)^{n/2}\,|\boldsymbol{\Sigma_k}|^{1/2}} \exp\left\{-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu}_k)^T \boldsymbol{\Sigma_k}^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_k)\right\}. \tag{3.30}$$

$\boldsymbol{\mu}_k$ es la media de la clase k, adicionalmente en el LDA, asumimos que las m clases tienen una matriz de covarianza común $\boldsymbol{\Sigma}_k = \boldsymbol{\Sigma}$, $\forall k$. De la comparación de las probabilidades a posteriori de dos clases arbitrarias k y l tras integrar las consideraciones anteriores, extraemos la función protagonista del modelo y que es además responsable de su nombre, nos referimos a las funciones de discriminante lineal o *linear discriminant*:

$$\delta_k(\boldsymbol{x}) \quad = \quad \boldsymbol{x}^T \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_k - \frac{1}{2}\boldsymbol{\mu}_k^T \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_k + \log\{\pi_k\}. \tag{3.31}$$

Para dos clases k y l, el límite de decisión corresponde al conjunto $P(Y = k \mid \boldsymbol{X} = \boldsymbol{x}) = P(Y = l \mid \boldsymbol{X} = \boldsymbol{x})$, que como $\delta_k(\boldsymbol{x})$ es lineal en $\boldsymbol{x}$. No debemos perder de vista que en el fondo permanece el problema de optimización subyacente a todas las técnicas, en el caso que nos ocupa la función objetivo corresponde a la función discriminante lineal, $\hat{y} = \text{argmax}_{k \in \{1,\ldots,m\}}\, \delta_k(\boldsymbol{x})$.

En la práctica, como en otras ocasiones, estimamos los parámetros de las distribuciones gaussianas utilizando los datos de entrenamiento $D = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{N} \subseteq \mathbb{R}^n \times \mathbb{R}$, como sigue:

$$\begin{cases} \hat{\pi}_k = \frac{N_k}{N}, \; N_k := \textit{número de observaciones de clase } k \\ \hat{\boldsymbol{\mu}}_k = \sum_{g_i = k} \frac{\boldsymbol{x}_i}{N_k}, \; g_i \in \{1, \ldots, m\} \\ \hat{\boldsymbol{\Sigma}} = \sum_{k=1}^{m} \sum_{g_i = k} \frac{(\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_k)(\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_k)^T}{N - m}. \end{cases} \tag{3.32}$$

Para dos clases k y l, la regla de decisión lineal de fronteras se efectúa considerando la siguiente condición:

$$\boldsymbol{x}^T \hat{\boldsymbol{\Sigma}}^{-1}(\hat{\boldsymbol{\mu}}_k - \hat{\boldsymbol{\mu}}_l) \quad > \quad \frac{1}{2}(\hat{\boldsymbol{\mu}}_k + \hat{\boldsymbol{\mu}}_l)^T \hat{\boldsymbol{\Sigma}}^{-1}(\hat{\boldsymbol{\mu}}_k - \hat{\boldsymbol{\mu}}_l) - \log\left\{\frac{N_k}{N_l}\right\}. \tag{3.33}$$

Si se cumple esta condición, el punto $\boldsymbol{x}$ pertenecerá a la clase k, de otro modo éste pertenecerá a la clase l.

En el problema general del discriminante, si no se suponemos que $\boldsymbol{\Sigma}_k$ sea igual para todas las clases, entonces aparecen dependencias cuadráticas de $\boldsymbol{x}$, lo que configura una función que llamamos *discriminante cuadrático* vinculada al QDA:

$$\delta_k(\boldsymbol{x}) \quad = \quad -\frac{1}{2}\log|\boldsymbol{\Sigma}_k| - \frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_k)^T\boldsymbol{\Sigma}_k^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_k) + \log\{\pi_k\}. \tag{3.34}$$

Las diferencias entre el QDA y el LDA son generalmente pequeñas, no obstante, aún cuando suele preferirse el QDA, éste requiere estimar $(m-1)\times(n(n+3)/2+1)$ parámetros; mientras que en el LDA sólo hacen falta $(m-1)\times(n+1)$ [18].

3.6. k-Means clustering

El *clustering* o el *análisis clustering* comprende un amplio abanico de modelos y métodos asociados cuyo objetivo central consiste en descubrir un número limitado de grupos o clústeres significativos en los datos disponible. Concretamente, dado un conjunto de datos de tamaño N, cosntituido por puntos en los que se observan n variables, el objetivo tradicional del clustering es identificar $k \ll N$ grupos tales que:

1. Cada grupo debe contener al menos una unidad de dato.

2. Los grupos son disjuntos.

A partir de aquí se puede definir una partición del conjunto de N datos en k grupos. Dado que es posible encontrar una gran cantidad de particiones, queda por explicar el sentido del adjetivo significativo. Un grupo es significativo si las unidades de datos de ese grupo son similares entre sí y diferentes de las unidades de otros grupos. El concepto de similitud o disimilitud entre unidades es fácil de entender, en la medida que todos podemos evaluar intuitivamente si dos unidades son similares o diferentes.

Una estrategia popular para la agrupación en clústeres, comienza definiendo una función de coste sobre un conjunto parametrizado de posibles agrupaciones. El objetivo por tanto del algoritmo de agrupación en clústeres es encontrar una partición de coste mínimo. Bajo este paradigma, la tarea de agrupación en clústeres se convierte en un problema de optimización. La función objetivo es una aplicación que tiene por entradas un conjunto de datos $D \subseteq \mathbb{R}^n$, y que retorna como solución de agrupación $\{C_1, \ldots, C_k\}$.

Cuando hablamos de la técnica de k-means clustering, en realidad estamos haciendo referencia a una familia de métodos basados en diferentes elecciones de la función objetivo. En la presente sección expondremos el caso concreto de una función que se suele denotar por $G_{k-means}$.

La premisa central de este algoritmo ampliamente utilizado en minería de datos, es que cada observación pertenece al grupo con el centroide más cercano. De modo que dado el conjunto de datos $D = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$ con $\boldsymbol{x}_m \in \mathbb{R}^n$, el objetivo del algoritmo *k-means* es dividir los datos en k grupos disjuntos $C_1, \ldots, C_k$ de tal forma que se minimice la suma de los errores cuadráticos internos, es decir las distancias cuadradas a los centroides de cada clase, $\boldsymbol{\mu}_m$,

$$\begin{aligned} G_{k-means} \quad &= \quad \sum_{m=1}^{k}\sum_{\boldsymbol{x}\in C_m}\|\boldsymbol{x}-\boldsymbol{\mu}_m\|^2 \\ \textit{donde} \qquad &\boldsymbol{\mu}_m = \frac{1}{|C_m|}\sum_{\boldsymbol{x}\in C_m}\boldsymbol{x}. \end{aligned} \tag{3.35}$$

El algoritmo puede inicializarse, elegiendo aleatoriamente k puntos como centroides, $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k$. A continuación, se asigna cada punto $\boldsymbol{x}_m$ al cluster cuyo centroide esté más cercano, lo que es equivalente a ir construyendo los clústeres a partir de los datos de D y sabiendo que éstos se definen como:

$$C_l = \left\{ x_m \in D \mid l = \underset{j \in \{1,\ldots,k\}}{\operatorname{argmin}} \|\boldsymbol{x}_m - \boldsymbol{\mu}_j\|,\ \forall m = 1, \ldots, N \right\}. \tag{3.36}$$

En el paso de actualización de la información, se recalculan los centroides otra vez como la media de los puntos ya asignados a cada cluster. Finalmente, se repiten los últimos pasos hasta que las asignaciones no cambien o el valor de $G_{k-means}$ converja.

Este algoritmo siempre converge en un número finito de pasos, sin embargo la solución óptima puede ser un mínimo local, por lo que suele ejecutarse varias veces con diferentes valores iniciales para los $\boldsymbol{\mu}_m$.

Tradicionalmente, la complejidad de este algoritmo se estima como $O(N \cdot k \cdot n \cdot t)$, siendo t el número total de iteraciones.

La selección del parámetro k, puede hacer usando distintos criterios, entre los más usados encontramos: el *método del codo* o *elbow method* que consiste en graficar el valor de $G_{k-means}$ en función del número de clústeres k, eligiendo el valor de este último como punto donde la disminución de $G_{k-means}$ se hace menos pronunciada; y el *coeficiente de silueta* o *silhouette* que mide lo bien que se adapta cada punto a su propio cluster en comparación con los demás [22], [17], [42].

Capítulo 4

Aplicaciones

Las bases de datos que hemos empleado proceden de dos repositorio independientes y de libre acceso en *Kaggle*. Estos datasets meticulosamente identificados para un tipo concreto de problema de aprendizaje automático, serán sometidos a un análisis que emplea o bien tres algoritmos regresión o bien tres algoritmos de clasificación clasificación, según sea su naturaleza.

4.1. Regresión

El dataset que aquí empleamos proviene de un conjunto de datos publicados originalmente por la Comisión de Taxis y Limusinas de Nueva York (TLC) y posteriormente disponibilizado en Kaggle (New York City Taxi Trip Duration) para una competencia cuyo objetivo era construir un modelo que permitiera predecir la duración total de los viajes en taxi en la ciudad de Nueva York.

Los datos están segmentados en dos archivos, de los cuales el primer está compuesto por 1 458 644 registros en formato *csv* y constituye el conjunto de entrenamiento, mientras el segundo corresponde al conjunto de prueba que consta de 625 134 entradas también en formato *csv*.

Las cuatro técnicas que hemos considerado oportuno probar para la base de datos elegida, son: regresión lineal, regresión ridge, regresión LASSO y regresión elastic net. Empleamos la herramienta PYTHON para implementar y ejecutar las rutinas de los algoritmos comentados, los resultados obtenidos se agrupan en el siguiente cuadro.

Técnica	RMSLE (%)
linear regression	61,4
ridge regression	61,4
LASSO regression	62,9
elastic net regression	62,4

Cuadro 4.1: Comparativa de ténicas de regresión.

La presión ha sido evaluada empleando el *root mean squared logarithmic error* (RMSLE), al igual que en la competición,

$$\varepsilon = \sqrt{\frac{1}{N}\sum_{k=1}^{N}\left(\log\{p_k+1\}-\log\{a_k+1\}\right)^2} \tag{4.1}$$

$$\textit{donde} \quad \begin{cases} p_k := \textit{predicción de la duración del viaje} \\ a_k := \textit{duración real del viaje.} \end{cases}$$

De los resultados obtenidos, nos llama la atención que las predicciones no mejoran con la regularización ridge, respecto de la regresión lineal. También es llamativo el hecho de que el desempeño de elastic net sea ligeramente inferior al de LASSO; esto podría atribuirse al peso que tiene que el método ridge subsumido en el elastic net.

Los scripts empleados pueden consultarse también en la página web kaggle usando siguiente enlace: Técnicas selectas de regresión.

4.2. Clasificación

El origen de datos en este caso es el dataset MNIST, ampliamente utilizado por investigadores y estudiantes para reconocimiento de patrones y aprendizaje automático. La base de datos MNIST se derivó de datos originales del *National Institute of Standards and Technology* (NIST) y fue disponibilizada en Kaggle (MNIST Dataset) gracias a Yann A. LeCun Chief AI Scientist en Meta.

Los datos están divididos en dos grupos, los de entrenamiento constituidos por 60 000 muestras y los de prueba por 10 000; dichas muestras no son otra cosa que dígitos manuscritos normalizados por tamaño.

Una aclaración adicional que juzgamos conveniente incluir, es que hemos decidido emplear la misma base de datos tanto para los algoritmos de clasificación seleccionados de aprendizaje supervisado y el único algoritmo de clasificación de aprendizaje no supervisado que hemos abordado. El objetivo último es comparar el desempeño de ambos tipos de técnicas.

Las tres técnicas que hemos estimado adecuado testar para la base de datos selccionada, son: k-nearest neighbours, support vector machine y k-means clustering.

Utilizamos la herramienta Python para implementar y ejecutar las rutinas de los algoritmos mencionados, los resultados obtenidos se alojan en la siguiente tabla.

Técnica	k	Precisión (%)
k-nearest neighbours	1	95
	3	94,2
	5	93,5
	7	93
	9	92,4
SVM	–	95,1
k-means clustering	10	57,8

Cuadro 4.2: Comparativa de ténicas de clasificación.

Resulta claro que de todas, la técnica que mejor desempeño ha tenido es la SVM, esto por supuesto era de esperarse dada la potencia de cálculo y la robustez que la caracterizan. Sin embargo, un hecho que puede ser llamativo es que incluso aumentando el número de vecinos cercanos, los resultados exhibidos por k-NN y k-means

clustering, apenas son comparables, siendo la precisión del método de aprendizaje no supervisado muy inferior al de las demás. Un detalle que no podemos dejar pasar, es la forma en que hemos calculado la precisión de k-means clustering, ésta se basa en una comparación de las predicciones con registros del conjunto de entrenamiento.

Los scripts empleados pueden consultarse también en la página web kaggle usando siguiente enlace: Técnicas selectas de clasificación.

4.3. Novedades y desafíos

El objetivo común de las disciplinas científicas es comprender las relaciones entre las cantidades observables y construir modelos que las codifiquen. Eventualmente, los resultados de cualquier modelo y las hipótesis que lo sustenta, deben contrastarse con observaciones de acuerdo con el célebre criterio de falsabilidad de Karl R. Popper; por tanto, los experimentos, las mediciones, las observaciones, en una palabra, los datos, desempeñan un papel fundamental en el quehacer científico.

En la última década, si bien existe una tendencia, sobre todo del mundo editorial, a inflar palabras de moda, es innegable que la cantidad sin precedentes de datos recopilados de casi cualquier tipo (preferencias de compra de clientes, registros de salud, colisiones de partículas de alta energía, resultados de simulaciones de superordenadores, lecturas meteorológicas, etc), hace que la época en la que vivimos sea única en la historia. Por supuesto, la disciplina que más se ha beneficiado con la explosión de la revolución de los datos es el aprendizaje automático, campo tradicionalmente considerado como un subconjunto de la inteligencia artificial.

El aprendizaje automático se encuentra hoy en apogeo, por el simple hecho de que los métodos, algoritmos y la tecnología, estudiados y diseñaados durante las últimas dos décadas, han comenzado a producir resultados inesperadamente buenos, gracias a la combinación única de disponibilidad de big data y potencia de cálculo.

Un campo que nos interesa especialmente es la física, en este ámbito el desarrollo de un modelo, a menudo se basa en el popular principio de la *navaja de Ockham* o *principio de parsimonia*, según el cuál se da preferencia al modelo más simple que pueda explicar los datos. Una consecuencia importante de esta filosofía, es que los pasos del proceso que conduce al desarrollo de la mayoría de modelos físicos son completamente inteligibles para la mente humana. Esta clase de modelos se conocen como *modelos de caja blanca*, lo que sugiere que cada componente incluidas las hipótesis son transparentes. A pesar de sus extraordinarios logros, el cerebro humano tiene una capacidad limitada para procesar datos, especialmente en grandes proporciones; es por tanto, que las relaciones entre las cantidades observables codificadas en los modelos de física de caja blanca no suelen explorar espacios de alta dimensión. Esta limitación no se traduce en un caracter "simple"; al contrario estos modelos pueden ser bastante complejos, requiriendo en ocasiones métodos numéricos avanzados para producir resultados comparables con las observaciones.

Por su parte, los métodos de aprendizaje automático suelen denominarse de *caja negra*, lo que significa que la estructura matemática subyacente y las relaciones entre variables son tan complejas que a menudo no resulta útil intentar comprenderlas, siempre que ofrezcan los resultados esperados. A diferencia de la forma de operar de los modelos de caja blanca, el machine learning se centra esencialmente en dos principios, la precisión y la robustez frente a nuevos datos, que se procuran mantener equilibrio para evitar el sobreajuste.

Cabe mencionar que existe un camino intermedio surgido recientemente y materializado en un tercer paradigma, llamado *modelos de caja gris*. La idea central de esta novedosa propuesta consiste en emplear modelos de

física reducida y calibrar los parámetros libres mediante técnicas de aprendizaje automático. A largo plazo, la esperanza que se alberga en este tipo de modelos, es que recojan las virtudes de preisión de los modelos de caja blanca y de rapidez de los caja negra, evitando al mismo tiempo sus respectivas limitaciones.

En los últimos cinco años la gran novedad sobre todo en medios no especializados, han sido las redes neuronales multicapa, mejor conocidas como *deep learning* o *aprendizaje profundo*. Esta potente herramienta es la tecnología detrás de los recientes éxitos en el reconocimiento de voz e imágenes, y del primer ordenador capaz de derrotar a un campeón mundial en el juego de Go. Si bien los medios de comunicación suelen centrarse en las aplicaciones más cotidianas del aprendizaje automático, como pueden ser los coches autónomos, la detección de fraudes en línea, las recomendaciones personalizadas y la traducción en tiempo real; nosotros creemos que es válido preguntarse si el machine learning podría cambiar el proceso de descubrimiento científico.

Resulta incontrovertible que existen numerosos desafíos en los distintos frentes de acción donde ya tiene cabida el aprendizaje automático, por no mencionar que día tras día son más las áreas técnicas y científicas que se rinden ante los encantos del panorama prometedor que advierten los métodos de machine learning para el futuro. Sin embargo, apelando nuevamente a nuestra inclinación hacia las ciencias físicas, nos gustaría destacar el rol fundamental del aprendizaje automático en el enfoque de modelo de caja gris, aplicado con éxito en un campo tan determinante como el *space weather*. Y es que son muchos los problemas de esta rama de la astrofísica, los que han encontrado soluciones satisfactorias empleando este tipo de técnicas.
Los escenarios donde el valor de las predicciones de los algoritmos de machine learning se hace evidente, son tan variados como complejos. Entre los más recurrentes, podemos destacar: valores de índices geomagnéticos, número de manchas solares, ocurrencia de erupciones solares, tiempo de propagación de un eyección de masa coronal, velocidad del viento solar y el flujo de partículas solares. Si bien estas magnitudes se sitúan más dentro de los límites del problema de regresión, encontramos también circunstancias que dan pie a la clasificación, como es el caso de: los distintos grupos de manchas solares y los tipos de viento solar de acuerdo con su origen [31], [44].

El interés concreto por estas aplicaciones, no es fortuito. Y es que existe evidencia sólida del poder destructivo de una amplia gama de fenómenos de meteorología espacial, que afectan de forma especial la infraestructura eléctrica y la tecnología satelital. Como consecuencia de esto, distintas publicaciones de análisis de riesgo han estimado que el impacto de un suceso extremo de esta índole, podría comprometer seriamente el estilo de vida moderno, altamente dependiente de esta red de servicios.

Adicionalmente, existen numerosos estudios que apuntan a la capacidad que tienen estos eventos de dañar gravemente la biología de las personas, bajo su rango de alcance. Por todo esto, consideramos justificada la afirmación de que las aplicaciones enfocadas al space weather deben figurar entre las prioridades a valorar por el aprendizaje automático [9], [8].

Capítulo 5

Conclusiones

Son numerosas las lecciones aprendidas a lo largo de todo el proceso que integra esta empresa que nos hemos propuesto llevar a cabo y que aquí se resume. Por supuesto, la presente memoria se queda corta al momento de valorar las distintas dinámicas de aprendizaje en las que me vi envuelto en este intento por sondear el campo de machine learning y sus aplicaciones; no obstante, estoy convencido que las diferentes actividades desarrolladas en este trabajo, que comprendieron, pero no se limitaron a la búsqueda de referencias bibliográficas como libros y artículos, la consulta de páginas web especializadas y el aprovechamiento de material disponibilizado por algunos de los exponentes más representativos del campo, han contribuido significativamente a mi crecimiento profesional.

Con el ánimo de sintetizar, identificamos dos puntos centrales a destacar que recogen la escencia de este proyecto y de los conocimientos adquiridos en su ejecución, estos son:

1. La profundización en algunas de las técnicas más distinguidas del aprendizaje supervisado y en una de las más conocidas de aprendizaje no supervisado.

   Además de haber tenido ocasión de indagar sobre los matices que enriquecen a cada método, tuve la oportunidad de poner a prueba una selección de estos, usando dos conjuntos de datos cuidadosamente identificados para los problemas de regresión y clasificación. Esta experiencia, supuso un paso crucial en el refuerzo de los conocimientos adquiridos.

2. La inmersión en el universo de las aplicaciones del aprendizaje automático, me permitió conocer el auténtico potencial transformador de esta herramienta, trascendiendo a las visiones parcializadas e incompletas de los medios de comunicación. A partir de esta probada capacidad revolucionaria, intuyo el enorme impacto que puede tener en disciplinas como la física, a tal punto que su irrupción puede suponer un punto de inflexión en la noble misión que históricamente se ha atribuido esta ciencia de descifrar los misterios del universo.

Bibliografía

[1] E. Alpaydin. *Introduction to machine learning.* MIT press, 2020.

[2] C. M. Bishop. Pattern recognition and machine learning, 2006.

[3] A. W. Bowman and A. Azzalini. *Applied smoothing techniques for data analysis: the kernel approach with S-Plus illustrations.* Clarendon Press-Oxford, 1997.

[4] U. Braga-Neto. *Fundamentals of pattern recognition and machine learning.* Springer, 2020.

[5] L. Breiman. Bagging predictors. *Machine learning*, 24:123–140, 1996.

[6] L. Breiman. Random forests. *Machine learning*, 45:5–32, 2001.

[7] L. Breiman. Consistency for a simple model of random forests. *University of California at Berkeley. Technical Report*, 670, 2004.

[8] E. Camporeale. The challenge of machine learning in space weather: Nowcasting and forecasting. *Space weather*, 17(8):1166–1207, 2019.

[9] E. Camporeale, S. Wing, and J. R. Johnson, editors. *Machine Learning Techniques for Space Weather.* Elsevier, 2018.

[10] F. Chollet. *Deep learning with Python.* Simon and Schuster, 2021.

[11] A. Christmann and I. Steinwart. *Support vector machines.* Springer, 2008.

[12] N. Cristianini and J. Shawe-Taylor. *An introduction to support vector machines and other kernel-based learning methods.* Cambridge university press, 2000.

[13] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. *The Annals of Statistics*, 32(2):407–499, 2004.

[14] D. A. Freedman. *Statistical models: theory and practice.* cambridge university press, 2009.

[15] J. Gareth, W. Daniela, H. Trevor, and T. Robert. *An introduction to statistical learning: with applications in R.* Spinger, 2013.

[16] R. Genuer and J.-M. Poggi. *Random forests with R.* Springer, 2020.

[17] P. Giordani, M. Brigida Ferraro, and F. Martella. *An Introduction to Clustering with R.* Springer, 2020.

[18] T. Hastie, R. Tibshirani, and J. Friedman. The elements of statistical learning: data mining, inference, and prediction, 2009.

[19] J. A. Hertz. *Introduction to the theory of neural computation*. Crc Press, 2018.

[20] T. K. Ho. Random decision forests. In *Proceedings of 3rd International Conference on Document Analysis and Recognition*, volume 1, pages 278–282 vol.1, 1995.

[21] T. K. Ho. The random subspace method for constructing decision forests. *IEEE transactions on pattern analysis and machine intelligence*, 20(8):832–844, 1998.

[22] A. K. Jain and R. C. Dubes. *Algorithms for clustering data*. Prentice-Hall, Inc., 1988.

[23] T. Jebara. *Machine learning: discriminative and generative*. Springer Science & Business Media, 2012.

[24] A. V. Joshi. Machine learning and artificial intelligence, 2023.

[25] D. Koller and N. Friedman. Probabilistic graphical models: Principles and techniques, 2009.

[26] R. Kruse, C. Borgelt, C. Braune, S. Mostaghim, M. Steinbrecher, F. Klawonn, and C. Moewes. *Computational intelligence*. Springer, 2011.

[27] W. Liu, J. C. Principe, and S. Haykin. *Kernel adaptive filtering: a comprehensive introduction*. John Wiley & Sons, 2011.

[28] R. S. Michalski, J. G. Carbonell, and T. M. Mitchell. *Machine Learning: An Artificial Intelligence Approach (Volume I)*, volume 1. Elsevier, 2014.

[29] T. Mitchell. Machine learning. *Publisher: McGraw Hill*, 1997.

[30] K. P. Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.

[31] M. Nair, R. Redmon, L.-Y. Young, A. Chulliat, B. Trotta, C. Chung, G. Lipstein, and I. Slavitt. Magnet—a data-science competition to predict disturbance storm-time index (dst) from solar wind data. *Space Weather*, 21(10):e2023SW003514, 2023.

[32] N. J. Nilsson. *Introduction to machine learning*. Department of Computer Sciences, Stanford University, 1996.

[33] W. H. Press. *Numerical recipes 3rd edition: The art of scientific computing*. Cambridge university press, 2007.

[34] J. R. Quinlan. Induction of decision trees. *Machine learning*, 1:81–106, 1986.

[35] C. R. Rao, Shalabh, H. Toutenburg, and C. Heumann. *Linear models and generalizations*. Springer, 2008.

[36] V. Roth and T. Vetter. *Pattern Recognition: 39th German Conference, GCPR 2017, Basel, Switzerland, September 12–15, 2017, Proceedings*, volume 10496. Springer, 2017.

[37] S. J. Russell and P. Norvig. *Artificial intelligence: a modern approach*. Pearson, 2016.

[38] B. Scholkopf and A. J. Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2018.

[39] G. Seber and C. Wild. *Nonlinear Regression*. Wiley Series in Probability and Statistics. Wiley, 2005.

[40] G. A. Seber. *The linear model and hypothesis*. Springer, 2015.

[41] G. Shakhnarovich, T. Darrell, and P. Indyk. *Nearest-Neighbors Methods in Learning and Vision. Theory and Practice*. The MIT Press, 2006.

[42] S. Shalev-Shwartz and S. Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

[43] J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

[44] F. Siciliano, G. Consolini, R. Tozzi, M. Gentili, F. Giannattasio, and P. De Michelis. Forecasting sym-h index: A comparison between long short-term memory and convolutional neural networks. *Space Weather*, 19(2):e2020SW002589, 2021.

[45] S. Theodoridis and K. Koutroumbas. *Pattern recognition*. Elsevier, 2006.

[46] B. Unhelker, H. M. Pandey, and G. Raj. *Applications of Artificial Intelligence and Machine Learning*. Springer, 2021.

[47] T. Weise. *Global optimization algorithms-theory and application*. Self-Published Thomas Weise `http://www.it-weise.de/`, 2009.

[48] C. K. Williams and C. E. Rasmussen. *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA, 2006.

[49] X. Yan and X. Su. *Linear regression analysis: theory and computing*. world scientific, 2009.

UNIVERSIDAD COMPLUTENSE MADRID

Declaración Responsable sobre Autoría y Uso Ético de

Herramientas de Inteligencia Artificial (IA)

Yo, HURTADO RESTREPO JUAN MANUEL

con DNI/NIE/PASAPORTE: 60339298V

declaro de manera responsable que el presente Trabajo de Fin de Grado (TFG) con el título

TÉNICAS SELECTAS DE MACHINE LEARNING Y APLICACIONES

es el resultado de mi trabajo intelectual personal y creativo, y ha sido elaborado de acuerdo con los principios éticos y las normas de integridad vigentes en la comunidad académica y, más específicamente, en la Universidad Complutense de Madrid.

Soy, pues, autor del material aquí incluido y, cuando no ha sido así y he tomado el material de otra fuente, lo he citado o bien he declarado su procedencia de forma clara -incluidas, en su caso, herramientas de inteligencia artificial-. Las ideas y aportaciones principales incluidas en este trabajo, y que acreditan la adquisición de competencias, son mías y no proceden de otras fuentes o han sido reescritas usando material de otras fuentes.

Asimismo, aseguro que los datos y recursos utilizados son legítimos, verificables y han sido obtenidos de fuentes confiables y autorizadas. Además, he tomado medidas para garantizar la confidencialidad y privacidad de los datos utilizados, evitando cualquier tipo de sesgo o discriminación injusta en el tratamiento de la información.

En Madrid, a 24/06/2025

Firmado por JUAN MANUEL HURTADO RESTREPO el día 24/06/2025 con un certificado emitido por AUTOFIRMA/CAMERFIRMA

FIRMA