Trabajo Fin de Máster

Algoritmos de aprendizaje basados en árboles de expansión mínima

Iria Rodríguez Acevedo

Máster en Técnicas Estadísticas

Curso 2022-2023

Propuesta de Trabajo Fin de Máster

Título en galego: Algoritmos de aprendizaxe baseados en árbores de expansión mínima
Título en español: Algoritmos de aprendizaje basados en árboles de expansión mínima
English title: Learning algorithms based on minimum spanning trees
Modalidad: Modalidad A
Autor/a: Iria Rodríguez Acevedo, Universidad de Santiago de Compostela
Director/a: Julio González Díaz, Universidad de Santiago de Compostela; Beatriz Pateiro López, Universidad de Santiago de Compostela
Breve resumen del trabajo: Un árbol de expansión mínima (mst) es un subconjunto de aristas de un grafo no dirigido, conexo y ponderado que conecta todos los vértices entre sí, sin ningún ciclo y con el mínimo peso total posible de las aristas. Este tipo de grafos han sido utilizados en aprendizaje no supervisado, en tareas de agrupamiento, debido a su capacidad para reconocer clústeres mediante la eliminación de aristas que se consideran inconsistentes en la definición de dichos clústeres. En este trabajo se pretende estudiar la utilización de este tipo de grafos en aprendizaje supervisado. En particular, se analizará mediante un estudio computacional el funcionamiento de una nueva propuesta de algoritmo de clasificación basado en árboles de expansión mínima. Junto con el análisis de dicha propuesta se hará una revisión de algunas de las técnicas más utilizadas en clasificación supervisada, como el método de k vecinos más próximos, incluyendo el estudio de los resultados teóricos que prueban la consistencia del método.

Don/doña Julio González Díaz, Titular de universidad de la Universidad de Santiago de Compostela, don/doña Beatriz Pateiro López, Titular de universidad de la Universidad de Santiago de Compostela, informan que el Trabajo Fin de Máster titulado

Algoritmos de aprendizaje basados en árboles de expansión mínima

fue realizado bajo su dirección por don/doña Iria Rodríguez Acevedo para el Máster en Técnicas Estadísticas. Estimando que el trabajo está terminado, dan su conformidad para su presentación y defensa ante un tribunal.

En Santiago de Compostela, a 5 de junio de 2023.

Firmado por GONZALEZ DIAZ JULIO - ***3267** el día 05/06/2023 con un certificado emitido por AC FNMT Usuarios

El/la director/a:
Don/doña Julio González Díaz

PATEIRO LOPEZ BEATRIZ - 46898344X
Firmado digitalmente por PATEIRO LOPEZ BEATRIZ - 46898344X
Fecha: 2023.06.05 14:45:15 +02'00'

El/la director/a:
Don/doña Beatriz Pateiro López

RODRIGUEZ ACEVEDO IRIA - 35582529B
Firmado digitalmente por RODRIGUEZ ACEVEDO IRIA - 35582529B
Fecha: 2023.06.05 14:37:47 +02'00'

El/la autor/a:
Don/doña Iria Rodríguez Acevedo

Declaración responsable. Para dar cumplimiento a la Ley 3/2022, de 24 de febrero, de convivencia universitaria, referente al plagio en el Trabajo Fin de Máster (Artículo 11, Disposición 2978 del BOE núm. 48 de 2022), **el/la autor/a declara** que el Trabajo Fin de Máster presentado es un documento original en el que se han tenido en cuenta las siguientes consideraciones relativas al uso de material de apoyo desarrollado por otros/as autores/as:

- Todas las fuentes usadas para la elaboración de este trabajo han sido citadas convenientemente (libros, artículos, apuntes de profesorado, páginas web, programas,...)
- Cualquier contenido copiado o traducido textualmente se ha puesto entre comillas, citando su procedencia.
- Se ha hecho constar explícitamente cuando un capítulo, sección, demostración,... sea una adaptación casi literal de alguna fuente existente.

Y, acepta que, si se demostrara lo contrario, se le apliquen las medidas disciplinarias que correspondan.

Índice general

Resumen

Resumen en español

El problema de clasificar una observación en función de sus características observables en cierta clase o categoría es una tarea fundamental del análisis de datos. Su gran cantidad de aplicaciones hace que sea objeto de estudio mediante el desarrollo de diferentes técnicas de clasificación.

El objetivo de este trabajo es presentar una nueva técnica de clasificación basada en árboles de expansión mínima, analizando su comportamiento mediante un exhaustivo estudio computacional. Con este fin, se hará en primer lugar un repaso de algunas de las técnicas de clasificación más empleadas en aprendizaje supervisado, como con la regla de k-vecinos más próximos o discriminación lineal. Nos centraremos en particular en el estudio teórico de la consistencia universal de la regla de k-vecinos más próximos.

English abstract

The classification problem of categorising an observation according to its observable characteristics is a fundamental task of data analysis. Its large number of applications makes it the object of study through the development of different classification techniques.

The aim of this work is to present a new classification technique based on minimum spanning trees, analysing its behaviour by means of an exhaustive computational study. To this end, some of the most commonly used classification techniques in supervised learning will be reviewed, such as the k-nearest neighbours rule or linear discrimination. In particular, we will focus on the theoretical study of the universal consistency of the k-nearest neighbours rule.

Introducción

Este trabajo se encuadra en la intersección de dos ramas de las matemáticas que, si bien son distintas, poseen relación: la estadística y la investigación operativa. Más concretamente, el objetivo del trabajo será presentar una nueva técnica de clasificación basada en árboles de expansión mínima.

Por una parte, el origen de los problemas de clasificación se remonta hasta épocas muy lejanas cuando los humanos comenzaban a clasificar objetos según el metal de fabricación, categorizar a los seres vivos (taxonomía) o mismo agrupar a los alimentos según su función o propiedades. A pesar de ello, desde el punto de vista de la estadística y del análisis de datos el estudio de los problemas de clasificación es más contemporáneo, impulsado sobre todo en los últimos tiempos por los avances tecnológicos y computacionales que permitieron desarrollar nuevas técnicas que hasta el momento eran inviables.

En los años 30 surgió una de las primeras reglas de clasificación de la mano del estadístico y biólogo Ronald Aylmer Fisher: el análisis lineal discriminante (Fisher 1936). Dicha técnica tiene como objetivo separar las clases mediante hiperplanos. Avanzando un poco en el tiempo, ya en los años 50, surgen otras técnicas de clasificación como k-vecinos más próximos (Fix y Hodges 1951) o el algoritmo Perceptrón (Rosenblatt 1958). Este último es el impulsor de lo que hoy en día conocemos como redes neuronales. En los años 70 y 80 se desarrollaron otras técnicas como los árboles de decisión, siendo pionero en este campo John Ross Quinlan con la invención de algoritmos de clasificación como ID3 (Quinlan 1979), que derivaron en la creación de los árboles de decisión (Quinlan 1986). En la década de los 90 surge la técnica de *support vector machines* (SVM) en los trabajos Boser et al. (1992) y Cortes y Vapnik (1995), impulsando así la clasificación binaria. A finales de la década de los 90 se empezaron a desarrollar técnicas de ensamblado, de forma que no se tenga en cuenta la clasificación devuelta por un solo clasificador. Uno de los clasificadores más conocidos que surgieron de aplicar esta idea son los bosques aleatorios o *random forests* (Breiman 2001).

En las últimas décadas, con el desarrollo de internet y de la computación, se ha experimentado un notable crecimiento en el estudio del conocido como *machine learning* o aprendizaje automático. Esta es una rama de la inteligencia artificial que se ocupa del desarrollo de algoritmos de aprendizaje, dentro de los cuales se encuentra la clasificación. Se impulsa entonces el estudio de nuevos métodos de clasificación, como aquellos basados en redes neuronales.

La gran cantidad de clasificadores que se han ido desarrollando a lo largo de la historia se debe en gran medida a la amplia variedad de aplicaciones que podemos encontrar en múltiples contextos, como pueden ser la detección de spam de un correo electrónico, en sistemas de detección de fraudes o en finanzas.

Por otra parte, encontrar el árbol de expansión mínima de un grafo es uno de los problemas combinatorios más conocidos en el campo de la optimización. Un árbol de expansión mínima es un subconjunto de aristas de un grafo no dirigido, conexo y ponderado que conecta todos los vértices entre sí, sin ningún ciclo y con el mínimo peso total posible de las aristas. Este concepto surgió a raíz del desarrollo de la teoría de grafos a principios del siglo veinte, durante el cual diferentes algoritmos para su obtención fueron surgiendo. El primero de ellos fue introducido por el matemático checo Otakar Boruvka en 1926 (Boruvka 1926). En 1930 el matemático de procedencia también checa Vojtech Jarník propuso otro algoritmo que posteriormente Robert Clay Prim modificó dando lugar a uno de los algoritmos para la obtención de árboles de expansión mínima más conocidos: el algoritmo

de Prim (Prim 1957). Casi de forma paralela, en 1956, Joseph B. Kruskal presentó otro algoritmo también muy extendido: el algoritmo de Kruskal (Kruskal 1956).

Todos estos algoritmos son voraces, es decir, son algoritmos que realizan elecciones localmente óptimas en cada etapa y dan como resultado una solución globalmente óptima. Su fácil implementación y rapidez hace que surjan diversas aplicaciones: en redes de comunicación, en rutas de transporte o para la creación de mapas. Hoy en día se continúan estudiando las propiedades de los árboles de expansión mínima, investigando nuevas aplicaciones que se valgan de ellas. Una de estas nuevas aplicaciones es precisamente la técnica de clasificación que presentaremos. La estructura de los árboles de mínimo coste nos permitirá captar la configuración de cada clase.

La organización de la memoria será la siguiente. Los tres primeros capítulos de la memoria son capítulos teóricos. El Capítulo 1 es un capítulo de preliminares, en el cual se presentarán los conceptos básicos de la clasificación. En la Sección 1.1 se define el clasificador de Bayes como aquel que minimiza la probabilidad de clasificar mal una observación, siendo esta probabilidad la que deriva en conceptos como consistencia de una regla de clasificación. En la Sección 1.2 se presentan, junto con sus principales características, algunos de los clasificadores que hemos comentado: discriminación lineal, árboles de decisión, regresión logística, *support vector machines*, bosques aleatorios y la regla del histograma cúbico.

El Capítulo 2 se centra en exclusiva en la regla de k-vecinos más próximos. Esta regla de clasificación es de fácil comprensión, por lo que es utilizada en multitud de casos en los que la interpretabilidad de los resultados es un factor relevante, priorizándola sobre otros métodos más complejos. La importancia de esta regla viene dada también por ser la primera para la cual se demostró su consistencia universal. En la Sección 2.1 se estudia el caso en el que el número de vecinos k se mantiene fijo, mostrando cómo se obtienen los principales resultados sobre la convergencia de la regla para este caso. En particular, en la Subsección 2.1.1 se proporciona una expresión explícita para la probabilidad asintótica de error, mientras que en la Subsección 2.1.2 se relaciona dicha probabilidad con la devuelta por el clasificador de Bayes. En la Subsección 2.1.3 se estudia el caso particular en el que la probabilidad de error del clasificador de Bayes es próximo a cero. Finalmente, en la Sección 2.2 se analiza el comportamiento de la regla cuando el número de vecinos tiende a infinito, probándose en este caso la consistencia universal de la misma mediante el Teorema de Stone (Stone 1977).

En el Capítulo 3 se presentarán formalmente los árboles de expansión mínima. Primeramente, en la Sección 3.1, se definirán una serie de conceptos sobre teoría de grafos. En la Sección 3.2 se introducen los algoritmos de Boruvka, de Kruskal y de Prim junto con un ejemplo para ilustrarlos.

Posteriormente, en el Capítulo 4 se presenta la nueva regla de clasificación basada en árboles de expansión mínima: *MST-Class* (*Minimun Spanning Tree Classifier*). En la Sección 4.1 se presenta la definición de la misma y en la Sección 4.2 se introduce una primera mejora del método: *MST-RClass* (*Minimum Spanning Tree Robust Classifier*). Esta última versión no solo devuelve mejores resultados sino que también es más eficiente computacionalmente. Por último, en la Sección 4.3 se presenta el estudio numérico realizado para evaluar el rendimiento de ambas variantes, comparándolo con el comportamiento de otras reglas mencionadas, como discriminación lineal o k-vecinos más próximos.

Finalmente, el Capítulo 5 contiene las conclusiones obtenidas gracias al estudio computacional comparativo. Asimismo, se explorarán posibles investigaciones futuras y mejoras del nuevo algoritmo.

En la Figura 1 puede apreciarse de forma gráfica la estructura de la memoria.

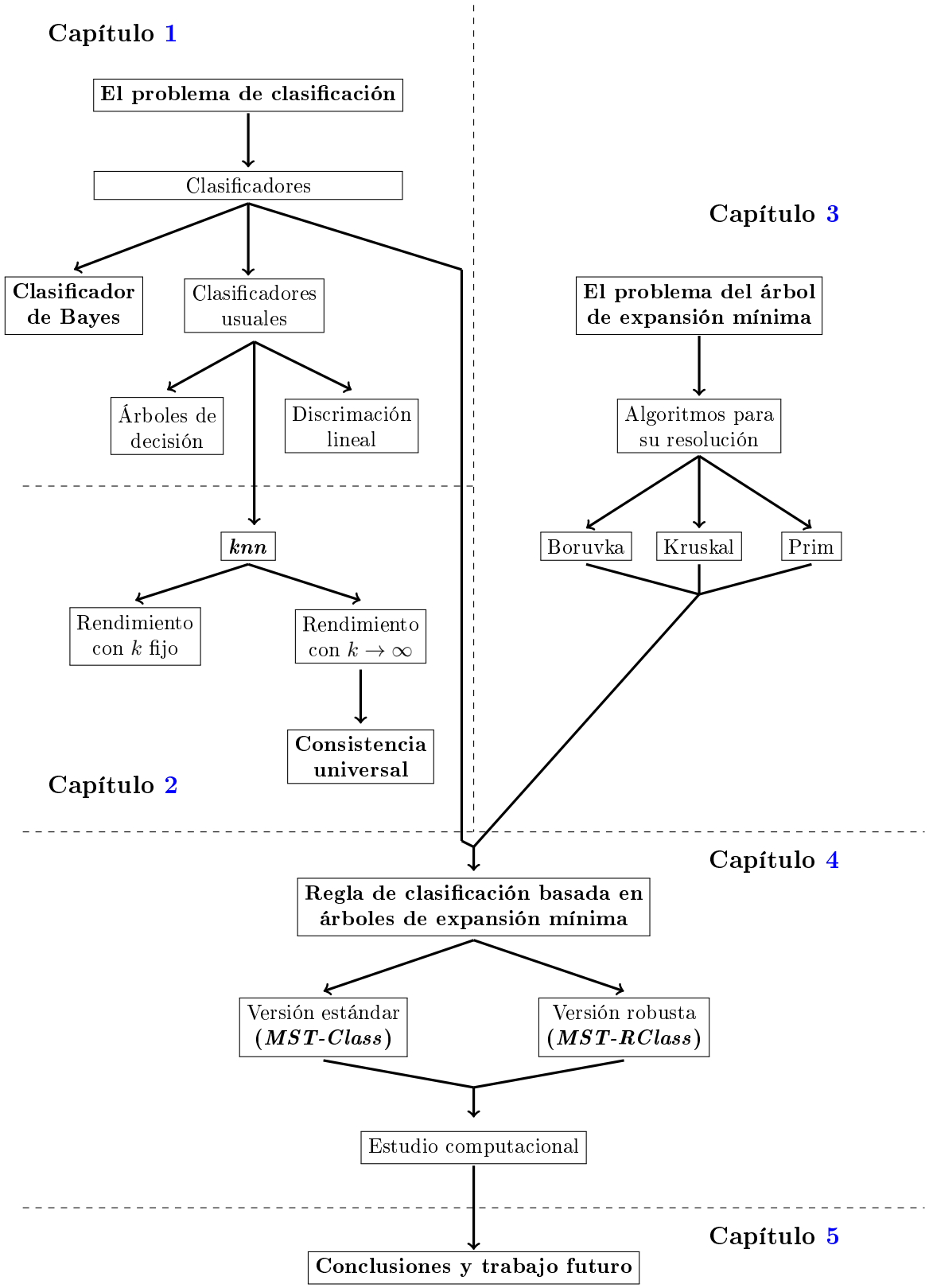


Figura 1: Estructura de la memoria.

Capítulo 1

Preliminares

El querer clasificar una observación en base a ciertas características observables de la misma es un problema que surge de forma habitual en muchas áreas de la vida: las finanzas, la sanidad, la industria, o mismo en la cocina de casa al querer determinar si una manzana está o no madura en función de su aspecto. Dentro de la estadística han surgido a lo largo de la historia muchos métodos para dar solución a estos problemas. Se distinguen dos vertientes: el aprendizaje supervisado y el no supervisado. La gran diferencia entre ambas es que la primera de ellas emplea datos de entrenamiento para los cuales la clasificación ya es conocida, lo que se conoce como datos etiquetados. Sin embargo, el aprendizaje no supervisado trata de reconocer los patrones inherentes a los conjuntos de datos no etiquetados y, en base a ellos, realizar una agrupación. Aunque de formas diferentes, el objetivo es el mismo: realizar una clasificación correcta.

En este capítulo se introducirán nociones básicas relacionadas con el aprendizaje supervisado. Se presentarán algunas definiciones como la de clasificador, regla discriminante y consistencia de una regla. Además, se presentará al clasificador de Bayes, el cual presenta la propiedad más deseable para un clasificador y a la que aspiran los demás clasificadores: minimizar la probabilidad de clasificar mal una observación. Sin embargo, veremos que determinar el clasificador de Bayes no es en la práctica factible, por lo que daremos una visión generalizada de los clasificadores más usuales.

1.1. Clasificador de Bayes

Para esta sección se han empleado, como referencias principales, los libros de Biau y Devroye (2015) y Devroye et al. (2013) como referencias, empleando la misma notación para presentar los diferentes conceptos claves.

Se denotará por x al vector d-dimensional que representa una observación dada y por y a la clase de una observación, que puede tomar valores en el conjunto $\{1, 2, \ldots, M\}$, donde M es el número de clases.

Definición 1.1.1. Sea x una observación e y su clase, un *clasificador* es una función

$$g(x) : \mathbb{R}^d \to \{1, \ldots, M\}$$

que representa la suposición mediante g de y. El clasificador cometerá un error cuando $g(x) \neq y$.

En general, consideraremos el vector aleatorio (X, Y) que toma valores en $\mathbb{R}^d \times \{1, \ldots, M\}$. Por tanto, si $g(X) \neq Y$ el clasificador g cometerá un error. La probabilidad de que esto ocurra la denotaremos de la siguiente forma:

$$L(g) = \mathbb{P}\{g(X) \neq Y\}.$$

Definición 1.1.2. Se define el *clasificador de Bayes* g^* como aquel que minimiza la probabilidad de error:

$$g^* = \underset{g:\mathbb{R}^d \to \{1,\dots,M\}}{\arg\min} \mathbb{P}\{g(X) \neq Y\}.$$

Definición 1.1.3. Se denomina *error de Bayes* o *riesgo de Bayes* L^* a la mínima probabilidad de error, es decir, $L^* = L(g^*)$.

Cabe destacar que el clasificador de Bayes depende de la distribución del vector aleatorio (X, Y), la cual es, mayoritariamente, desconocida. Por tanto, dicho clasificador es en general desconocido.

Como consecuencia, para construir un clasificador dispondremos de un conjunto de observaciones (X_i, Y_i), $i = 1, \dots, n$, que, de alguna forma, sea representativo de la distribución subyacente de (X, Y). En lo que sigue, se asumirá que $(X_1, Y_1), \dots, (X_n, Y_n)$ es una secuencia de pares aleatorios independientes, idénticamente distribuidos y con la misma distribución que (X, Y). Al clasificador basado en $(X_1, Y_1), \dots, (X_n, Y_n)$ se denota por g_n. Entonces, dado X, se deduce Y a partir de $g_n(X; X_1, Y_1, \dots, X_n, Y_n)$.

Para evaluar el comportamiento de g_n se emplea la probabilidad de error condicional:

$$L_n = L(g_n) = \mathbb{P}\{g_n(X; X_1, Y_1, \dots, X_n, Y_n) \neq Y \mid X_1, Y_1, \dots, X_n, Y_n\}.$$

Resulta claro que L_n depende de los datos, por lo que se trata de una variable aleatoria.

Definición 1.1.4. Una función $g_n : \mathbb{R}^d \times \{\mathbb{R}^d \times \{1, \dots, M\}\}^n \to \{1, \dots, M\}$ se denomina también *clasificador*. Una secuencia $\{g_n, n \geq 1\}$ se denomina *regla* (*de discriminación* o *discriminante*).

En base a la Definición 1.1.4 concluimos que los clasificadores son funciones y las secuencias de funciones son reglas.

Una de las características deseables en una buena regla discriminante es su consistencia.

Definición 1.1.5. Una regla de clasificación se dice *consistente* (o *asintóticamente eficiente al riesgo de Bayes*) para cierta distribución de (X, Y) si

$$\lim_{n\to\infty} \mathbb{E}(L_n) = L^*.$$

Como L_n es una variable aleatoria acotada entre L^* y 1 esta convergencia es equivalente a decir que la probabilidad de error condicional tiende al error Bayes en probabilidad cuando $n \to \infty$. Es decir, para todo $\epsilon > 0$

$$\lim_{n\to\infty} \mathbb{P}(L_n - L^* > \epsilon) = 0.$$

Esto nos garantiza que tomar más muestras hace que la distribución subyacente de (X, Y), que es desconocida, pueda ser reconstruida de forma aproximada.

Definición 1.1.6. Una regla se dice *fuertemente consistente* si L_n converge a L^* con probabilidad uno, en otras palabras, L_n converge de forma casi segura a L^*:

$$\mathbb{P}\left(\lim_{n\to\infty} L_n = L^*\right) = 1.$$

Observación 1.1.7. *Dado que convergencia casi segura implica convergencia en probabilidad, resulta claro que un regla fuertemente consistente es también consistente.*

Definición 1.1.8. Una regla se dice *universalmente (fuertemente) consistente* si es (fuertemente) consistente para toda distribución subyacente de (X, Y).

En lo que sigue se considerarán las siguientes restricciones:

- Consideraremos $M = 2$, es decir, Y puede tomar valores en $\{0, 1\}$, y un clasificador g_n será entonces una función con dominio $\mathbb{R}^d \times \{\mathbb{R}^d \times \{0, 1\}\}^d$ y rango $\{0, 1\}$.
- No se considerarán espacios infinitos, por lo que X debe ser un vector aleatorio evaluado en $\mathbb{R}^d$.

El par (X, Y) puede venir definido por el par (μ, η), donde, para cada conjunto de Borel medible $A \subseteq \mathbb{R}^d$:

$$\mu(A) = \mathbb{P}(X \in A),$$

y para cada $x \in \mathbb{R}^d$,

$$\eta(x) = \mathbb{P}(Y = 1 \mid X = x) = \mathbb{E}(Y \mid X = x).$$

En efecto, cada $C \subseteq \mathbb{R}^d \times \{0, 1\}$ puede expresarse de la siguiente forma:

$$C = (C \cap (\mathbb{R}^d \times \{0\})) \cup (C \cap (\mathbb{R}^d \times \{1\})) \coloneqq C_0 \times \{0\} \cup C_1 \times \{1\},$$

por tanto,

$$\begin{aligned}\mathbb{P}((X, Y) \in C) &= \mathbb{P}(X \in C_0, Y = 0) + \mathbb{P}(X \in C_1, Y = 1) \\ &= \int_{C_0} (1 - \eta(x))\mu(dx) + \int_{C_1} \eta(x)\mu(dx),\end{aligned}$$

por consiguiente, la distribución de (X, Y) está determinada por (μ, η). La función η se conoce como la *probabilidad a posteriori*.

El siguiente resultado nos muestra que el clasificador de Bayes en el caso de clasificación binaria se basa en clasificar una observación x en la clase $Y = 1$ cuando la probabilidad de que $Y = 1$ condicionada a x (la probabilidad a posteriori) es mayor que $\frac{1}{2}$.

Teorema 1.1.9 (Biau y Devroye (2015), Lema 17.1)**.** *El clasificador de Bayes g^* cuando $M = 2$ es de la siguiente forma:*

$$g^*(x) = \begin{cases} 1 & \text{si } \eta(x) > 1/2 \Leftrightarrow \mathbb{P}(Y = 1 \mid X = x) > \mathbb{P}(Y = 0 \mid X = x), \\ 0 & \text{en otro caso.} \end{cases}$$

Demostración. En efecto, veamos que g^* así definido es aquel que minimiza la probabilidad de error, es decir, que para toda función $g \colon \mathbb{R}^d \to \{0, 1\}$ se tiene que:

$$\mathbb{P}(g^*(X) \neq Y) \leq \mathbb{P}(g(X) \neq Y).$$

Dado $X = x$ y siendo I_A la indicadora de A:

$$\begin{aligned}&\mathbb{P}(g(X) \neq Y \mid X = x) \\ &\quad = 1 - \mathbb{P}(Y = g(X) \mid X = x) \\ &\quad = 1 - (\mathbb{P}(Y = 1, g(X) = 1 \mid X = x) + \mathbb{P}(Y = 0, g(X) = 0 \mid X = x)) \\ &\quad = 1 - (I_{\{g(x)=1\}}\mathbb{P}(Y = 1 \mid X = 1) + I_{\{g(x)=0\}}\mathbb{P}(Y = 0 \mid X = x)) \\ &\quad = 1 - (I_{\{g(x)=1\}}\eta(x) + I_{\{g(x)=0\}}(1 - \eta(x))).\end{aligned}$$

Por tanto, para cada $x \in \mathbb{R}^d$:

$$\begin{aligned}&\mathbb{P}(g(X) \neq Y \mid X = x) - \mathbb{P}(g^*(X) \neq Y \mid X = x) \\ &\quad = \eta(x)(I_{\{g^*(x)=1\}} - I_{\{g(x)=1\}}) + (1 - \eta(x))(I_{\{g^*(x)=0\}} - I_{\{g(x)=0\}}) \\ &\quad = \eta(x)(I_{\{g^*(x)=1\}} - I_{\{g(x)=1\}}) + (1 - \eta(x))(1 - I_{\{g^*(x)=1\}} - (1 - I_{\{g(x)=1\}})) \\ &\quad = (2\eta(x) - 1)(I_{\{g^*(x)=1\}} - I_{\{g(x)=1\}}) \\ &\quad \geq 0,\end{aligned}$$

donde la última desigualdad se deduce de la definición de g^*. Integrando a ambos lados con respecto a $\mu(dx)$ se obtiene el resultado deseado y g^* es el clasificador de Bayes. □

De la demostración del Teorema 1.1.9 se deduce que la probabilidad de error viene dada por:

$$L(g) = 1 - \mathbb{E}(I_{\{g(X)=1\}}\eta(X) + I_{\{g(X)=0\}}(1-\eta(X))),$$

y, por consiguiente, el error de Bayes L^* se puede expresar como:

$$L^* = 1 - \mathbb{E}(I_{\{\eta(X)>1/2\}}\eta(X) + I_{\{\eta(X)\leq 1/2\}}(1-\eta(X))).$$

Por tanto,

- Si $\eta(X) > 1/2$ entonces $\eta(X) > 1-\eta(X)$.
- Si $\eta(X) \leq 1/2$ entonces $\eta(X) \leq 1-\eta(X)$.

Así,

$$L^* = 1 - \mathbb{E}(\text{máx}\{\eta(X), 1-\eta(X)\}) = \mathbb{E}(\text{mín}\{\eta(X), 1-\eta(X)\}).$$

Además,

$$L^* = \frac{1}{2} - \frac{1}{2}\mathbb{E}(|2\eta(X)-1|).$$

En efecto,

- Si $2\eta(X)-1 > 0$ entonces $\eta(X) > 1/2 \implies \eta(X) > 1-\eta(X)$. Por tanto,

$$L^* = \frac{1}{2} - \frac{1}{2}\mathbb{E}(2\eta(X)-1) = 1 - \mathbb{E}(\eta(X)) = \mathbb{E}(1-\eta(X)).$$

- Si $2\eta(X)-1 \leq 0$ entonces $\eta(X) \leq 1/2 \implies \eta(X) \leq 1-\eta(X)$. Por tanto,

$$L^* = \frac{1}{2} - \frac{1}{2}\mathbb{E}(1-2\eta(X)) = \mathbb{E}(\eta(X)).$$

Es decir, $\frac{1}{2} - \frac{1}{2}\mathbb{E}(|2\eta(X)-1|) = \mathbb{E}(\text{mín}\{\eta(X), 1-\eta(X)\}) = L^*$. Además, si X tiene una densidad f entonces:

$$L^* = \int \text{mín}\{\eta(x), 1-\eta(x)\} f(x)\, dx = \int \text{mín}\{(1-p)f_0(x), pf_1(x)\}\, dx$$

donde $p = \mathbb{P}(Y=1)$, y $f_i(x)$ es la densidad de X dado $Y=i$, $i=0,1$. A p y a $1-p$ se les denomina *probabilidades de clase* y f_0, f_1 son *densidades condicionales de clase*.

A continuación mostraremos un ejemplo práctico que hemos elaborado para ilustrar algunos de los conceptos tratados hasta el momento. En él se verá que, supuesta conocida la probabilidad a posteriori, podremos determinar el clasificador Bayes. Además, se analizará el error Bayes en función de la distribución de X.

Ejemplo 1.1.10. Consideremos la variable aleatoria X que representa la temperatura ambiente y la variable aleatoria discreta Y, siendo $Y=1$ si se produce la floración de una especie e $Y=0$ en caso contrario. Supongamos además que la probabilidad a posteriori es conocida y tiene la siguiente expresión:

$$\eta(x) = \mathbb{P}(Y=1|X=x) = \mathbb{E}(Y|X=x) = \frac{1}{1+e^{-x}},$$

con $x \in [-10, 10]$.

En base al Teorema 1.1.9, el clasificador de Bayes g^* es de la siguiente forma:

$$g^*(x) = \begin{cases} 1 & \text{si } \eta(x) > 1/2, \\ 0 & \text{en otro caso.} \end{cases}$$

Entonces, como $\eta(x) > 1/2 \Leftrightarrow 2 > 1 + e^{-x} \Leftrightarrow 1 > e^{-x} \Leftrightarrow x > 0$:

$$g^*(x) = \begin{cases} 1 & \text{si } x > 0, \\ \\ 0 & \text{en otro caso.} \end{cases}$$

Veamos ahora cuál sería la expresión del error de Bayes según las dos fórmulas que hemos visto:

i)

$$\begin{aligned} L^* &= \frac{1}{2} - \frac{1}{2}\mathbb{E}(|2\eta(X) - 1|) = \frac{1}{2} - \frac{1}{2}\mathbb{E}\left(\left|\frac{2}{1+e^{-X}} - 1\right|\right) = \frac{1}{2} - \frac{1}{2}\mathbb{E}\left(\left|\frac{1-e^{-X}}{1+e^{-X}}\right|\right) \\ &= \begin{cases} \frac{1}{2} - \frac{1}{2}\mathbb{E}\left(\frac{e^{-X}-1}{1+e^{-X}}\right) & \text{si } X \leq 0, \\ \frac{1}{2} - \frac{1}{2}\mathbb{E}\left(\frac{1-e^{-X}}{1+e^{-X}}\right) & \text{si } X \geq 0. \end{cases} \end{aligned} \tag{1.1}$$

ii)

$$\begin{aligned} L^* &= \mathbb{E}(\text{mín}\{\eta(X), 1 - \eta(X)\}) = \mathbb{E}\left(\text{mín}\left\{\frac{1}{1+e^{-X}}, 1 - \frac{1}{1+e^{-X}}\right\}\right) \\ &= \mathbb{E}\left(\text{mín}\left\{\frac{1}{1+e^{-X}}, \frac{e^{-X}}{1+e^{-X}}\right\}\right) = \mathbb{E}\left(\frac{\text{mín}\{1, e^{-X}\}}{1+e^{-X}}\right) \\ &= \begin{cases} \mathbb{E}\left(\frac{1}{1+e^{-X}}\right) & \text{si } X \leq 0, \\ \mathbb{E}\left(\frac{e^{-X}}{1+e^{-X}}\right) & \text{si } X \geq 0. \end{cases} \end{aligned} \tag{1.2}$$

Podemos observar entonces que, a pesar de conocer la probabilidad a posteriori $\eta(x)$, el error de Bayes L^* sigue siendo desconocido. Supongamos entonces diferentes distribuciones para X y veamos que ambas fórmulas devuelven la misma probabilidad y cómo cambia en función de las distribuciones.

i) Supongamos en primer lugar que $X = 0$ (que la temperatura es de 0 grados) con probabilidad 1. Entonces:

- $L^* = \frac{1}{2} - \frac{1}{2}\mathbb{E}\left(\frac{e^{-X}-1}{1+e^{-X}}\right) = \frac{1}{2}$.
- $L^* = \mathbb{E}\left(\frac{1}{1+e^{-X}}\right) = \frac{1}{2}$.

Es decir, la mínima probabilidad de error es $1/2$.

ii) Supongamos que X es uniforme en $[-10, 10]$, es decir X tiene la siguiente densidad:

$$f(x) = \frac{1}{20} \quad x \in [-10, 10].$$

Entonces,

- Empleando la expresión (1.1) en primer lugar:

$$\begin{aligned} L^* &= \frac{1}{2} - \frac{1}{2}\left(\int_{-10}^{0} \frac{e^{-x}-1}{1+e^{-x}} \cdot \frac{1}{20}\, dx + \int_{0}^{10} \frac{1-e^{-x}}{1+e^{-x}} \cdot \frac{1}{20}\, dx\right) \\ &= \frac{1}{2} - \frac{1}{40}\left(-2\ln(|1+e^{-x}|) - x\Big|_{-10}^{0} + 2\ln(|1+e^{-x}|) + x\Big|_{0}^{10}\right) \\ &\approx 0.0693 \end{aligned}$$

- Empleando la expresión (1.2):

$$\begin{aligned} L^* &= \left(\int_{-10}^{0} \frac{1}{1+e^{-x}} \cdot \frac{1}{20}\, dx + \int_{0}^{10} \frac{e^{-x}}{1+e^{-x}} \cdot \frac{1}{20}\, dx \right) \\ &= \frac{1}{20} \left(\ln(|1+e^{-x}|) + x \Big|_{-10}^{0} - \ln(|1+e^{-x}|) \Big|_{0}^{10} \right) \\ &\approx 0.0693 \end{aligned}$$

Se aprecia así cómo el error Bayes ha disminuido mucho con respecto al caso anterior.

1.2. Clasificadores usuales

Obtener el clasificador de Bayes es una tarea complicada, ya que depende de la distribución de (X, Y). Como consecuencia, han surgido a lo largo del tiempo diferentes clasificadores que pueden ser empleados en la práctica sin el conocimiento de la distribución de (X, Y). A continuación presentamos algunos de ellos con sus principales características, centrándonos en especial en la discriminación lineal y en la regla de k-vecinos más próximos. El primero de ellos se trata de un método paramétrico muy conocido, mientras que la regla de k-vecinos más próximos es un método clásico, también muy extendido, pero no paramétrico. Ambos serán empleados en el estudio computacional posterior.

También comentaremos en menor detalle otros métodos cuyo uso está muy extendido, como los árboles de decisión, o la regresión logística, entre otros.

1.2.1. Regla de k-vecinos más próximos

La regla de k-vecinos más próximos surgió en 1951 (Fix y Hodges 1951), pero no es hasta 1977 cuando, gracias a Stone (Stone 1977), se consiguió probar su consistencia universal. De hecho, hasta ese momento se desconocía que ninguna regla lo fuese. Gracias a la demostración proporcionada se impulsó el estudio de la consistencia universal de otras reglas.

La regla de k-vecinos más próximos consiste en clasificar una observación en función de lo que dicta la mayoría de los k vecinos más próximos a él. La definición formal de esta regla la trataremos en el siguiente capítulo, así como la prueba de su consistencia universal, ya que, como hemos comentado, esta prueba fue pionera y, por tanto, muy relevante en el campo.

1.2.2. Discriminación lineal

Cuando una regla de clasificación divide el espacio mediante hiperplanos, asignando cada clase a uno de ellos, decimos que la regla es un método lineal de clasificación. Estas reglas no proporcionan generalmente probabilidades de error cercanas al error de Bayes. Aún así, son la base de otras muchas reglas de clasificación, como pueden ser los árboles de decisión. Es por ello por lo que hay que tenerlas en consideración. Otras reglas sin embargo no se encuadran dentro de la discriminación lineal, como es el caso de la regla de k-vecinos más próximos. Nos basaremos en Devroye et al. (2013) para introducir esta regla.

En el caso unidimensional la regla lineal discriminante se define como:

$$g(x) = \begin{cases} y' & \text{si } x \leq x', \\ 1 - y' & \text{en otro caso.} \end{cases}$$

siendo x' un punto de corte e $y' \in \{0, 1\}$ la clase, ambos dependiendo de los datos. Dentro del conjunto de clasificadores unidimensionales de esta forma existe uno que proporciona la mínima probabilidad

de error. Para saber cuál es necesitaríamos saber cuál es la distribución de (X, Y). Supongamos entonces que $\mathbb{P}(Y = 1) = p$. Además, las funciones de distribución condicionales de X se denotarán por $F_1(x) = \mathbb{P}(X \leq x|Y = 1)$ y $F_0(x) = \mathbb{P}(X \leq x|Y = 0)$. Por consiguiente, la regla lineal discriminante óptima escoge el punto de corte x^* y la clase y^* como:

$$(x^*, y^*) = \arg \min_{(x',y')} \mathbb{P}(g(X) \neq Y).$$

El corte definido por (x^*, y^*) se denomina *división teórica de Stoller*.

Para calcular la probabilidad de error hay que tener en cuenta que se clasifica mal cuando

a) $y = 0$ y se clasifica como $y = 1$. Esto se traduce en dos posibles casos:

- $y = 0$, $y' = 1$ y $x \leq x'$. La probabilidad de que esto suceda es

$$I_{\{y'=1\}}\mathbb{P}(X \leq x', Y = 0) = I_{\{y'=1\}}\mathbb{P}(Y = 0)\mathbb{P}(X \leq x'|Y = 0) = I_{\{y'=1\}}(1 - p)F_0(x').$$

- $y = 0$, $y' = 0$ y $x > x'$. La probabilidad de este suceso es

$$I_{\{y'=0\}}\mathbb{P}(X > x', Y = 0) = I_{\{y'=0\}}\mathbb{P}(Y = 0)\mathbb{P}(X > x'|Y = 0) = I_{\{y'=0\}}(1 - p)(1 - F_0(x')).$$

b) $y = 1$ y se clasifica como $y = 0$. A su vez se diferencian los siguientes casos:

- $y = 1$, $y' = 0$ y $x \leq x'$. La probabilidad de que esto suceda es

$$I_{\{y'=0\}}\mathbb{P}(X \leq x', Y = 1) = I_{\{y'=0\}}\mathbb{P}(Y = 1)\mathbb{P}(X \leq x'|Y = 1) = I_{\{y'=0\}}pF_1(x').$$

- $y = 1$, $y' = 1$ y $x > x'$. La probabilidad de este suceso es

$$I_{\{y'=1\}}\mathbb{P}(X > x', Y = 1) = I_{\{y'=1\}}\mathbb{P}(Y = 1)\mathbb{P}(X > x'|Y = 1) = I_{\{y'=1\}}p(1 - F_1(x')).$$

Por tanto, se define la mínima probabilidad de error dentro del conjunto de clasificadores lineales discriminantes unidimensionales como:

$$L = \inf_{(x',y')} \{I_{\{y'=0\}}[pF_1(x') + (1 - p)(1 - F_0(x'))] + I_{\{y'=1\}}[p(1 - F_1(x')) + (1 - p)F_0(x')]\}.$$

Veremos en el siguiente resultado que $L \leq 1/2$, lo cual quiere decir que siempre podremos encontrar una regla lineal discriminante para el caso unidimensional cuya probabilidad de error sea menor o igual que $1/2$.

Lema 1.2.1 (Devroye et al. (2013), Lema 4.1)**.** *$L \leq 1/2$, siendo además $L = \frac{1}{2}$ si y solo si $L^* = 1/2$.*

Demostración. Veamos en primer lugar que $L \leq 1/2$. Para ello consideremos los siguientes casos:

- Si $(x', y') = (-\infty, 0)$, entonces la probabilidad de error es $\mathbb{P}(Y = 0) = 1 - p$.
- Si $(x', y') = (-\infty, 1)$ la probabilidad de error es $\mathbb{P}(Y = 1) = p$.

Por tanto, $L \leq \text{mín}\{p, 1 - p\} \implies L \leq 1/2$.

Comprobemos ahora que $L = 1/2 \Leftrightarrow L^* = 1/2$. Si $L^* = 1/2$, como $L^* \leq L \leq 1/2$ se obtiene directamente que $L = 1/2$. Supongamos ahora que $L = 1/2$. Como consecuencia $p = 1/2$ y, para todo x,

$$\begin{aligned} pF_1(x) + (1 - p)(1 - F_0(x)) \geq 1/2 &\implies pF_1(x) - (1 - p)F_0(x) \geq p - 1/2 \\ p(1 - F_1(x)) + (1 - p)F_0(x) \geq 1/2 &\implies pF_1(x) - (1 - p)F_0(x) \leq p - 1/2 \end{aligned}$$

Así, $pF_1(x) - (1 - p)F_0(x) = p - 1/2$. Como $p = 1/2$, se obtiene que $F_1(x) = F_0(x)$, lo que implica que X e Y son variables independientes. Entonces,

$$\eta(x) = \mathbb{P}(Y = 1|X = x) = \mathbb{P}(Y = 1) = p = 1/2$$

para todo x. Se concluye por tanto que $L^* = 1/2$. □

Otra forma adecuada de expresar L se proporciona en el siguiente resultado. Gracias a esta nueva formulación se relaciona a la división teórica de Stroller con la distancia de Kolmogorov Smirnov entre las funciones de distribución condicionales: $\sup_x |F_1(x) - F_0(x)|$.

Lema 1.2.2 (Devroye et al. (2013), Lema 4.2)**.** *La mínima probabilidad de error puede expresarse como:*

$$L = \frac{1}{2} - \sup_x \left| pF_1(x) - (1-p)F_0(x) - p + \frac{1}{2} \right|.$$

En particular, si $p = 1/2$, se cumple que:

$$L = \frac{1}{2} - \frac{1}{2} \sup_x |F_1(x) - F_0(x)|.$$

Demostración. Recordemos que

$$L = \inf_{(x',y')} \{I_{\{y'=0\}}[pF_1(x') + (1-p)(1-F_0(x'))] + I_{\{y'=1\}}[p(1-F_1(x')) + (1-p)F_0(x')]\}.$$

Entonces,

$$\begin{aligned} L &= \inf_x \min\{pF_1(x) + (1-p)(1-F_0(x)), p(1-F_1(x)) + (1-p)F_0(x)\} \\ &= \inf_x \min\{[pF_1(x) - (1-p)F_0(x)] + (1-p), p - [pF_1(x) - (1-p)F_0(x)]\} \\ &= \frac{1}{2} - \sup_x \left| pF_1(x) - (1-p)F_0(x) - p + \frac{1}{2} \right| \end{aligned}$$

donde la última igualdad viene de aplicar que $\min\{a,b\} = (a+b-|a-b|)/2$. □

Para el caso de más de una dimensión, la regla lineal discriminante con pesos $a_0, a_1, \ldots, a_d$ viene dada por

$$g(x) = \begin{cases} 1 & \text{si } \sum_{i=1}^d a_i x^{(i)} + a_0 > 0, \\ 0 & \text{en otro caso.} \end{cases}$$

siendo $x = (x^{(1)}, \ldots, x^{(d)})$. La probabilidad de error se denota en este caso por $L(a, a_0)$, donde $a = (a_1, \ldots, a_d)$. Se define además

$$L = \inf_{a \in \mathbb{R}^d, a_0 \in \mathbb{R}} L(a, a_0).$$

En otras palabras, L es la menor probabilidad de error alcanzable dentro de las reglas lineales discriminantes. Al igual que ocurría en el caso unidimensional, y empleando un razonamiento similar, se llega al siguiente resultado que nos garantiza que L está acotada superiormente por $1/2$. Esto se traduce en que existe siempre una regla lineal discriminante cuya probabilidad de error sea menor o igual que $1/2$, independientemente de la dimensión en la que nos encontremos.

Teorema 1.2.3 (Devroye et al. (2013), Teorema 4.3)**.** *$L \leq 1/2$, siendo además $L = \frac{1}{2}$ si y solo si $L^* = 1/2$.*

Discriminación lineal de Fisher

Fisher propuso en 1936 una elección específica para el vector $a = (a_1, \ldots, a_d)$, motivada por un razonamiento muy intuitivo que veremos a continuación.

Denotemos por $\widehat{\mu}_1$ y $\widehat{\mu}_0$ a las medias muestrales de cada clase, es decir,

$$\widehat{\mu}_1 = \frac{\sum_{i:\, Y_i=1} X_i}{|\{i:\, Y_i = 1\}|} \qquad \text{y} \qquad \widehat{\mu}_0 = \frac{\sum_{i:\, Y_i=0} X_i}{|\{i:\, Y_i = 0\}|}.$$

Asimismo, se denotará por $\widehat{\sigma}_1^2$ y $\widehat{\sigma}_0^2$ a las varianzas muestrales de cada clase:

$$\widehat{\sigma}_1^2 = \sum_{i:\ Y_i=1} (a^T X_i - a^T \widehat{\mu}_1)^2 = a^T S_1 a \qquad \text{y} \qquad \widehat{\sigma}_0^2 = \sum_{i:\ Y_i=0} (a^T X_i - a^T \widehat{\mu}_0)^2 = a^T S_0 a,$$

siendo

$$S_1 = \sum_{i:\ Y_i=1} (X_i - \widehat{\mu}_1)(X_i - \widehat{\mu}_1)^T \qquad \text{y} \qquad S_0 = \sum_{i:\ Y_i=0} (X_i - \widehat{\mu}_0)(X_i - \widehat{\mu}_0)^T.$$

Entonces, la discriminación lineal de Fisher escoge aquel a tal que:

$$a = \arg\max_x \frac{(x^T \widehat{\mu}_1 - x^T \widehat{\mu}_0)^2}{\widehat{\sigma}_1^2 + \widehat{\sigma}_0^2} = \arg\max_x \frac{(x^T(\widehat{\mu}_1 - \widehat{\mu}_0))^2}{x^T(S_1 + S_0)x}.$$

Esto se traduce en escoger una dirección a que haga máxima la separación entre $a^T\widehat{\mu}_1$ y $a^T\widehat{\mu}_0$ con respecto a la dispersión muestral. Resolviendo el problema de maximización previo, se obtiene que el óptimo es:

$$a = (S_1 + S_0)^{-1}(\widehat{\mu}_1 + \widehat{\mu}_0).$$

Esta elección para a, a pesar de su interpretación, puede tener un mal comportamiento en la práctica. De hecho, existen ejemplos en los que, aún siendo $L = 0$, la discriminación lineal de Fisher obtiene una probabilidad de error cercana a 1.

Caso Normal

De especial interés es estudiar la discriminación lineal para el caso normal, ya que bajo este supuesto resulta que coincide con la regla de Bayes. Recordemos que la expresión de la densidad normal multivariante es la siguiente:

$$f(x) = (2\pi)^{-d/2}|\Sigma|^{-1/2} e^{-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)},$$

donde μ es el vector de medias (d dimensional) y Σ es la matriz de covarianzas (de dimensión $d \times d$).

Consideremos un problema de clasificación con dos clases posibles y donde X tiene densidad $(1 - p)f_0(x) + pf_1(x)$, siendo f_0 y f_1 densidades normales multivariantes con parámetros μ_0, μ_1 y Σ_0, Σ_1, respectivamente. Entonces, la regla de Bayes se escribe como:

$$g^*(x) = \begin{cases} 1 & \text{si } pf_1(x) > (1-p)f_0(x), \\ \\ 0 & \text{en otro caso.} \end{cases}$$

Si se toman logaritmos entonces:

$$\begin{aligned} & pf_1(x) > (1-p)f_0(x) \Leftrightarrow \\ & \log(pf_1(x)) > \log((1-p)f_0(x)) \Leftrightarrow \\ & -\frac{1}{2}\log(|\Sigma_1|) - \frac{1}{2}(x-\mu_1)^T\Sigma_1^{-1}(x-\mu_1) + log(p) > -\frac{1}{2}\log(|\Sigma_0|) - \frac{1}{2}(x-\mu_0)^T\Sigma_0^{-1}(x-\mu_0) + log(1-p) \Leftrightarrow \\ & (x-\mu_1)^T\Sigma_1^{-1}(x-\mu_1) - 2\log p + \log(|\Sigma_1|) < (x-\mu_0)^T\Sigma_0^{-1}(x-\mu_0) - 2\log(1-p) + \log(|\Sigma_0|). \end{aligned}$$

Teniendo en cuenta que $r_i^2 = (x-\mu_i)^T\Sigma_i^{-1}(x-\mu_i)$ es la distancia de Mahalanobis al cuadrado entre x y μ_i, entonces:

$$g^*(x) = \begin{cases} 1 & \text{si } r_1^2 < r_0^2 - 2\log((1-p)/p) + \log(|\Sigma_0|/|\Sigma_1|), \\ \\ 0 & \text{en otro caso.} \end{cases}$$

De la expresión anterior se pueden deducir los siguientes casos particulares:

- Si se supone que $\Sigma_0 = \Sigma_1 = \Sigma$ entonces la regla de Bayes es lineal:

$$g^*(x) = \begin{cases} 1 & \text{si } a^T x + a_0 > 0, \\ 0 & \text{en otro caso.} \end{cases}$$

donde $a^T = 2(\mu_1 - \mu_0)^T \Sigma^{-1}$ y $a_0 = 2\log(p/(1-p)) + \mu_0^T \Sigma^{-1} \mu_0 - \mu_1^T \Sigma^{-1} \mu_1$. Cuando $\Sigma_0 \neq \Sigma_1$ la regla de decisión es, en general, no lineal (análisis cuadrático discriminante cuando hay dos clases).

- En el caso de que las dos clases sean equiprobables, es decir, $p = 1/2$ y además $\Sigma_0 = \Sigma_1 = \Sigma$ entonces:

$$g^*(x) = \begin{cases} 1 & \text{si } r_1^2 < r_0^2, \\ 0 & \text{en otro caso.} \end{cases}$$

Es decir, se clasifica una observación en función de la clase cuya media está a una menor distancia de Mahalanobis de x.

1.2.3. Árboles de decisión

Los siguientes clasificadores que presentaremos son los árboles de decisión. Los árboles de decisión hacen una partición del espacio $\mathbb{R}^d$ en diferentes regiones. Existen diferentes métodos según la construcción de la partición de $\mathbb{R}^d$. En base a estas particiones se realiza la clasificación, más concretamente, en base a las hojas (nodos terminales) del árbol. Entre ellos, los más importantes son los árboles de clasificación binarios, que se caracterizan por tener únicamente dos hijos por nodo, lo que hace que sean más sencillos de manipular y actualizar.

Ejemplo 1.2.4. En la Figura 1.1 puede observarse una representación de un árbol de clasificación binario.

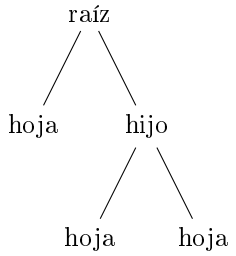


Figura 1.1: Árbol de clasificación.

A continuación presentaremos unas pinceladas sobre las características de los árboles de decisión, recogidas en Devroye et al. (2013), donde puede consultarse un análisis más extenso sobre ellos.

Definición 1.2.5. Si una hoja representa la región A, se dice que el clasificador g_n es *natural* si

$$g_n(x) = \begin{cases} 1 & \text{si } \sum_{i:\, X_i \in A} Y_i > \sum_{i:\, X_i \in A} (1 - Y_i),\ x \in A, \\ 0 & \text{en otro caso.} \end{cases}$$

Es decir, de entre todos los X_i en la región A dada por una hoja, se clasifica en función de lo que dicta la mayoría de los correspondientes Y_i. Los empates se deshacen a favor de la clase 0.

Consistencia

En lo que sigue se presentarán algunos resultados relacionados con la consistencia de una clase específica de árboles de decisión. Se supondrá que la forma del árbol está determinada en exclusiva por lo que dictan las X, dando por hecho que las etiquetas Y no tienen ningún papel a la hora de construir la partición del árbol. La notación empleada será la siguiente:

- Denotaremos por $\{A_1, \ldots, A_N\}$ a las regiones dadas por las hojas de un árbol.
- Por N_j se denota el número de X_i que se encuentran en la región A_j. Dado que $\{A_1, \ldots, A_N\}$ conforma una partición de $\mathbb{R}^d$, se tiene que $\sum_{j=1}^{N} N_j = n$.
- Denotemos por $\text{diam}(A_j)$ a la máxima distancia entre dos puntos de A_j:

$$\text{diam}(A_j) = \sup_{x,y \in A_j} \|x - y\|.$$

Las decisiones se tomarán en función de la mayoría, por lo que, para cada $x \in A_j$, $1 \leq j \leq N$, la regla es la siguiente:

$$g_n(x) = \begin{cases} 1 & \text{si } \sum_{i:\ X_i \in A_j} Y_i > \sum_{i:\ X_i \in A_j} (1 - Y_i), x \in A_j, \\ 0 & \text{en otro caso.} \end{cases} \tag{1.3}$$

Bajo ciertas condiciones naturales puede probarse la consistencia de este tipo de árboles de decisión, tal y como veremos en el Teorema 1.2.6.

Denotemos por $A(x)$ al conjunto de la partición $\{A_1, \ldots, A_N\}$ en el que se encuentra x y por $N(x)$ al número de datos que se encuentran en este conjunto:

$$N(x) = \sum_{i=1}^{n} I_{\{X_i \in A(x)\}}.$$

El siguiente teorema nos dice que probar la consistencia de la regla es posible cuando las regiones de la partición son lo suficientemente pequeñas (para que cambios locales de la distribución puedan ser detectados), pero al mismo tiempo lo suficientemente grandes para que contengan un número grande de puntos para que la clasificación sea efectiva. La demostración puede consultarse en Devroye et al. (2013).

Teorema 1.2.6 (Devroye et al. (2013),Teorema 6.1)**.** *Dado un árbol de clasificación natural cuya partición está determinada en exclusiva por lo que dictan las X y de la forma descrita en (1.3). Entonces, $\mathbb{E}(L_n) = L^*$ si*

i) $\text{diam}(A(X)) \to 0$ *en probabilidad,*

ii) $N(X) \to \infty$ *en probabilidad.*

Las condiciones impuestas por el Teorema 1.2.6 se verifican para un caso particular de árbol: la regla del k-espaciado en una dimensión. Definámosla formalmente.

Sea $k < n$ un entero positivo, denotemos por $X_{(1)}, \ldots, X_{(n)}$ a la muestra ordenada tal que

$$X_{(1)} \leq X_{(2)} \leq \cdots \leq X_{(n)}.$$

La partición de $\mathbb{R}$ en $N = \lceil n/k \rceil$ intervalos $A_1, \ldots, A_N$ se hace de la siguiente forma: para cada $j = 1, \ldots, N-1$, A_j es aquel intervalo que satisface que

$$X_{(k(j-1)+1)}, \ldots, X_{(kj)} \in A_j,$$

mientras que A_N (el intervalo situado más a la derecha) es aquel que:

$$X_{(k(N-1)+1)}, \ldots, X_{(n)} \in A_N.$$

Los extremos de todos los intervalos se toman, por simplicidad, como el punto medio entre el valor más a la derecha del intervalo inferior y el punto más a la izquierda del intervalo superior. La regla de clasificación del k-espaciado se define entonces como:

$$g_n(x) = \begin{cases} 1 & \text{si } \sum_{i=1}^n I_{\{X_i \in A(x), Y_i = 1\}} > \sum_{i=1}^n I_{\{X_i \in A(x), Y_i = 0\}}, \\ 0 & \text{en otro caso.} \end{cases} \tag{1.4}$$

Si n y k tienden a infinito de forma que $k/n \to 0$, es decir, el número de intervalos considerado tiende a infinito, entonces la regla del k-espaciado es una dimensión es consistente, tal y como muestra el siguiente resultado.

Teorema 1.2.7 (Devroye et al. (2013), Teorema 20.1)**.** *Sea g_n la regla descrita en (1.4). Si se asume que X tiene densidad f en $\mathbb{R}$, entonces, $\mathbb{E}(L_n) = L^*$ cuando $k \to \infty$ y $k/n \to 0$ cuando $n \to \infty$. Es decir, g_n es consistente.*

Para su demostración basta comprobar que se satisfacen las condiciones del Teorema 1.2.6. Se pueden consultar sus detalles en Devroye et al. (2013).

1.2.4. Regresión logística

A pesar de que, como su nombre indica, la regresión logística es un modelo de regresión, es habitual emplearlo como regla de clasificación aplicando un umbral. Esto se debe a que un modelo de regresión devuelve un valor continuo, mientras que una regla de clasificación da una suposición sobre a qué categoría pertenece un objeto. Esto sucede no solo con la regresión logística, sino en general con diferentes modelos de regresión. Presentaremos la regresión logística en base a Conde Amboage y Crujeiras Casais (2021).

Consideremos Y una variable dicotómica que toma toma los valores $Y = 1$ (éxito) e $Y = 0$ (fracaso). El modelo de regresión logística construye un modelo para

$$\pi(x) = \mathbb{P}(Y = 1 | X = x),$$

es decir, para la probabilidad de que $Y = 1$ condicionada a cada valor de la variable explicativa. Para poder considerar un modelo lineal, se aplica a $\pi(x)$ una función de enlace o función *link*. Es habitual emplear como enlace la función *logit*:

$$f(p) = \log\left(\frac{p}{1-p}\right), p \in [0, 1],$$

que transforma el intervalo $[0, 1]$ en toda la recta real. Sustituyendo p por $\pi(x)$ el modelo logístico es el siguiente:

$$\log\left(\frac{\pi(x, \beta)}{1 - \pi(x, \beta)}\right) = x'\beta,$$

donde x' es el vector que contiene los valores de las variables explicativas y β el vector de coeficientes que habría que estimar a través de la muestra (los modelos de regresión logística suelen ajustarse por máxima verosimilitud). Equivalentemente se puede formular de la manera siguiente:

$$\pi(x, \beta) = f^{-1}(x'\beta) = \frac{e^{x'\beta}}{1 + e^{x'\beta}}.$$

Así, si $x'\beta$ es positivo, la probabilidad de que $Y = 1$ cuando $X = x$ es superior a 0.5 (respetando la asíntota horizontal en 1), mientras que si es negativo entonces dicha probabilidad es inferior a 0.5 (respetando la asíntota horizontal en cero).

Un posible umbral para la discriminación logística podría ser $\pi(x, \beta) = 0.5$, clasificando como $Y = 1$ si $\pi(x, \beta)$ es mayor o igual que 0.5 y como $Y = 0$ en caso contrario, es decir:

$$g(x) = \begin{cases} 1 & \text{si } \pi(x,\beta) \geq 0.5 \Leftrightarrow x'\beta \geq 0, \\ 0 & \text{en otro caso.} \end{cases}$$

1.2.5. Otros clasificadores

Otros clasificadores usualmente empleados son los siguientes:

- **Support vector machine (SVM):**

  La técnica de *support vector machine* define un hiperplano que maximiza la distancia entre los puntos de las diferentes clases. Para presentar este método hemos recurrido a Friedman et al. (2009). Este clasificador se define como sigue:

  $$g(x) = \begin{cases} 1 & \text{si } a^T x + a_0 > 0, \\ 0 & \text{en otro caso,} \end{cases}$$

  donde a es un vector unitario, es decir, $\|a\| = 1$.

  Cuando las clases son separables (no se solapan en el espacio de características), la regla de clasificación (SVM) se puede formular de forma equivalente como el siguiente problema de optimización:

  $$\begin{aligned} \underset{a, a_0, \|a\|=1}{\text{maximizar}} \quad & M \\ \text{sujeto a} \quad & (x_i^T a + a_0) \geq M, \text{ para todo } i \text{ tal que } y_i = 1 \\ & (x_i^T a + a_0) \leq -M, \text{ para todo } i \text{ tal que } y_i = 0 \end{aligned}$$

  donde M representa la distancia del hiperplano a cada una de las clases de entrenamiento. Por tanto, $2M$ es el margen entre clases. Una forma más conveniente de formularlo es la siguiente:

  $$\begin{aligned} \underset{a, a_0}{\text{minimizar}} \quad & \|a\| \\ \text{sujeto a} \quad & (x_i^T a + a_0) \geq 1, \text{ para todo } i \text{ tal que } y_i = 1 \\ & (x_i^T a + a_0) \leq -1, \text{ para todo } i \text{ tal que } y_i = 0 \end{aligned}$$

  Nótese que la relación entre M y a es que $M = \frac{1}{\|a\|}$. Expresado de esta forma el problema puede verse como un problema de optimización convexo.

  Cuando las clases no son separables, una forma de lidiar con el solapamiento es maximizar M, pero permitiendo que algunos puntos estén en el lado equivocado del margen.

  Para ello se definen las variables de holgura $\xi = (\xi_1, \ldots, \xi_n)$ y se modifican las restricciones del problema de optimización de las siguientes posibles formas:

  - La primera de ellas mide el solapamiento en distancia real desde el margen:

    $$(x_i^T a + a_0) \geq M - \xi_i \text{ cuando } y_i = 1$$
    $$(x_i^T a + a_0) \leq -(M - \xi_i) \text{ cuando } y_i = 0$$

- La siguiente opción mide el solapamiento en distancia relativa:

$$(x_i^T a + a_0) \geq M(1 - \xi_i) \text{ cuando } y_i = 1$$
$$(x_i^T a + a_0) \leq -M(1 - \xi_i) \text{ cuando } y_i = 0$$

De forma habitual se escoge la segunda opción por proporcionar un problema de optimización convexo.

Se impone además que para todo i, $\xi_i \geq 0$ y $\sum_{i=1}^n \xi_i \leq c$ siendo c una constante. Como se produce un error de clasificación cuando $\xi_i > 1$, al acotar la suma por c se impone que el número de errores sobre la muestra de entrenamiento sea, como mucho, c.

En conclusión, el problema se formularía de la siguiente manera:

$$\begin{aligned} \text{minimizar} \quad & \|a\| \\ \text{sujeto a} \quad & (x_i^T a + a_0) \geq 1 - \xi_i, \text{ para todo } i \text{ tal que } y_i = 1 \\ & (x_i^T a + a_0) \leq -(1 - \xi_i), \text{ para todo } i \text{ tal que } y_i = 0 \\ & \xi_i \geq 0, \sum_{i=1}^n \xi_i \leq c. \end{aligned}$$

- **Random forest:**

La primera presentación de los bosques aleatorios se encuentra en Ho (1995), pero no fue hasta 2001 cuando se introdujeron de forma adecuada en Breiman (2001). La técnica de *random forest* o bosque aleatorio emplea un conjunto de árboles para la clasificación. De esta forma, utiliza los resultados devueltos por cada uno de los árboles y los agrega. Puede interpretarse de esta forma como una generalización de los árboles de clasificación aunque, como consecuencia, se pierde en interpretabilidad al añadir complejidad al modelo. Para llevar a cabo el proceso de agregación se emplea la técnica conocida como *bagging* o *agregación boostrap*. Esta técnica permite reducir la varianza de una función de predicción estimada. Consiste en entrenar a cada árbol con una muestra aleatoria de la muestra original y clasificar una observación en función de lo que dicta la mayoría de los árboles.

- **Regla del histograma cúbico:**

La regla del histograma cúbico hace una partición de $\mathbb{R}^d$ en cubos del mismo tamaño y clasifica una observación x en función de lo que dicta la mayoría de los Y_i del cubo en el que se encuentra x. Sea $\mathcal{P} = \{A_{n1}, A_{n2}, \dots\}$ una partición de $\mathbb{R}^d$ en cubos de tamaño $h_n > 0$ (regiones de la forma $\prod_{i=1}^d [k_i h_n, (k_i + 1)h_n]$ con k_i números enteros). Denotando por $A_n(x)$ al conjunto de la partición al que pertenece x. Se define formalmente la regla como:

$$g_n(x) = \begin{cases} 1 & \text{si } \sum_{i=1}^n I_{\{X_i \in A_n(x), Y_i = 1\}} > \sum_{i=1}^n I_{\{X_i \in A_n(x), Y_i = 0\}}, \\ 0 & \text{en otro caso.} \end{cases}$$

Bajo ciertos supuestos puede probarse la consistencia universal de dicha regla, como establece el siguiente teorema.

Teorema 1.2.8 (Devroye et al. (2013), Teorema 6.2)**.** *Si $h_n \to 0$ y $nh_n^d \to \infty$ cuando $n \to \infty$, entonces la regla del histograma cúbico es universalmente consistente.*

La prueba de este resultado consiste en verificar que se cumplen las condiciones del Teorema 1.2.6. La consistencia universal de forma general de la regla del histograma cúbico también fue probada.

Capítulo 2

Regla de k-vecinos más próximos

El algoritmo de k-vecinos más próximos es un clasificador no paramétrico de aprendizaje supervisado clásico que emplea la proximidad de una observación al conjunto de entrenamiento para determinar su clasificación. Es empleado en multitud de áreas de conocimiento no solo por ser uno de los primeros clasificadores, sino por ser también fácil de comprender y sencillo de implementar.

En este capítulo se dará una definición formal de la regla de k-vecinos más próximos y, una vez definida, el objetivo será estudiar la consistencia universal de la misma. En primer lugar, se analizará el comportamiento de la regla cuando se mantiene el número de vecinos fijo y n se hace tender a infinito. Se verá en este caso que la esperanza de L_n converge a un cierto valor L_{kNN} próximo al error de Bayes, presentando además algunas desigualdades relacionando L_{kNN} y L^*. Continuando con el estudio del rendimiento para k fijo, se ha estudiado el comportamiento de la regla cuando el error de Bayes es muy próximo a cero. Posteriormente se permitirá que k tienda a infinito tal que $k/n \to 0$ cuando $n \to \infty$, probando finalmente que en esta situación sí se da la convergencia universal. Para todo ello nos hemos basado tanto en Devroye et al. (2013) como en Biau y Devroye (2015). De estos libros hemos extraído los resultados que más nos interesan para nuestro objetivo. Asimismo, las demostraciones aquí mostradas siguen en muchos casos los pasos recogidos en ellos, pero han sido detalladas y explicadas de forma más extensa para su mejor comprensión.

Recordemos que $(X_1, Y_1), \ldots, (X_n, Y_n)$ es una secuencia de pares aleatorios independientes, idénticamente distribuidos y con la misma distribución que (X, Y). Además, al clasificador basado en $(X_1, Y_1), \ldots, (X_n, Y_n)$ lo denotábamos por g_n. Dado x, la regla de k-vecinos más próximos asigna a las k observaciones más cercanas a x un peso uniforme de $1/k$, mientras que al resto les asigna un peso nulo. De esta forma, en función de la clase mayoritaria entre las k observaciones más cercanas se decide la clasificación. Formalmente la regla se define como sigue.

Definición 2.0.1. Se define la regla de *k-vecinos más próximos*, *k-NN* o *knn* (*k-nearest neighbor rule*) como:

$$g_n(x) = \begin{cases} 1 & \text{si } \sum_{i=1}^n \omega_{ni} I_{\{Y_i=1\}} > \sum_{i=1}^n \omega_{ni} I_{\{Y_i=0\}}, \\ 0 & \text{en otro caso.} \end{cases}$$

donde $\omega_{ni} = 1/k$ si X_i se encuentra entre los k vecinos más próximos de x, es decir, si la distancia $\|x - X_i\|$ es una de las k menores entre $\|x - X_1\|, \ldots, \|x - X_n\|$, y $\omega_{ni} = 0$ en otro caso. En caso de empate en la distancia, el candidato con el índice más pequeño se considera más cercano a x.

Observación 2.0.2. *Para evitar posibles empates es conveniente escoger k un número impar.*

Observación 2.0.3. *De forma arbitraria se considerará la distancia Euclídea para definir los k vecinos más próximos de x. Sin embargo, muchas de las propiedades asintóticas que se tratarán siguen siendo válidas para muchas otras métricas.*

Una vez definida la regla el siguiente paso lógico es estudiar sus propiedades para concluir si presenta un comportamiento adecuado. Como ya hemos comentado, una de las características importantes para que una regla sea considerada conveniente es su consistencia. Es por ello por lo que se estudiará el comportamiento de k-NN cuando se hace tender a infinito el número de observaciones.

2.1. Rendimiento para k fijo

Comenzaremos estudiando el caso en el cual k se mantendrá fijo y además es impar. Para ello, en lo sucesivo se considerará $x \in \mathbb{R}^d$ fijo y se considerará al conjunto de datos $(X_1, Y_1), \ldots, (X_n, Y_n)$ reordenado en función de los valores de $\|X_i - x\|$ de forma creciente:

$$(X_{(1)}(x), Y_{(1)}(x)), \ldots, (X_{(n)}(x), Y_{(n)}(x)), \text{ o simplemente } (X_{(1)}, Y_{(1)}), \ldots, (X_{(n)}, Y_{(n)}).$$

Por tanto, se dirá que $X_{(k)}(x)$ es el k-vecino más próximo de x cuando no haya confusión, en otras palabras, la distancia de $X_{(k)}(x)$ a x es la k-ésima más pequeña de entre todas las distancias de los X_i a x.

Definición 2.1.1. Denotemos la medida de probabilidad para X por μ, y sea $S_{x,\epsilon}$ la bola cerrada centrada en x de radio $\epsilon > 0$. El conjunto de todas las x con $\mu(S_{x,\epsilon}) > 0$ para todo $\epsilon > 0$ se denomina *soporte de* X o simplemente μ y se denotará por $\text{sop}(\mu)$.

El primer resultado que presentaremos sobre el rendimiento de la regla de k-vecinos más próximos demuestra una propiedad que resulta muy intuitiva: bajo ciertas condiciones, cuando el número de observaciones tiende a infinito la distancia entre el k-vecino más próximo de x y x tiende a cero. Esta cualidad resultará fundamental a la hora de probar la consistencia de la regla.

Lema 2.1.2 (Devroye et al. (2013), Lema 5.1)**.** *Si $X_{(k)}(x)$ es el k-vecino más próximo de x, entonces se verifican estas dos propiedades:*

i) Si $x \in \text{sop}(\mu)$ y $\lim_{n\to\infty} k/n = 0$, entonces $\|X_{(k)}(x) - x\| \to 0$ con probabilidad uno.

ii) Si X es independiente de los datos y tiene medida de probabilidad μ, entonces $\|X_{(k)}(X) - X\| \to 0$ con probabilidad uno siempre que $k/n \to 0$.

Demostración. En primer lugar demostraremos la propiedad *i)* y, apoyándonos en ella, demostraremos la *ii)*.

i) Sea $\epsilon > 0$ y dado $x \in \text{sop}(\mu)$ entonces $\mu(S_{x,\epsilon}) > 0$. Además, $\|X_{(k)}(x) - x\| > \epsilon$ si y solo si

$$\frac{1}{n}\sum_{i=1}^{n} I_{\{X_i \in S_{x,\epsilon}\}} < \frac{k}{n}.$$

Por hipótesis tenemos que el lado derecho de la desigualdad converge a 0 cuando $n \to \infty$, y aplicando la ley fuerte de los grandes números el lado izquierdo de la desigualdad converge con probabilidad uno a la esperanza de $I_{\{X_i \in S_{x,\epsilon}\}}$, la cual es $\mu(S_{x,\epsilon}) > 0$. Por tanto, $\|X_{(k)}(x) - x\| \to 0$ con probabilidad uno.

ii) En primer lugar, nótese que $\mathbb{P}(X \in \text{sop}(\mu)) = 1$ y, por tanto, para todo $\epsilon > 0$ se tiene que:

$$\mathbb{P}(\|X_{(k)}(X) - X\| > \epsilon) = \mathbb{E}(I_{\{X \in \text{sop}(\mu)\}}\mathbb{P}(\|X_{(k)}(X) - X\| > \epsilon \mid X \in \text{sop}(\mu))).$$

Entonces, se puede aplicar el teorema de la convergencia dominada concluyendo que

$$\mathbb{E}(I_{\{X \in \text{sop}(\mu)\}}\mathbb{P}(\|X_{(k)}(X) - X\| > \epsilon \mid X \in \text{sop}(\mu))) \to 0.$$

Por consiguiente, $\|X_{(k)}(X) - X\| \to 0$ en probabilidad. Para probar la convergencia con probabilidad uno simplemente nos fijamos en que, cuando k no varía con n, entonces $\|X_{(k)}(X) - X\|$ es una sucesión monótona decreciente para $n \geq k$. Si $k \to \infty$ cuando $n \to \infty$ de forma que $\lim_{n\to\infty} k/n = 0$ entonces puede utilizare un argumento similar al ya visto. Empleando la notación $k = k_n$, $X_{(k_n,n)}(X) = X_{(k)}(X)$, entonces:

$$\sup_{m\geq n} \|X_{(k_m,m)}(X) - X\| \geq \|X_{(k_n,n)}(X) - X\|,$$

es una sucesión monótona decreciente de variables aleatorias. Por tanto, concluimos que converge en probabilidad y de forma casi segura, finalizando la demostración.

□

Observación 2.1.3. *Cabe destacar que el Lema 2.1.2 es válido no solo cuando k es fijo, sino siempre que $k \to \infty$ cuando $n \to \infty$ de forma que $\lim_{n\to\infty} k/n = 0$. Este aspecto permitirá emplear este resultado en la prueba de la consistencia universal de k-NN.*

A continuación introduciremos una regla auxiliar g'_n que nos permitirá seguir analizando el comportamiento asintótico de la regla de k-vecinos más próximos. En particular, veremos que el error asintótico de esta nueva regla coincide con el error asintótico de k-NN. La gran ventaja es que obtener la expresión de dicho error resulta mucho más fácil para el caso de g'_n. De esta forma se obtiene la expresión resultante para k-NN, que es el objetivo final.

Para definir g'_n debemos construir otra secuencia de datos que está fuertemente relacionada con la secuencia original pero que resultará más adecuada de analizar. En primer lugar, consideraremos la secuencia de pares aleatorios $(X_1, U_1), \ldots, (X_n, U_n)$ independientes, idénticamente distribuidos y con la misma distribución que (X, U), donde X tiene medida de probabilidad μ en los conjuntos de Borel de $\mathbb{R}^d$, y U es uniformemente distribuido en $[0, 1]$ e independiente de X. Si fijamos $Y_i = I_{\{U_i \leq \eta(X_i)\}}$, entonces $(X_1, Y_1), \ldots, (X_n, Y_n)$ son pares aleatorios, independientes, idénticamente distribuidos y con la misma distribución que (X, Y) determinada por (μ, η). Fijado $x \in \mathbb{R}^d$, se define

$$Y'_i(x) = I_{\{U_i \leq \eta(x)\}}.$$

Se obtienen entonces las secuencias $X_i, Y_i, Y'_i(x), U_i$, las cuales reordenando según los valores crecientes de $\|X_i - x\|$ derivan en nuevas secuencias $X_{(i)}(x), Y_{(i)}(x), Y'_{(i)}(x), U_{(i)}(x)$. En caso de que no haya confusión se eliminará el argumento x.

Definición 2.1.4. Una regla g_n se denomina *k-local* si, para $n \geq k$, es de la forma:

$$g_n(x) = \begin{cases} 1 & \text{si } \psi(x, Y_{(1)}(x), \ldots, Y_{(k)}(x)) > 0, \\ 0 & \text{en otro caso} \end{cases}$$

para cierta función ψ.

La regla de k-NN es k-local, simplemente tomando

$$\psi(x, Y_{(1)}, \ldots, Y_{(k)}) = \sum_{i=1}^{k} Y_{(i)}(x) - \frac{k}{2}.$$

Por tanto, empleando esta función ψ, la regla g'_n se define de la siguiente forma:

$$g'_n(x) = \begin{cases} 1 & \text{si } \psi(x, Y'_{(1)}(x), \ldots, Y'_{(k)}(x)) > 0, \\ 0 & \text{en otro caso} \end{cases}$$

Estudiar g'_n resulta ser más sencillo que estudiar g_n, aunque no tenga valor práctico al depender de $\eta(x)$ que es en general desconocido. La razón es que $Y'_{(1)}(x), \ldots, Y'_{(k)}(x)$ son independientes e idénticamente distribuidos mientras que $Y_{(1)}(x), \ldots, Y_{(k)}(x)$ no lo son. Esto se debe a que los $Y_{(i)}$ dependen de la ordenación de los $X_{(i)}$. Sin embargo, condicionado a x, los $Y'_{(i)}$ son independientes de la ordenación de los $X_{(i)}$, pues para su definición sólo se usa x.

La probabilidad de que g'_n y g_n clasifiquen de forma distinta una observación puede acotarse tal y como indica el siguiente lema. Esta acotación constituye el primer paso para demostrar que el error asintótico de esta nueva regla coincide con el error asintótico de k-NN.

Lema 2.1.5 (Devroye et al. (2013), Lema 5.2)**.** *Para todo x, $n \geq k$,*

$$\mathbb{P}(\psi(x, Y_{(1)}(x), \ldots, Y_{(k)}(x)) \neq \psi(x, Y'_{(1)}(x), \ldots, Y'_{(k)}(x))) \leq \sum_{i=1}^{k} \mathbb{E}(|\eta(x) - \eta(X_{(i)}(x))|)$$

y

$$\mathbb{P}(g_n(x) \neq g'_n(x)) \leq \sum_{i=1}^{k} \mathbb{E}(|\eta(x) - \eta(X_{(i)}(x))|).$$

Demostración. La demostración se sigue de las siguientes relaciones de contenido. Por una parte,

$$\begin{aligned} &\{\psi(x, Y_{(1)}(x), \ldots, Y_{(k)}(x)) \neq \psi(x, Y'_{(1)}(x), \ldots, Y'_{(k)}(x))\} \\ &\subseteq \{(Y_{(1)}(x), \ldots, Y_{(k)}(x)) \neq (Y'_{(1)}(x), \ldots, Y'_{(k)}(x))\}. \end{aligned}$$

Recordando que $Y'_i(x) = I_{\{U_i \leq \eta(x)\}}$ y que $Y_i = I_{\{U_i \leq \eta(X_i)\}}$, entonces,

$$\begin{aligned} &\{(Y_{(1)}(x), \ldots, Y_{(k)}(x)) \neq (Y'_{(1)}(x), \ldots, Y'_{(k)}(x))\} \\ &\subseteq \bigcup_{i=1}^{k} \{\eta(X_{(i)}(x)) \leq U_{(i)}(x) \leq \eta(x)\} \cup \bigcup_{i=1}^{k} \{\eta(x) \leq U_{(i)}(x) \leq \eta(X_{(i)}(x))\}, \end{aligned}$$

Además, teniendo en cuenta que los $U_{(i)}$ son uniformes en $[0, 1]$ y empleando la desigualdad de Boole se concluye la prueba. □

El siguiente resultado resultará clave para demostrar la consistencia universal de la regla de k-vecinos más próximos, tal y como veremos más adelante. No consta de una interpretación intuitiva, ya que servirá para garantizar que se cumple una condición técnica. Es por ello por lo que se omitirán los detalles de su demostración. Para consultar la prueba puede recurrirse a Stone (1977).

Lema 2.1.6 (Stone (1977))**.** *Sea X una variable aleatoria idénticamente distribuida a X_1 pero independiente de la secuencia de datos. Para cada función integrable f, cada n y cada $k \leq n$, se tiene:*

$$\sum_{i=1}^{k} \mathbb{E}(|f(X_{(i)}(X))|) \leq k\gamma_d \mathbb{E}(|f(X)|),$$

donde $\gamma_d \leq (1 + 2/\sqrt{2 - \sqrt{3}})^d - 1$ depende únicamente de la dimensión.

El siguiente teorema es un resultado auxiliar que será empleado para la demostración del lema que le sigue.

Teorema 2.1.7 (Devroye et al. (2013), Teorema A.8)**.** *Para toda medida de probabilidad ν en $\mathbb{R}^d$, el conjunto de funciones continuas con soporte compacto es denso en $L_p(\nu)$. Es decir, para cada $\epsilon > 0$ y cada $f \in L_p$ existe una función continua con soporte compacto g ($g \in L_p$ ya que todas las funciones continuas con soporte compacto están en L_p) tal que*

$$\int |f - g|^p \, d\nu < \epsilon.$$

El próximo resultado manifiesta que, para k fijo, $f(X_{(k)}(X))$ puede pensarse como $f(X)$ en la práctica, donde f es una función integrable. Esta propiedad constutuirá el último paso para demostrar que la probabilidad asintótica de error de g'_n coincide con la de k-NN. Se apoya en el Lema 2.1.6 y en el Teorema 2.1.7 para su demostración.

Lema 2.1.8 (Biau y Devroye (2015), Lema 18.1)**.** *Para toda función integrable f,*

$$\frac{1}{k}\sum_{i=1}^{k}\mathbb{E}(|f(X)-f(X_{(i)}(X))|)\to 0$$

cuando $n\to\infty$ siempre que $k/n\to 0$.

Demostración. Dado $\epsilon>0$, aplicando el Teorema 2.1.7 con $p=1$ existe una función $g\in L_1$ uniformemente continua (toda función continua en un compacto es uniformemente continua) que se anula en un conjunto acotado A, tal que

$$\int |f-g|\,d\nu<\epsilon\Rightarrow\mathbb{E}(|g(X)-f(X)|)<\epsilon. \tag{2.1}$$

Además, por ser g uniformemente continua para cada $\epsilon>0$, existe un $\delta>0$ tal que $\|x-z\|<\delta$ implica que $|g(x)-g(z)|<\epsilon$. Entonces,

$$\begin{aligned}&\frac{1}{k}\sum_{i=1}^{k}\mathbb{E}(|f(X)-f(X_{(i)}(X))|)\\&\quad\le\mathbb{E}(|f(X)-g(X)|)+\frac{1}{k}\sum_{i=1}^{k}\mathbb{E}(|g(X)-g(X_{(i)}(X))|)\\&\quad+\frac{1}{k}\sum_{i=1}^{k}\mathbb{E}(|g(X_{(i)}(X))-f(X_{(i)}(X))|)\end{aligned}$$

Al sumando $\frac{1}{k}\sum_{i=1}^{k}\mathbb{E}(|g(X_{(i)}(X))-f(X_{(i)}(X))|)$ podemos aplicarle el Lema 2.1.6, obteniendo así que

$$\frac{1}{k}\sum_{i=1}^{k}\mathbb{E}(|g(X_{(i)}(X))-f(X_{(i)}(X))|)\le\frac{1}{k}\sum_{i=1}^{k}k\gamma_d\mathbb{E}(|f(X)-g(X)|)=\gamma_d\mathbb{E}(|f(X)-g(X)|).$$

Por otra parte, el sumando $\frac{1}{k}\sum_{i=1}^{k}\mathbb{E}(|g(X)-g(X_{(i)}(X))|)$ podemos acotarlo teniendo en cuenta las dos casuísticas siguientes:

- Si $\|X-X_{(k)}(X)\|<\delta$ entonces $|g(X)-g(X_{(i)}(X))|<\epsilon$ para todo i. Entonces, se tiene que:
$$\frac{1}{k}\sum_{i=1}^{k}\mathbb{E}(|g(X)-g(X_{(i)}(X))|)\le\epsilon.$$
- Si $\|X-X_{(k)}(X)\|>\delta$ entonces, recordando que $\|g\|_\infty=\sup\{|g(x)|,\text{ con }x\text{ en el soporte de }g\}$, obtenemos:
$$\frac{1}{k}\sum_{i=1}^{k}\mathbb{E}(|g(X)-g(X_{(i)}(X))|)\le\|g\|_\infty\mathbb{P}(\|X-X_{(k)}(X)\|>\delta).$$

Por tanto,

$$\begin{aligned}\frac{1}{k}\sum_{i=1}^{k}\mathbb{E}(|f(X)-f(X_{(i)}(X))|)&\le(1+\gamma_d)\mathbb{E}(|f(X)-g(X)|)+\epsilon+\|g\|_\infty\mathbb{P}(\|X-X_{(k)}(X)\|>\delta)\\&\le(2+\gamma_d)\epsilon+o(1),\end{aligned}$$

donde la última desigualdad surge de aplicar el Lema 2.1.2 y (2.1). □

2.1.1. Probabilidad asintótica de error

Llegados a este punto estamos en condiciones de obtener una expresión de la probabilidad asintótica de error de la regla de k-NN apoyándonos en la relación existente entre la probabilidad de error asintótica de la regla basada en las Y_i y en la basada en las Y_i' definidas previamente.

Denotemos por $D_n' = ((X_1, Y_1, U_1), \ldots, (X_n, Y_n, U_n))$. Entonces, la probabilidad de error de la regla g_n basada en D_n' es:

$$\begin{aligned} L_n &= \mathbb{P}(g_n(X) \neq Y | D_n') \\ &= \mathbb{P}(\text{signo}(\psi(X, Y_{(1)}(X), \ldots, Y_{(k)}(X))) \neq \text{signo}(2Y-1) | D_n'). \end{aligned}$$

Se define análogamente

$$L_n' = \mathbb{P}(\text{signo}(\psi(X, Y_{(1)}'(X), \ldots, Y_{(k)}'(X))) \neq \text{signo}(2Y-1) | D_n').$$

Aplicando los Lemas 2.1.5 y 2.1.8 obtenemos que:

$$\begin{aligned} &\mathbb{E}(|L_n - L_n'|) \\ &\leq \mathbb{P}(\psi(X, Y_{(1)}(X), \ldots, Y_{(k)}(X)) \neq \psi(X, Y_{(1)}'(X), \ldots, Y_{(k)}'(X))) \\ &\leq \sum_{i=1}^{k} \mathbb{E}(|\eta(X) - \eta(X_{(i)}(X))|) \\ &= o(1). \end{aligned}$$

Como $L_n - L_n' \leq |L_n - L_n'|$, entonces $\mathbb{E}(L_n) - \mathbb{E}(L_n') = \mathbb{E}(L_n - L_n') \leq \mathbb{E}(|L_n - L_n'|) \leq o(1)$. Como consecuencia $\lim_{n\to\infty}(\mathbb{E}(L_n') - \mathbb{E}(L_n)) = 0$. Por tanto, sería únicamente necesario estudiar la regla g_n':

$$g_n'(x) = \begin{cases} 1 & \text{si } \psi(x, Y_{(1)}'(x), \ldots, Y_{(k)}'(x)) > 0, \\ 0 & \text{en otro caso.} \end{cases}$$

Equivalentemente:

$$g_n'(x) = \begin{cases} 1 & \text{si } \psi(x, Z_1, \ldots, Z_k) > 0, \\ 0 & \text{en otro caso} \end{cases}$$

donde $Z_i = Y_{(i)}'(x) = I_{\{U_i \leq \eta(x)\}}$. Entonces, $\mathbb{P}(Z_i = 1) = \mathbb{P}(U_i \leq \eta(x)) = \eta(x)$. Por tanto, $Z_1, \ldots, Z_k$ son variables independientes e idénticamente distribuidas a una Bernouilli de parámetro $\eta(x)$.

La utilidad de emplear L_n' se aprecia de forma clara considerando la regla del vecino más próximo, es decir, $k = 1$. Por una parte, $\psi(x, Z_1) = Z_1 - 1/2$. Por tanto,

$$g_n'(x) = \begin{cases} 1 & \text{si } Z_1 - 1/2 > 0 \iff Z_1 > 1/2 \iff Z_1 = 1, \\ 0 & \text{en otro caso} \end{cases}$$

Así,

$$\begin{aligned} \mathbb{E}(L_n') &= \mathbb{E}(\mathbb{P}(\text{signo}(Z_1 - 1/2) \neq \text{signo}(2Y-1) | D_n')) = \mathbb{E}(\mathbb{P}(Z_1 \neq Y | D_n')) = \mathbb{P}(Z_1 \neq Y) \\ &= \mathbb{E}(\mathbb{P}(Z_1 \neq Y | X)) = \mathbb{E}\left[\mathbb{P}(Z_1 = 1, Y = 0 | X) + \mathbb{P}(Z_0, Y = 1 | X)\right] = \mathbb{E}(2\eta(X)(1 - \eta(X))). \end{aligned}$$

De esta forma, se concluye el siguiente Teorema.

Teorema 2.1.9 (Devroye et al. (2013), Teorema 5.1)**.** *Para la regla del vecino más próximo se cumple que*

$$\lim_{n\to\infty} \mathbb{E}(L_n) = \mathbb{E}(2\eta(X)(1-\eta(X)) =: L_{NN},$$

independientemente de la distribución de (X, Y).

Se tiene entonces que el error asintótico de la regla del vecino más próximo es L_{NN}, pero, ¿cómo de grande es en comparación con L^*? Recoradando que $L^* = \mathbb{E}(\text{mín}\{\eta(X), 1-\eta(X)\})$ y teniendo en cuenta que

$$2\eta(X)(1-\eta(X)) \geq \text{mín}(\eta(X), 1-\eta(X))$$

entonces $L^* \leq L_{NN}$. Denotando por $A = \text{mín}\{\eta(X), 1-\eta(X)\}$, se cumple que:

$$\begin{aligned} A(1-A) &= \text{mín}\{\eta(X), 1-\eta(X)\}(1-\text{mín}\{\eta(X), 1-\eta(X)\}) \\ &= \text{mín}\{\eta(X), 1-\eta(X)\} \times \text{máx}\{\eta(X), 1-\eta(X)\} \\ &= \eta(X)(1-\eta(X)). \end{aligned}$$

Se obtienen entonces las siguientes desigualdades:

$$\begin{aligned} L^* \leq L_{NN} = \mathbb{E}(2\eta(X)(1-\eta(X))) &= 2\mathbb{E}(A(1-A)) \\ &\leq 2\mathbb{E}(A)\mathbb{E}(1-A) \\ &= 2L^*(1-L^*) \\ &\leq 2L^*, \end{aligned}$$

donde la desigualdad $2\mathbb{E}(A(1-A)) \leq 2\mathbb{E}(A)\mathbb{E}(1-A)$ se sigue de aplicar el siguiente resultado.

Teorema 2.1.10 (Devroye et al. (2013), Teorema A.19)**.** *Sea* X *una variable aleatoria de valor real,* $f(x)$ *una función monótona creciente de valor real y* $g(x)$ *una función mónotona decreciente de valor real. Entonces,*

$$\mathbb{E}(f(X)g(X)) \leq \mathbb{E}(f(X))\mathbb{E}(g(X)),$$

siempre que todas las esperanzas existan y sean finitas.

Recapitulando, tenemos que:

$$L^* \leq L_{NN} \leq 2L^*(1-L^*) \leq 2L^*.$$

Por tanto, L_{NN} es como mucho dos veces más grande que L^*, lo que significa que la regla del vecino más próximo es asintóticamente como mucho dos veces peor que la regla de Bayes. Puede observarse también que para errores de Bayes L^* pequeños estas desigualdades son especialmente convenientes.

Para k impar, se define L_{kNN} como:

$$L_{kNN} \coloneqq \mathbb{E}\left(\sum_{j=0}^{k} \binom{k}{j} \eta^j(X)(1-\eta(X))^{k-j} \left(\eta(X)I_{\{j<k/2\}} + (1-\eta(X))I_{\{j>k/2\}}\right)\right)$$

Demostraremos a continuación que el error asintótico de k-NN es exactamente L_{kNN}.

Teorema 2.1.11 (Biau y Devroye (2015), Teorema 18.1)**.** *Sea* k *un número fijo e impar. Para la regla de* k*-vecinos más próximos se cumple que*

$$\lim_{n\to\infty} \mathbb{E}(L_n) = L_{kNN}.$$

Demostración. Dado que $\lim_{n\to\infty}(\mathbb{E}(L'_n) - \mathbb{E}(L_n)) = 0$, debemos probar que

$$\lim_{n\to\infty} \mathbb{E}(L'_n) = L_{kNN}.$$

Para cada n,

$$\begin{aligned}
\mathbb{E}(L'_n) &= \mathbb{P}(g'_n(X) \neq Y) \\
&= \mathbb{P}(Z_1 + \cdots + Z_k > \frac{k}{2}, Y = 0) + \mathbb{P}(Z_1 + \cdots + Z_k < \frac{k}{2}, Y = 1) \\
&= \mathbb{P}(Z_1 + \cdots + Z_k > \frac{k}{2}, Z_0 = 0) + \mathbb{P}(Z_1 + \cdots + Z_k < \frac{k}{2}, Z_0 = 1),
\end{aligned}$$

siendo $Z_0, \ldots, Z_k$ variables independientes e idénticamente distribuidas a una Bernouilli de parámetro $\eta(X)$. Por tanto,

$$\begin{aligned}
&\mathbb{P}\Big(Z_1 + \cdots + Z_k > \frac{k}{2}, Z_0 = 0\Big) + \mathbb{P}\Big(Z_1 + \cdots + Z_k < \frac{k}{2}, Z_0 = 1\Big) \\
&= \mathbb{E}\left[\mathbb{P}\left(Z_1 + \cdots + Z_k > \frac{k}{2}, Z_0 = 0 \middle| X\right) + \mathbb{P}\left(Z_1 + \cdots + Z_k < \frac{k}{2}, Z_0 = 1 \middle| X\right)\right] \\
&= \mathbb{E}\left(\sum_{j=\lceil k/2\rceil}^{k} \binom{k}{j} \eta(X)^j (1-\eta(X))^{k-j}(1-\eta(X)) + \sum_{j=0}^{\lfloor k/2\rfloor} \binom{k}{j} \eta(X)^j (1-\eta(X))^{k-j}\eta(X)\right) \\
&= \mathbb{E}\left(\sum_{j=0}^{k} \binom{k}{j} \eta^j(X)(1-\eta(X))^{k-j}\left(\eta(X) I_{\{j<k/2\}} + (1-\eta(X)) I_{\{j>k/2\}}\right)\right) \\
&= L_{kNN}.
\end{aligned}$$

□

Teniendo en cuenta la expresión de L_{kNN}, se pueden obtener formulaciones análogas. Recordemos en primer lugar que si una variable aleatoria tiene distribución binomial con parámetros $n \in \mathbb{N}$ y p con $0 < p < 1$, $X \sim \text{Bin}(n, p)$, entonces:

$$\mathbb{P}(X = x) = \binom{n}{x} p^x (1-p)^{n-x}.$$

Por tanto,

$$\begin{aligned}
L_{kNN} &= \mathbb{E}\left(\eta(X)\mathbb{P}\Big(\text{Binomial}(k, \eta(X)) < \frac{k}{2}\Big| \ X\Big)\right) \\
&\quad + \mathbb{E}\left((1-\eta(X))\mathbb{P}\Big(\text{Binomial}(k, \eta(X)) > \frac{k}{2}\Big| \ X\Big)\right) \\
&= \mathbb{E}\left(\eta(X)\left[1 - \mathbb{P}\Big(\text{Binomial}(k, \eta(X)) > \frac{k}{2}\Big| \ X\Big)\right]\right) \\
&\quad + \mathbb{E}\left((1-\eta(X))\mathbb{P}\Big(\text{Binomial}(k, \eta(X)) > \frac{k}{2}\Big| \ X\Big)\right) \\
&= \mathbb{E}\left(\eta(X)\right) + \mathbb{E}\left((1-2\eta(X))\mathbb{P}\Big(\text{Binomial}(k, \eta(X)) > \frac{k}{2}\Big| \ X\Big)\right).
\end{aligned}$$

Además,

$$\begin{aligned}
L_{kNN} &= \mathbb{E}\left(\text{mín}\{\eta(X), 1-\eta(X)\}\right) \\
&\quad + \mathbb{E}\left((1 - 2\,\text{mín}\{\eta(X), 1-\eta(X)\})\mathbb{P}\Big(\text{Binomial}(k, \text{mín}\{\eta(X), 1-\eta(X)\}) > \frac{k}{2}\Big| \ X\Big)\right).
\end{aligned}$$

En efecto, si $\min\{\eta(X), 1-\eta(X)\} = \eta(X)$ la expresión es la misma que ya obtuvimos antes. Si $\min\{\eta(X), 1-\eta(X)\} = 1-\eta(X)$ veamos que podemos llegar a la misma expresión. Por una parte,

$$\begin{aligned}
&\mathbb{E}\,(1-\eta(X)) + \mathbb{E}\Big((1-2(1-\eta(X))\mathbb{P}\Big(\text{Binomial}(k, 1-\eta(X)) > \frac{k}{2}\Big|\ X\Big)\Big) \\
&= \mathbb{E}\,(1-\eta(X)) + \mathbb{E}\Big((2\eta(X)-1)\mathbb{P}\Big(\text{Binomial}(k, 1-\eta(X)) > \frac{k}{2}\Big|\ X\Big)\Big) \\
&= \mathbb{E}\,(1-\eta(X)) + \mathbb{E}\left((2\eta(X)-1)\left(\sum_{j>k/2}\binom{k}{j}(1-\eta(X))^j\eta(X)^{k-j}\right)\right).
\end{aligned}$$

Por otra parte, gracias al Binomio de Newton y a las propiedades de los números combinatorios se verifica que:

$$\begin{aligned}
&\left(\sum_{j>k/2}\binom{k}{j}(1-\eta(X))^j\eta(X)^{k-j}\right) + \left(\sum_{j>k/2}\binom{k}{j}\eta(X)^j(1-\eta(X))^{k-j}\right) \\
&= \left(\sum_{j>k/2}\binom{k}{j}(1-\eta(X))^j\eta(X)^{k-j}\right) + \left(\sum_{j<k/2}\binom{k}{j}(1-\eta(X))^j\eta(X)^{k-j}\right) \\
&= \left(\sum_{j=0}^{k}\binom{k}{j}(1-\eta(X))^j\eta(X)^{k-j}\right) \\
&= (1-\eta(X)+\eta(X))^k \\
&= 1.
\end{aligned}$$

Entonces,

$$\left(\sum_{j>k/2}\binom{k}{j}(1-\eta(X))^j\eta(X)^{k-j}\right) = 1 - \left(\sum_{j>k/2}\binom{k}{j}(1-\eta(X))^j\eta(X)^{k-j}\right).$$

Finalmente se obtiene que:

$$\begin{aligned}
&\mathbb{E}(1-\eta(X)) + \mathbb{E}\left((2\eta(X)-1)\left(\sum_{j>k/2}\binom{k}{j}(1-\eta(X))^j\eta(X)^{k-j}\right)\right) \\
&= \mathbb{E}(1-\eta(X)) + \mathbb{E}\left((2\eta(X)-1)\left(1-\left(\sum_{j>k/2}\binom{k}{j}(1-\eta(X))^j\eta(X)^{k-j}\right)\right)\right) \\
&= \mathbb{E}(\eta(X)) + \mathbb{E}\left((1-2\eta(X))\left(\sum_{j>k/2}\binom{k}{j}(1-\eta(X))^j\eta(X)^{k-j}\right)\right) \\
&= \mathbb{E}(\eta(X)) + \mathbb{E}\left((1-2\eta(X))\mathbb{P}\left(\text{Binomial}(k, \eta(X)) > \frac{k}{2}\Big|\ X\right)\right) \\
&= L_{kNN}.
\end{aligned}$$

Dado que estamos considerando que k es fijo, una pregunta que podríamos hacernos es cómo varía L_{kNN} cuando se cambia el valor de k. El Teorema 2.1.12 nos muestra que el error asintótico de la regla de k-vecinos más próximos (con k impar) disminuye a medida que aumentamos el número de vecinos, acercándose así a L^*.

Teorema 2.1.12 (Devroye et al. (2013), Teorema 5.4)**.** *Se verifica que:*

$$L^* \leq \cdots \leq L_{(2k+1)NN} \leq L_{(2k-1)NN} \leq \cdots \leq L_{3NN} \leq L_{NN} \leq 2L^*,$$

independientemente de la distribución considerada para (X, Y)*.*

Observación 2.1.13. *Cuando* $L^* = 0$ *se cumple que* $L_{NN} = L_{3NN} = L_{5NN} = \cdots = 0$*.*

2.1.2. Desigualdades para la probabilidad de error asintótica

A continuación se presentarán una serie de desigualdades para L_{kNN} que lo relacionen con L^*. Recordemos que $L_{kNN} = \mathbb{E}(\alpha_k(\eta(X)))$, siendo

$$\alpha_k(p) = \min\{p, 1-p\} + |1 - 2\min\{p, 1-p\}|\mathbb{P}\left(\text{Binomial}(k, \min\{p, 1-p\}) > \frac{k}{2}\right),$$

para k impar. El siguiente resultado nos servirá para probar la primera de las desigualdades.

Teorema 2.1.14 (Devroye et al. (2013), Teorema 8.1)**.** *[Desigualdad de Hoeffding] Sean* $X_1, \ldots, X_n$ *variables aleatorias independientes y acotadas tales que para cada* i*,* X_i *toma valores en* $[a_i, b_i]$ *con probabilidad 1. Denotemos por* S_n *a la suma de todas ellas:*

$$S_n = \sum_{i=1}^{n} X_i.$$

Entonces, para cada $\epsilon > 0$ *se tiene que*

$$\mathbb{P}(S_n - \mathbb{E}(S_n) \geq \epsilon) \leq e^{-2\epsilon^2 / \sum_{i=1}^{n}(b_i - a_i)^2}$$

y

$$\mathbb{P}(S_n - \mathbb{E}(S_n) \leq -\epsilon) \leq e^{-2\epsilon^2 / \sum_{i=1}^{n}(b_i - a_i)^2}.$$

Teorema 2.1.15 (Devroye et al. (2013), Teorema 5.6)**.** *Para la regla de* k*-vecinos más próximos con* k *impar se cumple que*

$$L_{kNN} \leq L^* + \frac{1}{\sqrt{ke}},$$

independientemente de la distribución de (X, Y)*.*

Demostración. Por una parte, teniendo en cuenta la definición de L_{kNN} se verifica que:

$$\begin{aligned} L_{kNN} - L^* &\leq \sup_{0 \leq p \leq 1/2} (1-2p)\mathbb{P}\left(\text{Binomial}(k, p) > \frac{k}{2}\right) \\ &= \sup_{0 \leq p \leq 1/2} (1-2p)\mathbb{P}\left(\text{Binomial}(k, p) - kp > \frac{k}{2} - kp\right). \end{aligned}$$

Considerando $X_1, \ldots, X_k$ variables de Bernoulli de parámetro p ($a_i = 0, b_i = 1$ para todo i), entonces S_k es una variable binomial de parámetros k, p con esperanza kp. Por consiguiente, podemos aplicar la desigualdad de Hoeffding del Teorema 2.1.14 con $\epsilon = \frac{k}{2} - kp$.

$$\begin{aligned} L_{kNN} - L^* &\leq \sup_{0 \leq p \leq 1/2} (1-2p)e^{-2k(\frac{1}{2}-p)^2} \\ &= \sup_{0 \leq p \leq 1} ue^{-ku^2/2}. \end{aligned}$$

Calculemos entonces el máximo de la función $g(u) = ue^{-ku^2/2}$ en $[0, 1]$:

$$g'(u) = e^{-ku^2/2} + ue^{-ku^2/2}(-ku) = (1 - ku^2)e^{-ku^2/2} = 0 \Leftrightarrow u = \frac{1}{\sqrt{k}}.$$

Por tanto, como $g(\frac{1}{\sqrt{k}}) = \frac{1}{\sqrt{ke}}$, se concluye el resultado. □

Presentamos a continuación dos desigualdades más relacionando L^* y L_{kNN} cuyas demostraciones omitiremos pues son similares a la anterior.

Teorema 2.1.16 (Devroye et al. (2013), Teorema 5.7)**.** *Para la regla de k-vecinos más próximos con k impar se cumple que*

$$L_{kNN} \leq L^* + \sqrt{\frac{2L_{NN}}{k}},$$

independientemente de la distribución de (X, Y).

Teorema 2.1.17 (Devroye et al. (2013), Teorema 5.8)**.** *Para la regla de k-vecinos más próximos con* $k \geq 3$ *impar se cumple, independientemente de la distribución de* (X, Y)*, que:*

$$L_{kNN} \leq L^* \left(1 + \frac{\gamma}{\sqrt{k}}(1 + O(k^{-1/6}))\right),$$

donde $\gamma = \sup_{r>0} 2r\mathbb{P}(N > r) = 0.33994241\ldots$, N *es normal* $(0, 1)$*, y* $O(\cdot)$ *se refiere a* $k \to \infty$.

2.1.3. L^* cercano a cero y $L^* = 0$

De especial interés es estudiar el comportamiento de la regla de k-NN cuando el error de Bayes es pequeño, ya que es entonces cuando se esperarían errores asintóticos pequeños para todo k. Recordemos que el error asintótico de la regla de k-vecinos más próximos es el siguiente:

$$\begin{aligned} L_{kNN} &:= \mathbb{E}\left(\sum_{j=0}^{k} \binom{k}{j} \eta^j(X)(1-\eta(X))^{k-j} \left(\eta(X) I_{\{j<k/2\}} + (1-\eta(X)) I_{\{j>k/2\}}\right)\right) \\ &= \mathbb{E}\left(\sum_{j>k/2} \binom{k}{j} \eta(X)^j (1-\eta(X))^{k-j+1} + \sum_{j<k/2} \binom{k}{j} \eta(X)^{j+1} (1-\eta(X))^{k-j}\right) \\ &= \sum_{j<k/2} \binom{k}{j} \mathbb{E}\left([\eta(X)(1-\eta(X))]^{j+1} [(1-\eta(X))^{k-2j-1} + \eta(X)^{k-2j-1}]\right), \end{aligned}$$

donde la última igualdad se sigue de las propiedades de los números combinatorios. Veamos con qué se corresponde para ciertos valores de k:

- $k = 1$

$$\begin{aligned} L_{NN} &= \binom{1}{0} \mathbb{E}\left([\eta(X)(1-\eta(X))]^1 [(1-\eta(X))^{1-1} + \eta(X)^{1-1}]\right) \\ &= \mathbb{E}(2\eta(X)(1-\eta(X))) \\ &= \mathbb{E}(\alpha_1(\eta(X))), \end{aligned}$$

siendo $\alpha_1(p) = 2p(1-p)$.

- $k = 3$

$$\begin{aligned}
L_{3NN} &= \binom{3}{0}\mathbb{E}\Big([\eta(X)(1-\eta(X))]^1[(1-\eta(X))^{3-1}+\eta(X)^{3-1}]\Big)\\
&+ \binom{3}{1}\mathbb{E}\Big([\eta(X)(1-\eta(X))]^{1+1}[(1-\eta(X))^{3-2-1}+\eta(X)^{3-2-1}]\Big)\\
&= \mathbb{E}\Big([\eta(X)(1-\eta(X))][(1-\eta(X))^2+\eta(X)^2]\Big)\\
&+ 6\mathbb{E}\Big([\eta(X)(1-\eta(X))]^2\Big)\\
&= \mathbb{E}\Big([\eta(X)(1-\eta(X))][(-2\eta(X)(1-\eta(X))+1))]\Big)\\
&+ 6\mathbb{E}\Big([\eta(X)(1-\eta(X))]^2\Big)\\
&= \mathbb{E}\Big(\eta(X)(1-\eta(X))\Big) + 4\mathbb{E}\Big([\eta(X)(1-\eta(X))]^2\Big)\\
&= \mathbb{E}(\alpha_3(\eta(X))),
\end{aligned}$$

  siendo $\alpha_3(p) = p(1-p) + 4[p(1-p)]^2$.

- $k = 5$

  Reagrupando términos al igual que en los anteriores casos se obtiene que:

$$\begin{aligned}
L_{5NN} &= \mathbb{E}(\eta(X)(1-\eta(X))) + \mathbb{E}([\eta(X)(1-\eta(X))]^2) + 12\mathbb{E}([\eta(X)(1-\eta(X))]^3)\\
&= \mathbb{E}(\alpha_5(\eta(X))),
\end{aligned}$$

  siendo $\alpha_3(p) = p(1-p) + [p(1-p)]^2 + 12[p(1-p)]^3$.

Este procedimiento puede hacerse de forma general para L_{kNN} con k impar, teniendo en cuenta que $p^a + (1-p)^a$ es función de $p(1-p)$ para un entero a. Cuando p tiende a 0 se tiene que:

$$\begin{aligned}
&\alpha_1(p) = 2p(1-p) \sim 2p,\\
&\alpha_3(p) = p(1-p) + 4p^2(1-p)^2 = p(1-p)(1+4p) = p + 3p^2 - 4p^3 \sim p + 3p^2,\\
&\alpha_5(p) = p(1-p) + p^2(1-p)^2 + 12p^3(1-p)^3 = p + 10p^3 - 35p^4 + 36p^5 - 12p^6 \sim p + 10p^3.
\end{aligned}$$

Por otra parte, $L^* = \mathbb{E}(\text{mín}\{\eta(X), 1-\eta(X)\}) = \mathbb{E}(\alpha_\infty(\eta(X)))$, donde $\alpha_\infty = \text{mín}\{p, 1-p\} \sim p$. Si se asume que $\eta(x) = p$ para todo x, entonces:

$$\begin{aligned}
&L_{NN} \sim 2L^*\\
&L_{3NN} \sim L^* + 3L^{*2}\\
&L_{5NN} \sim L^* + 10L^{*3}.
\end{aligned}$$

Por tanto, si por ejemplo se considera $L^* = 0.05$, entonces $L_{NN} \sim 0.1$, mientras que $L_{3NN} \sim 0,0575$ y $L_{5NN} \sim 0,05125$. Se observa que se ha experimentado una mayor disminución de L_{NN} a L_{3NN} que de L_{3NN} a L_{5NN}. Es decir, cuando L^* es pequeño, la regla de tres vecinos más próximos parece tener una menor probabilidad de error asintótica que la de un único vecino, pero poco se gana al aumentar el número de vecinos considerados a cinco. Es por este motivo por el que la regla de tres vecinos más próximos es altamente recomendada. Sin embargo, esta tendencia decreciente de L_{kNN} que observamos a medida que k crece no implica que la regla del vecino más próximo deba ser ignorada. En efecto, existen distribuciones para las que para todo n, la regla 1-NN es mejor que la regla k-NN para todo $k \geq 3$, como se muestra en el siguiente ejemplo.

Ejemplo 2.1.18. Considérense S_0 y S_1 esferas de radio 1 centradas en dos puntos que distan más de 2 unidades. Además, si $Y = 1$, entonces consideremos X uniforme en S_1, mientras que si $Y = 0$, entonces X es uniforme en S_0. A mayores, supongamos que las clases son equiprobables, es decir, $\mathbb{P}(Y = 1) = \mathbb{P}(Y = 0) = 1/2$. Dado n, entonces:

- Para la regla 1-NN:

$$\mathbb{E}(L_n) = \mathbb{P}(Y=0, Y_1=1, Y_2=1, \ldots, Y_n=1) + \mathbb{P}(Y=1, Y_1=0, Y_2=0, \ldots, Y_n=0) = \frac{1}{2^n}.$$

- Para la regla k-NN con $k \geq 3$ y k impar:

$$\begin{aligned}\mathbb{E}(L_n) &= \mathbb{P}\left(Y=0, \sum_{i=1}^{n} I_{\{Y_i=0\}} \leq \lfloor k/2 \rfloor\right) + \mathbb{P}\left(Y=1, \sum_{i=1}^{n} I_{\{Y_i=1\}} \leq \lfloor k/2 \rfloor\right) \\ &= \mathbb{P}\left(\text{Binomial}(n, 1/2) \leq \lfloor k/2 \rfloor\right) \\ &= \frac{1}{2^n} \sum_{j=0}^{\lfloor k/2 \rfloor} \binom{n}{j} > \frac{1}{2^n}.\end{aligned}$$

Por tanto, se concluye así que para este ejemplo la regla 1-NN es mejor que la regla k-NN para todo $k \geq 3$.

Por otra parte, teniendo en cuenta la Observación 2.1.13, se tiene que si $L^* = 0$, entonces

$$L_{NN} = L_{3NN} = L_{5NN} = \cdots = 0.$$

Por tanto, para cada k fijo, la regla de k-vecinos más próximos es consistente cuando $L^* = 0$.

2.1.4. Otras versiones de la regla estándar

Hasta el momento se ha considerado k impar, evitando así los empates. Además, el peso de los datos más cercanos a la observación considerada es siempre el mismo. Es por ello por lo que surgen variaciones de la regla de k-vecinos más próximos clásica. Si se quiere considerar k par existen extensiones en la literatura (consultar Devroye et al. 2013; Devijver 1978) con diferentes estrategias de desempate. Sin embargo, considerando k fijo, la probabilidad de error de la regla de k-vecinos no disminuye para los valores pares. Por tanto, en términos asintóticos, nada se gana al considerar un vecino más a la hora de decidir la clasificación. Asimismo surgen extensiones para el caso en el que las k observaciones más cercanas no tienen el mismo peso: la regla de k-vecinos más próximos ponderada (ver Devroye et al. 2013). Las ponderaciones pueden ser escogidas para cada caso y siguiendo diferentes criterios. Asintóticamente la regla de k-vecinos más próximos estándar presenta un mejor comportamiento. Sin embargo, puede ocurrir que para cierto tamaño de muestra emplear ponderaciones no uniformes resulte preferible. Es más, cuando k también crece con n las ponderaciones no uniformes son preferibles.

Otra variante de la regla de k-vecinos más próximos es la regla de (k,l)-vecinos más próximos, la cual difiere con la primera en que tan solo se clasifica en una clase cuando, como mínimo, haya l observaciones de dicha clase entre los k más próximos, siendo $l > k/2$. En caso contrario se produce una indecisión.

2.2. Rendimiento con k tendiendo a infinito

Una vez estudiado el caso en el que k se mantenía fijo, nos centraremos en el estudio del comportamiento de la regla cuando permitimos que k varíe con n de forma que $k/n \to 0$. En particular, veremos que bajo este supuesto la regla es universalmente consistente. Para demostrarlo se emplea el conocido como Teorema de Stone.

Considérese en primer lugar una regla basada en la estimación de la probabilidad a posteriori η de la siguiente forma:

$$\eta_n(x) = \sum_{i=1}^{n} I_{\{Y_i=1\}} W_{ni}(x) = \sum_{i=1}^{n} Y_i W_{ni}(x),$$

donde los pesos $W_{ni}(x) = W_{ni}(x, X_1, \ldots, X_n)$ son no negativos y suman uno, es decir,

$$\sum_{i=1}^{n} W_{ni}(x) = 1, W_{ni}(x) \geq 0.$$

La regla en cuestión se define de la siguiente manera:

$$g_n(x) = \begin{cases} 0 & \text{si } \sum_{i=1}^{n} I_{\{Y_i=1\}} W_{ni}(x) \leq \sum_{i=1}^{n} I_{\{Y_i=0\}} W_{ni}(x), \\ 1 & \text{en otro caso.} \end{cases}$$

Análogamente,

$$g_n(x) = \begin{cases} 0 & \text{si } \sum_{i=1}^{n} Y_i W_{ni}(x) \leq 1/2, \\ 1 & \text{en otro caso.} \end{cases}$$

Dado que se supone que los X_i que están más cerca de x aportan más información sobre $\eta(x)$, los pesos empleados suelen ser más grandes en una vecindad de X. Por tanto, η_n es aproximadamente una frecuencia relativa (ponderada) de los X_i que tienen etiqueta 1 entre los puntos en la vecindad de X.

Teorema 2.2.1 (Stone (1977)). *Si para cualquier distribución de X los pesos W_{ni} satisfacen estas tres condiciones:*

i) Para toda función medible no negativa f que satisface que $\mathbb{E}(f(X)) < \infty$ existe una constante c tal que

$$\mathbb{E}\left(\sum_{i=1}^{n} W_{ni}(X) f(X_i)\right) \leq c\mathbb{E}(f(X)).$$

ii) Para toda $a > 0$,

$$\lim_{n\to\infty} \mathbb{E}\left(\sum_{i=1}^{n} W_{ni}(X) I_{\{\|X_i - X\| > a\}}\right) = 0.$$

iii)

$$\lim_{n\to\infty} \mathbb{E}\left(\max_{1\leq i\leq n} W_{ni}(X) = 0\right).$$

Entonces, g_n es universalmente consistente.

Para comprender mejor los requerimientos que impone el Teorema de Stone para obtener una regla de clasificación universalmente consistente considérense las siguientes observaciones:

- La condición *i)* es una condición técnica.
- La condición *ii)* indica que solamente deben considerarse puntos de una vecindad de X cada vez más pequeña a la hora de hacer el promedio, ya que el peso de los X_i fuera de cualquier bola de radio fijo $a > 0$ centrada en X debe tender a cero.
- Por su parte, la condición *iii)* impone que ningún X_i tenga un peso demasiado grande y, como consecuencia, excesiva aportación a la estimación.

La aplicación directa del Teorema de Stone 2.2.1 a la regla de k-vecinos más próximos prueba su consistencia universal.

Teorema 2.2.2 (Biau y Devroye (2015), Corolario 19.1)**.** *Si $k \to \infty$ y $k/n \to 0$, entonces*

$$\mathbb{E}(L_n) \to L^*,$$

independientemente de la distribución considerada. Es decir, la regla de k-vecinos más próximos es universalmente consistente.

Demostración. Comprobemos que se cumplen las condiciones del Teorema de Stone 2.2.1. En primer lugar nótese que los pesos $W_{ni}(X)$ son iguales a $1/k$ si X_i se encuentra entre los k vecinos más próximos a X e igual a 0 en caso contrario. Por lo tanto, la condición *iii)* resulta inmediata al considerar $k \to \infty$. Para comprobar la condición *ii)* nótese que si:

$$\lim_{n\to\infty} \mathbb{P}(\|X_{(k)}(X) - X\| > \epsilon) = 0,$$

entonces

$$\lim_{n\to\infty} \mathbb{E}\left(\sum_{i=1}^{n} W_{ni}(X) I_{\{\|X_i - X\| > \epsilon\}}\right) = 0,$$

siendo $X_{(k)}(x)$ el k vecino más próximo de x entre $X_1, \ldots, X_n$. Por tanto, aplicando el Lema 2.1.2 concluimos que esto es cierto para todo $\epsilon > 0$ siempre que $k/n \to 0$, lo cual se tiene por hipótesis. Por último, hay que comprobar que para toda función medible no negativa f que satisface que $\mathbb{E}(f(X)) < \infty$ existe una constante c tal que

$$\mathbb{E}\left(\sum_{i=1}^{n} \frac{1}{k} I_{\{X_i \text{ es uno de los } k \text{ vecinos más próximos}\}} f(X_i)\right) \leq c\mathbb{E}(f(X)).$$

Esta condición se deduce directamente del Lema de Stone 2.1.6 tomando $c = \gamma_d$. Se ha demostrado así la consistencia universal de la regla de k-vecinos más próximos cuando $k \to \infty$ y $k/n \to 0$. □

Yendo un paso más allá puede probarse también que la regla de k-vecinos más próximos es fuertemente consistente siempre que se asuma la existencia de densidad para μ (para evitar empates en las distancias) tal y como indica el siguiente resultado. Su demostración puede encontrarse en Devroye et al. (2013) y fue realizada por Devroye y Györfi (1985), y Zhao (1987).

Teorema 2.2.3 (Devroye et al. (2013), Teorema 11.1)**.** *Supongamos que μ tiene densidad. Si $k \to \infty$ y $k/n \to 0$ entonces, para cada $\epsilon > 0$ existe un n_0 tal que para cada $n > n_0$*

$$\mathbb{P}(L_n - L^* > \epsilon) \leq 2e^{-n\epsilon^2/(72\gamma_d)},$$

donde γ_d es una constante que depende únicamente de la dimensión. Por tanto, la regla de k-NN es fuertemente consistente.

Bajo una estrategia adecuada de desempates se puede demostrar también la consistencia universal fuerte.

Capítulo 3

Árboles de Expansión Mínima

Dado un grafo no dirigido, el problema del árbol de expansión mínima es un problema combinatorio que consiste en encontrar un árbol que minimice la suma de los costes de los arcos que conectan a los nodos del grafo. Hay múltiples ámbitos en los que se aplican los árboles de expansión mínima. Una de las aplicaciones más clásicas es en la construcción de redes, como pueden ser redes eléctricas, redes telefónicas o redes de carreteras. Para este último caso se podrían considerar diferentes ciudades que se quieren conectar mediante la construcción de una red de carreteras, de forma que se minimice la longitud total de la red. Otra de las aplicaciones habituales de los árboles de expansión mínima es como aproximación de la solución óptima del problema del viajante de comercio.

En este capítulo presentaremos formalmente los árboles de expansión mínima, ya que la regla de clasificación que propondremos en el siguiente capítulo está basada en ellos. A pesar de todas las aplicaciones mencionadas de los árboles de expansión mínima este es un ámbito en el que no se han empleado todavía. Introduciremos una serie de definiciones y resultados básicos para después presentar tres algoritmos que se emplean para su obtención: el algoritmo de Prim, el de Kruskal y el de Boruvka.

3.1. Nociones básicas de teoría de grafos

Con la finalidad de hacer el trabajo más autocontenido se presentarán a continuación una serie de definiciones básicas sobre teoría de grafos. Para ello hemos seguido las referencias Ahuja et al. (1988) y González Díaz (2018).

Definición 3.1.1. Un *grafo* G es un par (N, M) donde N es un conjunto de elementos llamados nodos o vértices y M es un conjunto de elementos que representan arcos o aristas.

El número total de nodos de un grafo G se denotará por n y el número total de arcos por m, es decir, $|N| = n$ y $|M| = m$. Existen dos tipos de grafos dependiendo de cómo sean sus arcos, los cuales definimos a continuación.

Definición 3.1.2. Se dice que un grafo G es *dirigido* u *orientado* si $M \subset N \times N$, es decir, los arcos son pares ordenados. El arco (i, j) significa que este comienza en el nodo i y termina en el j.

De forma gráfica, que un grafo sea dirigido se representará empleando flechas.

Definición 3.1.3. Se dice que un grafo G es *no dirigido* si M está compuesto por subconjuntos de N de dos elementos. Por tanto, $\{i, j\}$ y $\{j, i\}$ representan el mismo arco.

En este caso, la representación del gráfico se hará sin flechas. No se considerarán grafos con aristas de la forma (i, i) o $\{i, i\}$, es decir, lazos.

Ejemplo 3.1.4. Consideremos el grafo dirigido $G = (N, M)$ donde $N = \{1, 2, 3, 4\}$ y

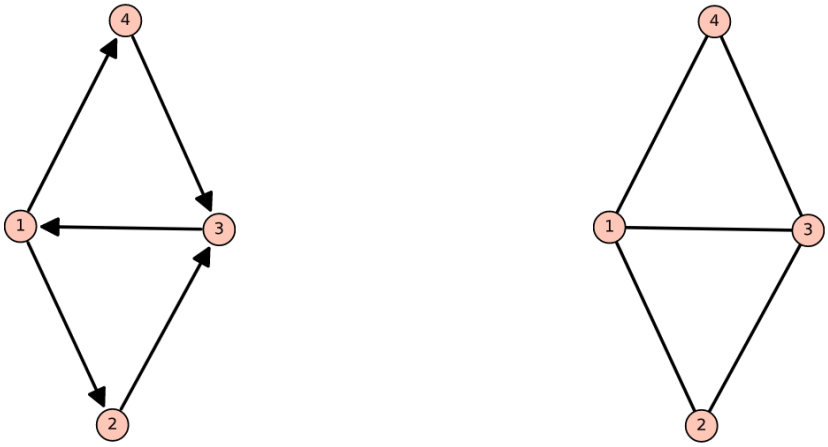
$$M = \{(1, 2), (2, 3), (1, 4), (4, 3), (3, 1)\}.$$

Una posible representación es la de la Figura 3.1a.

Un ejemplo de grafo no dirigido es $G' = (N', M')$ donde $N' = \{1, 2, 3, 4\}$ y

$$M' = \{\{1, 2\}, \{2, 3\}, \{1, 4\}, \{4, 3\}, \{3, 1\}\}.$$

Una posible representación es la de la Figura 3.1b.



(a) Grafo dirigido. (b) Grafo no dirigido.

Figura 3.1: Ejemplos de grafos.

Definición 3.1.5. Un grafo G se dice *completo* si todas las aristas posibles entre los nodos de G están presentes.

Observación 3.1.6. *Dado que no se permiten lazos en los grafos, si el grafo es completo y dirigido el número de aristas es* $n(n-1)$*, mientras que si fuera completo y no dirigido el número de arcos sería* $n(n-1)/2$.

Definición 3.1.7. Un subgrafo de un grafo $G = (N, M)$ es un grafo $G' = (N', M')$, donde $N' \subseteq N$ y $M' \subseteq M$. En otras palabras, un subgrafo G' es un grafo que tiene todos sus nodos y arcos en el grafo G.

Definición 3.1.8. Un subgrafo $G' = (N', M')$ de un grafo $G = (N, M)$ se dice que es un *subgrafo de expansión* si $G' = N$ y $M' \subset M$.

Definición 3.1.9. Si la arista (i, j) está presente en un grafo G entonces se dice que:

- Los nodos i y j son *adyacentes*.
- Los nodos i y j son *incidentes* con la arista (i, j).
- La arista (i, j) es *incidente* con los nodos i y j.
- El arco (i, j) *emana* del nodo i.
- El arco (i, j) es un *arco saliente* del nodo i y un *arco entrante* del nodo j.

Definición 3.1.10. Se define el *grado de un nodo* cualquiera de un grafo G como el número de arcos incidentes con él.

Definición 3.1.11. Sea G un grafo no dirigido y sea $(a_1, a_2, \ldots, a_r)$ una secuencia de aristas distintas en G. Si existen vértices $(v_0, v_1, \ldots, v_r)$ tales que, para $l \in \{1, 2, \ldots, r\}$, $a_l = \{v_{l-1}, v_l\}$, se dice que la secuencia es una *cadena*.

Definición 3.1.12. Una cadena en la cual $v_0 = v_r$ se dice *cadena cerrada*.

Definición 3.1.13. Un *camino* es una cadena en la que todos los nodos de la secuencia son distintos.

Definición 3.1.14. Una cadena cerrada en la que no hay más vértices coincidentes que el primero y el último se dice *circuito* o *ciclo*.

Las definiciones 3.1.11, 3.1.12, 3.1.13 y 3.1.14 pueden extenderse inmediatamente para grafos orientados.

Ejemplo 3.1.15. Consideremos en primer lugar el grafo 3.1a. La cadena $(1, 4, 3, 1)$ es una cadena cerrada donde no hay más nodos coincidentes que el primero y el último, por lo que se trata de un ciclo. Por su parte, la cadena $(1, 2, 3)$ es un camino (ya que todos los vértices son distintos) y la cadena $(1, 4, 3, 2)$ es no dirigida. Si consideramos el grafo 3.1b, la cadena $(1, 2, 3, 1, 4)$ no es un camino (ya que se repite un vértice).

Definición 3.1.16. Un grafo G se dice *conexo* si para cada par de nodos existe una cadena no dirigida que los une.

Definición 3.1.17. Un grafo se denomina *acíclico* si no contiene ningún ciclo.

Definición 3.1.18. Un grafo G se dice que es un *árbol* si es conexo y no contiene ciclos (no dirigidos), es decir, es un grafo acíclico conexo.

Observación 3.1.19. *La definición 3.1.18 puede adaptarse a grafos dirigidos, pero normalmente se aplica el concepto de árbol a grafos no dirigidos.*

Definición 3.1.20. Un *subárbol* de un árbol T es un subgrafo conexo de T.

Definición 3.1.21. Un nodo de un árbol con grado igual a uno se denomina *nodo hoja*.

Definición 3.1.22. Un *árbol de expansión* de un grafo G es un árbol que además es un subgrafo de expansión de G. Dicho árbol será un subgrafo conexo minimal en el sentido de que no habrá otro subgrafo conexo con menos aristas. Los arcos que pertenecen a un árbol de expansión se denominan *arcos arbóreos*, mientras que los restantes se denominan *arcos no arbóreos*.

Ejemplo 3.1.23. Si se considera de nuevo el grafo 3.1b, el subgrafo de expansión $G'' = (N'', M'')$ donde $N'' = \{1, 2, 3, 4\}$ y $M'' = \{\{1, 2\}, \{1, 3\}, \{1, 4\}\}$ es un árbol de expansión de G'. Una posible representación es la que encontramos en la Figura 3.2.

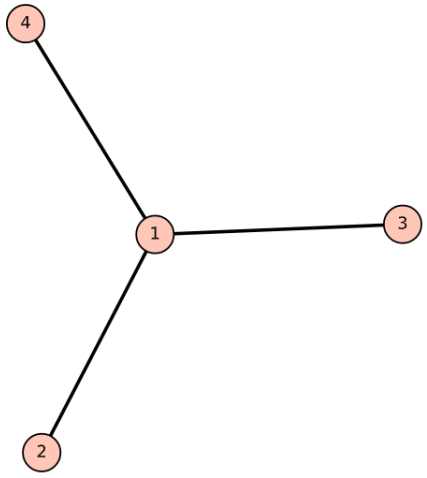


Figura 3.2: Árbol de expansión.

El siguiente resultado nos permite caracterizar a los árboles de diferentes formas, todas ellas equivalentes.

Proposición 3.1.24. *Dado un grafo no dirigido $G = (N, M)$, las siguientes propiedades son equivalentes:*

i) *G es un árbol.*

ii) *G es conexo y tiene $n - 1$ aristas.*

iii) *G no tiene ciclos y tiene $n - 1$ aristas.*

iv) *G no tiene ciclos y, si añadimos una arista cualquiera, se formaría un ciclo (y solo uno).*

v) *G es conexo y, si eliminamos una arista cualquiera, deja de serlo.*

vi) *Cada par de nodos de G están unidos por un único camino.*

vii) *G tiene al menos menos dos nodos hoja.*

viii) *G contiene un único camino entre dos nodos cualesquiera.*

3.2. El problema del árbol de expansión mínima

Una vez presentados todos estos conceptos sobre teoría de grafos estamos en condiciones de definir el problema del árbol de expansión mínima.

Definición 3.2.1. Dado un grafo no dirigido $G = (N, M)$ con costes c_k asociados a los arcos; el *problema del árbol de expansión mínima* consiste en encontrar un árbol de expansión de G de coste mínimo.

Dado un conjunto $X \subseteq N$, se denotará por $M(X)$ al conjunto de arcos formados por nodos de X. Teniendo en cuenta la Proposición 3.1.24, el problema del árbol de expansión mínima se puede expresar de la siguiente forma:

$$\begin{aligned} \text{minimizar} \quad & \sum_{k \in M} c_k f_k \\ \text{sujeto a} \quad & \sum_{k \in M} f_k = n - 1 \\ & \sum_{k \in M(X)} f_k \leq |X| - 1, \ X \subsetneq N \\ & f_k \in \{0, 1\}, \ k \in M. \end{aligned}$$

Analicemos la interpretación de la función objetivo y de las restricciones:

i) $f_k \in \{0, 1\}, \ k \in M$: estas restricciones nos indican que debemos elegir si se incluye el arco k en el árbol ($f_k = 1$) o no ($f_k = 0$), por lo que se trata de un problema de programación entera con variables binarias.

ii) Función objetivo $\sum_{k \in M} c_k f_k$: se suman todos los costes correspondientes a arcos para los cuales $f_k = 1$, es decir, los que se van a incluir en el árbol.

iii) $\sum_{k \in M} f_k = n - 1$: esta restricción nos indica que el número de arcos del árbol debe ser exactamente igual a $n - 1$.

iv) $\sum_{k \in M(X)} f_k \leq |X| - 1, \ X \subsetneq N$: estas restricciones nos garantizan que no se formen ciclos, ya que no habrá ningún subconjunto de N con tantas aristas como nodos.

Así, las restricciones *iii)* y *iv)* garantizan que se trate de un árbol por la Proposición 3.1.24.

A pesar de tratarse de un problema de programación entera, el siguiente resultado muestra que se trata de un problema más fácil de resolver de lo que puede parecer en primera instancia.

Proposición 3.2.2. *Toda solución básica factible del problema de programación lineal asociado al problema del árbol de expansión mínima tiene todos sus valores enteros.*

Como consecuencia de la Proposición 3.2.2, al resolver el problema relajado encontraríamos la solución del problema original entero. Sin embargo, el número de restricciones del problema del árbol de expansión mínima crece de forma exponencial con el número de nodos, ya que tendremos tantas restricciones de la forma $\sum_{k\in M(X)} f_k \leq |X| - 1$ como subconjuntos de N, lo cual es del orden de 2^n. Gracias a los algoritmos eficientes que se presentarán a continuación se concluirá finalmente que el problema del árbol de expansión mínima es un problema fácil de resolver, es decir, que los algoritmos resuelven el problema en tiempo polinomial. En particular, presentaremos los algoritmos de Boruvka, de Kruskal y de Prim.

3.2.1. Algoritmo de Boruvka

Históricamente, el primer algoritmo en aparecer fue el de Boruvka (Boruvka 1926). Para presentarlo debemos introducir previamente el concepto de componente conexa.

Definición 3.2.3. Dado un grafo $G = (N, M)$, decimos que $S \subset N$ es una *componente conexa* inducida por G si se cumplen las siguientes condiciones:

- Para cada $i, j \in S$, existe un camino en G de i a j.
- Para cada $i \in S$ y $j \in N\backslash S$, no hay ningún camino en G de i a j.

Denotaremos por $P(N, G)$ a la partición de N en componentes conexas inducida por G.

La idea del algoritmo es ir añadiendo de forma secuencial a cada componente conexa el arco con menor coste que la una con otras componentes conexas. El algoritmo finaliza cuando todos los vértices del grafo pertenecen a la misma componente. Se introduce de manera formal a continuación.

Algorithm 1 Algoritmo de Boruvka

Input: Grafo no dirigido $G = (N, M)$
Output: Árbol de expansión mínima de G.

1: **ETAPA 1**: Se define $G^0 = \emptyset$. Para cada nodo $i \in N$ se escoge otro nodo j_i tal que

$$c_{ij_i} = \min_{k\in N\backslash\{i\}} \{c_{ik}\}.$$

Si hay varios arcos que verifican esto, se selecciona uno indistintamente, siempre que no se formen ciclos. Se define $G^1 = \bigcup_{i\in N}\{\{i, j_i\}\}$. Se pueden dar dos situaciones:

- $P(N, G^1) = N$, en cuyo caso el algoritmo finaliza y G^1 es un árbol de mínimo coste.
- $P(N, G^1) \neq N$, en cuyo caso se va a la siguiente etapa.

2: **ETAPA p+1**: Se asume que se tiene definido G^p. Para cada $S \in P(N, G^p)$ se esocge $\{i, j_i\}$ tal que

$$c_{ij_i} = \min_{k\in S, l\in N\backslash S} \{c_{kl}\}.$$

Si hay varios arcos que verifican esto, se selecciona uno indistintamente sin formar ciclos. Se define $G^{p+1} = G^p \cup \left(\bigcup_{S\in P(N,G^p)}\{\{i, j_i\}\}\right)$. Se pueden dar dos situaciones:

- $P(N, G^{p+1}) = N$, en cuyo caso el algoritmo finaliza y G^{p+1} es un árbol de mínimo coste.
- $P(N, G^1) \neq N$, en cuyo caso el algoritmo continua en la etapa $p + 2$.

Ejemplo 3.2.4. Veamos en qué se traduce el Algoritmo de Boruvka aplicándolo al grafo de la Figura 3.3.

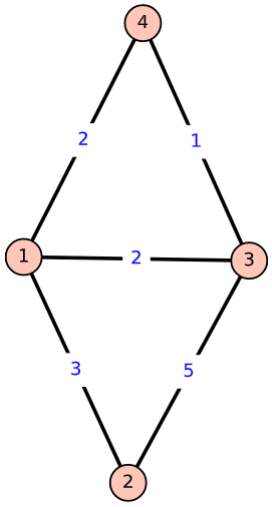


Figura 3.3: Grafo no dirigido con costes asociados a sus aristas.

- Etapa 1: Definimos $G^0 = \emptyset$. Para cada nodo $i \in N$ se escoge otro nodo j_i tal que

$$c_{ij_i} = \min_{k \in N \setminus \{i\}} \{c_{ik}\}.$$

  - Nodo 1: $c_{1j_1} = \min\{c_{12}, c_{14}, c_{13}\} = c_{13}$. Por tanto, $\{1, j_1\} = \{1, 3\}$.
  - Nodo 2: $c_{2j_2} = \min\{c_{21}, c_{23}\} = c_{21}$. Por tanto, $\{2, j_2\} = \{2, 1\}$.
  - Nodo 3: $c_{3j_3} = \min\{c_{34}, c_{31}, c_{32}\} = c_{34}$. Por tanto, $\{3, j_3\} = \{3, 4\}$.
  - Nodo 4: $c_{4j_4} = \min\{c_{41}, c_{43}\} = c_{43}$. Por tanto, $\{4, j_4\} = \{4, 3\}$.

  Se define $G^1 = \bigcup_{i \in N} \{\{i, j_i\}\}$. Como $P(N, G^1) = N$ el algoritmo finaliza y G^1 es un árbol de expansión mínima, representado en rojo la Figura 3.4.

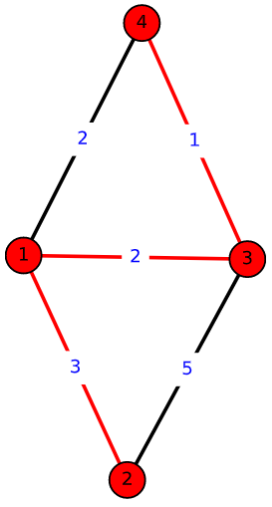


Figura 3.4: Árbol de expansión mínima resultante de aplicar el Algoritmo de Boruvka.

3.2.2. Algoritmo de Kruskal

El siguiente algoritmo que trataremos es el de Kruskal (Kruskal 1956). Este algoritmo, al igual que el de Boruvka, es un algoritmo voraz. Sin embargo, su manera de proceder es diferente: ordena los arcos del grafo de menor a mayor coste y los va añadiendo de forma secuencial y sin formar ciclos. El proceso termina cuando se hayan añadido $n-1$ aristas. A continuación se presentan sus etapas de manera formal.

Algorithm 2 Algoritmo de Kruskal

Input: Grafo no dirigido $G = (N, M)$
Output: Árbol de expansión mínima de G.

1: **ETAPA 1**: Se escoge un arco $\{i, j\} \in M$ tal que

$$c_{ij} = \min_{\{k,l\} \in M} \{c_{kl}\}.$$

Si hay varios arcos que verifican esto, se selecciona uno indistintamente. Se define $\{i^1, j^1\} = \{i, j\}$, $M = M \backslash \{\{i, j\}\}$ y $G^1 = \{\{i^1, j^1\}\}$.

2: **ETAPA p+1**: Se asume que se tiene definido M y G^p. Se escoge el arco $\{i, j\} \in M$ tal que

$$c_{ij} = \min_{\{k,l\} \in M} \{c_{kl}\}.$$

Si hay varios arcos que verifican esto, se selecciona uno indistintamente. Dos posibles escenarios:

- $G^p \cup \{\{i, j\}\}$ tiene un ciclo. Entonces $M = M \backslash \{\{i, j\}\}$ y G^p se queda igual. Se vuelve al principio de la etapa $p + 1$.
- $G^p \cup \{\{i, j\}\}$ no tiene ciclos. Entonces $\{i^{p+1}, j^{p+1}\} = \{i, j\}$, $M = M \backslash \{\{i, j\}\}$ y $G^{p+1} = G^p \cup \{\{i^{p+1}, j^{p+1}\}\}$. Se va a la etapa $p + 2$.

Cuando $|G^{p+1}| = n - 1$ el proceso termina.

El algoritmo de Kruskal termina tras $n - 1$ etapas. Así, G^{n-1} es el árbol de expansión mínima obtenido empleando el algoritmo de Kruskal.

Ejemplo 3.2.5. Veamos cómo se aplica el Algoritmo de Kruskal al grafo de la Figura 3.3.

- Etapa 1: Escogemos un arco $\{i, j\}$ tal que

$$c_{ij} = \min_{\{k,l\} \in M} \{c_{kl}\}.$$

Escogemos entonces el arco $\{3, 4\}$. Por tanto, $\{i^1, j^1\} = \{3, 4\}$, $M = M \backslash \{\{3, 4\}\}$ y $G^1 = \{\{3, 4\}\}$, representado en rojo en la Figura 3.5a.

- Etapa 2: Escogemos un arco $\{i, j\}$ tal que

$$c_{ij} = \min_{\{k,l\} \in M} \{c_{kl}\}.$$

Se escoge entonces el arco $\{1, 4\}$. $G^1 \cup \{\{1, 4\}\}$ no tiene ciclos, por lo que se define $\{i^2, j^2\} = \{1, 4\}$, $M = M \backslash \{\{1, 4\}\}$ y $G^2 = G^1 \cup \{1, 4\}$, representado en rojo en la Figura Figura 3.5b.

- Etapa 3: Escogemos un arco $\{i, j\}$ tal que

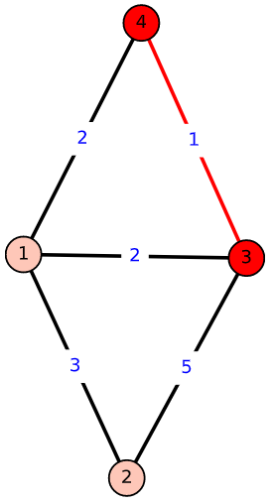
$$c_{ij} = \min_{\{k,l\} \in M} \{c_{kl}\}.$$

Se escoge entonces el arco $\{1,3\}$. $G^2 \cup \{\{1,3\}\}$ tiene un ciclo, como se puede ver en la Figura 3.5c. Como consecuencia, se define $M = M \backslash \{\{1,3\}\}$ y se vuelve al principio de la etapa 3. Escogemos un arco $\{i,j\}$ tal que

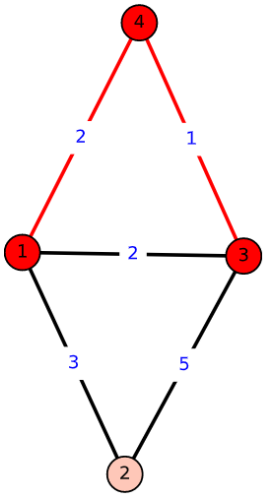
$$c_{ij} = \min_{\{k,l\} \in M} \{c_{kl}\}.$$

Se escoge entonces el arco $\{1,2\}$. $G^2 \cup \{\{1,2\}\}$ no tiene ciclos. Entonces, $\{i^3, j^3\} = \{1,2\}$, $M = M \backslash \{\{1,2\}\}$ y $G^3 = G^2 \cup \{1,2\}$, representado en rojo en la Figura 3.5d.

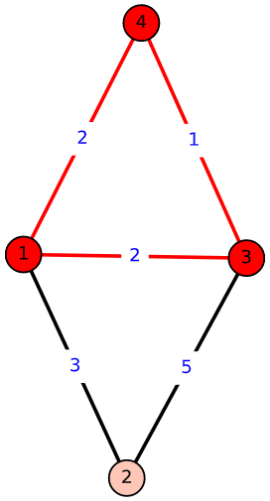
Por tanto, $G^3 = \{\{3,4\},\{1,4\},\{1,2\}\}$ es un árbol de expansión mínima proporcionado por el algoritmo de Kruskal, representado en la Figura 3.5d.



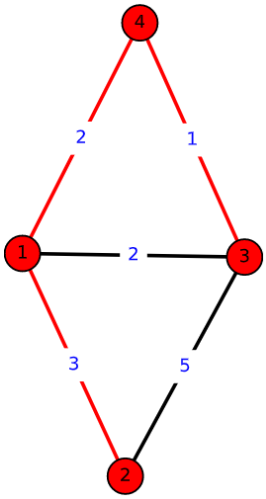
(a) Etapa 1 del Algoritmo de Kruskal.



(b) Etapa 2 del Algoritmo de Kruskal.



(c) Etapa 3 del Algoritmo de Kruskal.



(d) Etapa 3 del Algoritmo de Kruskal.

Figura 3.5: Etapas del Algoritmo de Kruskal.

3.2.3. Algoritmo de Prim

Por último presentaremos el Algoritmo de Prim (Prim 1957), el cual también es un algoritmo voraz. Para construir el árbol de expansión mínima el Algoritmo de Prim va eligiendo etapa a etapa

la arista de menor coste que una un nodo del árbol en construcción con uno que aún no se encuentre en él, procurando que no se formen ciclos. Una vez seleccionada, agrega la nueva arista al árbol con el vértice correspondiente. Es posible iniciar este proceso en cualquier nodo del grafo. Se describen a continuación las diferentes etapas.

Algorithm 3 Algoritmo de Prim

Input: Grafo no dirigido $G = (N, M)$
Output: Árbol de expansión mínima de G.

1: **ETAPA 1**: Se escoge un nodo arbitrario i y se escoge un arco $\{i, j\}$ tal que

$$c_{ij} = \min_{j \in N \setminus \{i\}} \{c_{ij}\}.$$

Si hay varios arcos que verifican esto, se selecciona uno indistintamente. Se define $X^1 = \{i, j\}$ y $G^1 = \{\{i, j\}\}$.

2: **ETAPA p+1**: Se asume que se tiene definido $X^p \subset N$ y G^p. Se escoge un arco $\{k, l\}$ con $k \in X^p$ y $l \in N \setminus X^p$ tal que

$$c_{kl} = \min_{i \in X^p, j \in N \setminus X^p} \{c_{ij}\}.$$

Si hay varios arcos que verifican esto, se selecciona uno indistintamente. Se define $X^{p+1} = X^p \cup \{l\}$ y $G^{p+1} = G^p \cup \{\{k, l\}\}$. Cuando $X^{p+1} = N$ el proceso termina.

El algoritmo de Prim se completa en $n-1$ etapas y G^{n-1} es el árbol de expansión mínima obtenido empleando dicho algoritmo.

Ejemplo 3.2.6. Apliquemos el Algoritmo de Prim al grafo de la Figura 3.3.

- Etapa 1: Escogemos el nodo 1 y se toma un arco $(1, j)$ tal que

$$c_{1j} = \min_{j \in N \setminus \{1\}} \{c_{1j}\}.$$

En caso de que hubiera varios arcos satisfaciendo esta condición, se elige uno de ellos. En efecto, hay dos arcos que la satisfacen, el $\{1, 4\}$ y el $\{1, 3\}$. Escogemos el arco $\{1, 4\}$. Por tanto, $X^1 = \{1, 4\}$ y $G^1 = \{\{1, 4\}\}$, representado en rojo en la Figura 3.6a.

- Etapa 2: Se escoge un arco $\{k, l\}$ con $k \in X^1 = \{1, 4\}$ y $l \in N \setminus X^1 = \{2, 3\}$ tal que

$$c_{kl} = \min_{i \in X^1, j \in N \setminus X^1} \{c_{ij}\}.$$

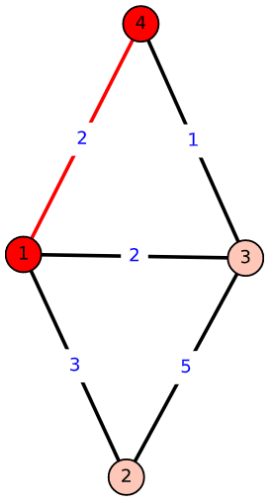
Se escoge entonces el arco $\{3, 4\}$. Así, $X^2 = \{1, 4, 3\}$ y $G^2 = \{\{1, 4\}, \{3, 4\}\}$, representado en rojo en la Figura 3.6b.

- Etapa 3: Se escoge un arco $\{k, l\}$ con $k \in X^2 = \{1, 4, 3\}$ y $l \in N \setminus X^1 = \{2\}$ tal que

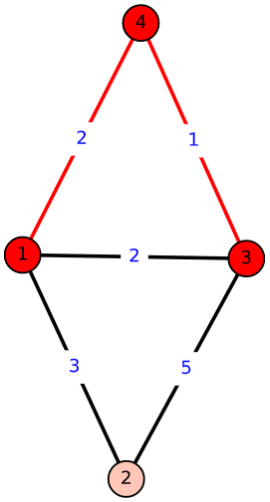
$$c_{kl} = \min_{i \in X^2, j \in N \setminus X^2} \{c_{ij}\}.$$

Se escoge entonces el arco $\{1, 2\}$. Así, $X^2 = \{1, 4, 3, 2\}$ y $G^3 = \{\{1, 4\}, \{3, 4\}, \{1, 2\}\}$.

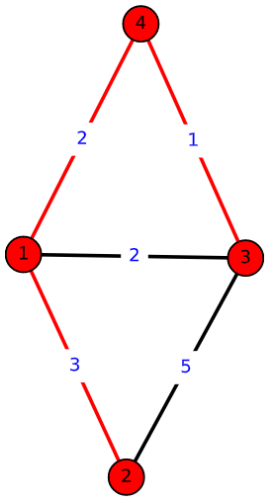
Por tanto, $G^3 = \{\{1, 4\}, \{3, 4\}, \{1, 2\}\}$ es el árbol de expansión mínima proporcionado por el algoritmo de Prim, representado en la Figura 3.6c.



(a) Etapa 1 del Algoritmo de Prim.



(b) Etapa 2 del Algoritmo de Prim.



(c) Etapa 3 del Algoritmo de Prim.

Figura 3.6: Etapas del Algoritmo de Prim.

Como acabamos de ver en el ejemplo, asociado a un mismo grafo pueden existir varios árboles de expansión mínima, pero todos ellos tendrán el mismo coste.

Es importante comentar que la complejidad computacional de estos algoritmos depende de ciertos detalles de la implementación en los que no vamos a entrar. Sin embargo, en ningún caso su complejidad computacional es peor que $O(n^2)$, que es el valor que tomaremos de referencia en el resto de la memoria.

Capítulo 4

Regla basada en árboles de expansión mínima

Tal y como acabamos de ver, existen muchas aplicaciones de los árboles de expansión mínima. Una de las principales razones es su fácil construcción gracias a algoritmos voraces como el de Prim. Sin embargo, no hemos encontrado en la literatura referencias que utilicen árboles de expansión mínima para definir reglas de clasificación. Sí que hemos identificado trabajos en los que se utilizan árboles de expansión mínima en aprendizaje no supervisado, véase Zahn (1971). Lo que se hace en ese caso es calcular el árbol de expansión mínima de las observaciones y después, si se quieren formar k clústers, se suprimen las $k-1$ aristas con coste más alto del mismo. Las k componentes conectadas resultantes constituyen los clústers. Nos parece natural pensar que también se podría explotar el uso de árboles de expansión mínima en el contexto de aprendizaje supervisado. En este sentido, la regla de clasificación binaria que presentaremos en este capítulo constituye una contribución metodológica novedosa de este trabajo. Dado un conjunto de entrenamiento, la idea del método se basa en la construcción de árboles de expansión mínima para cada una de las clases, asignando una nueva observación a aquella clase cuyo árbol de expansión se vea menos afectado al introducir la nueva observación en ella. La forma de medir esta conformidad se llevará a cabo mediante lo que se denominará separación de un grafo. Dicha separación se define como el cociente entre el coste de los árboles de expansión mínima del grafo y el número de observaciones. Así, la conformidad de cada clase será el cociente entre la separación del grafo y la separación del mismo una vez añadida la nueva observación. A mayor conformidad, más afín será, lo que determinará la clasificación. Dicho procedimiento es fácilmente extendible a más de dos clases.

En primer lugar, se dará la definición formal de la regla. A continuación se presentará una primera mejora del método, haciéndolo más robusto frente a observaciones atípicas y que, al mismo tiempo, lo hace mucho más eficiente desde el punto de vista computacional. Por último, se muestra un estudio numérico de la regla junto con las conclusiones obtenidas.

4.1. Definición de la regla

Considérese el vector aleatorio (X, Y) que toma valores en $\mathbb{R}^d \times \{0, 1\}$ y supongamos que disponemos de $(X_1, Y_1), \ldots, (X_n, Y_n)$, una secuencia de pares aleatorios independientes e idénticamente distribuidos a (X, Y). Se denotará por n_p al número de observaciones de la muestra tales que $Y_i = 1$ (observaciones positivas) y por n_n al número de observaciones tales que $Y_i = 0$ (observaciones negativas).

El primer paso de la regla consiste en normalizar todas las variables explicativas (en total d por ser X de esta dimensión), restándoles su media (muestral) y dividiendo por la desviación típica (muestral). De esta forma tendremos a todas ellas en la misma escala.

El segundo paso es definir los grafos correspondientes a cada clase y calcular árboles de expansión mínima para cada uno:

- Por GP denotaremos al grafo no dirigido completo cuyos nodos vienen dados por las observaciones de la clase 1. Los costes de las aristas vienen dados por las distancias euclideas entre ellas. Denotaremos por c_p al coste de cualquier árbol de expansión mínima de GP (todos ellos tienen el mismo coste).

- De forma análoga, por GN denotaremos al grafo no dirigido completo cuyos nodos vienen dados por las observaciones de la clase 0. Los costes de las aristas vienen dados por las distancias euclideas entre ellas. Denotaremos por c_n al coste de cualquier árbol de expansión mínima de GN.

El siguiente paso es definir una variable que mida la separación entre observaciones en cada grafo. Para el primero de ellos se define:

$$SEP(GP) = \frac{c_p}{n_p}.$$

Para el segundo grafo por su parte:

$$SEP(GN) = \frac{c_n}{n_n}.$$

Es decir, se divide el coste del árbol entre el número de observaciones. De esta forma, cuanto menor sea el valor de SEP más semejantes serán entre sí las observaciones de la clase en cuestión.

Dada una observación x, para clasificarla en la clase de las observaciones positivas o en la de las negativas se procede como sigue:

- Se definen los grafos GP^x y GN^x como los grafos resultantes tras añadir x tanto a la clase de las observaciones positivas como al de las negativas.

- Se calculan $SEP(GP^x) = \frac{c_p^x}{n_p+1}$ y $SEP(GN^x) = \frac{c_n^x}{n_n+1}$, siendo c_p^x y c_n^x los costes de cualquier árbol de expansión mínima de GP^x y GN^x, respectivamente.

- Como último paso se define una variable que mide la mejora o empeoramiento de la separación en cada uno de los grafos al añadir la nueva observación:

$$CONF_P^x = \frac{SEP(GP)}{SEP(GP^x)} \quad \text{y} \quad CONF_N^x = \frac{SEP(GN)}{SEP(GN^x)}.$$

  Estas variables se denominarán *conformidad en la clase 1* y *conformidad en la clase 0*, respectivamente. A mayor valor, mayor conformidad, es decir, menor separación entre los datos en GP^x/GN^x relativa a la separación en GP/GN.

Se define entonces la regla de clasificación basada en árboles de expansión mínima, de forma abreviada *MST-Class* (*Minimum Spanning Tree Classifier*), como:

$$g_n(x) = \begin{cases} 1 & \text{si } CONF_P^x > CONF_N^x, \\ 0 & \text{si } CONF_P^x < CONF_N^x. \end{cases}$$

En el caso en el que $CONF_P^x = CONF_N^x$ se aleatoriza la clasificación.

4.2. Método robusto

Una primera mejora que se ha incluido en el método viene motivada por intentar solventar un posible problema de robustez de la regla frente a observaciones atípicas. Se pueden consultar resultados más detallados sobre este aspecto en el estudio de simulación que se presenta en la sección 4.3, pero a efectos ilustrativos, presentamos a continuación un pequeño ejemplo en el que se hace evidente el problema. Supongamos la siguiente situación en la que la observación (X_4, Y_4) está mal clasificada en la clase 1 en vez de en la 0:

$$(X_1, Y_1) = ((0,0),0), (X_2, Y_2) = ((1,0),0), (X_3, Y_3) = ((0,1),0), (X_4, Y_4) = ((0.5,0.5),1),$$
$$(X_5, Y_5) = ((5,5),1), (X_6, Y_6) = ((6,5),1), (X_7, Y_7) = ((1,1),0).$$

En primer lugar, reescalemos todas las variables explicativas, restándoles su media muestral y dividiendo por la desviación típica muestral. El resultado es el siguiente conjunto de datos:

$$(X_1, Y_1) = ((-0.7743842, -0.7995481),0), (X_2, Y_2) = ((-0.3728517, -0.7995481),0),$$
$$(X_3, Y_3) = ((-0.7743842, -0.3518012),0), (X_4, Y_4) = ((-0.5736180, -0.5756746),1),$$
$$(X_5, Y_5) = ((1.2332786, 1.4391866),1), (X_6, Y_6) = ((1.6348112, 1.4391866),1),$$
$$(X_7, Y_7) = ((-0.3728517, -0.3518012),0).$$

En la Figura 4.1 se han representado las observaciones de cada clase. Puede apreciarse que la observación (X_4, Y_4), al estar etiquetada con $Y_4 = 1$, se encuentra en una zona en la que las observaciones pertenecen a la otra clase.

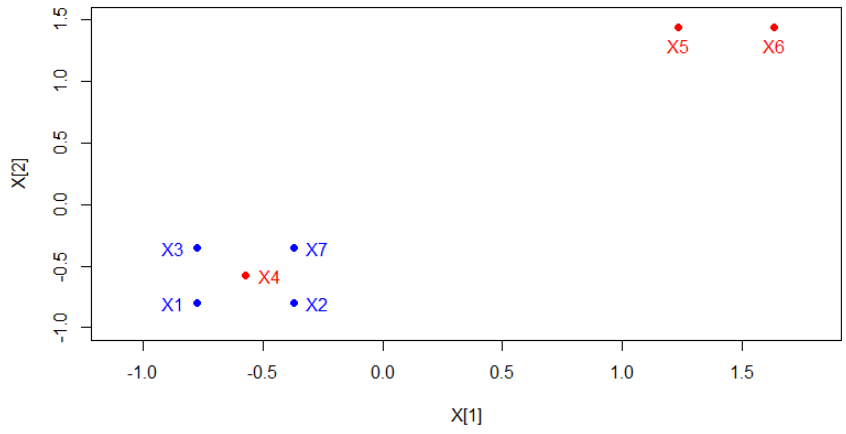


Figura 4.1: Representación de las observaciones: en rojo las de la clase 1 y en azul las de la 0.

Consideremos ahora como muestra de entrenamiento el conjunto $(X_i, Y_i), i = 1, \ldots, 6$ y veamos qué clasificación resulta para la observación (X_7, Y_7) según la regla *MST-Class*. Para calcular los árboles de mínimo coste en primer lugar se calculan las distancias entre las X_i:

$$d((X_1, X_2)) = 0.4015326, d((X_1, X_3)) = 0.4477469, d((X_2, X_3)) = 0.6014198$$
$$d((X_4, X_5)) = 2.7063889, d((X_4, X_6)) = 2.9894523, d((X_5, X_6)) = 0.4015326.$$

En las Figuras 4.2 y 4.3 hemos representado estos valores.

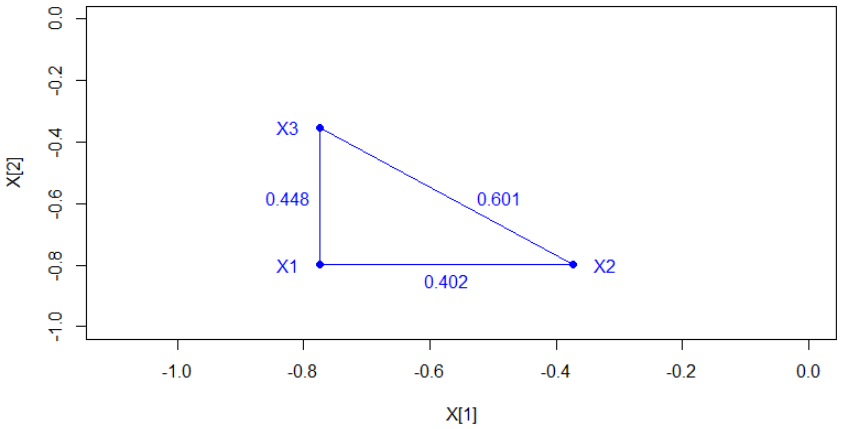


Figura 4.2: Representación de las distancias entre las observaciones de la clase 0.

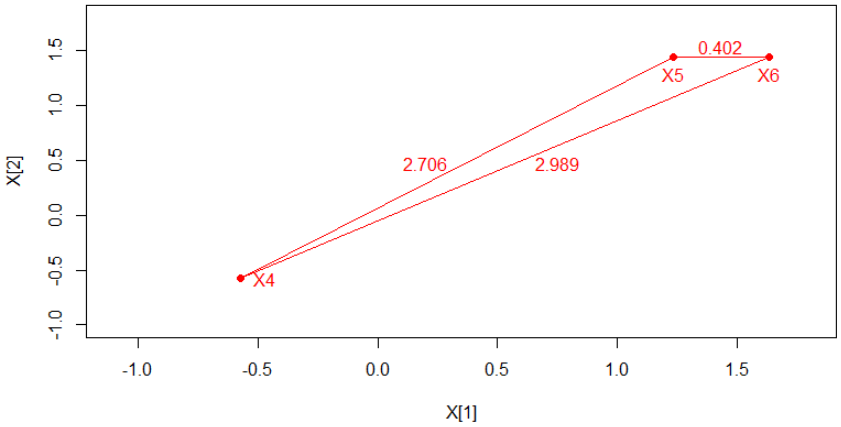


Figura 4.3: Representación de las distancias entre las observaciones de la clase 1.

Los árboles de mínimo coste se calculan a simple vista en este ejemplo pequeño, resultando los árboles representados en la Figura 4.4 y en la Figura 4.5 .Así, $c_n = 0.4015326 + 0.4477469 = 0.8492795$ y $c_p = 2.7063889 + 0.4015326 = 3.107921$. Por tanto,

$$SEP(GP) = \frac{c_p}{n_p} = \frac{3.107921}{3} = 1.035974,$$

$$SEP(GN) = \frac{c_n}{n_n} = \frac{0.8492795}{3} = 0.2830932.$$

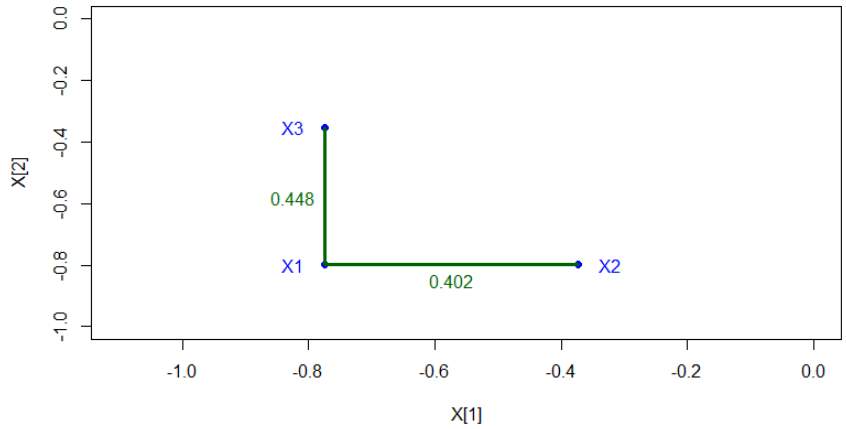


Figura 4.4: Representación del árbol de expansión mínima para la clase 0, en verde.

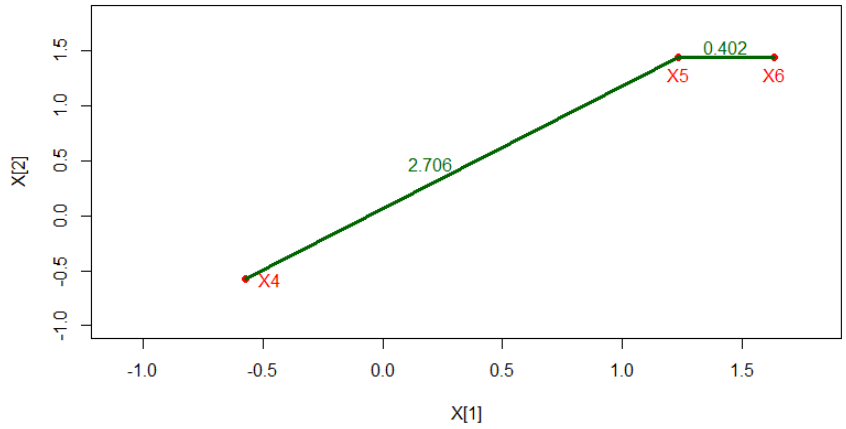


Figura 4.5: Representación del árbol de expansión mínima para la clase 1, en verde.

Calculemos ahora los grafos GP^{X_7} y GN^{X_7}. De nuevo, se calculan las distancias entre las X_i:

$$d((X_1, X_2)) = 0.4015326, d((X_1, X_3)) = 0.4477469, d((X_1, X_7)) = 0.6014198,$$
$$d((X_2, X_3)) = 0.6014198, d((X_2, X_7)) = 0.4477469, d((X_3, X_7)) = 0.4015326$$
$$d((X_4, X_5)) = 2.7063889, d((X_4, X_6)) = 2.9894523, d((X_4, X_7)) = 0.3007099,$$
$$d((X_5, X_6)) = 0.4015326, d((X_5, X_7)) = 2.4056791, d((X_6, X_7)) = 2.6904177.$$

En las Figuras 4.6 y 4.7 hemos representado estas distancias.

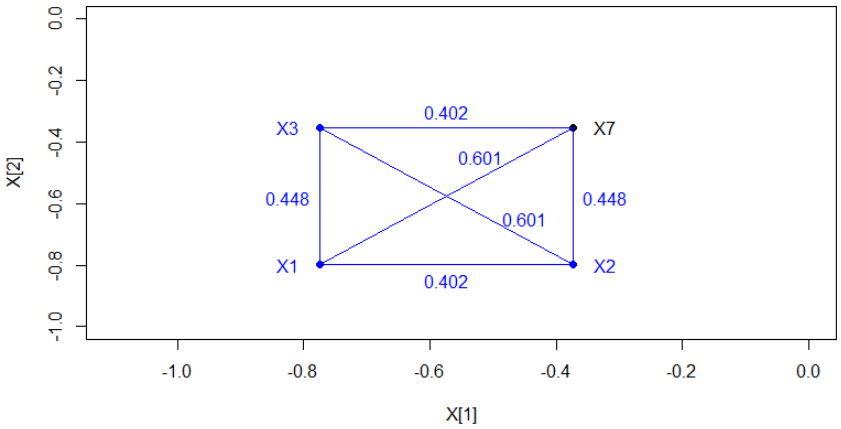


Figura 4.6: Representación de las distancias entre las observaciones de la clase 0 añadiendo X_7.

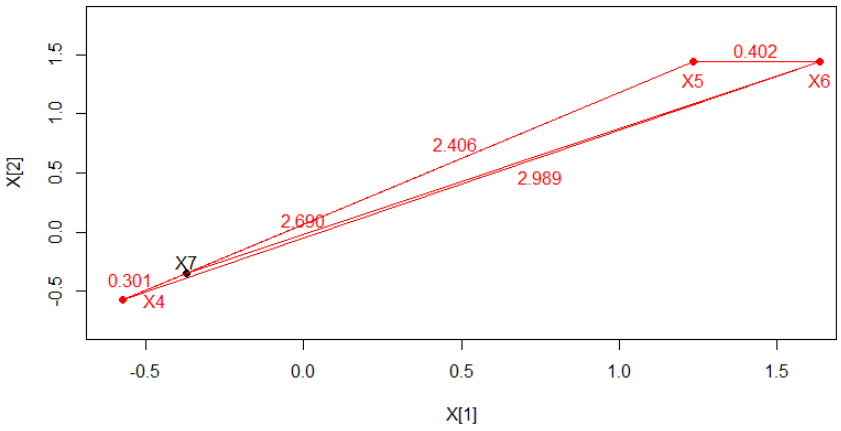


Figura 4.7: Representación de las distancias entre las observaciones de la clase 1 añadiendo X_7.

De nuevo, podemos calcular los árboles de mínimo coste a simple vista, resultando los árboles representados en la Figura 4.8 y en la Figura 4.9. Así, $c_n^{X_7} = 0.4015326 + 0.4015326 + 0.4477469 = 1.250812$ y $c_p^{X_7} = 0.3007099 + 2.4056791 + 0.4015326 = 3.107922$. Por consiguiente,

$$SEP(GP^{X_7}) = \frac{c_p}{n_p} = \frac{3.107921}{4} = 0.7769804,$$

$$SEP(GN^{X_7}) = \frac{c_n}{n_n} = \frac{1.250812}{4} = 0.312703.$$

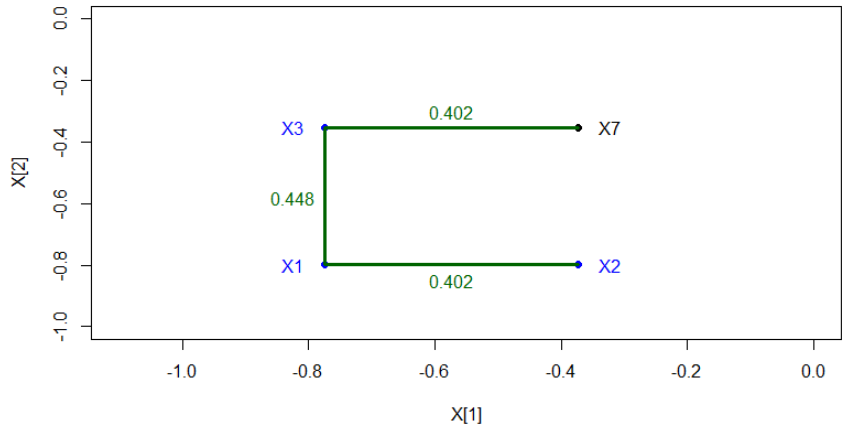


Figura 4.8: Representación (en verde) del árbol de expansión mínima para la clase 0 añadiendo la nueva observación.

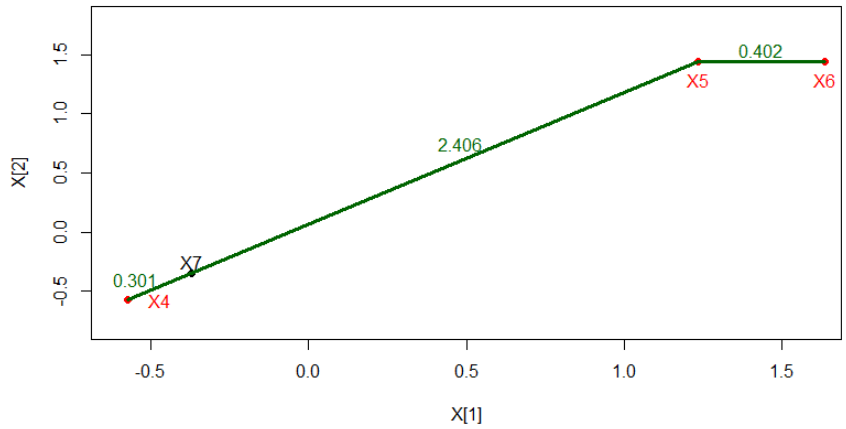


Figura 4.9: Representación (en verde) del árbol de expansión mínima para la clase 1 añadiendo la nueva observación.

Como último paso se calculan las variables de conformidad:

$$CONF_P^{X_7} = \frac{SEP(GP)}{SEP(GP^{X_7})} = \frac{1.035974}{0.7769804} = 1.333334$$

$$CONF_N^{X_7} = \frac{SEP(GN)}{SEP(GN^{x})} = \frac{0.2830932}{0.312703} = 0.9053102.$$

Por tanto, la observación X_7 se clasificaría en la clase $Y = 1$, cuando realmente $Y_7 = 0$ y todo como consecuencia de tener X_4 mal etiquetado. Para mitigar este problema y además hacer que la regla de clasificación use más información que simplemente la información sobre el grafo completo se procede como se describe a continuación.

Dada una observación x:

i) Se definen aleatoriamente l submuestras de la muestra de entrenamiento. Para que en dichas submuestras las clases estén balanceadas se procede como sigue:

- Se define $n \coloneqq \text{mín}\{n_p, n_n\}$.
- Se escogen aleatoriamente $\sqrt{n}$ observaciones positivas y $\sqrt{n}$ observaciones negativas.

Esta elección de tamaño para las submuestras es de especial importancia dado el orden de complejidad de los algoritmos para el problema del árbol de expansión mínima que ya hemos comentado en el Capítulo 3. Calcular los árboles de expansión mínima necesarios tiene una complejidad de $O(n_p^2)$ para la clase de las observaciones positivas, mientras que para las negativas es de $O(n_n^2)$. Por tanto, si pasamos a calcularlos con l submuestras de tamaño $\sqrt{n}$ en cada clase, la complejidad computacional para cada submuestra pasa a ser de $O((\sqrt{n})^2) = O(n)$ en ambas clases. De esta forma, el tiempo de cálculo pasa de ser cuadrático en el número de observaciones a ser lineal en el número de observaciones (que habría que multiplicar por l, que para conjuntos de datos grandes será mucho más pequeño que n_p y n_n).

Esta mejora computacional es especialmente relevante en contextos *bigdata*, en los que pasar de una complejidad cuadrática a una lineal tiene un gran impacto.

ii) Para cada una de las l submuestras se calculan los valores $CONF_P^x$ y $CONF_N^x$.

iii) Como siguiente paso se calcula la media ponderada de estos valores.

iv) Finalmente se realiza la clasificación en base a esta medida agregada de conformidad.

A este clasificador lo denotaremos de forma abreviada por *MST-RClass* (*Minimum Spanning Tree Robust Classifier*).

Volviendo al ejemplo, esto resultaría en que muchas de las submuestras tomadas dejarían fuera a X_4. Por tanto, en esos casos la conformidad de X_7 con las observaciones azules ($Y_i = 0$) será mucho mejor que con las rojas ($Y_i = 1$). Como consecuencia, al promediar los valores de conformidad devueltos por todas las submuestras seguramente acabaríamos realizando la clasificación de forma correcta.

4.3. Estudio computacional

Para llevar a cabo el estudio computacional se ha tenido que implementar de cero tanto la regla de clasificación base como la robusta. Para ello se ha elaborado el código necesario empleando el software R (R Core Team 2023). Cabe mencionar que la librería empleada para calcular el árbol de expansión mínima es la librería *igraph* (Csárdi et al. 2023), la cual utiliza en su función *mst* el algoritmo de Prim. Asimismo, se ha elaborado el código necesario para generar todas las gráficas presentes en el trabajo empleando la función *plot3d* de la librería *rgl* (Adler et al. 2023) y la función *ggplot* de la librería *ggplot2* (Chang et al. 2023). El cálculo de las medidas de rendimiento que emplearemos para comparar los distintos métodos ha sido realizado también empleando el software R. Por otra parte, todos los conjuntos de datos generados han sido almacenados en achivos *.csv*. Además, tanto en las ejecuciones realizadas como en el código generador de los diferentes conjuntos de datos se han establecido semillas empleando la función *set.seed* del paquete básico de R (R Core Team 2023), de forma que se puedan reproducir todas las ejecuciones de nuevo si fuese preciso.

Dado que el número de ejecuciones a realizar era elevado y el tiempo de las mismas era demasiado extenso como para llevarlas a cabo en un ordenador estándar, todas ellas se han realizado en el superordenador Finisterrae III, proporcionado por el Centro de Supercomputación de Galicia (CESGA). Salvo para cuando se ha querido comparar el rendimiento de las diferentes configuraciones del método no se han empleado nodos exclusivos, y el tiempo de ejecución máximo establecido así como la capacidad de memoria RAM del nodo se han adaptado a cada ejecución (conjuntos de datos con tamaños más pequeños requieren menos tiempo y capacidad).

4.3.1. Comparaciones entre distintas configuraciones de *MST-RClass* y *MST-Class*

El primer estudio que se ha realizado ha sido de tipo comparativo entre diferentes configuraciones de la versión robusta y la versión estándar de la regla. En particular, se han considerado 3 configuraciones distintas para *MST-RClass* variando el número de submuestras (todas ellas con $\sqrt{n}$ observaciones positivas y $\sqrt{n}$ negativas, $n = \text{mín}\{n_n, n_p\}$):

- Escogiendo $l = 10$ submuestras (*MST-RClass_10*).
- Tomando 50 submuestras: $l = 50$ (*MST-RClass_50*).
- Seleccionando $l = 100$ submuestras (*MST-RClass_100*).

Para testear el rendimiento de estas configuraciones hemos generado en una primera instancia conjuntos de datos con varias clases, donde la distribución subyacente de cada clase es una distribución normal multivariante preestablecida. En particular, se han generado 100 conjuntos de datos por cada una de las configuraciones. Se han escogido varios tamaños para el número de observaciones de cada clase en cada conjunto de datos generado: 300, 600, 1500 y 3000. Denotaremos por n_i al tamaño de la clase i. Además, las diferentes configuraciones surgen de ir realizando cambios progresivos sobre los parámetros de dos distribuciones normales que constituyen una estructura de referencia. Para la primera de las clases dichos parámetros son $\mu_1 = (1, 2)$, y

$$\Sigma_1 = \begin{pmatrix} 5 & 2 \\ 2 & 5 \end{pmatrix},$$

mientras que para la segunda clase $\mu_2 = (2, 3)$, y

$$\Sigma_2 = \begin{pmatrix} 5 & 2 \\ 2 & 5 \end{pmatrix}.$$

Los cambios que se han aplicado son los siguientes:

- Separar la media de la segunda clase en el eje x, manteniendo iguales las matrices de covarianzas.
- Separar la media de la segunda clase en el eje y, también manteniendo iguales las matrices de covarianzas.
- Separar la media en ambos ejes, manteniendo iguales las matrices de covarianzas.
- Aumentar las varianzas de la segunda clase, manteniendo iguales las medias.
- Movimientos de rotación mediante cambios en la matriz de covarianzas de la segunda clase, combinándolo también con cambios en la media de la segunda clase.

De forma aleatoria, dentro de cada conjunto de datos se ha seleccionado una muestra de entrenamiento con una proporción del 70 % del total, dejando un 30 % como muestra test sobre la que se han evaluado los resultados de clasificación. Al final del clasificado se calcula la proporción de bien clasificados para ese conjunto de datos. Después de repetir el proceso para los 100 conjuntos de datos de cada configuración se hace un resumen sobre todas las proporciones obtenidas.

Los resultados de las ejecuciones se han recogido en forma de tablas en el Capítulo 1, Sección 1.1, del archivo del siguiente repositorio: https://nubeusc-my.sharepoint.com/:f:/g/personal/iria_rodriguez_acevedo_rai_usc_es/EhBgIE6rW85JsgwaDfEKDz4BCzOjo1QEUA7GNf-kZVcKJA?e=d8eXs0.

Debido a la gran extensión de los resultados, aquí solo mostraremos algunos de ellos para ejemplificar las conclusiones.

Además del resumen sobre las proporciones de bien clasificados se ha añadido una fila más a las tablas titulada *Elapsed*. Esta fila recoge el tiempo medio, en segundos, de las 100 ejecuciones de cada configuración de datos. Para medir dicho tiempo se han solicitado nodos exclusivos en el CESGA. Además, se ha empleado la función *proc.time* incluida en el paquete básico de R, que devuelve un vector de dimensión 5 con diferentes tiempos medidos como el de usuario, el de sistema y el real transcurrido. El tiempo de usuario es el tiempo de CPU relacionado con la ejecución del código, mientras que el tiempo de sistema se relaciona con procesos como abrir o cerrar archivos. Para nuestro estudio hemos empleado el tiempo real transcurrido o *real elapsed time*.

Con respecto a las proporciones de bien clasificados podemos concluir que el método robusto proporciona mucho mejores resultados que la regla original, aún cuando no se está en presencia de datos atípicos o *outliers*. Esto constituye una gran ventaja, ya que no solo devuelve mejores resultados sino que además será más eficiente computacionalmente, tal y como veremos. Además, a mayor número de submuestras mejores resultados se obtuvieron. En particular, se observa un mayor impacto al considerar $l = 50$ en vez de $l = 10$ que el observado al considerar 100 submuestras en vez de 50.

Con respecto a los tiempos de ejecución se ha observado lo siguiente:

- Cuando el tamaño del conjunto de datos es pequeño ($n_1 = n_2 = 300$) tan solo se aprecia una reducción del tiempo transcurrido con respecto a la regla original cuando el número de submuestras es pequeño, es decir, $l = 10$. Sin embargo, cuando $l = 50$ o $l = 100$ el tiempo transcurrido se eleva, incluso triplicando el tiempo de la regla base. Esto se relaciona con la complejidad computacional del método robusto que hemos comentado anteriormente, el cual hemos visto que será particularmente bueno cuando l sea notablemente menor que n_p y n_n.

  Véase por ejemplo la Tabla de resultados 4.1.

- $\mu_1 = (1, 2)$, $\mu_2 = (10, 11)$, $\Sigma_1 = \begin{pmatrix} 5 & 2 \\ 2 & 5 \end{pmatrix}$, $\Sigma_2 = \begin{pmatrix} 5 & 2 \\ 2 & 5 \end{pmatrix}$, $n_1 = n_2 = 300$.

	MST-Class	*MST-RClass_ 10*	*MST-RClass_ 50*	*MST-RClass_ 100*
Min.	0.9444444	0.9722222	0.9777778	0.9722222
1st Qu.	0.9833333	0.9888889	0.9888889	0.9888889
Median	0.9888889	0.9944444	0.9944444	0.9944444
Mean	0.9863889	0.9918889	0.9925000	0.9926111
3rd Qu.	0.9944444	0.9958333	1.0000000	1.0000000
Max.	1.0000000	1.0000000	1.0000000	1.0000000
Elapsed	26.4640000	11.3150000	45.9550000	91.6110000

Tabla 4.1: Tabla de resultados.

Se pasa de tener un tiempo de 26.464 segundos con la regla inicial a 91.611 segundos con el método robusto tomando $l = 100$, mientras que con $l = 10$ el tiempo es menor: 11.315 segundos.

- A medida que se aumenta el tamaño de los conjuntos de datos se aprecia más la diferencia (en términos de tiempo) de emplear la regla robusta frente a la original. Aunque tomar solamente 10 submuestras siga proporcionando el menor tiempo, escoger 100 submuestras sigue siendo mucho más rápido que ejecutar el método inicial.

- Mientras que el aumento del tamaño de los conjuntos de datos parece resultar en un aumento exponencial de los tiempos de ejecución para la técnica base, para el método robusto esto no es así, lo cual es mucho más deseable.

- Al pasar de $l = 50$ a $l = 100$ submuestras el tiempo de ejecución se multiplica por 2.

Considerando todas estas observaciones se ha concluido que, aunque el método con $l = 100$ devuelve las mejores clasificaciones, escoger $l = 50$ resulta en tiempos computacionales mucho menores y proporcionando resultados muy próximos. Por tanto, de aquí en adelante se ha escogido $l = 50$, aunque este es un valor para el que hay que tener en cuenta el tamaño del conjunto de datos para evitar situaciones en las que puede que esta cantidad sea demasiado grande o quizás demasiado pequeña.

4.3.2. Comparaciones con otras reglas

El siguiente paso en el estudio realizado ha sido medir la competitividad de la regla frente a otras reglas como k-vecinos más próximos con $k = 15$ (*knn*), discriminación lineal (*lda*), discriminación cuadrática (*qda*) y la propia regla de Bayes (*bayes*) que, tal y como se ha visto en el Capítulo 1, para el caso de dos clases con distribuciones normales es conocida su expresión explícita. Para comparar estas reglas se han empleado los mismos conjuntos de datos del apartado anterior. Los resultados se han recogido en forma de tablas para su consulta en el Capítulo 1, Sección 1.2 del archivo del repositorio https://nubeusc-my.sharepoint.com/:f:/g/personal/iria_rodriguez_acevedo_rai_usc_es/EhBgIE6rW85JsgwaDfEKDz4BCzOjo1QEUA7GNf-kZVcKJA?e=d8eXs0. Se ha añadido además una última columna (*MST-RClass_ 50*) con los resultados del método robusto con $l = 50$ submuestras (todas ellas con $\sqrt{n}$ observaciones positivas y $\sqrt{n}$ negativas).

Las conclusiones generales obtenidas son las siguientes:

- La regla original obtiene generalmente peores resultados que el resto de métodos, aunque la diferencia no suele ser grande.

- Cuando el resto de métodos se comporta mal *MST-Class* también lo hace. Véase por ejemplo la siguiente configuración en la que las medias de ambas clases están muy próximas y tienen la misma matriz de covarianzas:

  - $\mu_1 = (1, 2)$, $\mu_2 = (2, 3)$, $\Sigma_1 = \begin{pmatrix} 5 & 2 \\ 2 & 5 \end{pmatrix}$, $\Sigma_2 = \begin{pmatrix} 5 & 2 \\ 2 & 5 \end{pmatrix}$, $n_1 = n_2 = 1500$.

	MST-Class	*knn*	*lda*	*qda*	*bayes*	*MST-RClass_50*
Min.	0.4844444	0.5433333	0.5711111	0.5700000	0.5722222	0.5344444
1st Qu.	0.5030556	0.5622222	0.5977778	0.5952778	0.5966667	0.5763889
Median	0.5166667	0.5727778	0.6044444	0.6055556	0.6050000	0.5827778
Mean	0.5169000	0.5729000	0.6049000	0.6046444	0.6060778	0.5833222
3rd Qu.	0.5280556	0.5833333	0.6125000	0.6133333	0.6136111	0.5933333
Max.	0.5688889	0.6155556	0.6366667	0.6388889	0.6388889	0.6144444

Tabla 4.2: Tabla de resultados.

Figura 4.10: Representación gráfica de uno de los conjuntos de datos de la configuración dada.

Puede verse en la Figura 4.10 que ambas clases están muy solapadas, y es por ello que se obtienen malos resultados, siendo la regla de Bayes la que mejores resultados devuelve en media (véase la Tabla 4.2).

Sin embargo, cuando no se da un solapamiento tan grande se puede observar que el rendimiento de *MST-Class* mejora, lo cual era de esperar. Como muestra de ello se tiene la siguiente configuración, con representación en la Figura 4.11 y resultados recogidos en la Tabla 4.3.

- $\mu_1 = (1,2)$, $\mu_2 = (10,3)$, $\Sigma_1 = \begin{pmatrix} 5 & 2 \\ 2 & 5 \end{pmatrix}$, $\Sigma_2 = \begin{pmatrix} 5 & 2 \\ 2 & 5 \end{pmatrix}$, $n_1 = n_2 = 600$.

	MST-Class	*knn*	*lda*	*qda*	*bayes*	*MST-RClass_50*
Min.	0.9388889	0.9638889	0.9611111	0.9638889	0.9638889	0.9638889
1st Qu.	0.9638889	0.9750000	0.9770833	0.9750000	0.9750000	0.9750000
Median	0.9694444	0.9833333	0.9833333	0.9833333	0.9833333	0.9833333
Mean	0.9703056	0.9812222	0.9823333	0.9821111	0.9823889	0.9812222
3rd Qu.	0.9777778	0.9861111	0.9868056	0.9868056	0.9888889	0.9861111
Max.	0.9944444	0.9972222	0.9972222	0.9972222	0.9972222	1.0000000

Tabla 4.3: Tabla de resultados.

Figura 4.11: Representación gráfica de uno de los conjuntos de datos de la configuración dada.

En efecto, vemos que la proporción de bien clasificados aumenta considerablemente para todas las medidas de posición.

- La diferencia observada entre la regla *MST-Class* y el resto de clasificadores se ve mitigada empleando *MST-RClass*, siendo incluso en algunos casos mejor. Por ejemplo, veamos los resultados obtenidos con la siguiente configuración (Tabla 4.4) representada en la Figura 4.12.

  - $\mu_1 = (1,2)$, $\mu_2 = (2,7)$, $\Sigma_1 = \begin{pmatrix} 5 & 2 \\ 2 & 5 \end{pmatrix}$, $\Sigma_2 = \begin{pmatrix} 5 & -2 \\ -2 & 5 \end{pmatrix}$, $n_1 = n_2 = 300$.

	MST-Class	*knn*	*lda*	*qda*	*bayes*	*MST-RClass_50*
Min.	0.7055556	0.7944444	0.8055556	0.8166667	0.7611111	0.8166667
1st Qu.	0.7666667	0.8555556	0.8611111	0.8611111	0.8319444	0.8611111
Median	0.7916667	0.8722222	0.8722222	0.8833333	0.8555556	0.8777778
Mean	0.7910000	0.8710000	0.8739444	0.8766667	0.8525556	0.8752222
3rd Qu.	0.8166667	0.8902778	0.8888889	0.8902778	0.8777778	0.8944444
Max.	0.8666667	0.9277778	0.9277778	0.9333333	0.9166667	0.9277778

Tabla 4.4: Tabla de resultados.

Figura 4.12: Representación gráfica de uno de los conjuntos de datos de la configuración dada.

Se observa que, en media, pasamos de obtener un porcentaje del 79.1 % de bien clasificados a un 87.52 %, siendo este porcentaje mayor que el del resto de reglas salvo para la discriminación cuadrática.

- A medida que aumentamos el tamaño de las clases puede observarse que, en líneas generales, se reducen muy ligeramente las proporciones para todas las reglas, lo cual puede estar motivado porque estamos clasificando un mayor número de observaciones.

4.3.3. Pruebas con outliers

Como siguiente paso se ha querido ver el comportamiento de la configuración del método robusto escogida frente a situaciones para las que originalmente fue pensado: conjuntos de datos con datos atípicos o *outliers*. Para ello se han escogido algunas de las configuraciones iniciales para

generar los datos, pero a un 4 % del total se les ha cambiado la etiqueta de su clase, es decir, son *outliers* (entendiendo, en este contexto, *outliers* como datos mal etiquetados). En concreto, dichos *outliers* estarán solamente en una de las dos clases. Nuevamente, los resultados se han recogido para su consulta en forma de tablas en el Capítulo 1, Sección 1.3 del archivo del repositorio https://nubeusc-my.sharepoint.com/:f:/g/personal/iria_rodriguez_acevedo_rai_usc_es/EhBgIE6rW85JsgwaDfEKDz4BCzOjo1QEUA7GNf-kZVcKJA?e=d8eXs0.

Las conclusiones tras observar los resultados son las siguientes:

- La regla original se ve muy afectada, tal y como se vaticinaba, por los datos atípicos, llegando a clasificar de forma incorrecta en casi la mitad de las ocasiones.

- El método robusto por su parte consigue mejorar estos resultados, proporcionando casi siempre porcentajes de bien clasificados competitivos con otras reglas. Véase por ejemplo el siguiente conjunto de resultados recogidos en la Tabla 4.5 para la configuración representada en la Figura 4.13.

- $\mu_1 = (1,2)$, $\mu_2 = (10,3)$, $\Sigma_1 = \begin{pmatrix} 5 & 2 \\ 2 & 5 \end{pmatrix}$, $\Sigma_2 = \begin{pmatrix} 5 & 2 \\ 2 & 5 \end{pmatrix}$, $n_1 = n_2 = 300$.

	MST-Class	*knn*	*lda*	*qda*	*bayes*	*MST-RClass_50*
Min.	0.4239130	0.9347826	0.9510870	0.9565217	0.9239130	0.9565217
1st Qu.	0.4945652	0.9728261	0.9782609	0.9728261	0.9565217	0.9728261
Median	0.5190217	0.9782609	0.9836957	0.9782609	0.9619565	0.9782609
Mean	0.5133696	0.9788043	0.9823370	0.9796739	0.9608152	0.9797283
3rd Qu.	0.5326087	0.9891304	0.9891304	0.9891304	0.9673913	0.9836957
Max.	0.5652174	1.0000000	1.0000000	1.0000000	0.9782609	1.0000000

Tabla 4.5: Tabla de resultados.

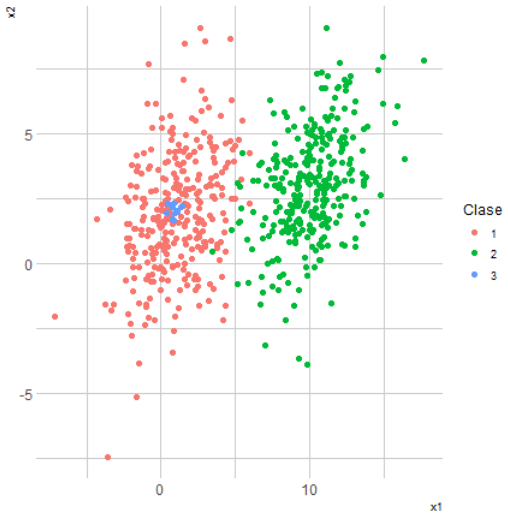


Figura 4.13: Representación gráfica de uno de los conjuntos de datos para la configuración dada, en azul los outliers.

Se observa en la Tabla 4.5 que, de obtener un 51.33696 % de bien clasificados en media se pasa a obtener un 97.97283 %, lo cual es una diferencia considerable.

4.3.4. Pruebas en grafos

Líneas de metro

Después de realizar el estudio con datos normales se ha procedido a estudiar otros conjuntos de datos con naturalezas distintas. Dado que *MST-Class* está basado en árboles de expansión mínima, la primera propuesta que se barajó fue considerar datos cuyas clases definieran, de forma natural, grafos. Es por esto por lo que se han considerado en primera instancia las redes de metro de Madrid. En particular, se han considerado diferentes líneas de metro con sus posiciones geográficas como variables explicativas. El objetivo entonces sería clasificar una ubicación dada en alguna de las líneas consideradas. Para ello se han extraído del repositorio `https://idealista.carto.com/tables/paradas_metro_madrid/public` las diferentes paradas de cada línea de metro de Madrid y, para disponer de un conjunto de datos más denso para cada clase, se generaron de forma aleatoria más paradas. Dichas paradas se situaron en las rectas que unen dos paradas consecutivas.

En esta ocasión, dado que no generamos los datos según una distribución dada, las 100 ejecuciones para cada par de líneas de metro se diferencian en la aleatoriedad del conjunto de entrenamiento escogido (que sigue siendo del 70 %). Una vez generados los datos, el estudio realizado fue el siguiente:

- Se comparó la versión robusta de *MST-RClass* con la regla de *knn*, escogiendo para esta última distintos números de vecinos: 1 (*knn1*), 5 (*knn5*), 15 (*knn15*), 21 (*knn21*), 31 (*knn31*), 41 (*knn41*) y $\sqrt{m}$ (*knn_sqrt*) siendo m el tamaño de la muestra de entrenamiento. Nótese que se han escogido cantidades impares para evitar empates.

  También se comparó con las reglas de discriminación lineal (*lda*) y discriminación cuadrática (*qda*) que asumen normalidad en la distribución de las clases. Cabe mencionar que para la versión robusta (*MST-RClass*) se escogió $l = 50$ submuestras, todas ellas con $3\sqrt{n}$ observaciones positivas y $3\sqrt{n}$ negativas (recordemos que $n = \text{mín}\{n_p, n_n\}$). Por tanto, cuando nos refiramos a *MST-RClass* sin especificar nada más estaremos haciendo referencia a dicha configuración.

- Asimismo se realizó un estudio comparativo entre diferentes configuraciones del método robusto, variando el número de submuestras y su tamaño, pero manteniendo la complejidad computacional. Las configuraciones consideradas han sido:

  i) $l = 10$ con $3\sqrt{n}$ observaciones positivas y $3\sqrt{n}$ negativas (*l=10,n=3sqrt*).

  ii) $l = 50$ con $3\sqrt{n}$ observaciones positivas y $3\sqrt{n}$ negativas (*l=50,n=3sqrt*).

  iii) $l = 100$ con $3\sqrt{n}$ observaciones positivas y $3\sqrt{n}$ negativas (*l=100,n=3sqrt*).

  iv) $l = 10$ con $5\sqrt{n}$ observaciones positivas y $5\sqrt{n}$ negativas (*l=10,n=5sqrt*).

  v) $l = 50$ con $5\sqrt{n}$ observaciones positivas y $5\sqrt{n}$ negativas (*l=50,n=5sqrt*).

  vi) $l = 100$ con $5\sqrt{n}$ observaciones positivas y $5\sqrt{n}$ negativas (*l=100,n=5sqrt*).

  vii) $l = 10$ con $10\sqrt{n}$ observaciones positivas y $10\sqrt{n}$ negativas (*l=10,n=10sqrt*).

  viii) $l = 50$ con $10\sqrt{n}$ observaciones positivas y $10\sqrt{n}$ negativas (*l=50,n=10sqrt*).

  ix) $l = 100$ con $10\sqrt{n}$ observaciones positivas y $10\sqrt{n}$ negativas (*l=100,n=10sqrt*).

- Por último, se hicieron representaciones gráficas de las zonas en las que la configuración robusta *l=50,n=3sqrt* y *knn* con 15 vecinos daban lugar a errores de clasificación.

Todos los resultados se pueden consultar en el Capítulo 2, Sección 2.1, del archivo del repositorio https://nubeusc-my.sharepoint.com/:f:/g/personal/iria_rodriguez_acevedo_rai_usc_es/EhBgIE6rW85JsgwaDfEKDz4BCzOjo1QEUA7GNf-kZVcKJA?e=d8eXs0.

Las conclusiones son las siguientes:

- A medida que se aumenta el número de vecinos para *knn* se observa un peor comportamiento, siendo entonces la regla de un vecino más próximo la que proporciona mejores resultados. Esto es un ejemplo más de que, tal y como veíamos en la Subsección 2.1.3, la regla del vecino más próximo no debe ser ignorada.

- Frente a la opción de *knn* con un vecino obtenemos resultados muy semejantes, siendo mejor *MST-RClass* (en media) en la mitad de las ocasiones.

- La discriminación lineal (*lda*) y la discriminación cuadrática (*qda*) obtienen siempre peores resultados que *MST-RClass* y que *knn* para cualquier número de vecinos, lo cual era de esperar al no estar bajo las hipótesis en las que se sustentan dichos métodos.

- Del análisis comparativo entre las diferentes configuraciones para *MST-RClass* concluimos que la regla es mucho menos sensible que *knn* a cambios en sus parámetros, a pesar de que sí que se observen algunas diferencias.

- Parece tener un mayor impacto el tamaño de las submuestras que el número de las mismas en las proporciones obtenidas. Se observa que a mayor tamaño de las submuestras el porcentaje de bien clasificados aumenta, pero se trata de un porcentaje pequeño, del 1 % como mucho.

- La configuración de *MST-RClass* que devuelve mayores proporciones para cada caso es siempre mejor que *knn* con un único vecino.

- Las zonas en las que se observa que tanto *knn* con $k = 15$ como *MST-RClass* cometen errores de clasificación son las intersecciones de las líneas de metro. Además, parece que los puntos mal clasificados por *knn* dibujan una cruz sobre el plano.

Todas estas conclusiones pueden apreciarse en los resultados para las líneas 6 y 2:

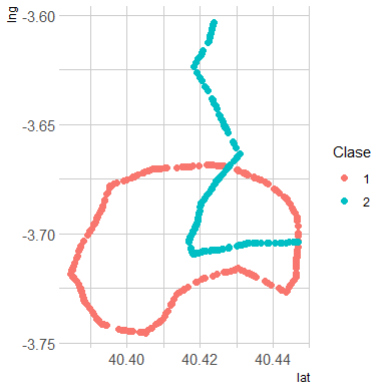


Figura 4.14: Representación gráfica de las líneas de metro consideradas.

	MST-RClass	*knn*1	*knn*5	*knn*15	*knn*21
Min.	0.9487179	0.9615385	0.9358974	0.9166667	0.8782051
1st Qu.	0.9743590	0.9871795	0.9743590	0.9599359	0.9407051
Median	0.9871795	0.9871795	0.9807692	0.9679487	0.9551282
Mean	0.9830769	0.9898718	0.9825000	0.9705128	0.9532692
3rd Qu.	0.9887821	0.9935897	0.9871795	0.9807692	0.9679487
Max.	1.0000000	1.0000000	1.0000000	1.0000000	1.0000000

Tabla 4.6: Tabla de resultados.

	*knn*31	*knn*41	*knn*_sqrt	*lda*	*qda*
Min.	0.8333333	0.7884615	0.8910256	0.5512821	0.7051282
1st Qu.	0.8958333	0.8589744	0.9487179	0.5961538	0.7500000
Median	0.9102564	0.8782051	0.9679487	0.6153846	0.7692308
Mean	0.9089744	0.8783974	0.9604487	0.6128846	0.7703205
3rd Qu.	0.9246795	0.8974359	0.9743590	0.6346154	0.7884615
Max.	0.9871795	0.9423077	1.0000000	0.6730769	0.8333333

Tabla 4.7: Tabla de resultados.

	$l = 10, n = 3sqrt$		$l = 50, n = 3sqrt$		$l = 100, n = 3sqrt$
Min.	0.9487179	Min.	0.9487179	Min.	0.9487179
1st Qu.	0.9743590	1st Qu.	0.9743590	1st Qu.	0.9743590
Median	0.9871795	Median	0.9871795	Median	0.9871795
Mean	0.9830128	Mean	0.9830769	Mean	0.9831410
3rd Qu.	0.9935897	3rd Qu.	0.9887821	3rd Qu.	0.9935897
Max.	1.0000000	Max.	1.0000000	Max.	1.0000000

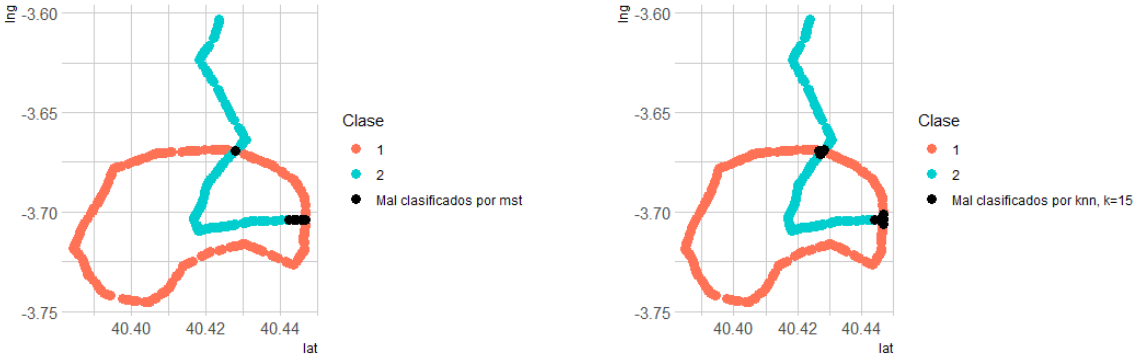
Tabla 4.8: Tabla de resultados.

	$l = 10, n = 5sqrt$		$l = 50, n = 5sqrt$		$l = 100, n = 5sqrt$
Min.	0.9487179	Min.	0.9551282	Min.	0.9551282
1st Qu.	0.9743590	1st Qu.	0.9807692	1st Qu.	0.9807692
Median	0.9871795	Median	0.9871795	Median	0.9871795
Mean	0.9831410	Mean	0.9875641	Mean	0.9871154
3rd Qu.	0.9935897	3rd Qu.	0.9935897	3rd Qu.	0.9935897
Max.	1.0000000	Max.	1.0000000	Max.	1.0000000

Tabla 4.9: Tabla de resultados.

	$l = 10, n = 10sqrt$		$l = 50, n = 10sqrt$		$l = 100, n = 10sqrt$
Min.	0.9679487	Min.	0.9679487	Min.	0.9679487
1st Qu.	0.9935897	1st Qu.	0.9871795	1st Qu.	0.9871795
Median	0.9935897	Median	0.9935897	Median	0.9935897
Mean	0.9939744	Mean	0.9932051	Mean	0.9933333
3rd Qu.	1.0000000	3rd Qu.	1.0000000	3rd Qu.	1.0000000
Max.	1.0000000	Max.	1.0000000	Max.	1.0000000

Tabla 4.10: Tabla de resultados.



(a) Mal clasificadas por *MST-RClass*.

(b) Mal clasificadas por *knn* con 15 vecinos.

Figura 4.15: Representación, en negro, de las observaciones mal clasificadas por *MST-RClass* y *lnn*.

Puede observarse en las Tablas 4.6 y 4.7 que, en media, se obtuvo una proporción de bien clasificados de 0.9830769 para *MST-RClass*, mientras que para *knn* con un vecino obtuvo 0.9898718. Cuando se aumenta el número de vecinos dicha proporción cae hasta 0.8783974. Por su parte, *lda* y *qda* obtuvieron valores mucho más bajos: 0.6128846 y 0.7703205, respectivamente.

Por otro lado, en las Tablas 4.8, 4.9 y 4.10 se observa lo ya comentado sobre la sensibilidad de la técnica. Además, encontramos configuraciones para las que obtenemos mejores resultados que *knn* con un vecino, como por ejemplo escogiendo *l=10, n=10sqrt*.

En la Figura 4.15 se aprecia que las zonas de mal clasificado de ambas técnicas son coincidentes (las intersecciones), siendo más grande la zona de *knn*.

Trayectorias de aviones

En una segunda instancia se han considerado diferentes trayectorias de aviones. Sus posiciones geográficas se han empleado como variables explicativas y como clases los identificadores de los aviones. El objetivo de la clasificación es entonces determinar, dada una coordenada, a qué avión pertenece ese punto de la trayectoria. El repositorio empleado se encuentra en Ghosh et al. (2021). Los datos de las trayectorias fueron recopilados entre el 18 de septiembre de 2020 y el 23 de abril de 2021 en el Aeropuerto Regional de Pittsburgh-Butler, situado al norte de la ciudad de Pittsburgh, Pensilvania. Además, los datos fueron segmentados en *frames*. Un *frame* comienza cuando al menos un avión está activo o entra en el umbral de detección y termina cuando todos los aviones han abandonado las inmediaciones o están inactivos. Para nuestro estudio se han seleccionado *frames* que involucran tan solo a una aeronave.

De nuevo, dado que no generamos los datos según una distribución dada, las 100 ejecuciones para cada par de trayectorias se diferencian en la aleatoriedad del conjunto de entrenamiento escogido (que sigue siendo del 70 %).

El estudio computacional realizado es exactamente el mismo que el efectuado para las líneas de metro. Los resultados se pueden consultar en el Capítulo 2, Sección 2.2 del archivo del repositorio https://nubeusc-my.sharepoint.com/:f:/g/personal/iria_rodriguez_acevedo_rai_usc_es/EhBgIE6rW85JsgwaDfEKDz4BCzOjo1QEUA7GNf-kZVcKJA?e=d8eXs0.

Como conclusiones obtenemos que:

- De nuevo, se observa que la mejor opción para *knn* es escoger un único vecino.
- Tanto la discriminación cuadrática como la discriminación lineal obtienen resultados muy por debajo de *MST-RClass* y *knn* para cualquiera de sus configuraciones.

- *MST-RClass* ($l = 50$ con $3\sqrt{n}$ observaciones positivas y $3\sqrt{n}$ negativas) es mejor que *knn* con un único vecino (en media) en un tercio de las ocasiones, siendo los resultados para ambas técnicas más distantes en general que lo observado para las líneas de metro.

- Una vez más, se aprecia que variar los parámetros de *MST-RClass* no conlleva grandes cambios en los resultados como sí ocurre con *knn*. De todas formas, los cambios observados son suficientes para que, nuevamente, exista alguna configuración para *MST-RClass* que obtiene mejores resultados que *knn* con un vecino para todos los pares de trayectorias considerados.

- Las zonas en las que *MST-RClass* y *knn* cometen errores de clasificación son aquellas en las que las trayectorias de ambos aviones están próximas.

4.3.5. Pruebas en toros

El siguiente paso realizado en el estudio computacional fue considerar toros para generar las clases. Esto fue motivado por la idea de extender lo visto para grafos en la sección anterior. Las características de los toros nos hacían pensar que obtendríamos un buen comportamiento para *MST-RClass* al igual que ocurría con las líneas de metro y las trayectorias de aviones y, al mismo tiempo, tendríamos una distribución subyacente conocida para cada clase. De esta forma, se generaron 100 conjuntos de datos para cada una de las 8 configuraciones diferentes escogidas para los toros. Cada toro generado (que no es lo mismo que cada clase ya que algunas configuraciones consideradas tienen más de un toro en la misma clase) contiene 1000 coordenadas.

El análisis llevado a cabo es el mismo que el efectuado para las líneas de metro y las trayectorias de avión, con la salvedad de que en esta ocasión no se han podido representar los puntos mal clasificados por las reglas, ya que en las 100 ejecuciones para cada configuración los datos varían (lo que permanece igual es la distribución). Los resultados pueden ser consultados en su totalidad en el Capítulo 3, Sección 3.1 del archivo del repositorio https://nubeusc-my.sharepoint.com/:f:/g/personal/iria_rodriguez_acevedo_rai_usc_es/EhBgIE6rW85JsgwaDfEKDz4BCzOjo1QEUA7GNf-kZVcKJA?e=d8eXs0.

Las conclusiones obtenidas difieren ligeramente de las de los datos con grafos:

- Para *knn* no se obtiene que la regla con un único vecino devuelva los mejores resultados, sino que dependiendo de la configuración la mejor elección para el número de vecinos varía.

- Los resultados para *MST-RClass* son muy buenos, siendo mejores que *knn* para cualquier k en casi todas las situaciones.

Observemos por ejemplo los siguientes resultados. Para esta configuración se han escogido 4 clases con un toro cada una, representados en la Figura 4.16. Como puede observarse en las Tablas 4.11 y 4.12 *MST-RClass* obtiene los mejores resultados. Además, *lda* y *qda* devuelven proporciones muy inferiores al resto como venía sucediendo anteriormente. Con respecto a la comparativa de las diferentes configuraciones de *MST-RClass* recogida en las Tablas 4.13, 4.14 y 4.15 se observa el mismo comportamiento ya mencionado para las líneas de metro.

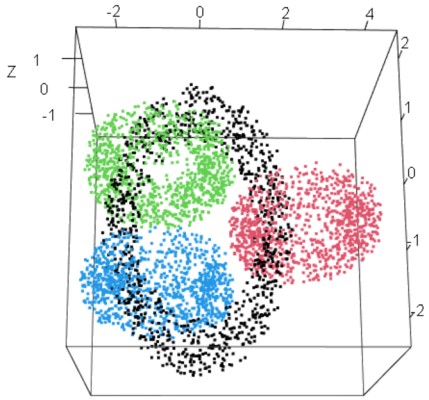


Figura 4.16: Representación gráfica de uno de los conjuntos de datos para la configuración dada, una clase en negro, otra en rojo, otra en verde y otra en azul.

	MST-RClass	*knn*1	*knn*5	*knn*15	*knn*21
Min.	0.9641667	0.9650000	0.9691667	0.9691667	0.9683333
1st Qu.	0.9791667	0.9731250	0.9766667	0.9766667	0.9766667
Median	0.9825000	0.9766667	0.9787500	0.9800000	0.9800000
Mean	0.9818750	0.9762417	0.9791583	0.9795083	0.9796250
3rd Qu.	0.9852083	0.9791667	0.9825000	0.9825000	0.9825000
Max.	0.9925000	0.9883333	0.9908333	0.9883333	0.9891667

Tabla 4.11: Tabla de resultados.

	*knn*31	*knn*41	*knn*_sqrt	*lda*	*qda*
Min.	0.9650000	0.9691667	0.9641667	0.6533333	0.8766667
1st Qu.	0.9758333	0.9750000	0.9733333	0.6700000	0.8916667
Median	0.9791667	0.9783333	0.9766667	0.6770833	0.8979167
Mean	0.9787167	0.9780000	0.9764333	0.6771250	0.8981917
3rd Qu.	0.9816667	0.9808333	0.9793750	0.6841667	0.9041667
Max.	0.9900000	0.9891667	0.9891667	0.7025000	0.9191667

Tabla 4.12: Tabla de resultados.

$l = 10, n = 3sqrt$		$l = 50, n = 3sqrt$		$l = 100, n = 3sqrt$	
Min.	0.9666667	Min.	0.9641667	Min.	0.9633333
1st Qu.	0.9766667	1st Qu.	0.9791667	1st Qu.	0.9791667
Median	0.9800000	Median	0.9825000	Median	0.9825000
Mean	0.9800333	Mean	0.9818750	Mean	0.9819333
3rd Qu.	0.9825000	3rd Qu.	0.9852083	3rd Qu.	0.9850000
Max.	0.9916667	Max.	0.9925000	Max.	0.9916667

Tabla 4.13: Tabla de resultados.

$l = 10, n = 5sqrt$		$l = 50, n = 5sqrt$		$l = 100, n = 5sqrt$	
Min.	0.9650000	Min.	0.9716667	Min.	0.9691667
1st Qu.	0.9783333	1st Qu.	0.9791667	1st Qu.	0.9800000
Median	0.9808333	Median	0.9820833	Median	0.9825000
Mean	0.9808333	Mean	0.9819417	Mean	0.9822750
3rd Qu.	0.9833333	3rd Qu.	0.9850000	3rd Qu.	0.9852083
Max.	0.9908333	Max.	0.9908333	Max.	0.9908333

Tabla 4.14: Tabla de resultados.

$l = 10, n = 10sqrt$		$l = 50, n = 10sqrt$		$l = 100, n = 10sqrt$	
Min.	0.9683333	Min.	0.9716667	Min.	0.9700000
1st Qu.	0.9783333	1st Qu.	0.9791667	1st Qu.	0.9791667
Median	0.9808333	Median	0.9820833	Median	0.9820833
Mean	0.9809000	Mean	0.9819417	Mean	0.9817417
3rd Qu.	0.9841667	3rd Qu.	0.9850000	3rd Qu.	0.9850000
Max.	0.9908333	Max.	0.9908333	Max.	0.9908333

Tabla 4.15: Tabla de resultados.

4.3.6. Pruebas en planos

En el siguiente conjunto de datos considerado las clases vienen dadas por planos que se cortan en $\mathbb{R}^3$. Cada plano considerado contiene 1000 puntos. En la Figura 4.17 se encuentra la representación del conjunto de datos. El estudio que se hizo es exactamente igual al realizado para los diferentes conjuntos de datos de las líneas de metro, centrándonos esta vez solamente en *MST-RClass* y *knn*.

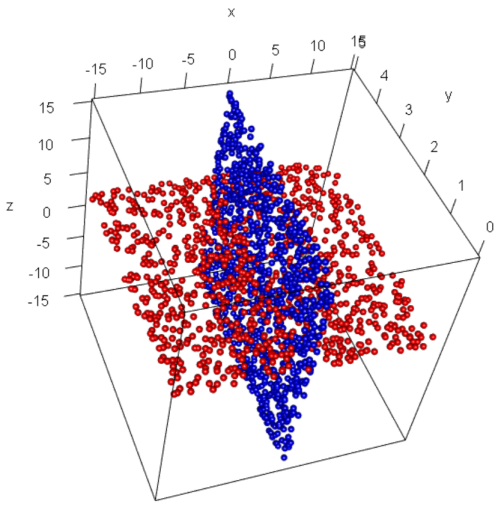


Figura 4.17: Representación del conjunto de datos, una clase representada en rojo y otra en azul.

Los resultados obtenidos se recogieron en las Tablas 4.16, 4.17, 4.18, 4.19 y 4.20. Por una parte, en las Tablas 4.16 y 4.17 puede observarse que *knn* presenta de nuevo un peor comportamiento a medida que se aumenta el número de vecinos. Además, *MST-RClass* presenta proporciones mayores que *knn* con un único vecino salvo para la medida del mínimo. Por otra parte, en las Tablas 4.18, 4.19 y 4.20 se observa una vez más un buen comportamiento del método robusto, siendo poco sensible a cambios en sus parámetros.

	MST-RClass	*knn*1	*knn*5	*knn*15	*knn*21
Min.	0.9683333	0.9750000	0.9616667	0.9550000	0.9483333
1st Qu.	0.9816667	0.9816667	0.9750000	0.9683333	0.9650000
Median	0.9850000	0.9850000	0.9791667	0.9733333	0.9708333
Mean	0.9847000	0.9845500	0.9790000	0.9732500	0.9701667
3rd Qu.	0.9883333	0.9866667	0.9816667	0.9783333	0.9750000
Max.	0.9966667	0.9950000	0.9916667	0.9916667	0.9866667

Tabla 4.16: Tabla de resultados.

	$knn31$	$knn41$	$knn_$sqrt
Min.	0.9483333	0.9483333	0.9466667
1st Qu.	0.9650000	0.9616667	0.9629167
Median	0.9683333	0.9666667	0.9675000
Mean	0.9688667	0.9666667	0.9671333
3rd Qu.	0.9733333	0.9716667	0.9716667
Max.	0.9866667	0.9816667	0.9816667

Tabla 4.17: Tabla de resultados.

$l = 10, n = 3sqrt$		$l = 50, n = 3sqrt$		$l = 100, n = 3sqrt$
Min.	0.9650000	Min.	0.9683333	Min.	0.9716667
1st Qu.	0.9766667	1st Qu.	0.9816667	1st Qu.	0.9816667
Median	0.9800000	Median	0.9850000	Median	0.9850000
Mean	0.9801667	Mean	0.9847000	Mean	0.9857167
3rd Qu.	0.9850000	3rd Qu.	0.9883333	3rd Qu.	0.9900000
Max.	0.9950000	Max.	0.9966667	Max.	0.9950000

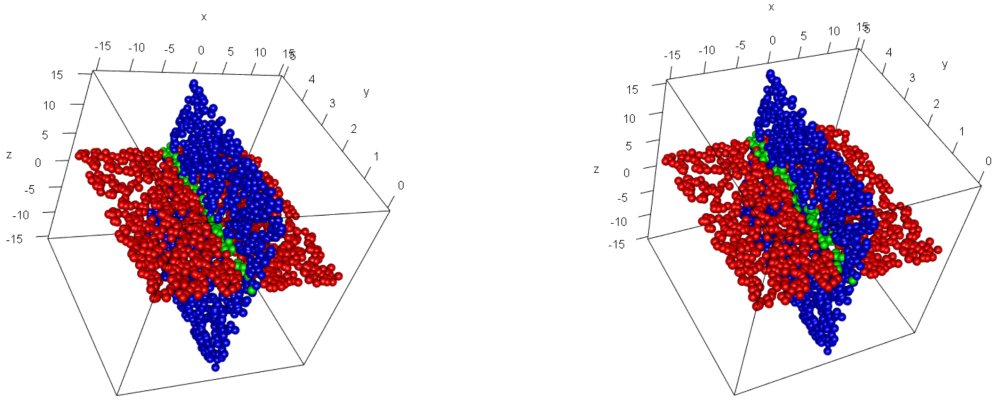
Tabla 4.18: Tabla de resultados.

$l = 10, n = 5sqrt$		$l = 50, n = 5sqrt$		$l = 100, n = 5sqrt$
Min.	0.9716667	Min.	0.9750000	Min.	0.9750000
1st Qu.	0.9816667	1st Qu.	0.9833333	1st Qu.	0.9833333
Median	0.9850000	Median	0.9858333	Median	0.9866667
Mean	0.9845500	Mean	0.9864333	Mean	0.9869000
3rd Qu.	0.9866667	3rd Qu.	0.9900000	3rd Qu.	0.9900000
Max.	0.9950000	Max.	0.9983333	Max.	0.9983333

Tabla 4.19: Tabla de resultados.

$l = 10, n = 10sqrt$		$l = 50, n = 10sqrt$		$l = 100, n = 10sqrt$	
Min.	0.9766667	Min.	0.9733333	Min.	0.9750000
1st Qu.	0.9833333	1st Qu.	0.9850000	1st Qu.	0.9850000
Median	0.9866667	Median	0.9883333	Median	0.9883333
Mean	0.9867167	Mean	0.9875167	Mean	0.9876333
3rd Qu.	0.9900000	3rd Qu.	0.9900000	3rd Qu.	0.9900000
Max.	0.9966667	Max.	0.9983333	Max.	1.0000000

Tabla 4.20: Tabla de resultados.



(a) Mal clasificadas *MST-RClass*.

(b) Mal clasificadas por *knn* con 15 vecinos.

Figura 4.18: Representación, en verde, de las observaciones mal clasificadas por *MST-RClass* y *knn*.

En la Figura 4.18 vemos que *MST-RClass* y *knn* cometen fallos de clasificación en la intersección de ambos planos.

4.3.7. Pruebas en planchas en espiral

El último estudio realizado se corresponde con conjuntos de datos en los que una de las clases viene dada por un plano en $\mathbb{R}^3$ y la otra por una plancha en espiral que corta al plano. Se escogieron 4 configuraciones distintas de las espirales para generar los conjuntos de datos. Tanto los planos como las espirales se generaron de forma que tuvieran 1000 puntos cada uno. El análisis realizado es el mismo que el ejecutado para los planos. Los resultados pueden ser consultados en su totalidad en el Capítulo 3, Sección 3.3 del archivo del repositorio https://nubeusc-my.sharepoint.com/:f:/g/personal/iria_rodriguez_acevedo_rai_usc_es/EhBgIE6rW85JsgwaDfEKDz4BCzOjo1QEUA7GNf-kZVcKJA?e=d8eXs0.

Las conclusiones que extraemos son las siguientes:

- La mejor configuración para *knn* es escoger $k = 1$.
- En comparación con *knn* con un único vecino *MST-RClass* ($l = 50$ con $3\sqrt{n}$ observaciones

positivas y $3\sqrt{n}$ negativas) devuelve peores resultados. Escogiendo la configuración de *MST-RClass* que mejores resultados obtiene la diferencia entre ambos métodos se hace más pequeña.

- Las zonas en las que *knn* y *MST-RClass* clasifican de forma errónea difieren en el siguiente sentido: *MST-RClass* clasifica mal las observaciones del plano próximas a las intersecciones de este con la espiral, mientras que *knn* clasifica de forma equivocada también en las intersecciones pero en puntos de la espiral.

- Una vez más se reafirma la robustez de *MST-RClass* frente a cambios en sus parámetros, resultando en alguna configuración que aumentar el tamaño de las submuestras no ha sido beneficioso.

Observemos por ejemplo los siguientes resultados del conjunto de datos representado en la Figura 4.19. Como puede observarse en las Tablas 4.21 y 4.22 *MST-RClass* obtiene peores resultados (en media) que *knn* con $k = 1$ o $k = 5$, pero para más de 5 vecinos son mejores. La comparativa de las diferentes configuraciones de *MST-RClass* está recogida en las Tablas 4.23, 4.24 y 4.25. Puede apreciarse que al pasar de considerar 100 submuestras con $3\sqrt{n}$ observaciones positivas y $3\sqrt{n}$ negativas a escogerlas con $5\sqrt{n}$ observaciones positivas y $5\sqrt{n}$ negativas, las proporciones obtenidas en la media pasan de 0.9260667 a 0.8935833.

En la Figura 4.20 se aprecia de forma clara el comportamiento antes mencionado sobre las zonas de clasificación errónea de ambas técnicas.

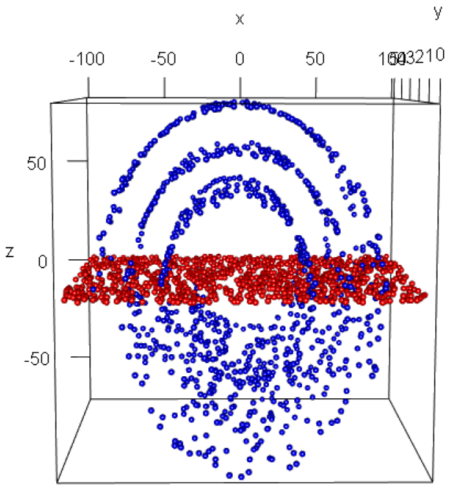


Figura 4.19: Representación del conjunto de datos, una clase representada en rojo y otra en azul.

	MST-RClass	*knn*1	*knn*5	*knn*15	*knn*21
Min.	0.8666667	0.9266667	0.9216667	0.8800000	0.8666667
1st Qu.	0.9150000	0.9483333	0.9333333	0.9045833	0.8950000
Median	0.9233333	0.9533333	0.9400000	0.9116667	0.9033333
Mean	0.9224667	0.9532000	0.9391833	0.9108167	0.9016500
3rd Qu.	0.9333333	0.9583333	0.9450000	0.9166667	0.9083333
Max.	0.9516667	0.9700000	0.9616667	0.9350000	0.9300000

Tabla 4.21: Tabla de resultados.

	*knn*31	*knn*41	*knn*_sqrt
Min.	0.8616667	0.8550000	0.8550000
1st Qu.	0.8812500	0.8700000	0.8750000
Median	0.8883333	0.8800000	0.8833333
Mean	0.8880500	0.8786667	0.8823167
3rd Qu.	0.8950000	0.8866667	0.8900000
Max.	0.9200000	0.9183333	0.9183333

Tabla 4.22: Tabla de resultados.

$l = 10, n = 3sqrt$		$l = 50, n = 3sqrt$		$l = 100, n = 3sqrt$	
Min.	0.8516667	Min.	0.8666667	Min.	0.8733333
1st Qu.	0.8929167	1st Qu.	0.9150000	1st Qu.	0.9166667
Median	0.9066667	Median	0.9233333	Median	0.9266667
Mean	0.9053500	Mean	0.9224667	Mean	0.9260667
3rd Qu.	0.9200000	3rd Qu.	0.9333333	3rd Qu.	0.9366667
Max.	0.9500000	Max.	0.9516667	Max.	0.9566667

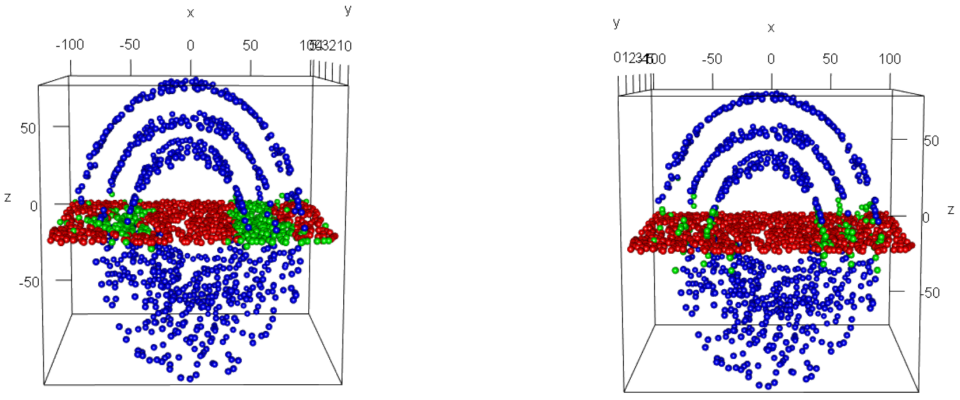
Tabla 4.23: Tabla de resultados.

$l = 10, n = 5sqrt$		$l = 50, n = 5sqrt$		$l = 100, n = 5sqrt$	
Min.	0.8550000	Min.	0.8683333	Min.	0.8750000
1st Qu.	0.8883333	1st Qu.	0.9000000	1st Qu.	0.9000000
Median	0.8983333	Median	0.9091667	Median	0.9100000
Mean	0.8977500	Mean	0.9087000	Mean	0.9107833
3rd Qu.	0.9100000	3rd Qu.	0.9187500	3rd Qu.	0.9220833
Max.	0.9416667	Max.	0.9500000	Max.	0.9516667

Tabla 4.24: Tabla de resultados.

$l = 10, n = 10sqrt$		$l = 50, n = 10sqrt$		$l = 100, n = 10sqrt$	
Min.	0.8450000	Min.	0.8583333	Min.	0.8550000
1st Qu.	0.8795833	1st Qu.	0.8850000	1st Qu.	0.8850000
Median	0.8908333	Median	0.8950000	Median	0.8950000
Mean	0.8896333	Mean	0.8939333	Mean	0.8935833
3rd Qu.	0.9016667	3rd Qu.	0.9033333	3rd Qu.	0.9050000
Max.	0.9333333	Max.	0.9300000	Max.	0.9233333

Tabla 4.25: Tabla de resultados.



(a) Mal clasificadas *MST-RClass*.

(b) Mal clasificadas por *knn* con 15 vecinos.

Figura 4.20: Representación, en verde, de las observaciones mal clasificadas por *MST-RClass* y *knn*.

Capítulo 5

Conclusiones y trabajo futuro

En este Trabajo Fin de Máster se revisan algunas de las técnicas de clasificación más empleadas en aprendizaje supervisado. Se analiza en detalle la regla de *knn* y se propone un nuevo algoritmo de clasificación, *MST-Class*, basado en árboles de expansión mínima. Además, se presenta una versión robusta del mismo, denominada *MST-RClass*. La principal conclusión extraída del estudio computacional realizado es que *MST-RClass* obtiene buenos resultados, y se muestra especialmente competitivo con *knn* cuando se aplica a conjuntos de datos cuya naturaleza define grafos en cada clase. Además, *MST-RClass* permite reducir considerablemente el tiempo computacional de las ejecuciones con respecto a *MST-Class*, realizando a su vez mejores clasificaciones, no solo en presencia de *outliers*. Otra de las conclusiones importantes es que la versión robusta es poco sensible a cambios en sus parámetros, al contrario de lo que hemos visto que ocurre con *knn* cuando variamos el número de vecinos.

Como trabajo futuro, el objetivo principal será demostrar la consistencia teórica del método *MST-RClass*, ya que esta propiedad es de especial relevancia para un algoritmo de clasificación. Consideramos que el estudio de la prueba de consistencia para knn será de gran ayuda en este sentido. Por otra parte, se consideran otras posibles versiones de *MST-RClass*. Por ejemplo, en esta memoria todas las variables explicativas han tenido el mismo peso en la clasificación, cuando realmente puede darse el caso en el que algunas sean más informativas que otras. Así, podría llevarse a cabo algún tipo de selección de variables antes de aplicar el procedimiento de *MST-RClass*. Aunque se podrían emplear técnicas estándar de selección de variables, se pretende introducir un nuevo procedimiento basado también en árboles de expansión mínima.

Las conclusiones obtenidas en este trabajo sientan las bases para investigaciones futuras y mejoras en el algoritmo *MST-RClass*, lo que podría ampliar su aplicabilidad a diversos escenarios y problemas de clasificación.

Bibliografía

[1] Adler D., Bolker B., Csárdi G., Demont Y, Eddelbuettel D., Fernandez i Marin X., Gebhardt A., Helffrich G., Krylov I., Ming C., Murdoch D., Oleg N., Ooms J., R Core Team, Senger A., Stein M., Strzelecki A., Sumner M., The authors of knitr, The authors of Shiny, Ulrich J., Urbanek S. (2023) *rgl: 3D Visualization Using OpenGL.* R package version 1.1.3. https://cran.r-project.org/web/packages/rgl/index.html

[2] Ahuja R. K., Magnanti T. L., Orlin J. B. (1988). Network flows.

[3] Biau G., Devroye L. (2015). Lectures on the nearest neighbor method (Vol. 246). Cham, Switzerland: Springer International Publishing.

[4] Boruvka O. (1926). O jistém problému minimálním (About a certain minimal problem), (3) 37-58 (Czech, German summary).

[5] Boser B. E., Guyon I. M., Vapnik V. N. (1992). A training algorithm for optimal margin classifiers. In Proceedings of the fifth annual workshop on Computational learning theory (pp. 144-152).

[6] Chang W., Dunnington D., Henry L., Lin Pedersen T., Takahashi K., Wickham H., Wilke C., Woo K., Yutani H. (2023) *ggplot2: Create Elegant Data Visualisations Using the Grammar of Graphics.* R package version 3.4.2. https://cran.r-project.org/web/packages/ggplot2/index.html

[7] Conde Amboage M., Crujeiras Casais R. M. (2021). Apuntes de la asignatura Regresión Generalizada y Modelos Mixtos.

[8] Cortes C., Vapnik V. (1995). Support vector machine. Machine learning, 20(3), 273-297.

[9] Csárdi G., Horvát S., Müller K., Nepusz T., Noom D., Salmon M., Traag V., Zanini F. (2023). *igraph: Network Analysis and Visualization.* R package version 1.4.2. https://cran.r-project.org/web/packages/igraph/index.html

[10] Devijver P. A. (1978). A note on ties in voting with the k-NN rule. Pattern Recognition, 10(4), 297-298.

[11] Devroye L., Györfi L., Lugosi G. (2013). A probabilistic theory of pattern recognition (Vol. 31). Springer Science & Business Media.

[12] Devroye L., Györfi L. (1985). Nonparametric density estimation. The L_1 View.

[13] Fisher R. A. (1936). The use of multiple measurements in taxonomic problems. Annals of eugenics, 7(2), 179-188.

[14] Fix E., Hodges J. (1951). Discriminatory analysis. Nonparametric discrimination:Consistency properties. Technical Report 4, Project Number 21-49-004, USAF School of Aviation Medicine, Randolph Field, TX.

[15] González Díaz J. (2018). Apuntes de la asignatura Programación Lineal y Entera.

[16] Friedman J. H., Hastie T., Tibshirani R. (2009). The elements of statistical learning: data mining, inference, and prediction (Vol. 2, pp. 1-758). New York: springer.

[17] Ho T. K. (1995). Random decision forests. In Proceedinzs of 3rd international conference on document analysis and recognition (Vol. 1, pp. 278-282). IEEE.

[18] Kruskal J. B. (1956). On the shortest spanning subtree of a graph and the traveling salesman problem. Proceedings of the American Mathematical society, 7(1), 48-50.

[19] Ghosh S., Moon B., Oh J., Patrikar J., Scherer S. (2021): TrajAir: A General Aviation Trajectory Dataset. Carnegie Mellon University. Dataset. https://doi.org/10.1184/R1/14866251.v1

[20] Prim R. C. (1957). Shortest connection networks and some generalizations. The Bell System Technical Journal, 36(6), 1389-1401.

[21] Quinlan J. R. (1979). Discovering rules by induction from large collections of examples. Expert systems in the micro electronics age.

[22] Quinlan J. R. (1986). Induction of decision trees. Machine learning, 1, 81-106.

[23] R Core Team (2023). *R: A language and environment for statistical computing.* R Foundation for Statistical Computing, Vienna, Austria. https://www.R-project.org/

[24] Rosenblatt F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. Psychological review, 65(6), 386.

[25] Stone C. J. (1977). Consistent nonparametric regression. The annals of statistics, 595-620.

[26] Zahn C. T. (1971). Graph-theoretical methods for detecting and describing gestalt clusters. IEEE Transactions on computers, 100(1), 68-86.

[27] Zhao L. C. (1987). Exponential bounds of mean error for the nearest neighbor estimates of regression functions. Journal of Multivariate Analysis, 21(1), 168-178.