**UNIVERSIDAD DE SALAMANCA**

**FACULTAD DE CIENCIAS**

DEPARTAMENTO DE ESTADÍSTICA


**TRABAJO DE FIN DE GRADO**

# USO DE LOS ÁRBOLES DE DECISIÓN CON ENTROPÍA EN LA MINERÍA DE DATOS. CREACIÓN DE UNA APLICACIÓN WEB CON R PARA CLASIFICAR O PREDECIR DATOS REALES

Tutor: MIGUEL RODRÍGUEZ ROSA

**Autor: GUILLERMO ALBERTO GONZÁLEZ MORALES**

Grado en Estadística

Curso académico 2022-23

**UNIVERSIDAD DE SALAMANCA**

**FACULTAD DE CIENCIAS**

DEPARTAMENTO DE ESTADÍSTICA

**TRABAJO DE FIN DE GRADO**

# USO DE LOS ÁRBOLES DE DECISIÓN CON ENTROPÍA EN LA MINERÍA DE DATOS. CREACIÓN DE UNA APLICACIÓN WEB CON R PARA CLASIFICAR O PREDECIR DATOS REALES

Firmado por:

Autor: Guillermo Alberto González Morales

Tutor: Miguel Rodríguez Rosa

Grado en Estadística

Curso académico 2022-23

Certificado del tutor TFG Grado en Estadística

D. Miguel Rodríguez Rosa, profesor del Departamento de Estadística de la Universidad de Salamanca,

HACE CONSTAR:

Que el trabajo titulado "*Uso de los árboles de decisión con entropía en la minería de datos. Creación de una aplicación web con R para clasificar o predecir datos reales*", que se presenta, ha sido realizado por D. Guillermo Alberto González Morales, con DNI 32892339Q y constituye la memoria del trabajo realizado para la superación de la asignatura Trabajo de Fin de Grado en Estadística en esta Universidad.

Salamanca, a 3 de julio de 2023.

Fdo.: Miguel Rodríguez Rosa

# Índice general

# Summary

In this project, in order to understand in the best possible way what a decision tree is, and to get the most out of it, we start at the beginning. That is, laying the foundations of data mining, the process of knowledge discovery in databases and everything surrounding decision trees. Next, we make clear what our main objectives are and what we are pursuing with this work. Then we explain in depth our database, its origin and the changes we have made to it. Once this explanation is finished, we go on to explain in detail the whole quantitative process of creating a decision tree and its different variants, as well as how to act when we encounter problems such as the nature of the variables or even missing data. However, the tree is not built with pencil and paper, but by software, so in the next step we develop all the necessary software to create our algorithm and our application in R, this is when we can start using our application through a manual and consulting this essay when we want to have more information about decision trees.

# Introducción

En la actualidad, se genera un volumen de datos desmesurado en comparación a lo que se producía años atrás. La minería de datos surge como fruto de la necesidad de extraer valor de la masiva cantidad de datos que actualmente se almacenan en los sistemas informáticos de organizaciones, compañías, entidades gubernamentales o personas físicas. Estos datos ya no son tratados como producto final, sino que se abordan desde la perspectiva de insumos, siendo explotados y aprovechados para generar un bien final, el conocimiento, el cual se utiliza en la toma de decisiones aplicadas al campo al que pertenecen los datos. Sin embargo, en numerosas ocasiones de la vida real, los datos se convierten en conocimiento mediante una evaluación a juicio personal por el especialista correspondiente, la cual se lleva a cabo de forma manual. De todas formas, este proceso suele ser lento, costoso y sobre todo subjetivo, además de inaplicable a situaciones donde el volumen se incrementa de manera exponencial. Otra forma de actuar que se esta quedando obsoleta es la de operar sobre la base de datos operacional junto al procesamiento transaccional en línea (On-Line Transaction Processing, OLTP) a través de consultas ejecutadas con lenguajes generalistas de consulta como SQL, ya que solo posibilita el acceso a información sintetizada mediante informes, que por otra parte es poco versátil y en especial poco escalable a grandes volúmenes de datos. Como consecuencia, surge una nueva arquitectura indispensable hoy en día en la minería de datos, que se conoce como el almacén de datos (data warehouse), el cual consiste en un repositorio de fuentes disímiles de datos, unificadas y estructuradas en un esquema que simplifica su análisis con el objetivo de apoyar la toma de decisiones, además engloba las operaciones de procesamiento analítico en línea (OLAP). Igualmente, aunque las herramientas OLAP aceptan análisis descriptivos y de síntesis que favorecen la creación de otros datos más refinados, no crean patrones o reglas que se puedan emplear en otros datos.

Por tanto, una vez aceptadas las carencias que el potencial humano tiene para convertir el gigantesco volumen de datos existente en información valiosa, acompañado de las falencias que los procedimientos clásicos poseen, emerge una nueva serie de instrumentos y procedimientos para favorecer la extracción de conocimiento fructífero desde la información disponible, la cual se comprende bajo el apelativo de minería de datos, y que se diferencia principalmente en que no procura datos, sino conocimiento. En resumidas cuentas, aunque existen muchas formas distintas de definir la minería de datos, una definición válida podría ser la aportada por Clark y Boswell (2000), que describieron la minería de datos como “El proceso de extraer conocimiento útil y comprensible, previamente desconocido, desde grandes cantidades de datos almacenados en distintos formatos”. De esta manera, podemos destacar en esencia dos desafíos a los que la minería de datos debe plantar cara, por un lado, trabajar con grandes volúmenes de datos enfrentándose a los numerosos problemas que ello conlleva, como intratabilidad, volatilidad o datos ausentes, y por otro lado designar correctamente las técnicas oportunas para el análisis, de modo que se adquiera el conocimiento de la manera más útil y eficaz posible, sin olvidarse, eso sí, de que los resultados deben ser fácilmente interpretables para el usuario final, el cual no tiene por qué ser especialista

en la materia para poder comprenderlos.

Sin embargo, no es exactamente lo mismo la minería de datos que el proceso de extracción del conocimiento (Knowledge Discovery in Databases, KDD), aunque ambos términos van estrechamente ligados, puesto que la minería de datos forma parte del KDD, el cual se compone de varias fases, siendo aquella una etapa más de este proceso. Podríamos definir el KDD como "el proceso no trivial de identificar patrones válidos, novedosos, potencialmente útiles y, en última instancia, comprensibles a partir de los datos" (Fayyad et al., 1996).

Como dice la definición, el KDD no solo consiste en obtener patrones como la minería de datos, sino que también incluye un proceso de evaluación e interpretación de los mismos. Cabe destacar que el KDD es además un procedimiento iterativo e interactivo, el primero porque para obtener conocimiento de calidad, normalmente son imprescindibles varias iteraciones, de manera que la salida de una fase nos conduce a otras anteriores, y el segundo porque siempre precisa de un usuario que manipule y guíe el proceso. Estas fases son principalmente cinco:

- Integración y recopilación: En esta primera fase se define la fuente o fuentes de información que puedan ser susceptibles de utilización, así como dónde obtenerlas. Acto seguido, se trata de integrar toda la información en un mismo formato, mediante transformaciones de los datos y solucionando cualquier problema de inconsistencias. Esta tarea resulta mucho más sencilla si se hace uso de un almacén de datos, ya que nos permitirá explorar y examinar los datos con anterioridad, de modo que podamos identificar aspectos a analizar.
- Selección y limpieza: Al provenir de todo tipo de fuentes, muchas veces los datos contienen valores incorrectos o ausentes, es en esta etapa donde nos deshacemos de los primeros y acordamos la estrategia a seguir con los segundos. También se visualizan los datos más esquematizados con el fin de poder desechar las variables más irrelevantes, es decir se realiza tanto la criba horizontal de registros, como la vertical de atributos. De tal forma, se pretende facilitar la tarea a la minería de datos, que es la siguiente etapa.
- Minería de datos: Una vez finalizada la parte de preparación de datos, la cual englobaría las dos primeras fases, llega la fase más crítica, la de minería de datos, donde tenemos varias tareas por delante. En primer lugar, debemos especificar la tarea a utilizar, como por ejemplo clasificación, regresión, agrupamiento, correlaciones o reglas de asociación. En este trabajo vamos a centrarnos en usar una tarea de clasificación. Posteriormente, elegiremos el modelo que más nos convenga en función de si queremos obtenerlo en forma de patrones, reglas, etc. Nosotros haremos uso de los árboles de decisión, pero también existen técnicas de inferencia estadística, redes neuronales, algoritmos genéticos, inducción de reglas, aprendizaje bayesiano, etc. A continuación, definiremos el algoritmo de minería que lleve a cabo el cometido y alcance el tipo de modelo que se busca. En nuestro caso haremos uso del algoritmo con entropía para árboles de decisión, aunque también existe como alternativa el algoritmo que se basa en el índice de Gini. Finalmente nos dispondremos a construir el modelo, donde será necesario probar diferentes opciones hasta dar con el que resulte más efectivo, es aquí donde nos percataremos del carácter iterativo del proceso, ya que en numerosas ocasiones nos veremos obligados a volver hacia las primeras fases para efectuar algunos cambios que nos ayuden a encontrar el mejor modelo. Además, para lograr una predicción robusta es necesario tener bien delimitadas las fases de entrenamiento estimando el modelo con una parte de los datos, y la de validación, contrastándolo con el resto de datos.
- Evaluación e interpretación: En esta etapa se valorará la calidad de los patrones obtenidos por el algoritmo, lo cual no será tarea fácil ya que se ha de comprobar que posean tres cua-

lidades principales: Precisión, fácil comprensión y utilidad. Existe más de una técnica de evaluación, dependiendo de la naturaleza de la tarea, es decir, si esta es de clasificación, reglas de asociación, regresión, agrupamiento, etc. Algunos ejemplos pueden ser la validación simple, la validación cruzada con *n* pliegues o el bootstrapping. En nuestro caso, como la validación simple está incorporada en el propio algoritmo de árboles de decisión con entropía, es la que usaremos. Conviene señalar que a la hora de interpretar los resultados se debe contrastar el conocimiento que el modelo nos brinda, con el que ya tenemos sobre la materia en cuestión, de manera que podamos detectar y resolver cualquier problema subyacente.

- Difusión y uso: Una vez validado el modelo, llegamos a la última fase, donde podemos hacer uso de este con dos propósitos: El de que un especialista sugiera conductas que se fundamenten en el modelo y los resultados del mismo, o utilizar el modelo aplicándolo a diversos conjuntos de datos.

Centrándonos más en la fase de la minería de datos, es necesario enfatizar que en cuanto al conocimiento extraído, este puede ser en formato de relaciones, patrones, reglas, o incluso en forma de una breve sinopsis, de manera que dichos formatos establecen el modelo de los datos en cuestión. Estos modelos pueden ser de dos tipos:

- Supervisados o predictivos: Son aquellos que, a partir de unos atributos conocidos, son capaces de predecir el valor de un atributo en un conjunto de datos. Para ello, parten de datos cuya categoría es conocida para inducir una relación entre dicha categoría y otro grupo de atributos, de esta manera predicen datos donde cuya categoría a priori sea desconocida.
- No supervisados o descriptivos: Tratan de averiguar patrones y tendencias en datos sin referencia a resultados etiquetados. A diferencia de los anteriores, no conocemos los valores de salida, por lo que no se puede entrenar al algoritmo. En este caso lo que se pretende es descubrir la estructura subyacente de los datos.

Ambos modelos se utilizan dependiendo de la tarea que se quiera abordar, los primeros se ponen en práctica para acometer tareas de clasificación, priorización o regresión, mientras que hacemos uso de los segundos cuando nos enfrentamos a tareas de agrupamiento, correlaciones, reglas de asociación, dependencias funcionales o detección de valores anómalos. A su vez, existe un amplio abanico de métodos diferentes para resolver cada tarea, y curiosamente también un mismo método puede resolver varias tareas diferentes. Algunos de ellos son los algebraicos, bayesianos, las tablas de contingencia, los relacionales, las redes neuronales, las máquinas de soporte vectorial, los estocásticos, los basados en distancias, y los que nosotros vamos a utilizar en este trabajo, los árboles de decisión.

Resulta significativo señalar que las diferentes técnicas de minería de datos, gracias a su heterogeneidad, se pueden aplicar a cualquier tipo de datos, entre los que es factible subrayar tres grandes grupos, los estructurados originarios de las bases de datos relacionales, otras clases de datos estructurados (espaciales, textuales, temporales o multimedia), y datos no estructurados procedentes de repositorios o de la web. Por su sencillez, para el trabajo que nos ocupa basta con aplicar nuestro algoritmo a los primeros mencionados, los datos estructurados originarios de las bases de datos. Aunque con modificaciones del algoritmo también se podría aplicar a cualquier de los otros dos tipos de datos.

Ahora una vez definidos los términos principales de este trabajo veremos su estructura:

Tras un primer apartado con los objetivos del trabajo, nuestro primer capítulo tratará de explicar de manera detallada nuestra base de datos, su origen, tamaño, tema, cada una de sus variables por separado y el preprocesamiento necesario para facilitar su posterior procesamiento.

En nuestro segundo capítulo trataremos de desarrollar y explicar todo el proceso cuantitativo que hay detras de la creación de un árbol de decisión binario con entropía, así como los diferentes tipos de árboles en función de la naturaleza de sus variables y los diferentes criterios de división que pueden adoptar. Además también hablaremos de cómo afrontar algunas adversidades como son los datos faltantes o las variables numéricas.

El siguiente capítulo se centrará en explicar cómo utilizar nuestra aplicación, así como describir e interpretar el código que hemos desarrollado para elaborar tanto nuestro árbol de decisión como nuestra aplicación en R.

Nuestro capítulo final tratará de exponer unas breves conclusiones finales fruto de la elaboración de todo el trabajo.

Por último, se ofrecerá la bibliografía utilizada para la realización de este trabajo, en esta encontraremos diversos libros, artículos, y páginas web.

# Objetivos

En este trabajo se perseguirán los siguientes objetivos:

- Desarrollar el algoritmo de los árboles de decisión con entropía mediante la minería de datos.
- Llevar a cabo la búsqueda de un conjunto de datos que cumpla las condiciones necesarias para aplicar dicho algoritmo, llevando a cabo el preprocesamiento requerido para hacer que estos sean adecuados de cara al ajuste del modelo (limpiando la base de datos, y dividiéndola, por un lado, en datos de testeo y, por otro, en datos de entrenamiento).
- Describir los datos visualmente y de manera que sirva de entrada para la toma de decisiones.
- Crear una aplicación web mediante el lenguaje de programación R para desarrollar en ella el procedimiento de los árboles de decisión, así como dibujar el resultado, y utilizarla para clasificar los datos o hacer predicciones.

# Capítulo 1

# Datos reales

Nuestra meta era dar con una base de datos que cumpliese el requisito de contar con una clara variable dependiente categórica de dos categorías, ya que es lo que pretendemos predecir con nuestro árbol de decisión, es decir, para un determinado conjunto de datos, predecir una categoría o la otra. En cuanto a las variables independientes, no nos concierne su naturaleza, sí que nos preocupa más su número, ya que para obtener resultados más fiables es necesaria una base de datos con varias variables explicativas. Para ello, hemos decidido acudir a Kaggle, una comunidad en línea de científicos de datos y profesionales del aprendizaje automático. Esta plataforma permite a los usuarios descubrir y compartir conjuntos de datos, así como investigar y desarrollar modelos en un entorno virtual dedicado a la ciencia y la minería de datos.

Finalmente, la base de datos seleccionada se denomina en Kaggle como "Diabetes prediction dataset" (Kaggle, 2023), la cual se trata de un conjunto de datos médicos y demográficos de pacientes, junto con su estado diabético (positivo o negativo), que cuenta con 100.000 registros y 9 variables, de las cuales una es dependiente y ocho son independientes, en la Figura 1.1 podemos apreciar una vista previa. Los datos incluyen características como la edad, el sexo, el índice de masa corporal (IMC), la hipertensión, las cardiopatías, el historial de tabaquismo, el nivel de HbA1c (prueba de hemoglobina glicosilada) y el nivel de glucosa en sangre. Este conjunto de datos puede utilizarse para crear modelos de minería de datos que permitan predecir la diabetes en pacientes basándose en su historial médico y en información demográfica. Además de encajar perfectamente con nuestras pretensiones, esto puede ser útil para que los profesionales sanitarios identifiquen a los pacientes que pueden estar en riesgo de desarrollar diabetes, de manera que puedan desarrollar planes de tratamiento personalizados. Además, el conjunto de datos puede ser utilizado por los investigadores para explorar las relaciones entre diversos factores médicos y demográficos, así como la probabilidad de desarrollar diabetes. A continuación, describiremos las variables una por una:

- Diabetes: La diabetes es la variable objetivo que se predice, con dos categorías que indican la presencia o ausencia de diabetes, codificadas con valor 1 para representar la presencia de diabetes y con valor 0 para representar la ausencia de diabetes.
- Nivel de glucosa en sangre (blood_glucose_level): Se refiere a la cantidad de glucosa en el torrente sanguíneo en un momento dado. Los niveles elevados de glucosa en sangre son un indicador clave de la diabetes. Se mide en una escala entre 80 y 300 en esta base de datos.
- Prueba de hemoglobina glicosilada (HbA1c_level): El nivel de HbA1c (Hemoglobina A1c) es una medida del nivel medio de azúcar en sangre de una persona en los últimos 2-3 meses. Los niveles más altos indican un mayor riesgo de desarrollar diabetes. En la mayoría de los casos, un nivel de HbA1c superior al 6,5 % indica diabetes.

- Índice de masa corporal (bmi): Es una medida de la grasa corporal basada en el peso y la estatura. Los valores más altos de IMC están relacionados con un mayor riesgo de diabetes. El intervalo del IMC en este conjunto de datos va de 10,16 a 71,55. Un IMC inferior a 18,5 es bajo peso, de 18,5 a 24,9 es normal, de 25 a 29,9 es sobrepeso y 30 o más es obesidad.
- Historial de tabaquismo (smoking_history): Los antecedentes de tabaquismo también se consideran un factor de riesgo para la diabetes y pueden agravar las complicaciones asociadas a la enfermedad. Se consideran las categorías de "Sin información" ("No info"), "Nunca" ("never"), "Fumador habitual" ("current"), "ex fumador de largo plazo" ("former"), "fumador esporádico" ("ever") y "ex fumador de corto plazo" ("not current").
- Enfermedad cardiaca (heart_disease): Es otra afección médica que se asocia a un mayor riesgo de desarrollar diabetes. Cuenta con dos categorías, tener enfermedad cardiaca o no tenerla, codificadas con valor 1 y con valor 0 respectivamente.
- Hipertensión (hypertension): Es una condición médica en la que la presión arterial es persistentemente elevada. Tiene dos categorías indicando si se tiene o no se tiene hipertensión, codificadas con valores 1 y 0 respectivamente.
- Edad (age): La edad es un factor importante, ya que la diabetes se diagnostica con más frecuencia en adultos mayores. Los valores de esta variable en nuestra base de datos van de 0 a 80 años.
- Género (gender): El género se refiere al sexo biológico del individuo, que puede influir en su susceptibilidad a la diabetes. En él se distinguen dos categorías, "Female" (femenino) y "male" (masculino).

Las historias clínicas electrónicas (HCE) son la principal fuente de datos. Las HCE son versiones digitales de los historiales médicos de los pacientes que contienen información sobre su historial médico, diagnóstico, tratamiento y resultados. Los proveedores sanitarios, como hospitales y clínicas, recogen y almacenan los datos de las HCE como parte de su práctica clínica habitual. El uso de HCE como fuente de datos tiene varias ventajas. En primer lugar, las HCE contienen una gran cantidad de datos de pacientes, incluida información demográfica y clínica, que puede utilizarse para desarrollar modelos de aprendizaje automático precisos. En segundo lugar, las HCE ofrecen una visión longitudinal de la salud de un paciente a lo largo del tiempo, que puede utilizarse para identificar patrones y tendencias en su estado de salud. Por último, las HCE se utilizan ampliamente en la práctica clínica, lo que hace que este conjunto de datos adquiera relevancia para los entornos sanitarios del mundo real.

En cuanto al preprocesamiento de la base de datos, ha sido necesario depurarla, ya que presentaba valores faltantes en alguna variable, contando con 18 pacientes con algún valor no válido. En el resto de la base de datos no ha sido necesario hacer ningún cambio, esto es debido a que la base de datos estaba previamente trabajada en su mayoría para garantizar su coherencia y eliminar cualquier información irrelevante o incompleta. Tras el preprocesamiento, la base de datos queda finalmente conformada por registros de 99.982 pacientes en 9 variables. Es importante mencionar que para el buen procesamiento de nuestra base de datos hemos procedido a sustituir, mediante excel, los puntos por comas, así como etiquetar todas las variables cualitativas que estaban codificadas numéricamente. En la Figura 1.1 podemos observar una parte de la base de datos recién extraída de Kaggle, mientras que en la Figura 1.2 tenemos la misma una vez aplicados los cambios.

| | A | B | C | D | E | F | G | H | I |
|---|---|---|---|---|---|---|---|---|---|
| 1 | gender | age | hypertension | heart_disease | smoking_history | bmi | HbA1c_level | blood_glucose_level | diabetes |
| 2 | 1 | 80.0 | 0 | 1 | 2 | 25.19 | 6.6 | 140 | 0 |
| 3 | 1 | 54.0 | 0 | 0 | 1 | 27.32 | 6.6 | 80 | 0 |
| 4 | 2 | 28.0 | 0 | 0 | 2 | 27.32 | 5.7 | 158 | 0 |
| 5 | 1 | 36.0 | 0 | 0 | 3 | 23.45 | 5.0 | 155 | 0 |
| 6 | 2 | 76.0 | 1 | 1 | 3 | 20.14 | 4.8 | 155 | 0 |
| 7 | 1 | 20.0 | 0 | 0 | 2 | 27.32 | 6.6 | 85 | 0 |
| 8 | 1 | 44.0 | 0 | 0 | 2 | 19.31 | 6.5 | 200 | 1 |
| 9 | 1 | 79.0 | 0 | 0 | 1 | 23.86 | 5.7 | 85 | 0 |
| 10 | 2 | 42.0 | 0 | 0 | 2 | 33.64 | 4.8 | 145 | 0 |

Figura 1.1: Diabetes prediction dataset (10 primeros registros). Fuente: Kaggle.

| | A | B | C | D | E | F | G | H | I |
|---|---|---|---|---|---|---|---|---|---|
| 1 | gender | age | hypertensio | eart_disease | moking_histo | bmi | HbA1c_level | od_glucose_l | diabetes |
| 2 | Female | 80 | No | Yes | never | 25,19 | 6,6 | 140 | Non diabetic |
| 3 | Female | 54 | No | No | No Info | 27,32 | 6,6 | 80 | Non diabetic |
| 4 | Male | 28 | No | No | never | 27,32 | 5,7 | 158 | Non diabetic |
| 5 | Female | 36 | No | No | current | 23,45 | 5 | 155 | Non diabetic |
| 6 | Male | 76 | Yes | Yes | current | 20,14 | 4,8 | 155 | Non diabetic |
| 7 | Female | 20 | No | No | never | 27,32 | 6,6 | 85 | Non diabetic |
| 8 | Female | 44 | No | No | never | 19,31 | 6,5 | 200 | Diabetic |
| 9 | Female | 79 | No | No | No Info | 23,86 | 5,7 | 85 | Non diabetic |
| 10 | Male | 42 | No | No | never | 33,64 | 4,8 | 145 | Non diabetic |

Figura 1.2: Diabetes prediction dataset (10 primeros registros) con etiquetas. Fuente: Elaboración propia a partir de Kaggle.

# Capítulo 2

# Desarrollo

Antes de centrarnos en los árboles de decisión, resulta conveniente para su comprensión una explicación más detallada del concepto de aprendizaje supervisado. En primer lugar, es necesario mencionar que esta técnica de la minería de datos es una rama del aprendizaje automático, más conocido por su connotación en inglés "Machine learning", un método de análisis que permite a los algoritmos aprender de manera iterativa de los datos, de modo que estos sistemas se mejoren de forma autónoma con el tiempo sin intervención humana, por ende, posibilita la predicción de comportamientos, respuestas y situaciones. Con el objetivo de realizar tareas específicas, entrena un algortimo haciendo uso de un conjunto de datos etiquetados. El aprendizaje supervisado predice resultados conocidos a partir de modelos. Por ejemplo, en nuestro trabajo, el proceso de aprendizaje supervisado consiste en clasificar pacientes con o sin diabetes a partir de otras variables conocidas. Para ello, es primordial que los datos de entrenamiento estén correctamente etiquetados. Como rama del aprendizaje automático, el aprendizaje supervisado permite que los algoritmos aprendan de datos históricos de entrenamiento y los apliquen a entradas desconocidas para así lograr la salida correcta. A su vez, los dos tipos de tareas más populares que se engloban dentro del aprendizaje supervisado, son la regresión y la clasificación. La primera consiste en entrenar a un algoritmo con el fin de predecir una salida utilizando un rango continuo de valores posibles. En la regresión, el valor de salida es una función de los parametros de entrada, por tanto el algoritmo debe hallar una relación funcional entre estos. En nuestro caso, nos vamos a enfocar en la segunda, ya que nuestra variable respuesta es categórica, y por tanto nuestro objetivo es desarrollar un árbol de decisión. En el supuesto caso de que nuestra variable respuesta fuese continua, la tarea a desarrollar sería de regresión y el árbol se denominaría, como la tarea indica, árbol de regresión.

## 2.1. Clasificación: Conceptos básicos

La clasificación extrae modelos que describen clases de datos. Estos modelos, denominados clasificadores, predicen etiquetas de clase categóricas. La investigación reciente en minería de datos se ha basado en estos trabajos y ha desarrollado técnicas escalables de clasificación y predicción capaces de manejar grandes cantidades de datos residentes en disco. Sin embargo, la mayoría de los algoritmos se basan en la memoria y, a menudo, en los datos. La clasificación tiene una gran cantidad de aplicaciones, como la detección de fraudes, el marketing selectivo, la predicción del rendimiento, la fabricación y, en nuestro caso, el diagnóstico médico (Han & Kamber, 2006).

### 2.1.1. ¿Que es la clasificación en minería de datos?

Empecemos por algunos ejemplos. Un vendedor de seguros necesita analizar sus datos para saber qué clientes son "potenciales compradores" y cuáles no lo son, "no potenciales compradores", para la aseguradora. Un responsable de préstamos de una entidad bancaria necesita un modelo para averiguar qué solicitantes de préstamos son "seguros" y cuáles son "arriesgados" para la entidad bancaria. Un investigador médico quiere analizar datos sobre la enfermedad de Parkinson para predecir qué tratamiento debe recibir un paciente entre tres tratamientos específicos. En cada uno de estos ejemplos, la tarea de análisis de datos es la clasificación, en la cual se construye un modelo o clasificador de modo que sea capaz de predecir etiquetas de clase o categorías, como "seguro" o "arriesgado" para los datos de la solicitud de préstamo; "sí" o "no" para los de marketing; o "tratamiento A", "tratamiento B" o "tratamiento C" para los datos médicos. Estas categorías pueden representarse mediante valores discretos, en los cuales el orden entre los valores no tiene ningún significado. Por ejemplo, los valores 1, 2 y 3 pueden utilizarse para representar los tratamientos A, B y C, sin que exista un orden implícito.

### 2.1.2. ¿Cómo funciona la clasificación?

La clasificación de datos es un proceso de dos pasos, el primero consta de un proceso de aprendizaje en el que se construye el modelo de clasificación, mientras que en el segundo se utiliza el modelo construido para predecir etiquetas de clase. En la Figura 2.1 se muestra un proceso donde los datos se han simplificado con fines ilustrativos, en realidad, cabe esperar que se tengan en cuenta muchos más atributos. En primer lugar, se construye un clasificador que describe un conjunto predeterminado de clases o conceptos de datos, esta es la etapa de aprendizaje o fase de entrenamiento, en la que un algoritmo de clasificación construye el clasificador a partir de analizar y aprender de un conjunto de entrenamiento formado por tuplas de la base de datos, así como sus etiquetas de clase asociadas. Una tupla, $X$, está representada por un vector de atributos $n-$dimensional, $X = (x_1, x_2, \ldots, x_n)$, el cual representa $n$ medidas realizadas en la tupla a partir de $n$ atributos de la base de datos, respectivamente, $A_1, A_2, \ldots, A_n$. Cada atributo representa una "característica" de $X$. De hecho, en la literatura sobre reconocimiento de patrones, en numerosas ocasiones se hace uso del término vector de características en lugar de vector de atributos. Se supone que cada tupla, $X$, pertenece a una clase predefinida, determinada por otro atributo de la base de datos denominado etiqueta de clase. Esta etiqueta de clase tiene valores discretos y no está ordenada, además es nominal en el sentido de que cada valor sirve como categoría o clase. Las tuplas individuales que componen el conjunto de entrenamiento se denominan tuplas de entrenamiento y se extraen aleatoriamente de la base de datos analizada. Asimismo, en el contexto de la minería de datos, las tuplas también pueden denominarse muestras, ejemplos, instancias, puntos de datos u objetos.

De acuerdo a lo señalado previamente, podemos notar que en la clasificación se proporciona la etiqueta de clase de cada tupla de entrenamiento, este paso ilustra perfectamente el aprendizaje supervisado, de modo que podemos afirmar que el aprendizaje del clasificador es supervisado en el sentido de que se le dice a qué clase pertenece cada tupla de entrenamiento. Esto contrasta notoriamente con el aprendizaje no supervisado, en el cual no se conoce la etiqueta de clase de cada tupla de entrenamiento, y el número o conjunto de clases a aprender puede no conocerse de antemano. Por ejemplo, si no dispusiéramos de los datos de decisión de un préstamo para el conjunto de entrenamiento, podríamos utilizar la agrupación para intentar determinar grupos de tuplas similares, que podrían corresponder a grupos de riesgo dentro de los datos de solicitud de préstamo. Este primer paso del proceso de clasificación también puede considerarse como el aprendizaje de una asignación o función, $Y = f(X)$, la cual puede predecir la etiqueta de clase asociada $Y$ de una tupla dada $X$. Desde este punto de vista, pretendemos aprender una asignación o función que separe las

clases de datos. Generalmente, esta asignación se representa en forma de reglas de clasificación, árboles de decisión o fórmulas matemáticas. En nuestro caso, el mapeo se representará como un árbol de decisión que identificará los pacientes como diabéticos o no diabéticos. El árbol de decisión nos proporcionará una solución muy visual, mientras que el uso de reglas de clasificación proporciona una representación más comprimida de los datos.

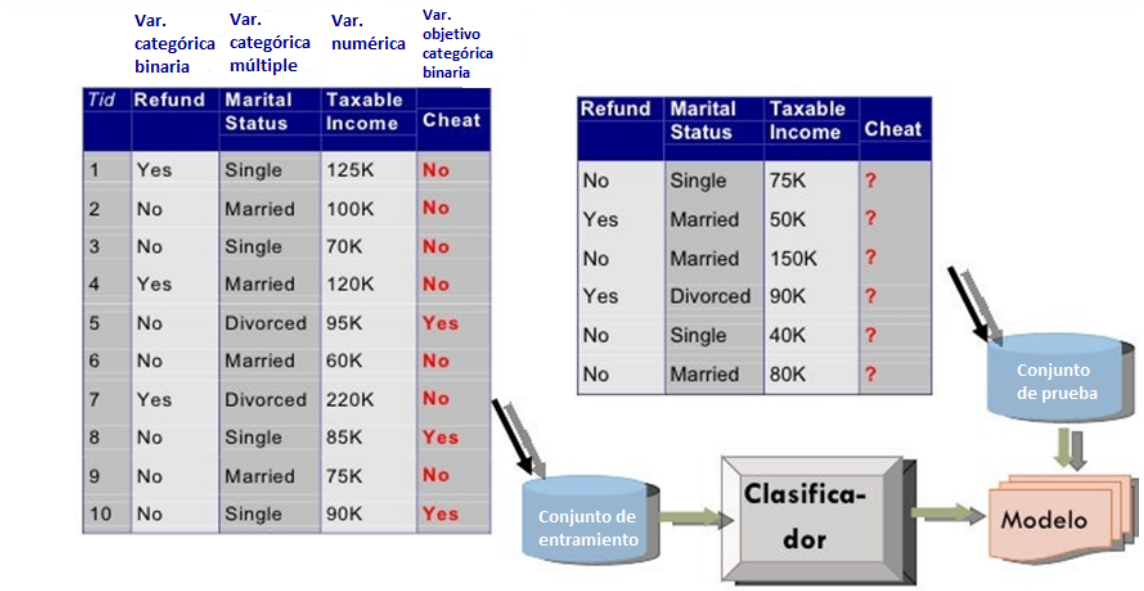



Figura 2.1: Ejemplo de Clasificación. Fuente: Propia, basada a partir de: https://shorturl.at/ltzU6

### 2.1.3. La precisión en la clasificación

Resulta esencial llevar a cabo no solo la estimación, sino la precisión predictiva del clasificador para obtener buenos resultados. Si utilizáramos el conjunto de entrenamiento para medir la precisión del clasificador, esta estimación probablemente sería demasiado optimista, ya que el clasificador tiende a sobreajustar los datos, es decir, durante el proceso de aprendizaje puede incorporar algunas anomalías particulares de los datos de entrenamiento que no están presentes en el conjunto general de datos. Por lo tanto, se utiliza un conjunto de testeo, formado por tuplas de testeo y sus etiquetas de clase asociadas, las cuales son independientes de las tuplas de entrenamiento, lo que significa que no se han utilizado para construir el clasificador. La precisión de un clasificador en un conjunto de testeo determinado, es el porcentaje de tuplas del conjunto de testeo que el clasificador clasifica correctamente. La etiqueta de clase asociada a cada tupla de testeo se compara con la predicción de clase del clasificador aprendido para esa tupla. Si la precisión del clasificador se considera aceptable, el clasificador puede utilizarse para clasificar futuras tuplas de datos cuya etiqueta de clase se desconozca, también llamados datos desconocidos. De esta manera, si la precisión de nuestro clasificador es aceptable, podremos usar nuestro árbol, el cual ha sido construido gracias al conjunto de entrenamiento de datos de pacientes previos, para predecir si un paciente va a tener o no diabetes.

## 2.2. Clasificación mediante el uso de árboles

En relación a los algoritmos basados en árboles es posible diferenciar dos grandes grupos, los árboles de decisión y los de regresión. Si bien ambos se utilizan para aprender patrones de clasificación y predicción a partir de datos, además de expresar la relación de las variables de atributo $X$

con una variable objetivo $Y$, $Y = f(X)$, en forma de árbol. Los primeros clasifican el valor objetivo categórico de un registro de datos utilizando sus valores de atributo. Mientras que, en cambio, los segundos, a partir de estos valores de atributo, predicen el valor objetivo numérico de un registro de datos. En este caso, primero definiremos un árbol de decisión binario y proporcionaremos el algoritmo de este a partir de un conjunto de datos con todas las variables tanto de atributo como objetivo categóricas. A continuación, se describe el método de aprendizaje de un árbol de decisión no binario. Más adelante, se introducen conceptos adicionales para manejar variables de atributo numéricas como en el caso de nuestra base de datos, así como valores perdidos de variables de atributo. Y por último, nos centraremos en los árboles de regresión, es decir aquellos que tienen una variable objetivo numérica. Es decir, con el objetivo de facilitar la comprensión de los algoritmos partiremos del caso más sencillo para ir añadiendo complejidad de manera gradual. Por tanto, aunque el árbol de decisión que nosotros necesitamos para nuestros datos sea de tipo binario y para variables independientes tanto categóricas como numéricas, describiremos todos los diferentes casos que se pueden dar dependiendo de la naturaleza de las variables de la base de datos.

### 2.2.1. El árbol de decisión binario

En este apartado presentaremos los elementos que componen un árbol de decisión, el razonamiento que se ha de seguir para obtener un árbol de decisión con la mínima longitud de descripción, así como los principales criterios de división a la hora de conformar el árbol. Por último, ilustraremos la construcción descendente de un árbol de decisión.

#### Elementos de un árbol de decisión

La Figura 2.2 presenta una parte del conjunto de datos de un sistema de fabricación. El conjunto incluye nueve variables independientes para la calidad de las piezas y una variable dependiente para el fallo del sistema, y hacemos uso de él como conjunto de datos de entrenamiento para instruir a un árbol de decisión binario a clasificar si el sistema es defectuoso o no mediante el uso de los valores de las nueve variables de calidad. La figura 2.3 muestra el árbol de decisión binario resultante de modo que se aprecien sus elementos. Como se observa en dicha figura, un árbol de decisión binario es un gráfico con nodos, el nodo raíz en la parte superior del árbol está formado por todos los registros de datos del conjunto de datos de entrenamiento. Para el conjunto de datos de detección de fallos del sistema, el nodo raíz contiene un conjunto con los 10 registros de datos del conjunto de datos de entrenamiento, $\{1, 2, \ldots, 10\}$. Es importante resaltar que debemos de tener en cuenta que los números del conjunto de datos son los números de instancia. El nodo raíz se divide en dos subconjuntos, $\{2, 4, 8, 9, 10\}$ y $\{1, 3, 5, 6, 7\}$, utilizando la variable de atributo, $x_7$, y sus dos valores categóricos, $x_7 = 0$ y $x_7 = 1$. Todos los casos del subconjunto se dividen en dos subgrupos, de forma que todas las instancias del subconjunto $\{2, 4, 8, 9, 10\}$ tienen $x_7 = 0$ y todas las instancias del subconjunto $\{1, 3, 5, 6, 7\}$ tienen $x_7 = 1$. Cada subconjunto se representa como un nodo en el árbol de decisión, en el cual se utiliza una expresión booleana para expresar $x_7 = 0$ por "$x_7 = 0$ es VERDADERO", y $x_7 = 1$ por "$x_7 = 0$ es FALSO". $x_7 = 0$ se denomina condición de división o criterio de división, y sus valores VERDADERO y FALSO son los que nos permiten una división binaria del conjunto en el nodo raíz en dos ramas con un nodo al final de cada rama. Cada uno de los dos nuevos nodos pueden dividirse aún más utilizando una de las variables independientes restantes en el criterio de división. Sin embargo, un nodo no puede seguir dividiéndose si los registros del conjunto de datos de ese nodo tienen el mismo valor que la variable dependiente, de manera que este nodo se convierte en un nodo hoja del árbol. Podemos afirmar entonces que, excepto el nodo raíz y los nodos hoja, todos los demás nodos de los árboles de decisión reciben la denominación de nodos internos.
El árbol de decisión puede clasificar un registro utilizando los valores de atributo en el registro de

datos. Por ejemplo, es posible apreciar que el registro 10 se comprueba primero con la primera condición de división en el nodo raíz. A continuación, con $x_7 = 0$, el registro pasa a la rama izquierda, lo mismo con $x_8 = 0$ y lo mismo luego con $x_9 = 0$. Finalmente, el registro termina en el nodo hoja de más a la izquierda, donde toma el valor objetivo para este nodo hoja, $y = 0$, el cual lo clasifica como no defectuoso.

| Máquina averiada | Variables independientes categóricas | | | | | | | | | Variable objetivo |
|---|---|---|---|---|---|---|---|---|---|---|
| | Calidad de las piezas | | | | | | | | | Fallo del sistema |
| | $X_1$ | $X_2$ | $X_3$ | $X_4$ | $X_5$ | $X_6$ | $X_7$ | $X_8$ | $X_9$ | $y$ |
| 1 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 |
| 2 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 |
| 3 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 1 |
| 4 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 |
| 5 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 |
| 6 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 |
| 7 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 |
| 8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |
| 9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| 10 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Figura 2.2: Conjunto de datos para la detección de fallos de sistema. Fuente: Propia, basada a partir de: Ye (2013).

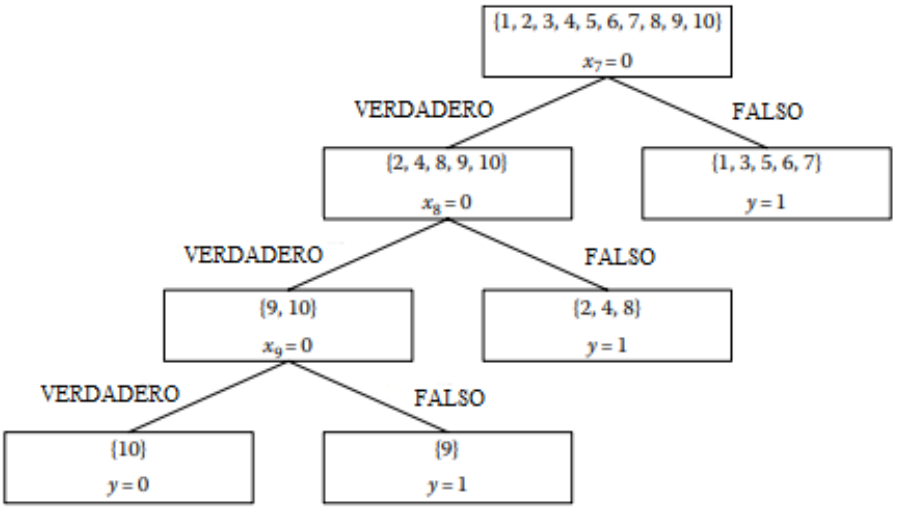



Figura 2.3: Árbol de decisión para la detección de fallos de sistema. Fuente: Propia, basada a partir de: Ye (2013).

### Árbol de decisión con longitud de descripción mínima

Partiendo del nodo raíz con todos los registros del conjunto de datos de entrenamiento, hay nueve formas posibles de dividirlo utilizando las nueve variables independientes del ejemplo de manera individual en la condición de división. Para cada nodo al final de una rama, desde la división del nodo raíz, hay ocho formas posibles de dividir el nodo, utilizando, una vez más, individualmente cada una de las ocho variables de atributo restantes, de manera que este proceso continúa pudiendo dar lugar a numerosos árboles de decisión posibles. Todos estos árboles de decisión posibles

difieren en su tamaño y complejidad. En otras palabras, un árbol de decisión puede ser tan grande que tenga tantos nodos hoja como registros en el conjunto de datos de entrenamiento, de forma que cada nodo hoja contendría cada registro. No obstante, debe ser uno solo el árbol de decisión que se utilice para representar la relación entre las variables independientes y la variable objetivo, y este será el árbol de decisión más pequeño que pueda retener dicha relación, es decir, el árbol que requiera la mínima longitud de descripción. Dados tanto el árbol más pequeño como uno más grande que clasifican correctamente todos los registros del conjunto de datos de entrenamiento, se espera que el árbol de decisión más pequeño generalice los patrones de clasificación mejor que el más grande, de manera que los patrones de clasificación mejor generalizados permiten clasificar mejor más puntos de datos, incluidos los que no están en el conjunto de datos de entrenamiento. Ahora considere un árbol de decisión grande que tenga tantos nodos hoja como registros de datos en el conjunto de datos de entrenamiento, con cada nodo hoja conteniendo cada registro de datos. A pesar de que este algoritmo clasifica correctamente todos los registros de datos de entrenamiento, su rendimiento puede ser deficiente a la hora de clasificar nuevos registros no incluidos en dicho conjunto. Estos nuevos registros poseen conjuntos de valores de atributos diferentes de los de los registros del conjunto de datos de entrenamiento y, por tanto, no siguen las mismas rutas hacia los nodos hoja del árbol de decisión. Necesitamos, pues, un árbol de decisión que capture patrones de clasificación generalizados para la relación mencionada anteriormente. Cuanto más generalizada sea dicha relación, menor será su longitud de descripción, dado que elimina las diferencias específicas entre los registros de datos individuales. Por lo tanto, se puede concluir que cuanto más pequeño sea un árbol de decisión, más capacidad de generalización se espera que tenga.

### Criterios de divisón

Para lograr el objetivo de hallar un árbol de decisión con la mínima longitud de descripción, es necesario conocer de antemano cómo dividir correctamente. Tomemos como ejemplo la creación de un árbol de decisión a partir del conjunto de datos de la Figura 2.2. Como hemos mencionado con anterioridad, existen nueve formas posibles de dividir el nodo raíz haciendo uso de las nueve variables de atributo de forma independiente, como se muestra en la Figura 2.2. Sin embargo, no estamos seguros de cuál de los nueve criterios de división deberíamos utilizar. Un enfoque común de la selección de criterios de división consiste en seleccionar el que produzca los subconjuntos más homogéneos. Un conjunto de datos homogéneo es aquel cuyos registros tienen el mismo valor objetivo. Asimismo, se dispone de varias medidas de homogeneidad de los datos, entre las que destacan la de entropía de la información y la del índice de Gini.
La entropía de la información se introdujo originalmente para medir el número de bits de información necesarios para codificar datos, y se connota como:

$$Entropia\,(D) = \sum_{i=1}^{c} - P_i \log_2 P_i \tag{2.1}$$

$$- 0 \log_2 0 = 0 \tag{2.2}$$

$$\sum_{i=1}^{c} P_i = 1, \tag{2.3}$$

donde $D$ indica un conjunto de datos determinado, $c$ es el número de valores objetivo diferentes y $P_i$ es la probabilidad de que un registro del conjunto de datos tenga el $i-$ésimo valor objetivo.
Un valor de entropía se sitúa en el intervalo $\left[0, \log_2 c\right]$. Por ejemplo, dado el conjunto de datos de

la Figura 2.2, tenemos $c = 2$ (para dos valores objetivo, $y = 0$ e $y = 1$),

$P_1 = 9/10$ (9 de los 10 registros con $y = 1$) $= 0.9$, $P_2 = 1/10$ (1 de los 10 registros con $y = 0$) $= 0.1$:

$$Entropia\,(D) = \sum_{i=1}^{2} - P_i \log_2 P_i = -\,0.9 \log_2 0.9 - \;0.1 \log_2 0.1 = 0.47.$$

A continuación, como ejemplo, desarrollaremos el criterio de división binaria para el nodo raíz mediante el calculo de la entropía de la información para el conjunto de datos de la Figura 2.2. En cada subconjunto calcularemos la entropía de la información media de la división:

$$x_1 = 0 : VERDADERO\ o\ FALSO \qquad \{2,3,4,5,6,7,8,9,10\},\ \{1\}$$
$$Entropia\,(S) = \frac{9}{10} Entropia\,(D_{verdadero}) + \frac{1}{10} Entropia\left(D_{falso}\right) =$$
$$= \frac{9}{10}\left(-\frac{8}{9}\log_2\frac{8}{9} - \frac{1}{9}\log_2\frac{1}{9}\right) + \frac{1}{10}\,0 = 0.45$$

$$x_2 = 0 : VERDADERO\ o\ FALSO \qquad \{1,3,4,5,6,7,8,9,10\},\ \{2\}$$
$$Entropia\,(S) = \frac{9}{10} Entropia\,(D_{verdadero}) + \frac{1}{10} Entropia\left(D_{falso}\right) =$$
$$= \frac{9}{10}\left(-\frac{8}{9}\log_2\frac{8}{9} - \frac{1}{9}\log_2\frac{1}{9}\right) + \frac{1}{10}\,0 = 0.45$$

$$x_3 = 0 : VERDADERO\ o\ FALSO \qquad \{1,2,4,5,6,7,8,9,10\},\ \{3\}$$
$$Entropia\,(S) = \frac{9}{10} Entropia\,(D_{verdadero}) + \frac{1}{10} Entropia\left(D_{falso}\right) =$$
$$= \frac{9}{10}\left(-\frac{8}{9}\log_2\frac{8}{9} - \frac{1}{9}\log_2\frac{1}{9}\right) + \frac{1}{10}\,0 = 0.45$$

$$x_4 = 0 : VERDADERO\ o\ FALSO \qquad \{1,5,6,7,8,9,10\},\ \{2,3,4\}$$
$$Entropia\,(S) = \frac{7}{10} Entropia\,(D_{verdadero}) + \frac{3}{10} Entropia\left(D_{falso}\right) =$$
$$= \frac{7}{10}\left(-\frac{6}{7}\log_2\frac{6}{7} - \frac{1}{7}\log_2\frac{1}{7}\right) + \frac{3}{10}\,0 = 0.41$$

$$x_5 = 0 : VERDADERO\ o\ FALSO \qquad \{2,3,4,6,7,8,9,10\},\ \{1,5\}$$
$$Entropia\,(S) = \frac{8}{10} Entropia\,(D_{verdadero}) + \frac{2}{10} Entropia\left(D_{falso}\right) =$$
$$= \frac{8}{10}\left(-\frac{7}{8}\log_2\frac{7}{8} - \frac{1}{8}\log_2\frac{1}{8}\right) + \frac{2}{10}\,0 = 0.43$$

$$x_6 = 0 : VERDADERO\ o\ FALSO \qquad \{1,2,4,5,7,8,9,10\},\ \{3,6\}$$
$$Entropia\,(S) = \frac{8}{10} Entropia\,(D_{verdadero}) + \frac{2}{10} Entropia\left(D_{falso}\right) =$$
$$= \frac{8}{10}\left(-\frac{7}{8}\log_2\frac{7}{8} - \frac{1}{8}\log_2\frac{1}{8}\right) + \frac{2}{10}\,0 = 0.43$$

$$x_7 = 0 : VERDADERO\ o\ FALSO \qquad \{2,4,8,9,10\},\ \{1,3,5,6,7\}$$
$$Entropia\,(S) = \frac{5}{10} Entropia\,(D_{verdadero}) + \frac{5}{10} Entropia\left(D_{falso}\right) =$$
$$= \frac{5}{10}\left(-\frac{4}{5}\log_2\frac{4}{5} - \frac{1}{5}\log_2\frac{1}{5}\right) + \frac{5}{10}\,0 = 0.36$$

$$x_8 = 0 : VERDADERO\ o\ FALSO \qquad \{1,5,6,7,9,10\},\ \{2,3,4,8\}$$
$$Entropia\,(S) = \frac{6}{10} Entropia\,(D_{verdadero}) + \frac{4}{10} Entropia\left(D_{falso}\right) =$$
$$= \frac{6}{10}\left(-\frac{5}{6}\log_2\frac{5}{6} - \frac{1}{6}\log_2\frac{1}{6}\right) + \frac{4}{10}\,0 = 0.39$$

$$x_9 = 0 : VERDADERO\ o\ FALSO \qquad \{2,3,4,6,7,8,10\},\ \{1,5,9\}$$
$$Entropia\,(S) = \frac{7}{10} Entropia\,(D_{verdadero}) + \frac{3}{10} Entropia\left(D_{falso}\right) =$$
$$= \frac{7}{10}\left(-\frac{6}{7}\log_2\frac{6}{7} - \frac{1}{7}\log_2\frac{1}{7}\right) + \frac{3}{10}\,0 = 0.41$$

En la Figura 2.4 podemos apreciar cómo el valor de la entropía de la información cambia con $P_1$ $\left(\text{y } P_2 = 1 - P_1\right)$ cuando $c = 2$. Concretamente tenemos:

$$P_1 = 0.5,\ P_2 = 0.5,\ Entropia\,(D) = 1$$
$$P_1 = 0,\ P_2 = 1,\ Entropia\,(D) = 0$$
$$P_1 = 1,\ P_2 = 0,\ Entropia\,(D) = 0$$

Si todos los registros de un conjunto de datos toman un mismo valor objetivo, tenemos que $P_1 = 0$, $P_2 = 1$ o $P_1 = 1$, $P_2 = 0$, y el valor de la entropía de la información es 0, es decir, necesitamos 0 bits de información porque ya conocemos el valor objetivo que toman todos los registros. Por lo tanto, el valor de entropía de 0 indica que el conjunto de datos es homogéneo con respecto al valor objetivo. Si la mitad de los registros de un conjunto de datos toma un valor objetivo y la otra mitad toma otro valor objetivo, tenemos que $P_1 = 0.5, P_2 = 0.5$, y el valor de la entropía de la información es 1, lo que significaría que necesitamos 1 bit de información para transmitir cuál es el valor objetivo. En este caso, el valor de entropía de 1 indica que el conjunto de datos totalmente heterogéneo. Por ende, cuando hacemos uso de la entropía de la información como medida de la

homogeneidad de los datos, cuanto menor es el valor de entropía, más homogéneo es el conjunto de datos con respecto al valor objetivo. En definitiva, basándonos en esto, es importante señalar que tras dividir un conjunto de datos en varios subconjuntos, se utiliza la siguiente fórmula para calcular la entropía de información media de los subconjuntos:

$$Entropia\,(S) = \sum_{v \in Valores(S)} \frac{|D_v|}{|D|} Entropia\,(D_v)\,, \tag{2.4}$$

Donde $S$ indica la división, $Valores\,(S)$ hace referencia a un conjunto de valores que se utilizan en el criterio de división, $v$ es un valor genérico de $Valores\,(S)$, $D$ expresa el conjunto de datos que se divide, $|D|$ denota el número de registros del conjunto de datos $D$, $D_v$ es el subconjunto resultante de la división haciendo uso del valor $v$, y por último, $|D_v|$ indica el número de registros de datos del conjunto $D_v$.

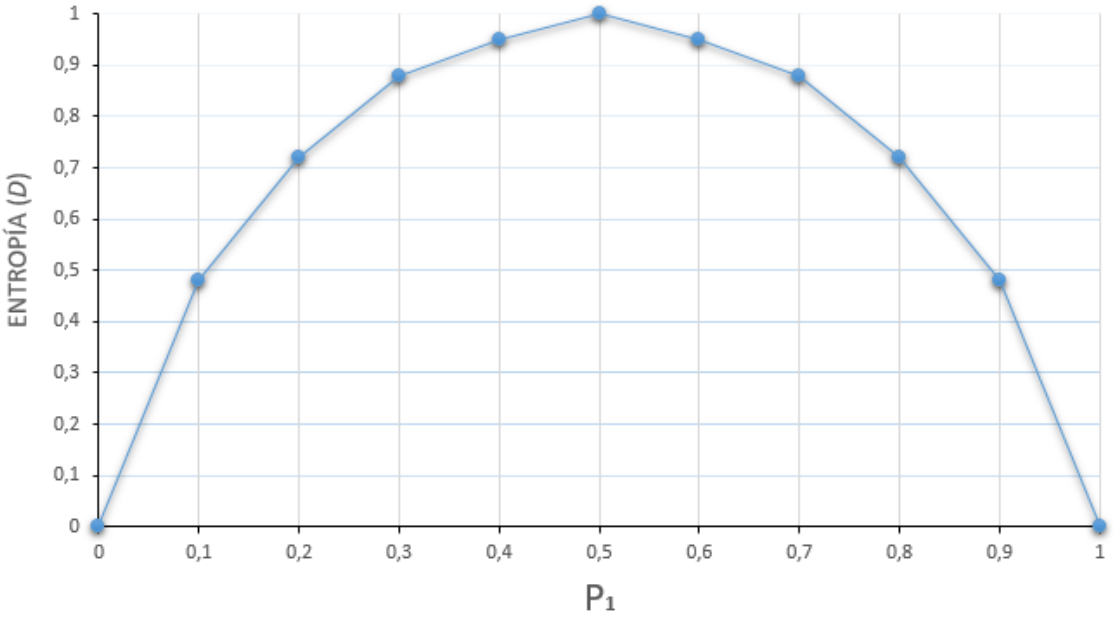



Figura 2.4: Entropía de información. Fuente: Propia, basada a partir de: Ye (2013).

Por ejemplo, el nodo raíz del árbol de decisión para la Figura 2.3 tiene el conjunto de datos, $D = \{1, 2, \ldots, 10\}$, cuyo valor de entropía es 0.47 como se ha mostrado anteriormente. Utilizando el criterio de división, $x_1 = 0 : VERDADERO\ o\ FALSO$, el nodo raíz se divide en dos subconjuntos: $D_{falso} = \{1\}$, que es homogéneo, y $D_{verdadero} = \{2, 3, 4, 5, 6, 7, 8, 9, 10\}$, que es heterogéneo con ocho registros que toman el valor objetivo de 1 y un registro que adquiere el valor objetivo de 0. Por tanto, la entropía media de los dos subconjuntos después de la división es de:

$$Entropia\,(S) = \frac{9}{10} Entropia\,(D_{verdadero}) + \frac{1}{10} Entropia\,\left(D_{falso}\right) =$$
$$\frac{9}{10}\left(-\frac{8}{9}\log_2\frac{8}{9} - \frac{1}{9}\log_2\frac{1}{9}\right) + \frac{1}{10}\,0 = 0.45$$

Dado que esta entropía media de los subconjuntos tras la división es mejor que la entropía original $Entropia\,(D) = 0.47$, se puede afirmar que la división mejora la homogeneidad de los datos. Como podemos observar en el ejemplo anterior, donde hemos calculado la entropía media de los subcon-

juntos después de cada una de las otras ocho divisiones del nodo raíz, entre las nueve divisiones posibles, es la que utiliza el criterio $x_7 = 0 : VERDADERO\ o\ FALSO$, la que produce la menor entropía media de la información, lo que indica que los subconjuntos son más homogéneos. Por lo tanto, el criterio de división $x_7 = 0 : VERDADERO\ o\ FALSO$ será el seleccionado para dividir el nodo raíz, lo que da lugar a dos nodos internos, como se muestra en la Figura 2.3. De esta manera, el nodo con el subconjunto $x_7 = 0 : VERDADERO\ o\ FALSO$ es el nodo raíz, del que parten los demás nodos. Además, el nodo interno con el subconjunto $\{2, 4, 8, 9, 10\}$ no es homogéneo, lo que implica que el árbol de decisión se seguirá ampliando con más divisiones hasta que todos los nodos de hoja sean homogéneos, como se puede apreciar.

Por otra parte, aunque la entropía de la información será el criterio de división implementado en nuestro árbol de decisión, como hemos señalado con anterioridad, no es el único a tener en cuenta, por lo que consideramos relevante describir otro criterio de gran popularidad para medir la homogeneidad de los datos como es el basado en el índice de Gini, el cual se define de la siguiente manera:

$$Gini\,(D) = 1 - \sum_{i=1}^{c} P_i^2 \tag{2.5}$$

Que con el conjunto de datos de la Figura 2.2, para $c = 2$, $P_1 = 0.9$, y $P_2 = 0.1$, un ejemplo sería:

$$Gini\,(D) = 1 - \sum_{i=1}^{2} P_i^2 = 1 - 0.9^2 - 0.1^2 = 0.18$$

Para $c = 2$, con diferentes valores de $P_i$, el índice de Gini tomaría los siguientes valores:

$$P_1 = 0.5,\ P_2 = 0.5,\ Gini\,(D) = 1 - 0.5^2 - 0.5^2 = 0.5$$
$$P_1 = 0,\ P_2 = 1, Gini\,(D) = 1 - 0^2 - 1^2 = 0$$
$$P_1 = 1,\ P_2 = 0, Gini\,(D) = 1 - 1^2 - 0^2 = 0$$

De este modo, cuanto menor sea el valor del índice de Gini, más homogéneo será el conjunto de datos.
Por otro lado, cuando la intención es obtener el valor medio del índice de Gini a raíz de los subconjuntos de datos tras una división, este se calcularía de la misma forma que lo hacíamos con la entropía de la información:

$$Gini\,(S) = \sum_{v\ \in\ Valores(S)} \frac{|D_v|}{|D|} Gini\,(D_v)\,, \tag{2.6}$$

A continuación, desarrollaremos el criterio de división binaria para el nodo raíz mediante el cálculo del índice de Gini para el conjunto de datos de la Figura 2.2. En cada subconjunto, al igual que en el caso de la entropía de la información, calcularemos el índice de Gini medio de la división. Es decir, mostraremos el valor medio del índice Gini de los subconjuntos después de cada una de las nueve divisiones del nodo raíz para el conjunto de datos de entrenamiento de la Figura 2.2. Entre las nueve divisiones posibles, el criterio de división de $x_7 = 0 : VERDADERO\ o\ FALSO$ produce el valor medio más bajo, indicando, de este modo, los subconjuntos más homogéneos. En consecuencia, se selecciona el criterio de división $x_7 = 0 : VERDADERO\ o\ FALSO$ para dividir

el nodo raíz. De esta manera, podemos afirmar que el índice de Gini produce la misma división que la entropía de la información:

$$x_1 = 0 : VERDADERO\ o\ FALSO \qquad \{2,3,4,5,6,7,8,9,10\},\ \{1\}$$
$$Gini(S) = \frac{9}{10}Gini(D_{verdadero}) + \frac{1}{10}Gini\left(D_{falso}\right) =$$
$$= \frac{9}{10}\left(1 - \left(\frac{1}{9}\right)^2 - \left(\frac{8}{9}\right)^2\right) + \frac{1}{10}\,0 = 0.18$$

$$x_2 = 0 : VERDADERO\ o\ FALSO \qquad \{1,3,4,5,6,7,8,9,10\},\ \{2\}$$
$$Gini(S) = \frac{9}{10}Gini(D_{verdadero}) + \frac{1}{10}Gini\left(D_{falso}\right) =$$
$$= \frac{9}{10}\left(1 - \left(\frac{1}{9}\right)^2 - \left(\frac{8}{9}\right)^2\right) + \frac{1}{10}\,0 = 0.18$$

$$x_3 = 0 : VERDADERO\ o\ FALSO \qquad \{1,2,4,5,6,7,8,9,10\},\ \{3\}$$
$$Gini(S) = \frac{9}{10}Gini(D_{verdadero}) + \frac{1}{10}Gini\left(D_{falso}\right) =$$
$$= \frac{9}{10}\left(1 - \left(\frac{1}{9}\right)^2 - \left(\frac{8}{9}\right)^2\right) + \frac{1}{10}\,0 = 0.18$$

$$x_4 = 0 : VERDADERO\ o\ FALSO \qquad \{1,5,6,7,8,9,10\},\ \{2,3,4\}$$
$$Gini(S) = \frac{7}{10}Gini(D_{verdadero}) + \frac{3}{10}Gini\left(D_{falso}\right) =$$
$$= \frac{7}{10}\left(1 - \left(\frac{6}{7}\right)^2 - \left(\frac{1}{7}\right)^2\right) + \frac{3}{10}\,0 = 0.17$$

$$x_5 = 0 : VERDADERO\ o\ FALSO \qquad \{2,3,4,6,7,8,9,10\},\ \{1,5\}$$
$$Gini(S) = \frac{8}{10}Gini(D_{verdadero}) + \frac{2}{10}Gini\left(D_{falso}\right) =$$
$$= \frac{8}{10}\left(1 - \left(\frac{7}{8}\right)^2 - \left(\frac{1}{8}\right)^2\right) + \frac{2}{10}\,0 = 0.175$$

$$x_6 = 0 : VERDADERO\ o\ FALSO \qquad \{1,2,4,5,7,8,9,10\},\ \{3,6\}$$
$$Gini(S) = \frac{8}{10}Gini(D_{verdadero}) + \frac{2}{10}Gini\left(D_{falso}\right) =$$
$$= \frac{8}{10}\left(1 - \left(\frac{7}{8}\right)^2 - \left(\frac{1}{8}\right)^2\right) + \frac{2}{10}\,0 = 0.175$$

$$x_7 = 0 : VERDADERO\ o\ FALSO \qquad \{2,4,8,9,10\},\ \{1,3,5,6,7\}$$

$$Gini(S) = \frac{5}{10} Gini(D_{verdadero}) + \frac{5}{10} Gini\left(D_{falso}\right) =$$

$$= \frac{5}{10}\left(1 - \left(\frac{4}{5}\right)^2 - \left(\frac{1}{5}\right)^2\right) + \frac{5}{10}\,0 = 0.16$$

$$x_8 = 0 : VERDADERO\ o\ FALSO \qquad \{1,5,6,7,9,10\},\ \{2,3,4,8\}$$

$$Gini(S) = \frac{6}{10} Gini(D_{verdadero}) + \frac{4}{10} Gini\left(D_{falso}\right) =$$

$$= \frac{6}{10}\left(1 - \left(\frac{5}{6}\right)^2 - \left(\frac{1}{6}\right)^2\right) + \frac{4}{10}\,0 = 0.167$$

$$x_9 = 0 : VERDADERO\ o\ FALSO \qquad \{2,3,4,6,7,8,10\},\ \{1,5,9\}$$

$$Gini(S) = \frac{7}{10} Gini(D_{verdadero}) + \frac{3}{10} Gini\left(D_{falso}\right) =$$

$$= \frac{7}{10}\left(1 - \left(\frac{6}{7}\right)^2 - \left(\frac{1}{7}\right)^2\right) + \frac{3}{10}\,0 = 0.17$$

**Construcción descendente del algoritmo de un árbol de decisión binario**

Una vez comprendido el concepto de criterio de división y sus diferentes posibilidades, es el momento de ejemplificar la construcción de manera descendiente del algoritmo de un árbol de decisión binario completo. Para ello se han de implementar los siguientes pasos:

- En primer lugar, se ha de identificar el nodo raíz, el cual incluye todos los registros del conjunto de datos de entrenamiento, con el fin de seleccionarlo para llevar a cabo el proceso de división.
- Seguidamente, debemos de designar un criterio de división para llevar a cabo la mejor fragmentación posible y particionar el conjunto de datos de entrenamiento en el nodo seleccionado en dos nodos con dos subconjuntos de registros.
- Por último, será necesario verificar si se cumple el criterio de parada, el cual consiste en detenerse cuando cada nodo hoja posea datos homogéneos, dicho de otra manera, cuando contenga un conjunto de registros con el mismo valor objetivo. Es importante señalar que en el mundo real una gran proporción de los conjuntos de datos son muy heterogéneos, lo que añade complejidad a la obtención de conjuntos de datos homogéneos en los nodos hoja. De ahí que el criterio de parada se establezca a menudo con la intención de que la medida de homogeneidad de los datos sea inferior a un valor umbral, por ejemplo, $Entropia(D) < 0.1$. En caso de que se cumpla esta condición, la construcción del árbol habrá llegado a su fin. Ante lo contrario, será preciso volver al segundo paso para seguir seleccionando un nodo a dividir.

A continuación, como ejemplo, se muestra la construcción de un árbol de decisión binario completo para el conjunto de datos de la Figura 2.2. En primer lugar, vamos a utilizar la entropía de la información como medida de la homogeneidad de los datos. Tal y como aparece en la Figura 2.3, el conjunto de datos del nodo raíz se divide en dos subconjuntos, $\{2, 4, 8, 9, 10\}$, y $\{1, 3, 5, 6, 7\}$, el cual ya es homogéneo con el valor objetivo $y = 1$, por lo que no necesita una división. En el caso del subconjunto $D = \{2, 4, 8, 9, 10\}$, la entropía de la información sería:

$$Entropia\,(D) = \sum_{i=1}^{c} - P_i \log_2 P_i = -\frac{1}{5}\log_2 \frac{1}{5} - \frac{4}{5}\log_2 \frac{4}{5} = 0.72$$

A la hora de hacer uso de las variables independientes para dividir $D$, a excepción de $x_7$ que se ha utilizado para particionar el nodo raíz, las otras ocho variables explicativas, $x_1, x_2, x_3, x_4, x_5, x_6, x_8$ y $x_9$, se pueden usar para dividir $D$. Sin embargo, $x_1 = 0$, $x_3 = 0$, $x_5 = 0$ y $x_6 = 0$ no producen una división de $D$, al contrario que $x_2$, $x_4$, $x_8$ y $x_9$. Dado que el criterio de división, $x_8 = 0 : VERDADERO\ o\ FALS\,0$, produce la entropía media más pequeña, se selecciona este criterio de división para dividir $D = \{2, 4, 8, 9, 10\}$ en $\{9, 10\}$ y $\{2, 4, 8\}$, donde este último conjunto ya es homogéneo con el valor objetivo $y = 1$, por lo que no necesita una división, mientras que para el subconjunto, $D = \{9, 10\}$ la entropía es:

$$Entropia\,(D) = \sum_{i=1}^{c} - P_i \log_2 P_i = -\frac{1}{2}\log_2 \frac{1}{2} - \frac{1}{2}\log_2 \frac{1}{2} = 1$$

Ahora ya son dos, $x_7$ y $x_8$, las variables que se han utilizado, por tanto, son siete las variables independientes $x_1$, $x_2$, $x_3$, $x_4$, $x_5$, $x_6$ y $x_9$, de las que se puede hacer uso para dividir $D$. Por un lado, $x_1 = 0\, x_2 = 0$, $x_3 = 0$, $x_4 = 0$, $x_5 = 0$, $x_6 = 0$ no producen una división de $D$. Sin embargo, el criterio de división de $x_9 = 0 : VERDADERO\ o\ FALS\,O$, produce dos subconjuntos, $\{9\}$ con el valor objetivo de $y = 1$, y $\{10\}$ con el valor objetivo de $y = 0$, los cuales son homogéneos y no necesitan una división. De esta forma, dado que todos los nodos del árbol de decisión son homogéneos, la construcción del árbol se detiene, siendo los últimos nodos los nodos hoja, con el árbol de decisión completo que se puede observar en la Figura 2.3. A continuación, se muestra el cálculo de la entropía de la información para las divisiones que utilizan $x_2$, $x_4$, $x_8$ y $x_9$:

$$x_2 = 0 : VERDADERO\ o\ FALS\,O \qquad \{4, 8, 9, 10\},\ \{2\}$$

$$Entropia\,(S) = \frac{4}{5} Entropia\,(D_{verdadero}) + \frac{1}{5} Entropia\,\left(D_{falso}\right) =$$

$$= \frac{4}{5}\left(-\frac{3}{4}\log_2 \frac{3}{4} - \frac{1}{4}\log_2 \frac{1}{4}\right) + \frac{1}{5}\,0 = 0.64$$

$$x_4 = 0 : VERDADERO\ o\ FALS\,O \qquad \{8, 9, 10\},\ \{2, 4\}$$

$$Entropia\,(S) = \frac{3}{5} Entropia\,(D_{verdadero}) + \frac{2}{5} Entropia\,\left(D_{falso}\right) =$$

$$= \frac{3}{5}\left(-\frac{2}{3}\log_2 \frac{2}{3} - \frac{1}{3}\log_2 \frac{1}{3}\right) + \frac{2}{5}\,0 = 0.55$$

$$x_8 = 0 : VERDADERO\ o\ FALS\,O \qquad \{9, 10\},\ \{2, 4, 8\}$$

$$Entropia\,(S) = \frac{2}{5}Entropia\,(D_{verdadero}) + \frac{3}{5}Entropia\,\left(D_{falso}\right) =$$
$$= \frac{2}{5}\left(-\frac{1}{2}\log_2\frac{1}{2} - \frac{1}{2}\log_2\frac{1}{2}\right) + \frac{3}{5}\,0 = 0.4$$

$$x_9 = 0 : VERDADERO\ o\ FALSO \qquad \{2,4,8,10\},\ \{9\}$$
$$Entropia\,(S) = \frac{4}{5}Entropia\,(D_{verdadero}) + \frac{1}{5}Entropia\,\left(D_{falso}\right) =$$
$$= \frac{4}{5}\left(-\frac{3}{4}\log_2\frac{3}{4} - \frac{1}{4}\log_2\frac{1}{4}\right) + \frac{1}{5}\,0 = 0.64$$

En segundo lugar, desarrollaremos otro ejemplo de la construcción de un árbol de decisión binario, pero en este caso haciendo uso del índice de Gini como medida de homogeneidad de los datos. Como se ha descrito anteriormente, el conjunto de datos del nodo raíz se divide en dos subconjuntos, $\{2,4,8,9,10\}$ y $\{1,3,5,6,7\}$, donde este último ya es homogéneo con el valor objetivo $y = 1$, por lo que no precisa de una división. No obstante, a diferencia de como ocurría en el ejemplo anterior el subconjunto $D = \{2,4,8,9,10\}$ sí que es susceptible de particionarse:

$$Gini\,(D) = 1 - \sum_{i=1}^{c} P_i^2 = 1 - \left(\frac{4}{5}\right)^2 - \left(\frac{1}{5}\right)^2 = 0.32$$

Ahora hemos de tener en cuenta que $x_1 = 0,\ x_3 = 0,\ x_5 = 0$ y $x_6 = 0$ no producen una división de $D$, de ello se encargarán el resto de variables restantes. Por otro lado, una vez comprobado que el criterio de división, $x_8 = 0 : VERDADERO\ o\ FALS\,0$, produce el menor valor medio del índice de Gini de la división, este se selecciona para particionar $D = \{2,4,8,9,10\}$ en $\{9,10\}$ y $\{2,4,8\}$, de manera que $\{2,4,8\}$ ya es homogéneo con el valor objetivo $y = 1$, y no necesita ser dividido. En contraposición, el conjunto $D = \{9,10\}$ sí lo requiere, cuyo índice de Gini es:

$$Gini\,(D) = 1 - \sum_{i=1}^{c} P_i^2 = 1 - \left(\frac{1}{2}\right)^2 - \left(\frac{1}{2}\right)^2 = 0.5$$

En este momento, al igual que ocurría en el ejemplo de la entropía de la información, son $x_7$ y $x_8$ las variables que han sido empleadas para dividir el nodo raíz, contrariamente, las otras siete variables independientes, $x_1,\ x_2,\ x_3,\ x_4,\ x_5,\ x_6$ y $x_9$, se prestan a ser utilizadas para la partición de $D$, sin embargo, las 6 primeras, $x_1 = 0,\ x_2 = 0,\ x_3 = 0,\ x_4 = 0,\ x_5 = 0,\ x_6 = 0$ no sirven para la divisón de $D$. De modo que solo nos queda una, $x_9$, cuyo criterio de división $x_9 = 0 : VERDADERO\ o\ FALS\,O$, genera dos subconjuntos, $\{9\}$ con el valor objetivo de $y = 1$, y $\{10\}$ con el valor objetivo de $y = 0$, que son homogéneos y no necesitan ser particionados. Por lo tanto, como todos los nodos del árbol de decisión son homogéneos, la construcción del árbol de decisión llega a su fin, siendo todos los últimos nodos los nodos hoja, con el árbol de decisión completo, que como ya nos habremos percatado, es el mismo que el árbol de decisión generado al utilizar la entropía de la información como medida de homogeneidad de los datos. En los ejemplos a continuación podemos apreciar el cálculo de los valores del índice de Gini para las divisiones que utilizan $x_2,\ x_4,\ x_8$ y $x_9$:

$$x_2 = 0 : VERDADERO\ o\ FALS\,O \qquad \{4,8,9,10\},\ \{2\}$$
$$Gini\,(S) = \frac{4}{5}Gini\,(D_{verdadero}) + \frac{1}{5}Gini\,\left(D_{falso}\right) =$$

$$= \frac{4}{5}\left(1 - \left(\frac{3}{4}\right)^2 - \left(\frac{1}{4}\right)^2\right) + \frac{1}{5}\,0 = 0.3$$

$$x_4 = 0 : VERDADERO\ o\ FALSO \qquad \{8, 9, 10\},\ \{2, 4\}$$
$$Gini\,(S) = \frac{3}{5}Gini\,(D_{verdadero}) + \frac{2}{5}Gini\left(D_{falso}\right) =$$
$$= \frac{3}{5}\left(1 - \left(\frac{3}{4}\right)^2 - \left(\frac{1}{4}\right)^2\right) + \frac{2}{5}\,0 = 0.27$$

$$x_8 = 0 : VERDADERO\ o\ FALSO \qquad \{9, 10\},\ \{2, 4, 8\}$$
$$Gini\,(S) = \frac{2}{5}Gini\,(D_{verdadero}) + \frac{3}{5}Gini\left(D_{falso}\right) =$$
$$= \frac{2}{5}\left(1 - \left(\frac{1}{2}\right)^2 - \left(\frac{1}{2}\right)^2\right) + \frac{3}{5}\,0 = 0.2$$

$$x_9 = 0 : VERDADERO\ o\ FALSO \qquad \{2, 4, 8, 10\},\ \{9\}$$
$$Gini\,(S) = \frac{4}{5}Gini\,(D_{verdadero}) + \frac{1}{5}Gini\left(D_{falso}\right) =$$
$$= \frac{4}{5}\left(1 - \left(\frac{3}{4}\right)^2 - \left(\frac{1}{4}\right)^2\right) + \frac{1}{5}\,0 = 0.3$$

**Clasificación de datos mediante un árbol de decisión**

Un árbol de decisión se utiliza para clasificar un registro determinado propagándolo a un nodo hoja del árbol de decisión, de manera que se haga uso de los valores de las variables independientes al mismo tiempo que se le asigna el valor objetivo del nodo hoja en cuestión. De este modo, haciendo alusión a la instancia 10 de la Figura 2.2, en la Figura 2.5, se expone en color azul la ruta de paso de esta, donde va desde el nodo raíz a un nodo hoja, finalizando la ruta con el valor objetivo $y = 0$. Como resultado, el registro se clasifica como "sin fallo del sistema". Por otra parte, los valores objetivo de los registros del conjunto de datos de testeo de detección de fallos del sistema de la Figura 2.2, se obtienen utilizando el árbol de decisión de la Figura 2.3 y se muestran en la Figura 2.6. En la Figura 2.7 se muestra en azul la ruta que sigue el registro de testeo de la instancia 1 en la Figura 2.2. Es posible apreciar que va desde el nodo raíz, hasta un nodo hoja con el valor objetivo $y = 1$, por lo tanto, el registro en cuestión se clasifica como "fallo del sistema".

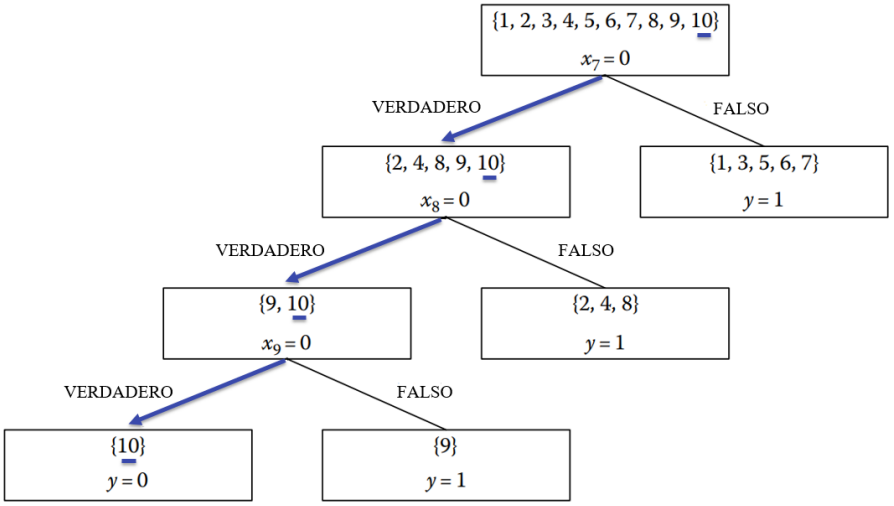



Figura 2.5: Clasificación de un registro sin fallo del sistema mediante el árbol de decisión para la detección de fallos del sistema. Fuente: Propia, basada a partir de: Ye (2013).

| Instancia (Máquina averiada) | Variables independientes categóricas | | | | | | | | | Variable objetivo | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Calidad de las piezas | | | | | | | | | Fallo del sistema | |
| | $X_1$ | $X_2$ | $X_3$ | $X_4$ | $X_5$ | $X_6$ | $X_7$ | $X_8$ | $X_9$ | *Valor real* | *Valor clasificado* |
| 1 (M1, M2) | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 |
| 2 (M2, M3) | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 |
| 3 (M1, M3) | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 4 (M1, M4) | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 |
| 5 (M1, M6) | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 |
| 6 (M2, M6) | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 |
| 7 (M2, M5) | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| 8 (M3, M5) | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 9 (M4, M7) | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 1 |
| 10 (M5, M8) | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| 11 (M5, M6) | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 12 (M3, M9) | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| 13 (M1, M8) | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 |
| 14 (M2,M3,M9) | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| 15 (M2,M3,M5) | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 16 (M1,M2,M3) | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

Figura 2.6: Clasificación de registros en el conjunto de datos de testeo para la detección de fallos del sistema. Fuente: Propia, basada a partir de: Ye (2013).

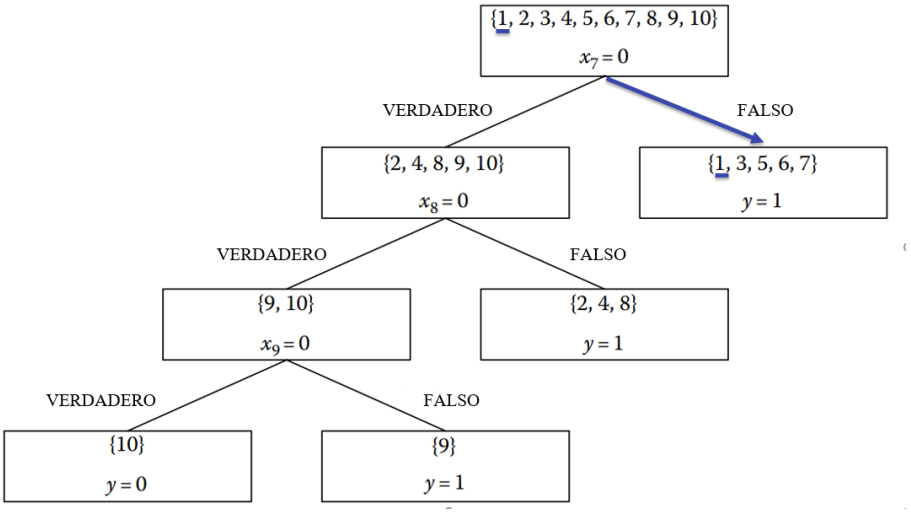



Figura 2.7: Clasificación de un registro para múltiples fallos de la máquina utilizando el árbol de decisión para la detección de fallos del sistema. Fuente: Propia, basada a partir de: Ye (2013).

### 2.2.2. El árbol de decisión no binario

En primer lugar, es relevante clarificar que denominamos no binarios a aquellos árboles de decisión cuya variable objetivo presenta más de dos categorías, por lo que la complejidad del árbol aumenta en relación con el binario, ya que estos solo poseen dos categorías en la variable dependiente. En el conjunto de datos de lentes de la Figura 2.8, la variable de atributo Edad, tiene tres valores categóricos: Joven, Pre-presbicia y Presbicia[1]. En caso de querer construir un árbol de decisión binario para este conjunto de datos, sería necesario convertir los tres valores categóricos de la variable Edad en dos valores categóricos si Edad se utiliza para dividir el nodo raíz. De este modo, podríamos tener Joven y Pre-presbicia juntos como una categoría, mientras que Presbicia comformaría la segunda categoría, por tanto contaríamos con Edad = Presbicia: VERDADERO o FALSO como criterio de división. Otra opción sería que Joven comformase una categoría, mientras que Pre-presbicia y Presbicia juntas formasen la otra categoría, teniendo Edad = Joven: VERDADERO O FALSO como criterio de división. Sin embargo, esto no será necesario si contruimos un árbol de decisión no binario, el cual permitirá dividir el conjunto de datos de un nodo en más de dos subconjuntos haciendo uso de cada uno de los múltiples valores categóricos para cada rama de la división. A continuación, mostraremos la construcción de un árbol de decisión no binario para el conjunto de datos de lentes de la Figura 2.8.

A la hora de construir el árbol, si se hace uso de la variable independiente Edad para dividir el nodo raíz del conjunto de datos de lentes, es posible utilizar los tres valores categóricos de Edad para particionar el conjunto de 24 registros en el nodo raíz empleando el criterio de división Edad = Joven, Pre-presbicia o Presbicia, como se presenta en la Figura 2.9. Como conjunto de datos de entrenamiento se hará uso del conjunto de 24 registros de la Tabla 2.8, *D*, en el nodo raíz del árbol de decisión no binario. En esta ocasión, la variable objetivo tiene tres valores categóricos, Sin contacto en 15 registros, Lentillas suaves en 5 registros y Lentillas duras en 4 registros. Asimismo, designando la entropía de la información como medida de homogeneidad de los datos, tenemos:

[1] Afección en la que el cristalino del ojo pierde su capacidad para enfocar, lo que dificulta al hecho de ver objetos cercanos.

$$Entropia\,(D) = \sum_{i=1}^{3} - P_i \log_2 P_i = -\frac{15}{24}\log_2\frac{15}{24} - \frac{5}{24}\log_2\frac{5}{24} - \frac{4}{24}\log_2\frac{4}{24} = 1.3261$$

Seguidamente, se muestra el cálculo de la entropía de la información para dividir el nodo raíz utilizando el criterio de división, Tasa de Producción de Lágrimas = Reducida o Normal, el cual genera un subconjunto homogéneo de $\{1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23\}$ y un subconjunto no homogéneo de $\{2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24\}$. Luego se expone otro cálculo de la entropía de la información para dividir el nodo con el conjunto de datos de $\{2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24\}$ utilizando el criterio de división, Astigmático = No o Sí, que da lugar a dos subconjuntos de $\{2, 6, 10, 14, 18, 22\}$ y $\{4, 8, 12, 16, 20, 24\}$. Después, el criterio de división, Edad = Joven, Pre-presbicia o Presbicia se utiliza para dividir el nodo con el conjunto de datos de $\{2, 6, 10, 14, 18, 22\}$, que produce a su vez otros tres subconjuntos: $\{2, 6\}, \{10, 14\}$ y $\{18, 22\}$. Estos subconjuntos se vuelven a dividir otra vez utilizando el criterio de división Prescripción de gafas = Miope o Hipermétrope, para generar nodos de hoja con conjuntos de datos homogéneos. Por otro lado, también se calcula la entropía de la información para dividir el nodo con el conjunto de datos de $\{4, 8, 12, 16, 20, 24\}$ haciendo uso del criterio de división, Prescripción de gafas = Miope o Hipermétrope, que en este caso da lugar a dos subconjuntos: $\{4, 12, 20\}$ y $\{8, 16, 24\}$. Estos subconjuntos se particionan a su vez con el criterio de división Edad = Joven, Pre-presbicia o Presbicia, para finalmente crear nodos hoja con conjuntos de datos homogéneos dando por terminada la construcción del árbol. Finalmente, en la Figura 2.9, podemos echar un vistazo al árbol de decisión no binario completo para el conjunto de datos de las lentes.

| Instancia | Variables independientes categóricas | | | | Variable objetivo |
|---|---|---|---|---|---|
| | *Edad* | *Graduación* | *Astigmatismo* | *Tasa de producción de lágrimas* | *Lentes* |
| 1 | Joven | Miope | No | Reducida | Sin contacto |
| 2 | Joven | Miope | No | Normal | Lentillas blandas |
| 3 | Joven | Miope | Sí | Reducida | Sin contacto |
| 4 | Joven | Miope | Sí | Normal | Lentillas duras |
| 5 | Joven | Hipermétrope | No | Reducida | Sin contacto |
| 6 | Joven | Hipermétrope | No | Normal | Lentillas blandas |
| 7 | Joven | Hipermétrope | Sí | Reducida | Sin contacto |
| 8 | Joven | Hipermétrope | Sí | Normal | Lentillas duras |
| 9 | Pre-presbicia | Miope | No | Reducida | Sin contacto |
| 10 | Pre-presbicia | Miope | No | Normal | Lentillas blandas |
| 11 | Pre-presbicia | Miope | Sí | Reducida | Sin contacto |
| 12 | Pre-presbicia | Miope | Sí | Normal | Lentillas duras |
| 13 | Pre-presbicia | Hipermétrope | No | Reducida | Sin contacto |
| 14 | Pre-presbicia | Hipermétrope | No | Normal | Lentillas blandas |
| 15 | Pre-presbicia | Hipermétrope | Sí | Reducida | Sin contacto |
| 16 | Pre-presbicia | Hipermétrope | Sí | Normal | Sin contacto |
| 17 | Presbicia | Miope | No | Reducida | Sin contacto |
| 18 | Presbicia | Miope | No | Normal | Sin contacto |
| 19 | Presbicia | Miope | Sí | Reducida | Sin contacto |
| 20 | Presbicia | Miope | Sí | Normal | Lentillas duras |
| 21 | Presbicia | Hipermétrope | No | Reducida | Sin contacto |
| 22 | Presbicia | Hipermétrope | No | Normal | Lentillas blandas |
| 23 | Presbicia | Hipermétrope | Sí | Reducida | Sin contacto |
| 24 | Presbicia | Hipermétrope | Sí | Normal | Sin contacto |

Figura 2.8: Conjunto de datos de lentes. Fuente: Propia, basada a partir de: Ye (2013).

- Nodo raíz:

$$\begin{aligned} \text{Edad} = \quad & \text{Joven}, && \{1,2,3,4,5,6,7,8\}, \\ & \text{Pre-presbicia}, && \{9,10,11,12,13,14,15,16\}, \\ & \text{o Presbicia} && \{17,18,19,20,21,22,23,24\} \end{aligned}$$

$$Entropia\,(S) = \frac{8}{24} Entropia\,(D_{Joven}) + \frac{8}{24} Entropia\,\left(D_{Pre-presbicia}\right) +$$

$$+ \frac{8}{24} Entropia\,(D_{Presbicia}) =$$

$$= \frac{8}{24}\left(-\frac{4}{8}\log_2\frac{4}{8} - \frac{2}{8}\log_2\frac{2}{8} - \frac{2}{8}\log_2\frac{2}{8}\right) +$$

$$+ \frac{8}{24}\left(-\frac{5}{8}\log_2\frac{5}{8} - \frac{2}{8}\log_2\frac{2}{8} - \frac{1}{8}\log_2\frac{1}{8}\right) +$$

$$+ \frac{8}{24}\left(-\frac{6}{8}\log_2\frac{6}{8} - \frac{1}{8}\log_2\frac{1}{8} - \frac{1}{8}\log_2\frac{1}{8}\right) = 1.2867$$

$$\begin{aligned} \text{Graduación} = \quad & \text{Miope}, && \{1,2,3,4,9,10,11,12,17,18,19,20\}, \\ & \text{o Hipermétrope} && \{5,6,7,8,13,14,15,16,21,22,23,24\} \end{aligned}$$

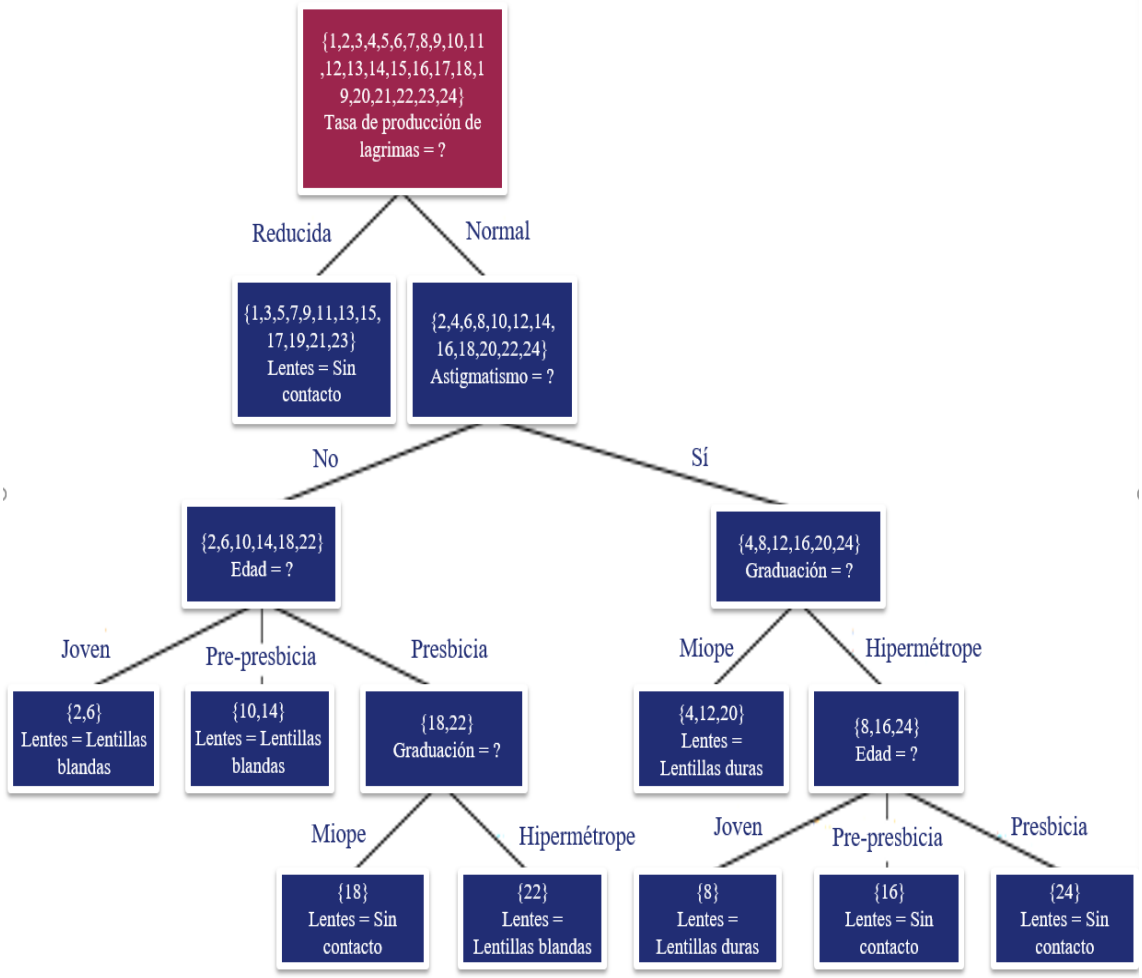



Figura 2.9: Árbol de decisión no binario para el conjunto de datos de lentes. Fuente: Propia, basada a partir de: Ye (2013).

$$Entropia\,(S) = \frac{12}{24} Entropia\,(D_{Miope}) + \frac{12}{24} Entropia\,(D_{Hipermetrope}) =$$
$$= \frac{12}{24}\left(-\frac{7}{12}\log_2\frac{7}{12} - \frac{2}{12}\log_2\frac{2}{12} - \frac{3}{12}\log_2\frac{3}{12}\right) +$$
$$+\frac{12}{24}\left(-\frac{8}{12}\log_2\frac{8}{12} - \frac{3}{12}\log_2\frac{3}{12} - \frac{1}{12}\log_2\frac{1}{12}\right) = 1.2866$$

$$\text{Astigmatismo} = \begin{array}{ll} \text{No}, & \{1,2,5,6,9,10,13,14,17,18,21,22\}, \\ \text{o Sí} & \{3,4,7,8,11,12,15,16,19,20,23,24\} \end{array}$$

$$Entropia\,(S) = \frac{12}{24} Entropia\,(D_{No}) + \frac{12}{24} Entropia\,(D_{Si}) =$$
$$= \frac{12}{24}\left(-\frac{7}{12}\log_2\frac{7}{12} - \frac{5}{12}\log_2\frac{5}{12} - \frac{0}{12}\log_2\frac{0}{12}\right) +$$
$$+\frac{12}{24}\left(-\frac{8}{12}\log_2\frac{8}{12} - \frac{4}{12}\log_2\frac{4}{12} - \frac{0}{12}\log_2\frac{0}{12}\right) = 0.9491$$

$$\begin{array}{rrl} \text{Tasa de producción de lágrimas} = & \text{Reducida}, & \{1,2,5,6,9,10,13,14,17,18,21,22\}, \\ & \text{o Normal} & \{5,6,7,8,13,14,15,16,21,22,23,24\} \end{array}$$

$$Entropia\,(S) = \frac{12}{24} Entropia\,(D_{Reducida}) + \frac{12}{24} Entropia\,(D_{Normal}) =$$

$$= \frac{12}{24}\left(-\frac{12}{12}\log_2\frac{12}{12} - \frac{0}{12}\log_2\frac{0}{12} - \frac{0}{12}\log_2\frac{0}{12}\right) +$$

$$+\frac{12}{24}\left(-\frac{3}{12}\log_2\frac{3}{12} - \frac{5}{12}\log_2\frac{5}{12} - \frac{4}{12}\log_2\frac{4}{12}\right) = 0.7773$$

- Nodo hoja $\{2,4,6,8,10,12,14,16,18,20,22,24\}$:

$$\begin{array}{rrl} \text{Edad} = & \text{Joven}, & \{2,4,6,8\}, \\ & \text{Pre-presbicia}, & \{10,12,14,16\}, \\ & \text{o Presbicia} & \{18,20,22,24\} \end{array}$$

$$Entropia\,(S) = \frac{4}{12} Entropia\,(D_{Joven}) + \frac{4}{12} Entropia\left(D_{Pre-presbicia}\right) +$$

$$+\frac{4}{12} Entropia\,(D_{Presbicia}) =$$

$$= \frac{4}{12}\left(-\frac{0}{4}\log_2\frac{0}{4} - \frac{2}{4}\log_2\frac{2}{4} - \frac{2}{4}\log_2\frac{2}{4}\right) +$$

$$+\frac{4}{12}\left(-\frac{1}{4}\log_2\frac{1}{4} - \frac{2}{4}\log_2\frac{2}{4} - \frac{1}{4}\log_2\frac{1}{4}\right) +$$

$$+\frac{4}{12}\left(-\frac{2}{4}\log_2\frac{2}{4} - \frac{1}{4}\log_2\frac{1}{4} - \frac{1}{4}\log_2\frac{1}{4}\right) = 1.3333$$

$$\begin{array}{rrl} \text{Graduación} = & \text{Miope}, & \{2,4,10,12,18,20\}, \\ & \text{o Hipermétrope} & \{6,7,14,16,22,24\} \end{array}$$

$$Entropia\,(S) = \frac{6}{12} Entropia\left(D_{Miope}\right) + \frac{6}{12} Entropia\left(D_{Hipermetrope}\right) =$$

$$= \frac{6}{12}\left(-\frac{1}{6}\log_2\frac{1}{6} - \frac{2}{6}\log_2\frac{2}{6} - \frac{3}{6}\log_2\frac{3}{6}\right) +$$

$$+\frac{6}{12}\left(-\frac{2}{6}\log_2\frac{2}{6} - \frac{3}{6}\log_2\frac{3}{6} - \frac{1}{6}\log_2\frac{1}{6}\right) = 1.4591$$

$$\begin{array}{rrl} \text{Astigmatismo} = & \text{No}, & \{2,6,10,14,18,22\}, \\ & \text{o Sí} & \{4,8,12,16,20,24\} \end{array}$$

$$Entropia\,(S) = \frac{6}{12} Entropia\,(D_{No}) + \frac{6}{12} Entropia\,(D_{Si}) =$$

$$= \frac{6}{12}\left(-\frac{1}{6}\log_2\frac{1}{6} - \frac{5}{6}\log_2\frac{5}{6} - \frac{0}{6}\log_2\frac{0}{6}\right) +$$

$$+\frac{6}{12}\left(-\frac{2}{6}\log_2\frac{2}{6}-\frac{0}{6}\log_2\frac{0}{6}-\frac{4}{6}\log_2\frac{4}{6}\right)=0.7842$$

- Nodo hoja $\{2,6,10,14,18,22\}$:

$$\begin{array}{rcc}\text{Edad} = & \text{Joven}, & \{2,6\},\\ & \text{Pre-presbicia}, & \{10,14\},\\ & \text{o Presbicia} & \{18,22\}\end{array}$$

$$Entropia(S)=\frac{2}{6}Entropia(D_{Joven})+\frac{2}{6}Entropia\left(D_{Pre-presbicia}\right)+$$
$$+\frac{2}{6}Entropia(D_{Presbicia})=$$
$$=\frac{2}{6}\left(-\frac{0}{2}\log_2\frac{0}{2}-\frac{2}{2}\log_2\frac{2}{2}-\frac{0}{2}\log_2\frac{0}{2}\right)+$$
$$+\frac{2}{6}\left(-\frac{0}{2}\log_2\frac{0}{2}-\frac{2}{2}\log_2\frac{2}{2}-\frac{0}{2}\log_2\frac{0}{2}\right)+$$
$$+\frac{2}{6}\left(-\frac{1}{2}\log_2\frac{1}{2}-\frac{1}{2}\log_2\frac{1}{2}-\frac{0}{2}\log_2\frac{0}{2}\right)=0.3333$$

$$\begin{array}{rcc}\text{Graduación} = & \text{Miope}, & \{2,10,18\},\\ & \text{o Hipermétrope} & \{6,14,22\}\end{array}$$

$$Entropia(S)=\frac{3}{6}Entropia\left(D_{Miope}\right)+\frac{3}{6}Entropia\left(D_{Hipermetrope}\right)=$$
$$=\frac{3}{6}\left(-\frac{1}{3}\log_2\frac{1}{3}-\frac{2}{3}\log_2\frac{2}{3}-\frac{0}{3}\log_2\frac{0}{3}\right)+$$
$$+\frac{3}{6}\left(-\frac{0}{3}\log_2\frac{0}{3}-\frac{3}{3}\log_2\frac{3}{3}-\frac{0}{3}\log_2\frac{0}{3}\right)=1.4591$$

- Nodo hoja $\{4,8,12,16,20,24\}$:

$$\begin{array}{rcc}\text{Edad} = & \text{Joven}, & \{4,8\},\\ & \text{Pre-presbicia}, & \{12,16\},\\ & \text{o Presbicia} & \{20,24\}\end{array}$$

$$Entropia(S)=\frac{2}{6}Entropia(D_{Joven})+\frac{2}{6}Entropia\left(D_{Pre-presbicia}\right)+$$
$$+\frac{2}{6}Entropia(D_{Presbicia})=$$
$$=\frac{2}{6}\left(-\frac{0}{2}\log_2\frac{0}{2}-\frac{0}{2}\log_2\frac{0}{2}-\frac{2}{2}\log_2\frac{2}{2}\right)+$$
$$+\frac{2}{6}\left(-\frac{1}{2}\log_2\frac{1}{2}-\frac{0}{2}\log_2\frac{0}{2}-\frac{1}{2}\log_2\frac{1}{2}\right)+$$

$$+\frac{2}{6}\left(-\frac{1}{2}\log_2\frac{1}{2}-\frac{0}{2}\log_2\frac{0}{2}-\frac{1}{2}\log_2\frac{1}{2}\right)=0.6667$$

$$\text{Graduación} = \begin{array}{ll} \text{Miope,} & \{4,12,20\},\\ \text{o Hipermétrope} & \{8,16,24\}\end{array}$$

$$Entropia\,(S)=\frac{3}{6}Entropia\left(D_{Miope}\right)+\frac{3}{6}Entropia\left(D_{Hipermetrope}\right)=$$
$$=\frac{3}{6}\left(-\frac{0}{3}\log_2\frac{0}{3}-\frac{0}{3}\log_2\frac{0}{3}-\frac{3}{3}\log_2\frac{3}{3}\right)+$$
$$+\frac{3}{6}\left(-\frac{2}{3}\log_2\frac{2}{3}-\frac{0}{3}\log_2\frac{0}{3}-\frac{1}{3}\log_2\frac{1}{3}\right)=0.4591$$

### 2.2.3. Tratamiento de variables independientes numéricas

Las variables de atributo numéricas suponen un escollo a la hora de confeccionar un árbol de decisión, hasta tal punto que son incompatibles con esto, por ello si estamos ante un conjunto de datos que cuenta con una o más de estas variables, antes de que hagamos uso de ellas en la construcción del árbol, nos veremos obligados a transformarlas en variables categóricas. Para efectuar esta transformación haremos uso del siguiente método: Vamos a suponer que una variable independiente numérica, $x$, posee los siguientes valores numéricos en el conjunto de datos de entrenamiento, $a_1, a_2, \ldots, a_k$, los cuales están ordenados de manera creciente, de este modo el punto medio de dos valores numéricos, $a_i$ y $a_j$, lo calcularemos así:

$$c_i = \frac{a_i + a_j}{2} \tag{2.7}$$

De esta manera, con la anterior fórmula podemos crear $k-1$ puntos de corte $c_i$ donde $i = 1, \ldots, k-1$, y consecuentemente $k+1$ categorías de $x$:

$$\begin{array}{ll} \text{Categoría 1:} & x \leq c_1 \\ \text{Categoría 2:} & c_1 < x \leq c_2 \end{array}$$

$$\ldots$$

$$\begin{array}{ll} \text{Categoría } k\text{:} & c_{k-1} < x \leq c_k \\ \text{Categoría } k+1\text{:} & c_k < x \end{array}$$

En consecuencia, con este procedimiento un valor numérico de $x$ se transforma en un valor categórico, de modo que pueda ser utilizado en la elaboración del árbol de decisión.

### 2.2.4. Tratamiento de valores faltantes

En conjuntos de datos provenientes del mundo real, es muy probable dar con registros donde falten valores para algunas determinadas variables independientes del conjunto. Por ejemplo, si hay variables de atributo de nombre, dirección y número de teléfono de clientes en una base de datos de una tienda, es posible que no tengamos el número de teléfono de un cliente concreto, es decir, pueden faltar los números de teléfono de algunos clientes. Por ello, tenemos que buscar una solución para tratar esta clase de registros. Una de ellas sería descartar el registro que posea valores omitidos, no obstante, cuando el conjunto de datos es pequeño, es posible que necesitemos todos

los registros del conjunto para construir un árbol de decisión. Por este motivo, para hacer uso de un registro con un valor que falta, cabe la posibilidad de estimar el valor faltante y utilizar este nuevo valor estimado como sustituto. De esta manera, para una variable de atributo categórica, el valor que falta puede estimarse como el valor que toman la mayoría de los registros del conjunto de datos que tienen el mismo valor objetivo que el registro con el valor faltante. Sin embargo, en el caso de una variable de atributo numérica, puede estimarse el valor omitido como la media de los valores procedentes de los registros que tienen el mismo valor objetivo que el registro con el valor faltante, es decir, se sigue un procedimiento muy similar en ambos casos. Cabe destacar que estas son solo algunas de las muchas maneras de tratar valores faltantes.

### 2.2.5. Tratamiento de una variable objetivo numérica

En nuestra base de datos la variable objetivo es categórica, por lo que el resultado de nuestro algoritmo será un árbol de decisión. Sin embargo, es importante mencionar que la variable dependiente también puede ser numérica, dando lugar en este caso a un árbol de regresión, por consiguiente el tratamiento para esta última será diferente.

Cuando trabajamos con una variable objetivo numérica no se pueden aplicar medidas de homogeneidad de los datos como la entropía de la información o el índice de Gini. En estos casos, para medir el grado de homogeneidad de los datos se utiliza la fórmula 2.8 con el objetivo de calcular la diferencia media cuadrática de los valores de un conjunto de datos con respecto a su valor medio, a lo que se denomina $R$. De este modo, cuanto menor es el valor $R$, más homogéneo es el conjunto de datos.

$$R(D) = \sum_{y \in D} (y - \bar{y})^2 \tag{2.8}$$

$$\text{donde } \bar{y} = \frac{\sum_{y \in D} y}{n}$$

A continuación, la siguiente fórmula 2.9 muestra el cálculo de la media cuadrática $R$ después de una división:

$$R(S) = \sum_{v \in Valores(S)} \frac{|D_v|}{|D|} R(D_v) \tag{2.9}$$

Por ejemplo, de un conjunto cualquiera de datos $D$ con 10 registros, una variable objetivo numérica y tres variables independientes numéricas, el valor $R$ de $D$ con los 10 registros en el nodo raíz del árbol de regresión se calcularía como:

$$\bar{y} = \frac{\sum_{y \in D} y}{n} = \frac{1 + 1 + 2.5 + 3 + 2 + 1.5 + 3 + 1.5 + 2 + 2.5}{10} = 2$$

$$R(D) = \sum_{y \in D} (y - \bar{y})^2 = (1-2)^2 + (1-2)^2 + (2.5-2)^2 + (3-2)^2 + (2-2)^2 +$$
$$+ (1.5-2)^2 + (3-2)^2 + (1.5-2)^2 + (2-2)^2 + (2.5-2)^2 = 4.8$$

La media de los valores objetivo de los registros en un nodo hoja suele tomarse como valor objetivo del nodo hoja. Considerando esto, cuando un registro recorre el árbol de decisión, para determinar el valor objetivo del registro, el valor objetivo del nodo hoja al que llega este registro se asigna a este último. Asimismo, es importante recalcar que el árbol de decisión, en esta ocasión, para una variable objetivo numérica, recibe el nombre de árbol de regresión.

## 2.3. Evaluación del clasificador

Una vez construido un clasificador, como en nuestro caso el árbol de decisión, para saber si en mayor o menor medida es fiable, necesitamos evaluar su precisión. Una evaluación exitosa es trascendental ya que no podemos utilizar un clasificador en el mundo real si no conocemos de antemano su precisión, puesto que de lo contrario podríamos llegar a obtener resultados erróneos. Existen numerosas formas de evaluar un clasificador, así como también muchas medidas. La medida más importante es la precisión de la clasificación, que es el número de instancias clasificadas correctamente en el conjunto de testeo dividido por el número total de instancias existentes en dicho conjunto. Otra medida también utilizada es la tasa de error, que es uno menos la precisión. Evidentemente, cuando poseemos más de un clasificador, se prefiere aquel con mayor precisión. Además, es posible hacer uso de pruebas de significación estadística para corroborar si la precisión de un clasificador es significativamente mejor que la de otro dados los mismos conjuntos de datos de entrenamiento y de testeo. A continuación, se exponen algunos de los métodos más utilizados para la evaluación de clasificadores (Liu, 2011).

### 2.3.1. Métodos de evaluación del clasificador

- **Método de retención o Holdout**: El conjunto de datos a disposición $D$ se particiona en dos subconjuntos disjuntos, es decir que no tienen ningún elemento en común, o lo que es lo mismo, que su intersección es vacía. Después de la división, por un lado tenemos el conjunto de entrenamiento $D_{Entrenamiento}$, y por otro lado el conjunto de testeo $D_{Testeo}$. Este último también recibe el nombre de conjunto de retención o de prueba. Este método se utiliza principalmente cuando el conjunto de datos a disposición $D$ es grande. De manera evidente, ambos conjuntos se utilizan con fines distintos, ya que hacemos uso del conjunto de entrenamiento, como su propio nombre dice, para entrenar al clasificador, en otras palabras, enseñarlo a clasificar correctamente, mientras que el conjunto de testeo se emplea para evaluar al anterior. Es importante no alterar este procedimiento, puesto que en caso de que el conjunto de entrenamiento se utilice en la evaluación, el clasificador podría sobreajustarse a los datos de entrenamiento, ya que dicho clasificador está sesgado hacia el conjunto de entrenamiento, lo que se traduce en que tendrá una precisión muy alta para este conjunto pero baja para el conjunto de testeo. En cambio, si hacemos uso del conjunto de testeo para la evaluación, este proporcionaría una estimación no sesgada para la precisión de la clasificación. En cuanto al porcentaje de datos que debe utilizarse para cada conjunto, este depende del tamaño de nuestro conjunto de datos $D$. No obstante, se suelen utilizar porcentajes del 50 % de los datos para cada uno, o bien dos tercios para el entrenamiento y un tercio para el de testeo. A la hora de dividir $D$ en estos dos conjuntos, podemos utilizar diferentes enfoques. Por un lado cabe la posibilidad de seleccionar aleatoriamente un conjunto de ejemplos de entrenamiento de $D$ para el aprendizaje, dejando el resto para ser utilizado como conjunto de testeo. Sin embargo, por otro lado, si los datos se recogen a lo largo del tiempo, podemos utilizar la primera parte de los datos como conjunto de entrenamiento y la última parte de estos como conjunto de testeo. En numerosos casos, este último es un enfoque más adecuado ya que cuando el clasificador se utiliza en el mundo real, este

enfoque refleja mejor los aspectos dinámicos y cambiantes. Este ha sido el método utilizado en nuestro trabajo, sin embargo, como queríamos utilizar la misma base de datos también para predecir, hemos decidido dividir nuestra base de datos original en tres partes iguales, una para entrenamiento, otra para testeo y otra para predicción.

- **Muestreo aleatorio múltiple**: En los casos en que el conjunto de datos que tenemos a disposición $D$ es lo suficientemente pequeño como para que el conjunto de testeo no sea representativo, hacer uso del método anterior puede ser poco fiable. Un método para resolver este problema consiste en realizar el muestreo aleatorio anterior $n$ veces. De modo que cada vez que lo realizamos, se produce un conjunto de entrenamiento diferente, así como uno de testeo distinto también, de esta manera se obtienen $n$ precisiones, y la precisión final estimada de los datos será la media de las $n$ precisiones.
- **Validación cruzada**: Otra opción válida cuando el conjunto de datos es pequeño, es utilizar el método de validación cruzada, o $n-$fold. En este método, el conjunto de datos disponible se divide en $n$ subconjuntos disjuntos del mismo tamaño para, a continuación, utilizar cada subconjunto como conjunto de testeo, de manera que los $n-1$ subconjuntos restantes se combinan como conjunto de entrenamiento para enseñar al clasificador. Este procedimiento se lleva a cabo $n$ veces, por lo que se obtienen $n$ precisiones, de este modo la precisión final estimada es la media de las $n$ precisiones obtenidas. Lo más común es que $n$ tenga el valor de 5 o 10.

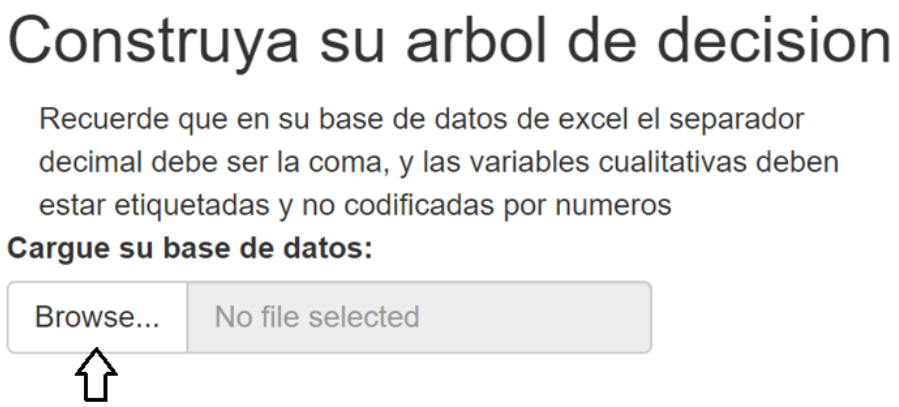
# Capítulo 3

# Software

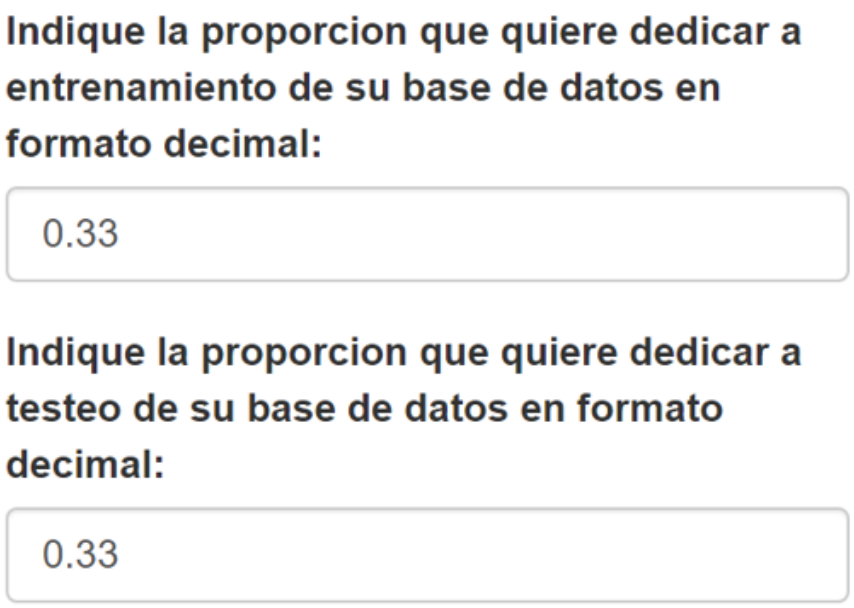
En este capítulo, para usuarios sin conocimientos de programación, mediante un manual, se detallarán los pasos a seguir para un correcto uso de la aplicación construida. Así como también para usuarios con mayores conocimientos en la materia, se describirá el código elaborado para crear nuestro árbol de decisión, y su correspondiente aplicación para el usuario. Es importante destacar que el lenguaje de programación utilizado ha sido únicamente R.

## 3.1. Manual de la interfaz gráfica

En primer lugar, pulse el botón "Browse" como se muestra en la imagen para examinar sus archivos, posteriormente seleccione la base de datos pertinente:



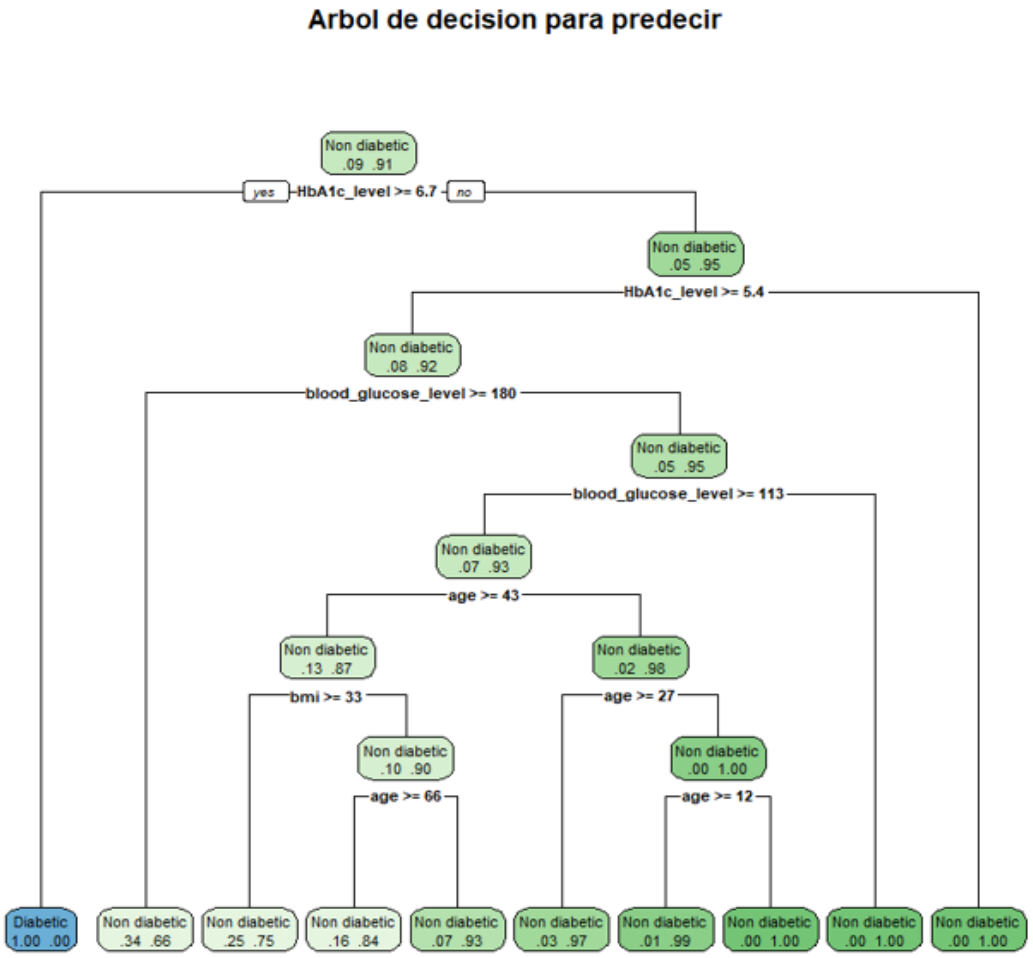

A continuación, rellene los siguientes campos con la información requerida. De esta manera, podrá particionar su base de datos dividiéndola en conjunto de entrenamiento, conjunto de testeo y conjunto de predicción. La proporción dedicada a este último del total de la base de datos original se calcula sola al indicar las dos anteriores. Asegúrese de que introduce un número de tipo decimal como en el ejemplo de la imagen, y nunca un porcentaje:




Se producirá entonces una salida que indicará, en primer lugar, la proporción dedicada a predicción de la base de datos original de la siguiente forma:

```
on dedicada a prediccion de su base de datos sera:  0.34"
```

Seguidamente podremos observar la representación gráfica de nuestro árbol de decisión. Acompañando al árbol, en la parte inferior, también podremos observar la bondad de ajuste del modelo:




```
ritmo clasifica correctamente el 95.32% de los registros"
```

Y para finalizar, la aplicación imprime la tabla de la base de datos con los valores de la variable respuesta predichos, los cuales podemos observar en el rectángulo rojo de la siguiente imagen:

| gender | age | hypertension | heart_disease | smoking_history | bmi | HbA1c_level | blood_glucose_level | respuesta |
|---|---|---|---|---|---|---|---|---|
| Female | 36.00 | No | No | current | 23.45 | 5.00 | 155.00 | Non diabetic |
| Male | 76.00 | Yes | Yes | current | 20.14 | 4.80 | 155.00 | Non diabetic |
| Female | 20.00 | No | No | never | 27.32 | 6.60 | 85.00 | Non diabetic |
| Female | 67.00 | No | No | never | 25.69 | 5.80 | 200.00 | Non diabetic |
| Female | 4.00 | No | No | No Info | 13.99 | 4.00 | 140.00 | Non diabetic |
| Male | 67.00 | No | Yes | not current | 27.32 | 6.50 | 200.00 | Non diabetic |
| Male | 43.00 | No | No | never | 26.08 | 6.10 | 155.00 | Non diabetic |
| Female | 53.00 | No | No | No Info | 31.75 | 4.00 | 200.00 | Non diabetic |

## 3.2. Descripción del código del algoritmo del árbol de decisión

Para llevar a cabo el proceso de desarrollo, representación y evaluación del algoritmo de nuestro árbol de decisión, hemos trabajado con RStudio realizando los pasos subsiguientes:

En primera instancia, haciendo uso de la función “library()”, hemos cargado las librerías “readxl” que posibilita la lectura de bases de datos provenientes de Microsoft Excel, “caret” una librería de clasificación y regresión de la que nos hemos servido para particionar nuestra base de datos, “rpart” que nos ha permitido crear nuestro árbol de decisión conforme a nuestras preferencias, y “rpart.plot” que da la opción de graficar nuestro árbol de una manera visual para el usuario.

A continuación, se procede a leer la base de datos con la función “read_excel()”, convirtiendo el objeto en “Data Frame” para facilitar los cálculos mediante la función “data.frame()” y guardando este en la variable “datos”, donde se almacena el objeto como “Data Frame”. Una vez leído el conjunto de datos, examinamos la estructura interna del objeto a través del “Global Environment” de RStudio y percibimos que la naturaleza de nuestras variables es la correcta. Gracias a la conversión de nuestro objeto en “Data Frame” las variables categóricas y numéricas son interpretadas como tal, de manera que se garantice tanto su buen procesamiento como su buena presentación. En nuestro siguiente paso nos ayudamos de la función “colnames()”, que nos devuelve los nombres de las columnas, es decir de nuestras variables, así como de “ncol()” que cuenta el número de columnas. Utilizando esta última como índice, cambiamos el nombre de nuestra variable dependiente “diabetes” a “respuesta”, ya que esto nos permitirá poder llamar a esta variable en el resto del código para cualquier base de datos diferente que el usuario decida analizar. Seguidamente, procedemos a dividir nuestra base de datos en tres partes iguales, una parte de entrenamiento, una de testeo y otra para realizar la predicción. Antes de empezar, fijamos una semilla con la función “set.seed()” para que las particiones sean siempre las mismas y no obtener árboles diferentes cada vez que ejecutemos el código. Para dividir la base de datos utilizamos la función “createDataPar-

tition()", como parámetro de división nos serviremos de la columna "respuesta" con el comando "datos$respuesta" y para, por ejemplo, conseguir una partición del 33 % para la base de datos de entrenamiento, damos el valor 0.33 al parámetro "p". De aquí obtenemos una serie de índices que guardamos en la variable "registros_entrenamiento" y que posteriormente usamos para crear la base de datos de entrenamiento llamada "datos_entrenamiento", a partir de aquí creamos una variable temporal "temp" que será la partición del 66 % de la base original con el resto de índices no usados para crear la de entrenamiento. Para crear la base de datos de testeo volvemos al primer paso, con la singularidad de que en esta ocasión utilizaremos "temp$respuesta" y le otorgaremos el valor 0.5 a "p", de manera que divida la base de datos "temp" en dos partes iguales, que serán la base de datos de testeo y la de predicción, de modo que cada una contenga aproximadamente el 33 % de los registros de la original. No obstante, necesitamos que la base de datos de predicción no contenga la última columna "respuesta", ya que es la que debe predecir nuestro árbol a partir de dicha base. Para ello, una vez más con la función "ncol()" calculamos el número de columnas que tiene la base de datos, ya que en este caso se conoce que son 9, sin embargo en otros trabajos es posible que no se puedan contar las variables fácilmente. De esta forma, como índice, hacemos uso de la variable donde hemos guardado el número de columnas "n_columnas", y eliminamos la última columna de la base de datos de predicción, que es la que contenía la variable objetivo.

Con las tres bases de datos elaboradas nos disponemos a crear las funciones de entrenamiento del modelo, testeo y predicción:

- Entrenamiento: Para esta función solo introducimos un parámetro que será la base de datos de entrenamiento. Dentro de esta empezamos calculando el número de filas de la base de datos de entrenamiento con la función "nrow()" que guardaremos en la variable "n_filas". Después creamos el árbol de decisión a partir de la función "rpart" donde indicamos que nuestra variable dependiente será "respuesta" y el resto de variables serán las independientes. Introducimos también como parámetros de la función nuestra base de datos de entrenamiento, el método, el cual señalamos que es "class", que hace alusión a la clasificación, ya que nuestro árbol será de decisión y no de regresión. "cp", que es el parámetro de complejidad, se utiliza como medida de tolerancia, de manera que si en algún punto del algoritmo el cambio de un paso al siguiente sigue estando por encima de esa tolerancia, el algoritmo no se detiene y sigue haciendo divisiones, el -1 en este caso se interpreta como una tolerancia de 0 y seguirá haciendo divisiones de manera indefinida, no se utiliza exactamente 0, sino un número por debajo (como -1) porque 0 en un entorno de programación es en realidad la precisión de la máquina, que puede ser $10^{-}16$, no un 0 absoluto. En cuanto a "minsplit", este parámetro se refiere al número mínimo de observaciones que deben existir en un nodo para que se intente una división, y como nosotros no queremos un árbol sumamente largo e imposible de interpretar, lo fijamos en el 10 % de las filas, es decir, del número de pacientes de la base de datos de entrenamiento, para ello multiplicamos 0.1 por "n_filas", y por último en "parms" informamos de que nuestro criterio de división es la entropía de la información. Ahora solo nos queda graficar el árbol, y para ello emplearemos la función "rpart.plot()", donde introduciremos como parámetro principal nuestro árbol, y lo retocaremos haciendo uso de "extra = 4" que en nuestro caso nos da información en cada nodo de la probabilidad de ser diabético o no diabético de los pacientes que se incluyen dentro de dicho nodo, "under = F" para que estas probabilidades aparezcan dentro y no debajo de cada nodo, "fallen.leaves = T" por cuestión de estética, para que todas las hojas del árbol lleguen hasta abajo y la interpretación de la variable respuesta sea más visual, y "main" para darle un título al árbol de decisión. Finalmente, con "return()" devolvemos en una lista el árbol de decisión creado, así como su gráfico.

- Testeo: En este caso, añadiremos un parámetro más a la función, que tendrá un total de dos parámetros, la base de datos de testeo y el árbol de decisión creado en la función de entrenamiento. Al inicio de la función calculamos el número de columnas de la base de datos de testeo, de manera que en el siguiente paso podamos eliminar la última columna de dicha base de datos para que esta pueda ser predicha con nuestro árbol. De este modo, haciendo uso de la función "predict()", aplicamos nuestro árbol a los datos de testeo sin la variable respuesta, y lo guardamos en el objeto "testeo". A continuación, creamos una matriz de confusión con el objetivo de calcular posteriormente el % de registros que el modelo clasifica correctamente, ya que dicha matriz compara los datos originales de la variable respuesta de la base de datos de testeo, con los que acabamos de predecir en el paso anterior, esto lo conseguimos utilizando la función "table()". Finalmente, calculamos la precisión global del modelo sumando los registros de la diagonal principal de la matriz de confusión, que son los bien clasificados, y dividimos entre todos los registros de la base de datos, multiplicamos por 100 para que salga porcentaje, y lo rendondeamos a 2 decimales, esto lo guardamos en la variable "precision" y lo devolvemos mediante "return()".
- Predicción: Una vez asegurada la buena precisión de nuestro árbol, creamos la tercera función, la de predicción, donde una vez más utilizaremos dos parámetros, los datos de predicción y el árbol de decisión. La función consiste en predecir la variable respuesta de la base de datos de predicción mediante la aplicación, sobre estos datos, del árbol creado previamente. Como en la función de testeo, haremos uso de la función "predict()", convirtiendo el resultado de esta en "Data Frame" y guardándolo en el objeto "prediccion". A continuación, mediante la función "cbind()" fusionamos el objeto "prediccion" con los datos de predicción, formando de este modo la última columna que contiene los nuevos datos predichos de la variable objetivo. Para finalizar, devolvemos el "Data Frame" resultante con "return()".

```
1  library(readxl)
2  library(caret)
3  library(rpart)
4  library(rpart.plot)
5
6  # leer la base de datos
7  datos<-data.frame(read_excel("diabetes_prediction_dataset.xlsx"))
8  colnames(datos)[ncol(datos)]<-"respuesta"
9
10 # Dividimos la base en tres partes iguales (Entrenamiento, testeo
       y predicci n)
11 set.seed(1)
12 registros_entrenamiento <- createDataPartition(datos$respuesta, p
       = 0.33, list = F)
13 datos_entrenamiento <- datos[registros_entrenamiento,]
14
15 # Variable temporal, partici n de la base de datos con el resto
     de registros no usados
16 # en la base de datos de entrenamiento, esta base de datos
     temporal ser  dividida a su vez en datos de testeo
17 # y datos de predicci n
18 temp <- datos[-registros_entrenamiento,]
19 registros_testeo <- createDataPartition(temp$respuesta, p = 0.5,
     list = F)
20 datos_testeo <- temp[registros_testeo,]
```

```
21 datos_prediccion <- temp[-registros_testeo,]
22 n_columnas<-ncol(datos)
23 datos_prediccion <- datos_prediccion[,-n_columnas]
24
25 # Entrenamiento
26 entrenamiento<-function(datos_entrenamiento){
27   n_filas<-nrow(datos_entrenamiento)
28   arbol_decision <- rpart(respuesta ~ ., data = datos_
       entrenamiento, method = "class", cp=-1, minsplit=0.1*n_filas
       , parms = list(split = "information"))
29   plot<-rpart.plot(arbol_decision, extra = 4, under= F, fallen.
       leaves= T, main = " rbol de decisi n para predecir")
30   return(list(arbol_decision,plot))
31 }
32 arbol_plot<-entrenamiento(datos_entrenamiento)
33
34 # Testeo
35 testeo<-function(datos_testeo,arbol){
36   n_columnas<-ncol(datos_testeo)
37   #Aplicamos nuestro rbol al conjunto de testeo
38   testeo <- predict(arbol, datos_testeo[,-n_columnas], type = "
       class")
39   #Creamos la matriz de confusi n para ver que % de registros ha
        clasificado bien el modelo
40   matriz_confusion <- table((datos_testeo[,n_columnas])[[1]],
       testeo)
41   #Calculamos la precisi n global del modelo sumando los
       registros de la diagonal principal
42   #de la matriz de confusi n que son los bien clasificados, y
       divisimos entre todos los registros de la base de datos,
43   #multiplicamos por 100 para que salga porcentaje, y lo
       rendondeamos a 2 decimales
44   precision <- round((sum(diag(matriz_confusion))/nrow(datos_
       testeo))*100,2)
45   return(cat(precision, "%"))
46 }
47 testeo(datos_testeo,arbol_plot[[1]])
48
49 # Prediccion
50 prediccion<-function(datos_prediccion,arbol){
51   prediccion <- data.frame("respuesta"=predict(arbol, datos_
       prediccion, type = "class"))
52   return(cbind(datos_prediccion,prediccion))
53 }
54 datos_prediccion<-prediccion(datos_prediccion,arbol_plot[[1]])
```

## 3.3. Descripción del código de la interfaz

Para desarrollar la interfaz gráfica en Rstudio nos hemos servido únicamente de la librería "shiny" (Chang, 2020), la cual hemos cargado mediante la orden "library()". Es necesario destacar que el código de nuestra interfaz gráfica se compone de dos funciones raíz, "ui" y "server".

La función "ui" construida mediante la función "fluidPage()", se encarga de crear la parte visible de la interfaz gráfica, la cual se compone por tres tipos de elementos, elementos que únicamente producen texto ("titlePanel()" y "mainPanel()"), instrucciones que sirven para crear objetos que piden inputs al usuario ("fileInput()" y "numericInput()"), e instrucciones que crean objetos que devuelven outputs al usuario. Más detalladamente:

- titlePanel(): Como su nombre indica, se encarga de crear el título de la interfaz gráfica.
- mainPanel(): Produce un texto en letra más pequeña que el título. En nuestro caso lo utilizaremos para advertir al usuario de cómo debe estar estructurada su base de datos a la hora de utilizar la aplicación.
- fileInput(): Su función es examinar dónde está la base de datos que quiere usar el usuario, así como cargar la misma. Como parámetros le damos el Id de nuestra base de datos, un texto para guiar al usuario, y la especificación de que lea bases de datos provenientes de excel.
- numericInput(): Se ocupa de pedir un número al usuario. En este caso lo utilizamos para que el usuario indique el porcentaje de la base de datos original que quiere dedicar a entrenamiento, y a testeo.
- verbatimTextOutput(): Devuelve un texto. Hacemos uso de él para indicar al usuario el porcentaje de la base de datos original que será dedicada a predicción. Este se calcula automáticamente al introducir la partición de la base de datos para entrenamiento y testeo.
- plotOutput(): Se encarga de devolver un gráfico. En nuestra aplicación devolverá el dibujo de nuestro árbol de decisión.
- tableOutput(): Devuelve una tabla. En nuestra caso esta será la base de datos de predicción con los valores de la variable respuesta predichos por el árbol de decisión.

```
ui <- fluidPage(
  titlePanel("Construya su arbol de decision"),
  mainPanel("Recuerde que en su base de datos ... y no
     codificadas por numeros"),
  fileInput("datos","Cargue su base de datos: ",accept=".xlsx"),
  numericInput("p_entrenamiento","Indique proporcion ... base de
     datos en formato decimal: ",value=NULL),
  numericInput("p_testeo","Indique la proporcion ... base de
     datos en formato decimal: ",value=NULL),
  verbatimTextOutput("p_prediccion"),
  plotOutput("plot"),
  verbatimTextOutput("precision"),
  tableOutput("prediccion")
)
```

Por otro lado, tenemos la función "server" cuya tarea es realizar todos los cálculos internos a través del código que hemos creado para realizar el algoritmo del árbol de decisión, de manera que recibe los inputs creados en la función "ui", y devuelve los outputs también hacia "ui". Para recibir los inputs utilizamos "input$" más el nombre del input y para devolverlos "output$" más el nombre

del output. Para ello, hacemos uso de las instrucciones “renderPrint()”, “renderPlot()” y “renderTable()”, con el fin de devolver texto, gráficos o tablas respectivamente.

Es importante mencionar que en nuestro algoritmo para crear el árbol de decisión, indicamos en el código que cada partición de la base de datos fuese de un 33 % aproximadamente. En la interfaz de usuario necesitamos que el usuario indique la proporción que desee, por tanto la parte dedicada a predicción la calculamos mediante la siguiente fracción: “ p = input$p_testeo/(1-input$p_entrenamiento) ”.

Finalmente ejecutamos nuestra aplicación mediante la función “shinyApp(ui, server)” donde introducimos como parámetros las dos funciones anteriormente explicadas.

```
server <- function(input, output, session){
  output$p_prediccion<-renderPrint({...})
  output$plot<-renderPlot({...})
  output$precision<-renderPrint({...})
  output$prediccion<-renderTable({...})
}
```

# Capítulo 4

# Conclusiones

## 4.1. Interpretación de los resultados

Por un lado, desarrollaremos una conclusión más técnica interpretando los resultados de nuestro estudio: En cuanto al análisis de la base de datos utilizada en nuestro trabajo, una vez aplicado nuestro algoritmo, obtenemos el correspondiente árbol de decisión, el cual está disponible en el Anexo. Observándolo detenidamente, antes de nada, en el nodo raíz podemos apreciar que el 91 % de los pacientes totales no son diabéticos, frente al 9 % que sí lo son.
La variable que utiliza nuestro árbol para llevar a cabo la primera división es el "HbA1c_level", es decir, la hemoglobina glicosilada, donde podemos observar dos ramificaciones, los pacientes cuyos niveles son superiores o iguales a 6.7, que son clasificados como diabéticos en el 100 % de los casos, y aquellos cuyo nivel de hemoglobina glicosilada es inferior a 6.7, cuyo 95 % es clasificado como no diabéticos y al 5 % restante como diabéticos.
La segunda división la vuelve a ejecutar con la misma variable, pero esta vez para un nivel de hemoglobina glicosilada mayor o igual a 5.4, donde clasifica al 100 % de los pacientes que están por debajo como no diabéticos, mientras que, de los que cumplen la condición, categoriza como no diabéticos al 92 % y como diabéticos al 8 %.
En este nodo ya utiliza otra variable para dividir que será "blood_glucose_level", es decir, el nivel de glucosa en sangre. Donde para los pacientes con un nivel mayor o igual a 180, considera no diabéticos al 66 % y diabéticos al 34 %, mientras que para los que tienen un nivel inferior a este, considera no diabéticos al 95 %, considerando diabéticos al 5 %.
En este cuarto nodo, una vez más, a la hora de dividir repite variable, fijando la condición en un nivel de glucosa en sangre mayor o igual a 113, donde para el grupo de pacientes que lo cumplen clasifica al 93 % como no diabéticos y a un 7 % como diabéticos, mientras que agrupa a todos los pacientes que no cumplen la condición como no diabéticos.
En este quinto nodo, utiliza la variable "edad" como variable de división, donde los pacientes mayores o iguales a 43 años son clasificados en un 87 % como no diabéticos y en un 13 % como diabéticos, mientras que los menores de esta edad los considera en un 98 % no diabéticos y en un 2 % diabéticos.
Ahora, por un lado, de entre los primeros (mayores o iguales a 43 años), considera a los que tienen un índice de masa corporal superior o igual a 33 no diabéticos en un 75 % y diabéticos en un 25 %, y a los que tienen un índice de masa corporal inferior, los considera no diabéticos en un 90 % y diabéticos en un 10 %, y de estos a su vez establece otra división considerando a los mayores o iguales a 66 años, no diabéticos en un 84 % y diabéticos en un 16 %, y a los menores de esta edad no diabéticos en un 93 % de los casos y diabéticos en un 7 %.
Por otro lado, entre los segundos en el quinto nodo (menores de 43 años), agrupa como no diabéti-

cos al 97 % de los que tienen más o igual a 27 años, y como diabéticos al 3 % de los que cumplen esta condición. Sin embargo, etiqueta como no diabéticos a cerca del 100 % de los que tienen menos de 27 años, y entre ellos establece la última división, clasificando a los mayores o iguales a 12 años como no diabéticos en un 99 % y por tanto como diabéticos al 1 %, mientras que agrupa como no diabéticos a todos los que tienen menos de 12 años.

## 4.2. Detalles a tener en cuenta sobre los árboles de decisión

Por otro lado, resulta necesario también, exponer unas conclusiones acerca de lo aprendido sobre los árboles de decisión a través de la elaboración de este trabajo:

- Deben ser visuales, sin demasiadas ramificaciones, aunque perdamos precisión.
- Resultan especialmente efectivos si nuestra variable respuesta es binaria, ya que la visibilidad aumenta.
- Son una muy buena opción cuando contamos con una base de datos con todas o casi todas las variables categóricas.
- Es esencial particionar la base de datos y utilizar cada subconjunto de esta adecuadamente, en particular, una parte dedicada al entrenamiento del árbol, de lo contrario podemos tener sobreajustes y problemas a la hora de predecir.
- Cobra también especial importancia contar con una base de datos limpia y preprocesada.
- Es importante la fase de evaluación, ya que debemos elegir modelos con una buena bondad de ajuste si queremos precisión en los resultados.
- Podemos aplicar nuestro mismo algoritmo a diferentes bases de datos obteniendo grandes resultados.
- Sin embargo, no parecen ser la mejor opción cuando trabajamos con variables con numerosas categorías.

# Bibliografía


Chang, W. (2020). *shiny: Web Application Framework for R* [Accessed: 15-06-2023]. https://cran.r-project.org/web/packages/shiny/index.html

Clark, P., & Boswell, R. (2000). *Practical Machine Learning Tools and Techniques with Java Implementation*. Morgan Kaufmann Publisher.

Fayyad, U. M., Piatetsky-Shapiro, G., & Smyth, P. (1996). From Data Mining to Knowledge Discovery: An Overview. *Advances in Knowledge Discovery and Data Mining. AAAI/MIT Press*, 1-34. https://doi.org/10.1609/aimag.v17i3.1230

Han, J., & Kamber, M. (2006). *Data Mining: Concepts and Techniques,* $2^{nd}$. University of Illinois at Urbana Champaign: Morgan Kaufmann.

Kaggle. (2023). *Diabetes prediction dataset* [Accessed: 05-07-2023]. https://www.kaggle.com/datasets/iammustafatz/diabetes-prediction-dataset

Liu, B. (2011). *Web data mining: exploring hyperlinks, contents, and usage data (Vol. 1)*. Berlin: Springer.

Ye, N. (2013). *Data mining: theories, algorithms, and examples*. CRC Press.

# Anexos

## Código

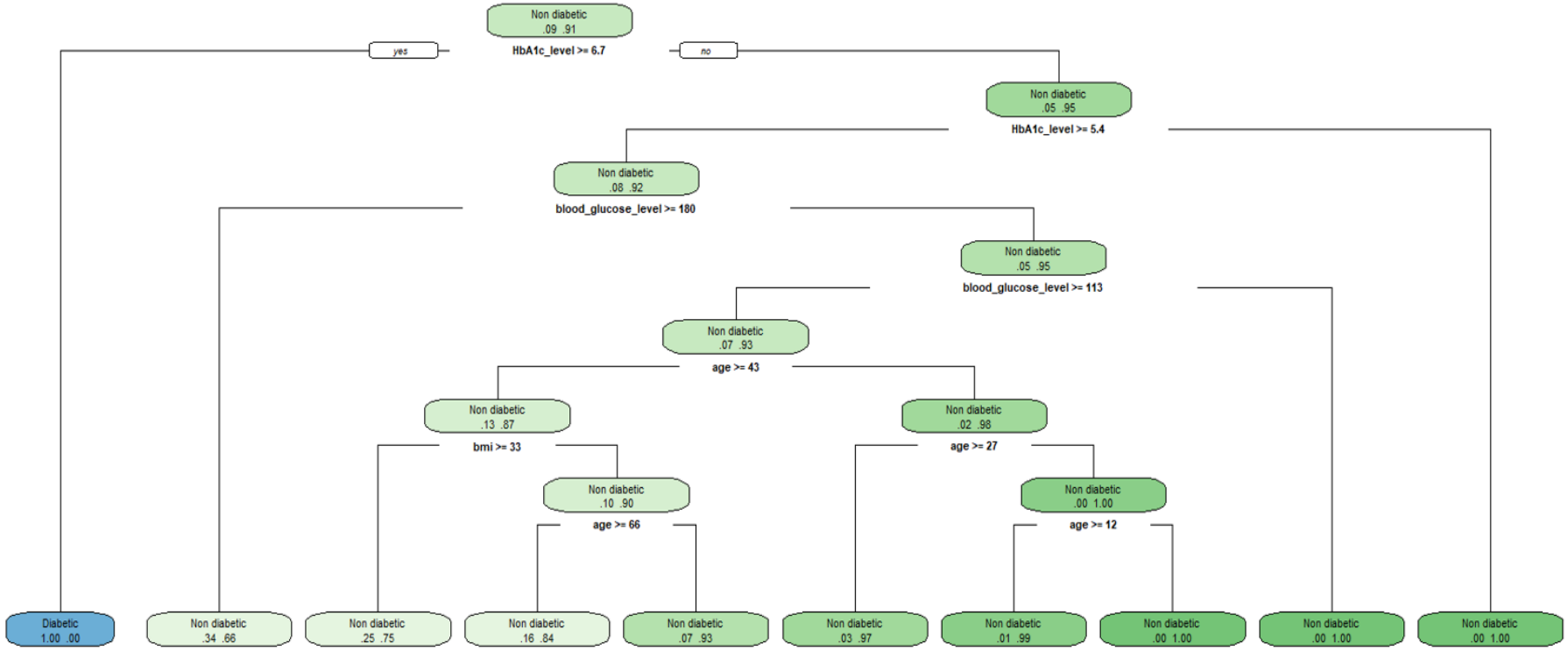
```
library(shiny)

ui <- fluidPage(
  titlePanel("Construya su arbol de decision"),
  mainPanel("Recuerde que en su base de datos de excel el
     separador decimal debe ser la coma, y las variables
     cualitativas deben estar etiquetadas y no codificadas por
     numeros"),
  fileInput("datos","Cargue su base de datos: ",accept=".xlsx"),
  numericInput("p_entrenamiento","Indique la proporcion que
     quiere dedicar a entrenamiento de su base de datos en
     formato decimal: ",value=NULL),
  numericInput("p_testeo","Indique la proporcion que quiere
     dedicar a testeo de su base de datos en formato decimal: ",
     value=NULL),
  verbatimTextOutput("p_prediccion"),
  plotOutput("plot"),
  verbatimTextOutput("precision"),
  tableOutput("prediccion")
)

server <- function(input, output, session) {
  output$p_prediccion<-renderPrint({
    paste("La proporcion dedicada a prediccion de su base de
       datos sera: ",1-input$p_entrenamiento-input$p_testeo)
  })
  output$plot<-renderPlot({
    library(readxl)
    library(caret)
    library(rpart)
    library(rpart.plot)

    # Leer la base de datos
    datos<-data.frame(read_excel(input$datos$datapath, 1))
    colnames(datos)[ncol(datos)]<-"respuesta"

    # Particion de la base de datos
```

```
30      set.seed(1)
31      registros_entrenamiento <- createDataPartition(datos$
            respuesta, p = input$p_entrenamiento, list = F)
32      datos_entrenamiento <- datos[registros_entrenamiento,]
33
34      # Entrenamiento
35      entrenamiento<-function(datos_entrenamiento){
36        n_filas<-nrow(datos_entrenamiento)
37        arbol_decision <- rpart(respuesta ~ ., data = datos_
              entrenamiento, method = "class", cp=-1, minsplit=0.1*n_
              filas, parms = list(split = "information"))
38        plot<-rpart.plot(arbol_decision, extra = 4, under= F,
              fallen.leaves= T, main = "Arbol de decision para
              predecir")
39        return(list(arbol_decision,plot))
40      }
41      arbol_plot<-entrenamiento(datos_entrenamiento)
42      print(arbol_plot[[2]])
43    })
44    output$precision<-renderPrint({
45      library(readxl)
46      library(caret)
47      library(rpart)
48      library(rpart.plot)
49
50
51      # Leer la base de datos
52      datos<-data.frame(read_excel(input$datos$datapath, 1))
53      colnames(datos)[ncol(datos)]<-"respuesta"
54
55      # Particion de la base de datos
56      set.seed(1)
57      registros_entrenamiento <- createDataPartition(datos$
            respuesta, p = input$p_entrenamiento, list = F)
58      datos_entrenamiento <- datos[registros_entrenamiento,]
59
60      # Variable temporal
61      temp <- datos[-registros_entrenamiento,]
62      registros_testeo <- createDataPartition(temp$respuesta, p =
            input$p_testeo/(1-input$p_entrenamiento), list = F)
63      datos_testeo <- temp[registros_testeo,]
64
65      # Entrenamiento
66      entrenamiento<-function(datos_entrenamiento){
67        n_filas<-nrow(datos_entrenamiento)
68        arbol_decision <- rpart(respuesta ~ ., data = datos_
              entrenamiento, method = "class", cp=-1, minsplit=0.1*n_
              filas, parms = list(split = "information"))
69        return(arbol_decision)
```

```
70      }
71      arbol_plot<-entrenamiento(datos_entrenamiento)
72
73      # Testeo
74      testeo<-function(datos_testeo,arbol){
75
76        n_columnas<-ncol(datos_testeo)
77
78        #Aplicamos nuestro arbol al conjunto de testeo
79        testeo <- predict(arbol, datos_testeo[,-n_columnas], type =
              "class")
80
81        #Creamos la matriz de confusion para ver que % de registros
              ha clasificado bien el modelo
82        matriz_confusion <- table(datos_testeo[,n_columnas], testeo
              )
83
84        #Calculamos la precision global del modelo sumando los
              registros de la diagonal principal
85        #de la matriz de confusion que son los bien clasificados, y
              dividimos entre todos los registros de la base de datos
              ,
86        #multiplicamos por 100 para que salga porcentaje, y lo
              rendondeamos a 2 decimales
87        precision <- round((sum(diag(matriz_confusion))/nrow(datos_
              testeo))*100,2)
88        return(precision)
89      }
90      paste("El algoritmo clasifica correctamente el ",testeo(datos
          _testeo,arbol_plot),"% de los registros",sep="")
91    })
92    output$prediccion<-renderTable({
93      library(readxl)
94      library(caret)
95      library(rpart)
96      library(rpart.plot)
97
98      # Leer la base de datos
99      datos<-data.frame(read_excel(input$datos$datapath, 1))
100     colnames(datos)[ncol(datos)]<-"respuesta"
101
102     # Particion de la base de datos
103     set.seed(1)
104     registros_entrenamiento <- createDataPartition(datos$
            respuesta, p = input$p_entrenamiento, list = F)
105     datos_entrenamiento <- datos[registros_entrenamiento,]
106
107     # Variable temporal
108     temp <- datos[-registros_entrenamiento,]
```

```
109       registros_testeo <- createDataPartition(temp$respuesta, p =
              input$p_testeo/(1-input$p_entrenamiento), list = F)
110       datos_testeo <- temp[registros_testeo,]
111       datos_prediccion <- temp[-registros_testeo,]
112       n_columnas<-ncol(datos)
113       datos_prediccion <- datos_prediccion[,-n_columnas]
114
115       # Entrenamiento
116       entrenamiento<-function(datos_entrenamiento){
117         n_filas<-nrow(datos_entrenamiento)
118         arbol_decision <- rpart(respuesta ~ ., data = datos_
              entrenamiento, method = "class", cp=-1, minsplit=0.1*n_
              filas, parms = list(split = "information"))
119         return(arbol_decision)
120       }
121       arbol_plot<-entrenamiento(datos_entrenamiento)
122
123       # Prediccion
124       prediccion<-function(datos_prediccion,arbol){
125         prediccion <- data.frame("respuesta"=predict(arbol, datos_
              prediccion, type = "class"))
126         return(cbind(datos_prediccion,prediccion))
127       }
128       datos_prediccion<-prediccion(datos_prediccion,arbol_plot)
129       print(datos_prediccion)
130     })
131   }
132
133   shinyApp(ui, server)
```

# Árbol de decisión para predecir la diabetes






54