Trabajo unidad 1

Alex Guaman - Evelyn Faican

Running Code

Capitulo 1

Es una esencia el aprendizaje ya que ocupa diversos algoritmos en donde la información es procesable, en donde el aprendizaje automático va adaptándose con la era actual.

- El lenguaje de R ofrece un conjunto de herramientas potente que nos ayudan a encontrar información sobre los datos.

Existen diferentes etapas en donde comienza desde:

- Juegos simples (tres en raya - ajedrez)

Pero fueron avanzando y diseñaron controles de:

- Comunicaciones - drones militares

Cada vez la evolución de las máquinas da un giro más fuerte, en donde las computadoras cada vez se vuelven más conscientes es así que los diferentes aprendizajes pueden verse representados por la Inteligencia artificial y por los diferentes medios de comunicación, es importante conocer que se aprenderá diversos campos como:

- Los orígenes y las aplicaciones prácticas del aprendizaje automático
- Cómo las computadoras definen y representan el conocimiento
- Los conceptos básicos que diferencian los enfoques de aprendizaje automático.

Los orígenes del aprendizaje automático

Se denominan sensores del cuerpo humano a los ojos, oídos, nariz, lengua y a los nervios en donde van saltando continuamente los diversos datos que nuestro cerebro los traduce en olores, sabores, texturas, imágenes e incluso el usando el lenguaje tenemos diversas experiencias que podemos compartir con otros. Los primeros bases de datos que fueron registrados fue la información del entorno observable en donde el ser humano primero observaba y luego registraba la observación en donde cada vez fueron automatizadas y sistemáticamente computarizadas.

La invención de los diversos sensores electrónicos ha contribuido con gran riqueza a los datos registrados en donde tienen especialidad de probar, oler, ver e incluso analizar siendo de Gran beneficio para la sociedad. Poco a poco la sociedad ha ido entrando en una era que se le denomina Big Data en donde las máquinas procesan directamente gran parte de la información parte desde tomar diferentes decisiones dándole sentido a todo. El campo de estudio nos ha brindado un algoritmo de información que nos permite transformar datos en acciones inteligentes a lo que cada vez un estímulo en el desarrollo se basan en métodos estadísticos para poderlos analizar con llevados como conjunto de datos y lo que nos permite recopilar cada vez datos interesantes.

Existe una pregunta que es conveninete analizarla y comprenderla: **¿Cómo aprenden las máquinas?** El científico informático M. Mitchell denomina a una máquina en base a su experiencia y su rendimiento que cada vez deben ir mejorando en torno a las experiencias similares en el futuro en donde transforman los datos en conocimiento procesable, en donde se divide en tres componentes esenciales:

Entrada de datos	Abstracción	Generalización
Utiliza la observación, el almacenamiento de memoria y el recuerdo para proporcionar una base fáctica para un razonamiento posterior.	Implica la traducción de datos en representaciones más amplias	Utiliza datos abstractos para formar una base para la acción.

Los tres componentes del aprendizaje están inextricablemente vinculados

Las estrategias de aprendizaje comúnmente utilizadas para crear un esquema o un mapa conceptual son similares a cómo una máquina realiza la abstracción del conocimiento.

Abstracción y representación del conocimiento

Existen conexiones abstractas que son la base de la representación del conocimiento en donde la formación de estructuras lógicas ayudan a convertir la información sensorial en bruto en una percepción significativa que durante el proceso de representación del conocimiento, la computadora resume las entradas sin procesar en un modelo que pódemos representarlo en varios modelos como:

- Ecuaciones
- Diagramas como árboles y gráficos
- Reglas lógicas if/else
- Agrupaciones de datos conocidas como clústeres

El proceso de ajustar un modelo particular a un conjunto de datos se conoce como capacitación en donde el proceso de aprendizaje no termina con el paso de abstracción de datos, en cambio generaliza el conocimiento a datos futuros considerando que el aprendizaje implica una especie de razonamiento inductivo que se transformado en una forma abstracta y que resume la información original.

- Es importante tener en cuenta que el modelo en sí mismo no proporciona datos adicionales, aunque a veces es interesante por sí solo.

La mayoría de los modelos no darán como resultado el desarrollo de teorías que sacudirán el pensamiento científico durante siglos. Aún así, su modelo podría dar como resultado el descubrimiento de relaciones no vistas previamente entre los datos.

Generalización

El términogeneralizacióndescribe el proceso de convertir el conocimiento abstracto en una forma que se puede utilizar para la acción. La generalización es un proceso algo vago que es un poco difícil de describir.

Ejemplo:

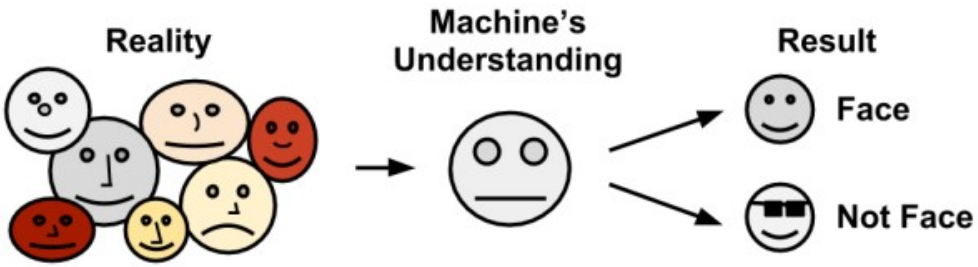
ALGORITMO	Heurística
Se utiliza aproximaciones y otrasreglas generales.	Se pretende encontrar información útil en un gran conjunto de datos.

ALGORITMO	Heurística
Los seres humanos utilizan rutinariamente las heurísticas para generalizar rápidamente la experiencia a los nuevos escenarios.	Se utiliza el instinto para tomar una decisión rápida antes de evaluar completamente las circunstancias, usando intuitivamente la heurística mental.
Puede dar lugar conclusiones ilógicas y erróneas.	Si las conclusiones son sistemáticamente imprecisas, se dice que el algoritmo tiene una inclinación.

Por ejemplo:

Suponga que un algoritmo de aprendizaje automático nos permita identificar caras y al encontrar dos círculos u ojos, colocados uno al lado del otro sobre una línea para la boca. Entonces, el algoritmo podría tener problemas con las caras que no se ajustan a su modelo o estar sesgado contra ellas. Esto puede incluir rostros con anteojos, girados en ángulo, mirando de lado o con tonos de piel más oscuros.

- De manera similar, podría estar sesgado hacia rostros con colores de ojos más claros u otras características que no se ajusten a su comprensión del mundo.



Evaluar el éxito del aprendizaje

El sesgo es un mal necesario asociado con el proceso de abstracción y generalización inherente a cualquier tarea de aprendizaje automático. Cada alumno tiene sus debilidades y está sesgado de una manera particular; no hay un modelo único para gobernarlos a todos.

- Una vez que un modelo ha sido entrenado en un conjunto de datos inicial, el modelo se prueba en un nuevo conjunto de datos y se juzga en qué medida su caracterización de los datos de entrenamiento se generaliza a los nuevos datos. Vale la pena señalar que es extremadamente raro que un modelo generalice perfectamente todos los casos imprevistos.

Los datos ruidosos son causados por eventos aparentemente aleatorios, como:

- Error de medición debido a sensores imprecisos que a veces suman o restan un bit de la lectura
- Problemas con los datos de informes, como que los encuestados informen respuestas aleatorias a las preguntas de la encuesta para terminar más rápido.
- Errores causados cuando los datos se registran incorrectamente, incluidos valores faltantes, nulos, truncados, codificados incorrectamente o dañados.

Pasos para aplicar el aprendizaje automático a sus datos

Pasos	Información
1. Recolectando datos:	Los datos estén escritos en papel, registrados en archivos de texto y hojas de cálculo, o almacenados en una base de datos SQL, deberá recopilarlos en un formato electrónico adecuado para el análisis.
2. Exploración y preparación de los datos.:	La calidad de cualquier proyecto de aprendizaje automático se basa en gran medida en la calidad de los datos que utiliza. Este paso en el proceso de aprendizaje automático tiende a requerir una gran cantidad de intervención humana.
3. Entrenamiento de un modelo en los datos:	Los datos hayan sido preparados para el análisis, es probable que tenga una idea de lo que espera aprender de los datos.
4. Evaluación del rendimiento del modelo:	Debido a que cada modelo de aprendizaje automático da como resultado una solución sesgada al problema de aprendizaje, es importante evaluar qué tan bien aprendió el algoritmo a partir de su experiencia.
5. Mejora del rendimiento del modelo:	Se necesita un mejor rendimiento, se hace necesario utilizar estrategias más avanzadas para aumentar el rendimiento del modelo.

Después de completar estos pasos, si el modelo parece estar funcionando satisfactoriamente, se puede implementar para la tarea prevista. Según sea el caso, puede utilizar su modelo para proporcionar datos de puntuación para predicciones.

Elegir un algoritmo de aprendizaje automático

La elección de un algoritmo de aprendizaje automático depende en gran medida del tipo de datos que está analizando y de la tarea propuesta en cuestión, a menudo es útil pensar en este proceso mientras recopila, explora y limpia sus datos.

Pensando en los datos de entrada

Todos los algoritmos de aprendizaje automático requieren datos de entrenamiento de entrada. El formato exacto puede diferir, pero en su forma más básica, los datos de entrada toman la forma de ejemplos y características.

- La unidad de observación tiene la forma de transacciones, personas, puntos de tiempo, regiones geográficas o medidas.
- Otras posibilidades incluyen combinaciones de estos, como años de persona, que denotarían casos en los que se realiza un seguimiento de la misma persona en varios puntos de tiempo.
- Las característica son un atributo de un ejemplo, que puede ser útil para aprender el concepto deseado.

Pensando en los tipos de algoritmos de aprendizaje automático

Los algoritmos de aprendizaje automático se pueden dividir en dos grupos principales:

- Estudiantes supervisados que se usan para construir modelos predictivos y estudiantes no supervisados que se usan para construir modelos descriptivos.
- El tipo que necesitará usar depende de la tarea de aprendizaje que espera lograr.

Modelo predictivo

Se usa para tareas que involucran, como su nombre lo indica, la predicción de un valor usando otros valores en el conjunto de datos. El algoritmo de aprendizaje intenta descubrir y modelar la relación entre los objetivo que se predice y las otras características.

- La característica objetivo que se va a predecir es una característica categórica conocida como clase y se divide en categorías llamadasniveles.

- Una clase puede tener dos o más niveles, y los niveles no tienen por qué ser necesariamente ordinales. Debido a que la clasificación se usa tanto en el aprendizaje automático, existen muchos tipos de algoritmos en torno a la clasificación.

Modelo descriptivo

Se utiliza para tareas que se beneficiarían de la información obtenida al resumir datos de formas nuevas e interesantes. A diferencia de los modelos predictivos que predicen un objetivo de interés; en un modelo descriptivo, ninguna característica es más importante que otra. De hecho, debido a que no hay un objetivo para aprender, el proceso de entrenamiento de un modelo descriptivo se llama aprendizaje sin supervisión.

- La tarea de modelado descriptivo de dividir un conjunto de datos en grupos homogéneos se llama agrupamiento. Esto a veces se usa para el análisis de segmentación que identifica grupos de personas con información similar de compras, donaciones o demográfica para que las campañas publicitarias se puedan adaptar a audiencias particulares.

Uso de R para el aprendizaje automático

Muchos de los algoritmos necesarios para el aprendizaje automático en R no se incluyen como parte de la instalación básica. Gracias a que R es un software gratuito de código abierto, no hay cargo adicional por esta funcionalidad. Una gran comunidad de expertos que contribuyeron al software agregaron los algoritmos necesarios para el aprendizaje automático a la base R. Una colección de funciones de R que se pueden compartir entre los usuarios se denomina paquete. Existen paquetes gratuitos para cada uno de los algoritmos de aprendizaje automático que se tratan en este libro.

Instalación de un paquete R

- La forma más directa de instalar un paquete es a través de la función instalar.paquetes().
- Para instalar el R Weka paquete, en el símbolo del sistema R simplemente escriba:

  > install.packages("RWeka")
- R luego se conectará a CRAN y se descargará el paquete en el formato correcto para su sistema operativo. Algunos paquetes como R Weka requieren que se instalen paquetes adicionales antes de que puedan usarse (estos se denominan dependencias). De manera predeterminada, el instalador descargará e instalará automáticamente cualquier dependencia.

- Las opciones de instalación predeterminadas son apropiadas para la mayoría de los sistemas. Sin embargo, en algunos casos, es posible que desee instalar un paquete en otra ubicación

  > install.packages("RWeka", lib="/path/to/library")

- La función de instalación también proporciona opciones adicionales para instalar desde un archivo local, instalar desde la fuente o usar versiones experimentales. Puede leer acerca de estas opciones en el archivo de ayuda usando el siguiente comando:

  > ?install.packages

Conclusión del capitulo 1

El aprendizaje automático se originó en la intersección de las estadísticas, la ciencia de bases de datos y la informática. Es una herramienta poderosa, capaz de encontrar información procesable en grandes cantidades de datos. Aún así, se debe tener precaución para evitar abusos comunes del aprendizaje automático en el mundo real.

- En términos conceptuales, el aprendizaje implica la abstracción de datos en una representación estructurada y la generalización de esta estructura en acción.
- En términos más prácticos, un aprendiz automático utiliza datos que contienen ejemplos y características del concepto que se va a aprender.
- Entre las muchas opciones, los algoritmos de aprendizaje automático se eligen en función de los datos de entrada y la tarea de aprendizaje.
- *R proporciona soporte para el aprendizaje automático en forma de paquetes creados por la comunidad.*
- *Estas potentes herramientas se pueden descargar gratis, pero es necesario instalarlas antes de poder usarlas.*

Capítulo 2

Gestión y comprensión de los datos

Uno de los primeros componentes clave de cualquier proyecto de aprendizaje automático es la gestión y comprensión de los datos recopilados. El algoritmo de aprendizaje es bueno si los datos de entrada son buenos y, en muchos casos, los datos de entrada son complejos, desordenados y poco fiables, es por eso que la mayor parte del esfuerzo invertido en proyectos de aprendizaje automático se dedica a la preparación y exploración de los datos.

Estructuras de datos en R

Las estructuras de datos de R que se utilizan con mayor frecuencia en el aprendizaje automático son vectores, factores, listas, matrices y marcos de datos. Cada uno de estos tipos de datos está especializado para una tarea de administración de datos específica, por lo que es importante comprender cómo interactuarán en su proyecto de R.

Vectores

La estructura de datos fundamental de R es la vector, que almacena un conjunto ordenado de valores llamado elementos. Un vector puede contener cualquier número de elementos.

Construyamos un conjunto de vectores que contengan datos sobre tres pacientes médicos.

Tipos de vectores:

- Entero (números sin decimales)
- Numérico (números con decimales)
- Carácter (datos de texto)
- Lógico (valores TRUE o FALSE).
- NULL, se utiliza para indicar la ausencia de cualquier valor
- NA, que indica un valor omitido.

Para crear vectores se usa la función de combinación c(), también se puede dar un nombre al vector utilizando el operador de flecha <-, que es el operador de asignación de R.

Por ejemplo, vamos a construir un vector. Se crea un un vector de caracteres llamado **nombre_sujeto,** un vector numérico llamado **temperatura** y un vector lógico **estado_gripe**; TRUE si tiene gripe, FALSE en caso contrario.

```
subject_name<- c("John Doe", "Jane Doe", "Steve Graves")
temperature<- c(98.1, 98.6, 101.4)
flu_status<- c(FALSE, FALSE, TRUE)
```

Se puede acceder a los registros contando el número del elemento en el conjunto, y rodeándolo con corchetes después del nombre del vector.

```
temperature[1]
```

```
[1] 98.1
```

Un rango de valores se puede obtener utilizando el operador dos puntos.

```
temperature[2:3]
```

```
[1]  98.6 101.4
```

Los elementos pueden excluirse especificando un número de elemento negativo.

```
temperature[-2]
```

```
[1]  98.1 101.4
```

Por último, es útil especificar un vector lógico que indique si cada valor debe incluirse.

```
temperature[c(TRUE, TRUE, FALSE)]
```

```
[1] 98.1 98.6
```

Factores

R proporciona una estructura de datos conocida comofactor específicamente para este propósito. Un factor es un caso especial de vector que se utiliza únicamente para representar variables nominales. En el conjunto de datos médicos que estamos construyendo, podríamos usar un factor para representargénero,porque utiliza dos categorías:MASCULINO y FEMENINO.
Para crear un factor a partir de un vector de caracteres, simplemente aplique la función factor().

```
gender <- factor(c("MALE", "FEMALE", "MALE"))
gender
```

```
[1] MALE   FEMALE MALE
Levels: FEMALE MALE
```

Podemos agregar niveles adicionales que pueden no aparecer en los datos. Supongamos que agregamos otro factor para el tipo de sangre:

```
blood <- factor(c("O", "AB", "A"),
levels = c("A", "B", "AB", "O"))
blood
```

```
[1] O  AB A
Levels: A B AB O
```

Tenga en cuenta que cuando definimos el factor sanguíneo para los tres pacientes, especificamos un vector adicional de cuatro posibles tipos de sangre usando la instrucción level =. Como resultado, aunque nuestros datos incluyen solo los tipos O, AB y A, los cuatro tipos se almacenan con el factor sanguíneo como lo indican los niveles de salida: A B AB O. El almacenamiento del nivel adicional permite la posibilidad de agregar datos con el otro tipo de sangre en el futuro.

Listas

Se usa para almacenar un conjunto ordenado de valores. Sin embargo, a diferencia de un vector que requiere que todos los elementos sean del mismo tipo, una lista permite recopilar diferentes tipos de valores. Debido a esta flexibilidad, las listas a menudo se usan para almacenar varios tipos de datos de entrada y salida y conjuntos de parámetros de configuración para modelos de aprendizaje automático.

```
subject_name[1]
```

```
[1] "John Doe"
```

```
temperature[1]
```

```
[1] 98.1
```

```
flu_status[1]
```

```
[1] FALSE
```

```
gender[1]
```

```
[1] MALE
Levels: FEMALE MALE
```

Una lista se crea utilizando la función list(), se tiene la opción de proporcionar nombres para cada valor en la secuencia de elementos. Los nombres permiten acceder posteriormente a los valores de la lista por su nombre, en lugar de por su posición numerada.

```
subject1 <- list(fullname = subject_name[1],
 temperature = temperature[1],
 flu_status = flu_status[1],
 gender = gender[1],
 blood = blood[1])
subject1
```

```
$fullname
[1] "John Doe"

$temperature
[1] 98.1

$flu_status
[1] FALSE

$gender
[1] MALE
Levels: FEMALE MALE

$blood
[1] O
Levels: A B AB O
```

Se puede acceder a una lista utilizando los mismos métodos que a un vector, los nombres dan una claridad adicional para acceder a los valores.

```
subject1[2]
```

```
$temperature
[1] 98.1
```

A menudo es más fácil acceder directamente a la temperatura, añadiendo un $ y el nombre del valor

```
subject1$temperature
```

```
[1] 98.1
```

Es posible obtener varios elementos de una lista especificando un vector de nombres:

```
subject1[c("temperature", "flu_status")]
```

```
$temperature
[1] 98.1

$flu_status
[1] FALSE
```

Marco de Datos (Data Frames)

Con mucho, la estructura de datos R más importante utilizada en el aprendizaje automático es lamarco de datos, una estructura análoga a una hoja de cálculo o base de datos ya que tiene filas y columnas de datos. En términos de R, un marco de datos puede entenderse como una lista de vectores o factores, cada uno de los cuales tiene exactamente el mismo número de valores. Debido a que el marco de datos es literalmente una lista de vectores, combina aspectos de vectores y listas. Vamos a crear un marco de datos para nuestro conjunto de datos de pacientes. Usando los vectores de datos de pacientes que creamos previamente, el marco de datos()la función los combina en un marco de datos:

```
pt_data <- data.frame(subject_name, temperature, flu_status,
 gender, blood, stringsAsFactors = FALSE)
```

Incluimos un parámetro adicional: stringsAsFactors = FALSE. Si no especificamos esta opción, R convertirá automáticamente cada vector de caracteres en un factor. Aquí, el campo subject_name no es un dato categórico; los nombres no son categorías de valores. Por lo tanto, establecer la opción stringsAsFactors en FALSE nos permite convertir a factores solo donde tiene sentido para el proyecto.

Cuando mostramos el marco de datos pt_data, vemos que la estructura es bastante diferente de las estructuras de datos con las que trabajamos anteriormente

```
pt_data
```

```
  subject_name temperature flu_status gender blood
1     John Doe        98.1      FALSE   MALE     O
2     Jane Doe        98.6      FALSE FEMALE    AB
3 Steve Graves       101.4       TRUE   MALE     A
```

Un marco de datos tiene dos dimensiones y, por lo tanto, se muestra en formato de matriz. En términos de aprendizaje automático, las columnas son las características y las filas son los ejemplos.

Para extraer columnas enteras (vectores) de datos, la forma más directa de extraer un solo elemento, es referirse a él por su nombre.

```
pt_data$subject_name
```

```
[1] "John Doe"     "Jane Doe"     "Steve Graves"
```

También similar a las listas, se puede usar un vector de nombres para extraer varias columnas de un marco de datos:

```
pt_data[c("temperature", "flu_status")]
```

```
  temperature flu_status
1        98.1      FALSE
2        98.6      FALSE
3       101.4       TRUE
```

También puede ingresar pt_data[2:3] para extraer las columnas de temperatura y flu_status, pero enumerar las columnas por nombre da como resultado un código R claro y fácil de mantener.

Por ejemplo, para extraer el valor de la primera fila y la segunda columna (temperatura de John Doe), ingresaría:

```
pt_data[1, 2]
```

```
[1] 98.1
```

Si desea más de una fila o columna de datos, puede hacerlo especificando vectores. La siguiente declaración extraerá datos de las filas 1 y 3, y de las columnas 2 y 4:

```
pt_data[c(1, 3), c(2, 4)]
```

```
  temperature gender
1        98.1   MALE
3       101.4   MALE
```

Para extraer todas las filas o columnas, deje en blanco la parte de la fila o la columna.

```
pt_data[, 1]
```

```
[1] "John Doe"     "Jane Doe"     "Steve Graves"
```

Para extraer todas las columnas de la primera fila:

```
pt_data[1, ]
```

```
  subject_name temperature flu_status gender blood
1     John Doe        98.1      FALSE   MALE     O
```

Para extraer todo:

```
pt_data[ , ]
```

```
  subject_name temperature flu_status gender blood
1     John Doe        98.1      FALSE   MALE     O
2     Jane Doe        98.6      FALSE FEMALE    AB
3 Steve Graves       101.4       TRUE   MALE     A
```

Se puede acceder a las columnas por nombre en lugar de por posición, y se pueden usar signos negativos para excluir filas o columnas de datos. Por lo tanto, la declaración:

```
pt_data[c(1, 3), c("temperature", "gender")]
```

```
  temperature gender
1        98.1   MALE
3       101.4   MALE
```

Es equivalente a:

```r
pt_data[-2, c(-1, -3, -5)]
```

```
  temperature gender
1        98.1   MALE
3       101.4   MALE
```

Matrices y Arreglos

Para crear una matriz, simplemente suministre un vector de datos almatriz()función, junto con un parámetro que especifica el número de filas (fila)o número de columnas (ncol). Por ejemplo, para crear una matriz de 2x2 que almacene las primeras cuatro letras del alfabeto, podemos usar el nrow parámetro para solicitar que los datos se dividan en dos filas:

```r
m <- matrix(c('a', 'b', 'c', 'd'), nrow = 2)
m
```

```
     [,1] [,2]
[1,] "a"  "c"
[2,] "b"  "d"
```

Esto es equivalente a la matriz producida usando ncol = 2:

```r
 m <- matrix(c('a', 'b', 'c', 'd'), ncol = 2)
m
```

```
     [,1] [,2]
[1,] "a"  "c"
[2,] "b"  "d"
```

Con seis valores, solicitar dos filas crea una matriz con tres columnas:

```r
m <- matrix(c('a', 'b', 'c', 'd', 'e', 'f'), nrow = 2)
m
```

```
     [,1] [,2] [,3]
[1,] "a"  "c"  "e"
[2,] "b"  "d"  "f"
```

Solicitar dos columnas crea una matriz con tres filas:

```r
m <- matrix(c('a', 'b', 'c', 'd', 'e', 'f'), ncol = 2)
m
```

```
     [,1] [,2]
[1,] "a"  "d"
[2,] "b"  "e"
[3,] "c"  "f"
```

Los valores de las matrices se pueden extraer utilizando la notación [fila, columna]. Se pueden solicitar filas o columnas enteras:

```r
m[1, ]
```

```
[1] "a" "d"
```

```r
m[, 1]
```

```
[1] "a" "b" "c"
```

El arreglo es una tabla de datos multidimensional. Tiene filas, columnas y cualquier cantidad de capas adicionales de valores

Gestión de datos con R

Uno de los desafíos que se enfrentan cuando se trabaja con conjuntos de datos masivos consiste en recopilar, preparar y administrar datos de una variedad de fuentes. Esta tarea se ve facilitada por las herramientas de R para cargar datos de muchos formatos comunes.

Guardar y cargar estructuras de datos de R

Cuando haya pasado mucho tiempo obteniendo un marco de datos en particular en el formato que desea, no debería necesitar volver a crear su trabajo cada vez que reinicia su sesión de R. Para guardar una estructura de datos en particular en un archivo que pueda volver a cargarse más tarde o transferirse a otro sistema, puede usar elahorrar()función

```
> save (x, y, z, file = "mydata.RData")
```

El comando load() recreará cualquier estructura de datos ya guardada que estuviera en un fichero .RData. Para cargar el archivo mydata.RData que guardamos en el código anterior, escriba:

```
> load ("mydata.RData")
```

Si necesita terminar su sesión de R, el comando save.image() escribirá toda su sesión en un archivo de forma sencilla.

Importar y guardar datos de archivos CSV

Es muy común que los datos disponibles públicamente se almacenen en archivos de texto. Los cuales se pueden leer en cualquier computadora o sistema operativo. También se pueden exportar e importar desde/hacia programas como Microsoft Excel.

A tabular(como en "tabla") el archivo de datos está estructurado en forma de matriz, de tal manera que cada línea de texto refleja un ejemplo, y cada ejemplo tiene el mismo número de características. Los valores de características en cada línea están separados por un símbolo predefinido conocido como delimitador. A menudo, la primera línea de un archivo de datos tabulares enumera los nombres de las columnas de datos. Esto se llama un en cabezamiento línea.

El formato de archivo de texto tabular más común es el archivo de valores separados por comas (CSV), que utiliza la coma como delimitador. Un ejemplo de archivo CSV es:

```
subject_name,temperature,flue_status,gender,blood_type

Steve Graves,101.4,TRUE,MALE,A

Jane Doe,98.6,FALSE,FEMALE,AB

John Doe,98.1,FALSE,MALE,O
```

Para cargar este archivo CSV en R, se usa read.csv() de la siguiente manera:

```
pt_data<-read.csv( "usedcars.csv", stringsAsFactors = FALSE)
```

Necesitamos usar el parámetro stringsAsFactors = FALSE para evitar que R convierta todas las variables de texto en factores.

Si sus datos residen fuera del directorio de trabajo de R, puede especificar la ruta, por ejemplo, /ruta/a/misdatos.csv al llamar a la función read.csv().

R asume que el archivo CSV incluye una línea de encabezado. En caso de no tenerlo, especifique la opción header = FALSE y R asignará nombres de funciones predeterminados en la forma V1, V2, como se muestra:

```
mydata <- read.csv("usedcars.csv", stringsAsFactors = FALSE,
header = FALSE)
```

La función read.csv() es un caso especial de la función read.table() que puede leer datos tabulares en muchas formas diferentes, como el valor separado por tabuladores (TSV).

Para guardar un marco de datos en un archivo CSV, utilice la función write.csv(). Si su marco de datos se llama pt_data, simplemente ingrese:

```
write.csv(pt_data, file = "usedcars.csv")
```

Esto escribirá un archivo CSV con el nombre pt_data.csv en la carpeta de trabajo de R.

Importación de datos de bases de datos SQL

Si sus datos se almacenan en unODBC(Conectividad de base de datos abierta)sql(lenguaje de consulta estructurado) base de datos como Oracle, MySQL, PostgreSQL, Microsoft SQL o SQLite, la RODBCpaquete creado por Brian Ripleyse puede utilizar para importar estos datos directamente en un marco de datos R.

```
> install.packages ("RODBC")
```

```
> library (RODBC)
```

Abrir una conexión llamada mydb a la base de datos con el DSN my_dsn:

```
> mydb <- odbcConnect("my_dsn")
```

Si su conexión ODBC requiere un nombre de usuario y una contraseña, deben especificarse:

```
> mydb <- odbcConnect ("my_dsn", uid = "username", pwd = "pasword")
```

Podemos usar la función sqlQuery() para crear un marco de datos R a partir de las filas de la base de datos extraídas por consultas SQL. Esta función permite especificar stringsAsFactors = FALSE.

```
> patient_query <- "select * from patient_data where alive = 1"
```

```
> patient_data <- sqlQuery (channel = mydb, query = patient_query, stringsAsFactors
= FALSE)
```

La variable de datos_paciente será un marco de datos con todas las filas seleccionadas mediante la consulta SQL almacenada en consulta_paciente.

Cuando haya terminado de usar la base de datos, la conexión se puede cerrar:

```
> odbcClose(mydb)
```

Esto cerrará la conexión mydb. Aunque R cerrará automáticamente las conexiones ODBC al final de una sesión de R, es una mejor práctica hacerlo explícitamente.

Explorar y comprender los datos

Después de recopilar datos y cargarlos en estructuras de datos R, el siguiente paso en el proceso de aprendizaje automático consiste en examinar los datos en detalle. Es durante este paso que comenzará a explorar las características y los ejemplos de los datos, y se dará cuenta de las peculiaridades que hacen que sus datos sean únicos. Cuanto mejor comprenda sus datos, mejor podrá hacer coincidir un modelo de aprendizaje automático con su problema de aprendizaje.

Dado que el conjunto de datos se almacena en formato CSV, podemos usar elleer.csv()función para cargar los datos en un marco de datos R:

```
usedcars <- read.csv("usedcars.csv", stringsAsFactors = FALSE)
```

La exploración de datos es un proceso fluido, los pasos se pueden imaginar como una especie de investigación en la que se responde a preguntas sobre los datos, estos pasos básicos de esta investigación se deberian aplicar a cualquier conjunto de datos que desee.

Explorando la estructura de los datos

Su investigación debe ser sobre cómo se organizan los datos. Si tiene suerte, su fuente le proporcionará unaDiccionario de datos, un documento que describe las características de los datos. En nuestro caso, los datos del coche usado no vienen con esta documentación, por lo que tendremos que crear la nuestra.\

El cadena () La función proporciona un método para mostrar la estructura de un marco de datos, o cualquier estructura de datos R, incluidos vectores y listas. Se puede utilizar para crear el esquema básico de nuestro diccionario de datos:

Para un comando tan simple, aprendemos una gran cantidad de información sobre el conjunto de datos. El número de observaciones a menudo se abrevia simplemente como n.

La declaración de 6 variables se refiere a las seis características que se registraron en los datos. Estas características se enumeran por nombre en líneas separadas. Mirando la línea de la característica llamada color, notamos algunos detalles adicionales:

```
$color : chr "Yellow" "Gray" "Silver" "Gray" ...
```

Después del nombre de la variable, chr nos dice que la función es de tipo carácter. En este conjunto de datos, tres de las variables son de carácter, mientras que tres se indican como int. Aunque este conjunto de datos incluye solo variables de caracteres y enteros, también es

probable que encuentre num cuando utilice datos que no sean enteros. Cualquier factor se enumeraría como Tipo de factor.

A veces, los conjuntos de datos tienen características con nombres y códigos sin sentido o simplemente un número como V1. Puede ser necesario realizar investigaciones adicionales para determinar qué representa realmente una característica.

Exploración de variables numéricas

Para investigar las variables numéricas de los datos, la función summary() muestra varios estadísticos de resumen comunes.

```
summary(usedcars$year)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   2000    2008    2009    2009    2010    2012
```

También podemos utilizar summary() para obtener estadísticas de resumen para varias variables numéricas al mismo tiempo:

```
summary(usedcars[c("price", "mileage")])
```

```
     price          mileage
 Min.   : 3800   Min.   :  4867
 1st Qu.:10995   1st Qu.: 27200
 Median :13592   Median : 36385
 Mean   :12962   Mean   : 44261
 3rd Qu.:14904   3rd Qu.: 55124
 Max.   :21992   Max.   :151479
```

Los seis estadísticos de resumen que proporciona la función summary() son herramientas sencillas, pero potentes para investigar los datos. Los estadísticos de resumen pueden dividirse en dos tipos: medidas de centro y medidas de dispersión.

Medición de la tendencia central: media y mediana

Las medidas de tendencia central son una clase de estadísticas usadas para identificar un valor que se encuentra en el medio de un conjunto de datos. Cuando algo se considera promedio, cae en algún lugar entre los extremos de la escala. En estadística, el promedio también se conoce como la media, una medida definida como la suma de todos los valores dividida por el número de valores.

```
(36000 + 44000 + 56000) / 3
```

```
[1] 45333.33
```

R también proporciona una función mean(), que calcula la media de un vector de números:

```
mean(c(36000, 44000, 56000))
```

```
[1] 45333.33
```

Aunque la media es la estadística más citada para medir el centro de un conjunto de datos, no siempre es la más adecuada. Otra medida de tendencia central de uso común es la mediana, que es el valor que se encuentra en la mitad de una lista ordenada de valores. R proporciona una función mediana():

```
median(c(36000, 44000, 56000))
```

```
[1] 44000
```

Nota: Si un conjunto de datos tiene un número par de valores, no hay un valor medio. En este caso, la mediana suele calcularse como el promedio de los dos valores en el centro de la lista ordenada.

La media y la mediana se ven afectadas de manera diferente por los valores que caen en los extremos del rango. La media es muy sensible a los valores atípicos (altos o bajos), por lo que es más probable que se desplace hacia arriba o hacia abajo por un pequeño número de valores extremos.

Medición de la dispersión: los cuartiles y el resumen de cinco números

Para medir la diversidad, necesitamos emplear otro tipo de estadísticas resumidas que\

se ocupan de ladesparramarde los datos, o qué tan apretado o suelto están espaciados los\

valores. Conocer la dispersión proporciona una idea de los máximos y mínimos de los datos, y si la mayoría de los valores son similares o diferentes a la media y la mediana.\

Elresumen de cinco numeroses un conjunto de cinco estadísticas que representan aproximadamente la dispersión de un conjunto de datos. Las cinco estadísticas se incluyen en el resultado delresumen() función. Escritos en orden, son:

1. Mínimo (Min.)
2. Primer cuartil, o Q1 (1er Qu.)
3. Mediana, o Q2 (Median)
4. Tercer cuartil, o Q3 (3rd Qu.)
5. Máximo (Max.)

El mínimo y el máximo son los valores más extremos del conjunto de datos, para encontrar estos valores se proporciona las funciones min() y max().

En R la función range() devuelve tanto el valor mínimo como el máximo. Combinando range() con la función de diferencia, diff() permite examinar el rango de datos con un solo comando:

```
range(usedcars$price)
```

```
[1]  3800 21992
```

```
diff(range(usedcars$price))
```

```
[1] 18192
```

El primer y tercer cuartil, Q1 y Q3, se refieren al valor por debajo o por encima del cual se encuentra una cuarta parte de los valores. Junto con la mediana (Q2), los cuartiles dividen un conjunto de datos en cuatro partes, cada una con el mismo número de valores.

El 50% medio de los datos entre Q1 y Q3 es de especial interés porque es una medida sencilla de la dispersión. La diferencia entre Q1 y Q3 se conoce como rango intercuartílico (IQR), y puede calcularse con la función IQR():

```r
IQR(usedcars$price)
```

```
[1] 3909.5
```

La función quantile() es una herramienta para identificar cuantiles para un conjunto de valores, por defecto, devuelve el resumen de cinco números. Permite especificar entre nueve algoritmos diferentes especificando el parámetro type.

```r
quantile(usedcars$price)
```

```
    0%     25%     50%     75%    100% 
3800.0 10995.0 13591.5 14904.5 21992.0 
```

Si especificamos un parámetro probs adicional con un vector que denota puntos de corte, podemos obtener cuantiles arbitrarios:

```r
quantile(usedcars$price, probs = c(0.01, 0.99))
```

```
     1%      99% 
5428.69 20505.00 
```

La función de secuencia seq() se utiliza para generar vectores de valores espaciados uniformemente.

```r
quantile(usedcars$price, seq(from = 0, to = 1, by = 0.20))
```

```
    0%    20%    40%    60%    80%   100% 
3800.0 10759.4 12993.8 13992.0 14999.0 21992.0 
```

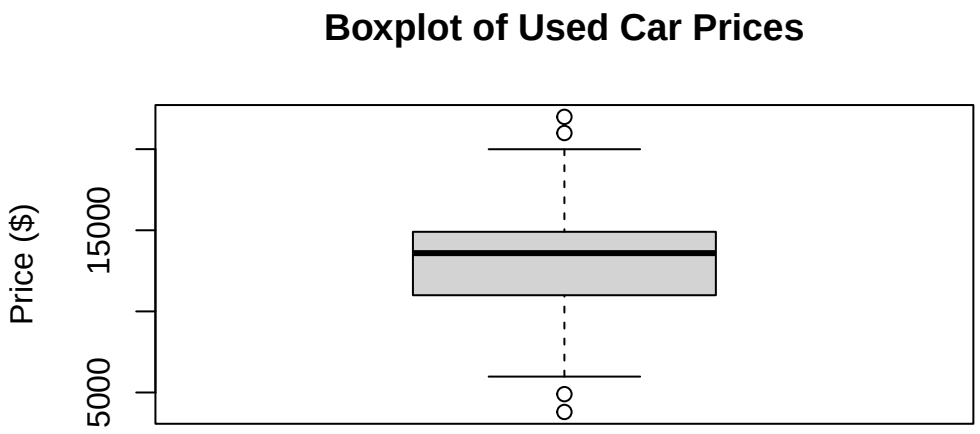
Una vez comprendido el resumen de cinco números, podemos volver a examinar.

Visualización de variables numéricas: diagramas de caja

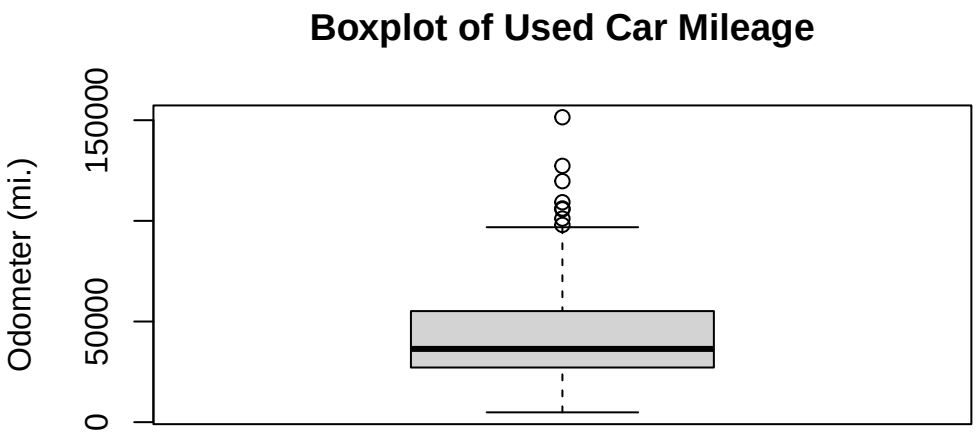
La visualización de variables numéricas puede ser útil para diagnosticar muchos problemas con los datos. El gráfico de caja muestra el centro y la dispersión de una variable numérica en un formato que le permite obtener rápidamente una idea del rango y el sesgo de una variable, o compararla con otras variables.

Para obtener un diagrama de caja para una variable, usaremos la función boxplot(). Se especifican un parámetros adicionales, main e ylab, para agregar el título a la figura y etiquetar el eje y:

```
boxplot(usedcars$price, main="Boxplot of Used Car Prices",
ylab="Price ($)")
```




```
boxplot(usedcars$mileage, main="Boxplot of Used Car Mileage",
ylab="Odometer (mi.)")
```



El diagrama de caja y patillas representa los valores de resumen de cinco números usando líneas horizontales. Las líneas horizontales que forman el cuadro en el medio de cada figura representan Q1, Q2 y Q3 cuando se lee el gráfico de abajo hacia arriba.

Nota: En diagramas de caja simples, el ancho de la caja y los bigotes es arbitrario y no ilustra ninguna característica de los datos. Es posible utilizar la forma y el tamaño de los recuadros para facilitar las comparaciones de los datos entre varios grupos.

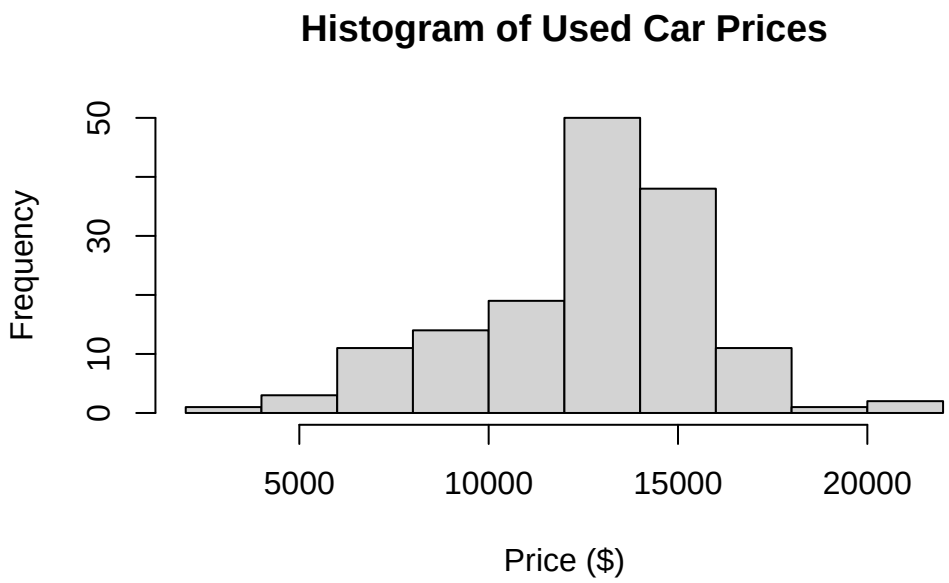
El mínimo y el máximo se ilustran con los bigotes que se extienden por debajo y por encima de la caja; sin embargo, es una convención permitir que los bigotes se extiendan hasta un mínimo o un máximo de 1,5 veces el IQR por debajo de Q1 o por encima de Q3. Los valores que superan este umbral se consideran valores atípicos y se indican como círculos o puntos.

Visualización de variables numéricas – Histogramas

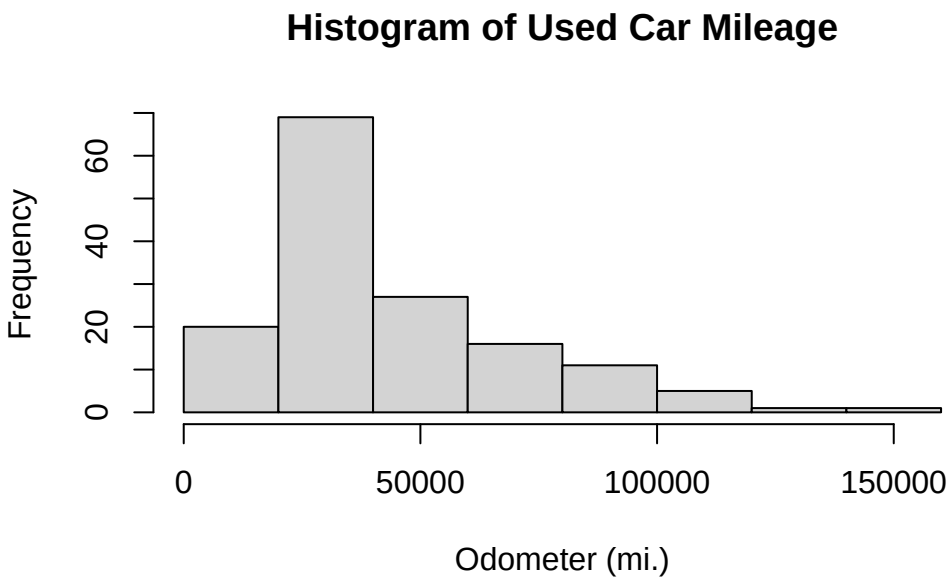
Un histograma representa gráficamente la dispersión de una variable numérica. Divide los valores de la variable en un número predefinido de porciones que actúan como contenedores de valores. Utiliza cualquier número de contenedores de idéntico ancho, pero permite estos contengan diferentes números de valores.

Podemos crear un histograma usando la función hist():

```
hist(usedcars$price, main = "Histogram of Used Car Prices",
xlab = "Price ($)")
```




```
hist(usedcars$mileage, main = "Histogram of Used Car Mileage",
xlab = "Odometer (mi.)")
```



El histograma se compone de una serie de barras con alturas que indican la frecuencia de los valores que caen dentro de cada uno de los contenedores. Las líneas verticales que separan las barras, (eje horizontal), indican los puntos inicial y final del rango de valores del contenedor.

Los histogramas de datos sesgados se ven estirados en uno de los lados:

La capacidad de diagnosticar rápidamente dichos patrones en nuestros datos es uno de los puntos fuertes del histograma como herramienta de exploración de datos.

Comprensión de datos numéricos - distribuciones uniforme y distribuciones normales

Los histogramas, los gráficos de caja y los estadísticos que describen el centro y la dispersión permiten examinar la distribución de los valores de una variable.

Si todos los valores tienen la misma probabilidad de ocurrir, se dice que la distribución es uniforme. Una distribución uniforme es fácil de detectar con un histograma porque las barras tienen aproximadamente la misma altura.

Es importante tener en cuenta que no todos los sucesos aleatorios son uniformes.

Medición de la dispersión: varianza y desviación estándar

Las distribuciones nos permiten caracterizar una gran cantidad de valores utilizando una menor cantidad de parámetros. La distribución normal se puede definir con dos datos: centro y dispersión. El centro se define por su valor medio y la dispersión se mide mediante la desviación estándar.

Para calcular la desviación estándar, primero obtenemos la varianza, (promedio de las diferencias al cuadrado entre cada valor y el valor medio). En notación matemática, la varianza se define mediante la siguiente fórmula:

$$\text{Var}(X) = \sigma = \frac{1}{norte}\sum_{norte}^{i=1}(x_i - \mu)^2$$

La desviación estándar:

$$\text{DesvStd}(X) = \sigma = \sqrt{\frac{1}{norte}\sum_{norte}^{i=1}(x_i - \mu)^2}$$

Para obtener la varianza y la desviación estándar en R se pueden utilizar las funciones var() y sd()

```
var(usedcars$price)
```

```
[1] 9749892
```

```
sd(usedcars$price)
```

```
[1] 3122.482
```

```
var(usedcars$mileage)
```

```
[1] 728033954
```

```
sd(usedcars$mileage)
```

```
[1] 26982.1
```

Al interpretar la varianza, los números más grandes indican que los datos se distribuyen más ampliamente alrededor de la media. La desviación estándar indica, en promedio, cuánto difiere cada valor de la media.

La desviación estándar se puede usar para estimar rápidamente qué tan extremo es un valor dado bajo el supuesto de que proviene de una distribución normal.

Exploración de variables categóricas

El conjunto de datos de coches usados tenía tres variables categóricas: modelo, color y transmisión. Como utilizamos el parámetro stringsAsFactors = FALSE al cargar los datos, R las ha dejado como variables de carácter (chr) en lugar de convertirlas automáticamente en factores. La función table() puede utilizarse para generar tablas unidireccionales

```
table(usedcars$year)
```

```
2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010 2011 2012
   3    1    1    1    3    2    6   11   14   42   49   16    1
```

```
table(usedcars$model)
```

```
 SE SEL SES
 78  23  49
```

```
table(usedcars$color)
```

```
 Black   Blue   Gold   Gray  Green    Red Silver  White Yellow
    35     17      1     16      5     25     32     16      3
```

La salida de la tabla enumera las categorías de la variable nominal y un recuento del número de valores que entran en esa categoría.

```
model_table <- table(usedcars$model)
prop.table(model_table)
```

```
       SE       SEL       SES
0.5200000 0.1533333 0.3266667
```

Supongamos que queremos mostrar los resultados en porcentajes con un solo decimal:

```
color_table <- table(usedcars$color)
color_pct <- prop.table(color_table) * 100
round(color_pct, digits = 1)
```

```
 Black   Blue   Gold   Gray  Green    Red Silver  White Yellow
  23.3   11.3    0.7   10.7    3.3   16.7   21.3   10.7    2.0
```

Medición de la tendencia central: la moda

En términos estadísticos, la moda de una característica es el valor que ocurre con más frecuencia (medida de tendencia central). Se usa para datos categóricos, ya que la media y la mediana no están definidas para variables nominales.

Una variable puede tener más de una moda; una variable con una moda es unimodal, con dos modas es bimodal. Los datos que tienen múltiples modos se denominan multimodales.

Los modos se utilizan en un sentido cualitativo para obtener una comprensión de los valores importantes en un conjunto de datos. Sin embargo, sería peligroso poner demasiado énfasis en la moda ya que el valor más común no es necesariamente una mayoría.

Pensar en las modas como valores comunes nos permite aplicar el concepto de moda estadística a los datos numéricos. Sería poco probable tener una moda para una variable continua, ya que es posible que no se repitan dos valores. Puede ser útil considerar la moda cuando se exploran datos numéricos, particularmente para examinar si los datos son multimodales o no.

Exploración de las relaciones entre variables

Se han examinado las variables de una en una, calculando sólo estadísticas univariantes. Durante nuestra investigación, nos planteamos preguntas que no pudimos responder en ese momento:

- ¿Implican los datos de precios que estamos examinando sólo coches de clase económica, o también hay coches de lujo con mucho kilometraje?
- ¿Permiten las relaciones entre el modelo y el color determinar el tipo de coche que estamos examinando?

Este tipo de preguntas pueden abordarse analizando la relación entre dos variables. Las relaciones de más de dos variables se denominan relaciones multivariantes.

Examen de las relaciones - bidireccionales tabulaciones cruzadas

Para examinar una relación entre dos variables nominales, se utiliza una tabulación cruzada bidireccional.

Una tabulación cruzada es similar a un gráfico de dispersión en el sentido de que permite examinar cómo varían los valores de una variable por los valores de otra.

Existen varias funciones para producir tablas bidireccionales en R, incluida table(), la función CrossTable() es quizás la más fácil de usar, ya que presenta los porcentajes de fila, columna y en una sola tabla.

```
> install.packages("gmodels")
```

Simplifiquemos nuestro proyecto reduciendo el número de niveles de la variable color. Esta variable tiene nueve niveles, pero realmente no necesitamos tanto detalle.

Crearemos una variable indicadora binaria, la cual indica si el color del coche es conservador o no, su valor será 1 si es verdadero, 0 en caso contrario.

```
usedcars$conservative <-
usedcars$color %in% c("Black", "Gray", "Silver", "White")
```

Puede que haya notado un nuevo comando aquí: el operador %in% devuelve TRUE o FALSE para cada valor en el vector a la izquierda del operador. En términos sencillos, puede traducir esta línea como "¿está el color del coche usado en el conjunto de negro, gris, plata y blanco?".

```
table(usedcars$conservative)
```

```
FALSE  TRUE
   51    99
```

Veamos ahora una tabulación cruzada para ver cómo varía la proporción de coches de color varía según el modelo, trataremos conservador como la variable dependiente (y).

El comando CrossTable() es por lo tanto:

```
> CrossTable(x=usedcars$model, y=usedcars$conservative)
```

Hay una gran cantidad de datos en la salida de CrossTable(). La leyenda en la parte superior indica cómo interpretar cada valor. Las proporciones indican la proporción de esa celda en relación con la estadística Chi-cuadrado, el total de la fila, el total de las columnas y el total de la tabla.

Los valores de Chi-cuadrado se refieren a la contribución de la celda en la prueba de Chi-cuadrado de Pearson para la independencia entre dos variables. Esta prueba mide la probabilidad de que la diferencia en los recuentos de celdas en la tabla se deba únicamente al azar. Si la probabilidad es muy baja, proporciona una fuerte evidencia de que las dos variables están asociadas.

Puede obtener los resultados de la prueba de chi-cuadrado agregando un parámetro adicional que especifique chisq = TRUE al llamar a la función CrossTable()