Capitulo 4

Daniel Villatoro

23/1/2020

#Trabajando con datos externos

Uno de los grandes ayudas de aprender R es construir varios ejemplos de datos, internet representa un invremento importante de fuentes para datasets largos, y ofrecen una corta pero util introduccion, a algunas maneras de como podemos obtener en internet los datos, en internet y meterlos a una sesion de R.

Uno de los mas populares y convenientes formatos de datos son los CVS esos archivos son facilmente confundidos con hojas de dispersion, los cuales no son de todo,

Uno de los formatos de archivo de datos más populares y convenientes es el archivo de valores separados por comas (CSV), estos archivos se confunden fácilmente con hojas de cálculo, que no son lo mismo y requieren diferentes utilidades para acceder; estas utilidades son fácilmente disponibles, pero es importante reconocer que no son los mismos que los necesarios para leer y escribir archivos CSV.

Existen muchos otros tipos de archivos y, aunque este libro no intenta considerarlos a todos, algunos otros tipos de archivos clave y sus usos principales, incluidos los archivos de texto simples, un formato especial para guardar y recuperar objetos R arbitrarios (archivos RDS), y algunos de los formatos de archivos gráficos más populares utilizados para guardar la salida gráfica para su uso por otros programas (por ejemplo, para preparar documentos o presentaciones de diapositivas). Una cuestión práctica importante que surge con frecuencia es la necesidad de fusionar datos

#Manejo de Archivos en R Por default todas estas funciones usadas para leer datos de manera externa dentro de nuestra sesion interactiva de R asume que esos archivos existen dentro de un directorio de trabajo y la funcion getwd() se utiliza para identificar dicho directorio, esta funcion regresa un texto string de esos identificadores en el directorio donde R podria apuntar a esos archivos para leer las funciones como read.csv o readLInes, es importante que la nota de los directorios donde esta arraigados en una estructura jerarquica y de los diferentes niveles son especificados en el nombre del directorio

getwd()

setwd("C:/NewDirectory")

La funcion list.files cuando es llamada sin parametros regresa una lista de todos los archivos contenidos en el directorio de trabajo, y posiblemente la lista de archivos con directorios alternativos, usando la opcion path para especificar el directorio designado o la lista de solo esos archivos que contienen caracteres strings con sus nombres por via pattern option

list.files(path = "C:/Users/Ron/Documents/IntroRbook/ExploringDataCode",pattern = "Three")

Aqui el path especifica los directorios alternativos que buscaremos, ,iemtras que pattern especifica que solo esos archivos con nombres contenidos con el string Three son regresados.

list.files("../ExploringDataCode", pattern = "Three")

La funcion file.info nos permite aprender detalles mas alla de los archivos esto es el nombre, incluyendo el tamaño, si se encuentra en el directorio o no si es ejecutable o no si es un archivo con permisos de leer y escribir y cuando fue creado o modificado

file.info("../ExploringDataCode/ThreeSigma.R")

Nosotros podemos crear y editar archivos usando las funciones file.create y file.edit, por que esas funciones son muy utiles para crear y ediatr en R algunos archivos de programa, finalmente los archivos pueden ser renombrados copiados o borrados con las funciones file.rename,file.copy y file.remove.

#Entrada Manual de datos

La entrada manual de datos es tediosa y propenssa a errores esto debe ser evitado de cualquier manera ya que es facil caprurar mal los numeros o caracteres, escribir campos incorrectos o brincarse entradas enteras.

#Entrada Manual de datos es mala pero aveces conveniente

La entrada manual de datos, avceces es necesaria y es una practica alternativa pero en general es una mala practica, por dos razones, la primera y las mas obia es el dataset que es demasiado extenso y el requerir una entrada manual puede ser inseguro, la segunda razon por que tiene a ser un error propenso con errores, esos errores pueden tomar muchas formas y en casos infortunados puedes ser muy dificl detectarlos, estos son unos ejemplos especificos:

Omitir digitos de datos numericos

Transponer digitos en datos numericos

Omitir o transponer cracteres de strings con caracteres largos

Omitir uno o varios campos de el registro

Capturar entradas correctas de valores en campos erroneos

Brincarse registos

Por ambas razones, es mejor evitar el ingreso manual de datos siempre que sea posible, especialmente si hay más de unos pocos registros involucrados. Por el contrario, dado que alguien ingresa inicialmente muchos datos manualmente al estar incorporado en un archivo o base de datos, también es extremadamente importante examinar los datos cuidadosamente para detectar cualquier error de calidad de datos que pueda estar presente.

#Interactuando con internet

En el otro extremo del espectro de entrada de datos, Internet es una fuente cada vez más importante de grandes volúmenes de datos. En los casos más simples, Internet proporciona enlaces a archivos que contienen los datos que queremos, lo que nos permite obtener datos siguiendo los tres pasos:

Encontrar una pagina web con un data set disponible

Entrar a links apropiados de descarga de datos y en el formato que ocupas

Leer el resultado de el archivo de datos en tu sesion de R.

#Trabajando con archivos CSV Uno de los mas convenientes formatos para intercambiar datos entre diferentes softwares de desarrollo es evalor coma separado o archivo CSV el cual esta organizado en filas, y cada fila con un registro y campo por separado por comas, esta diferente habilidad en el software de desarroll de escritura y lectura de archivos CSV es importante por que muy a menudo los datasets son colectados procesados y analizados por una cadena distinta de personas y organizaciones frecuentemente usado por diferentes utilidades de software, en casos favorables los archivos CSV pueden ser comunmente un conducto de datos en cada estado de esta cadena, esos archivos pueden tener sus limitaciones.

#Leyendo y escribiendo archivos CSV

En R una manera simple de leer y escribir archivos CSV es usando la funcion read.csv y write.csv respectivamente, se rquiere un parametro para ambas funciones read.csv y write.csv respectivamente, se requieren parametros para ambas funciones y es el nombre del archivo que sera leido desde done se encuentra o escrito, este nombre de archivo debe ser una cadena de caracteres que corresponda, por defecto a un archivo en el directorio de trabajo actual.

Por el contrario, es posible leer o escribir en archivos en otros directorios incluyendo la designación de ruta completa en el archivo nombre. Para la función read.csv, este nombre de archivo es el único parámetro requerido, aunque la función admite muchos parámetros opcionales.

La función write.csv requiere dos argumentos: la matriz o el marco de datos contener los datos que se escribirán y una cadena de caracteres que especifica el nombre de el archivo en el que se escribirán los datos. Por defecto, los datos se escriben en un archivo CSV en el directorio de trabajo actual, pero los datos se pueden escribir en archivos en otros directorios especificando una ruta completa.

Hojas de calculo y archivos csv no son la misma cosa

Si bien es cierto que el paquete de software de Excel puede leer y escribir archivos CSV simples,a continuacion tres caracteristicas que hacen la diferencia:

1. un archivo CSV es el archivo de datos simple descrito anteriormente, que puede leerse Microsoft Excel, junto con muchos otros programas, incluidos R, Python, y Microsoft Access, por nombrar solo algunos;
2. El programa de hoja de cálculo de Microsoft Excel es un paquete de software que puede hacer muchas cosas, incluyendo leer y escribir archivos CSV, realizan cómputos simples presentaciones y análisis de datos.
3. Un archivo de datos de Microsoft Excel contiene los datos sobre los cuales la hoja de cálculo se basa, junto con mucha información adicional, que incluye formato de visualización, el código requerido para cálculos internos, etc.

En entornos de software como R o Python, la principal diferencia práctica es entre un archivo CSV y un archivo de datos de hoja de cálculo es que las utilidades simples como la función read.csv, puede funcionar con archivos CSV, pero no con archivos de datos de Microsoft Excel u otros archivos de datos de hoja de cálculo, que son organizados de manera diferente

#Problemas potenciales con los archivos CSV

Los archivos CSV no están exentos de dificultades, primero, la posibilidad que la función read.csv puede cambiar los nombres de las variables, y segundo problema que puede surgir al leer archivos CSV que contienen cadenas de caracteres con comas incrustadas

#Trabajando con archivos de texto La funcion basica de R para leer archivos er readLines y esa funcion tiene diferentes funciones, la funcion especificamente puede ser usada para leer datos HTML y puede ser usada para examinar los nombres originales de las variables en archivos CSV para detectar la precencias de comas en el archivo.

La funcion nchar nos dice cuantos caracteres tiene el primer registro y el substr nos arroja una parte del regristro en dos componentes, el primero consiste en 1 atravez de 56 y el segundo consiste en los caracteres 57 atravez de los 84

autoMpgRecords <- readLines("UCIautoMpg.txt") x <- autoMpgRecords[1] nchar(x) ## [1] 84 substr(x, 1, 56) ## [1] "18.0 8 307.0 130.0 3504. 12.0 70 1" substr(x, 57, 84)

El simbolo á̄l principio de la segunda parte del regustro es un caracter tab y aqui divide en dos partes el registro el de la izquierda es es una secuencua de ocho numeros y el segundo representa los nombres de los vehiculos especificos, los nombres de todas esas variables estan disponibles dentro del UCI machine learning Repository.