Other Data Sources

Jim Harner

10/4/2020

2.7 Other Data Sources

This section covers more specialized sources. First, we cover importing spreadsheets since they are the *lingua franca* of BI. Then we cover dates and times and finally foreign formats such as SAS and SPSS.

2.7.1 Spreadsheets

Spreadsheets are widely used as a storage option. Microsoft Excel is commonly used spreadsheet software. The corresponding spreadsheet format is Microsoft Excel workbook.

Spreadsheet formats

Previously Excel workbooks used a binary format, XLS, which was difficult to decode. Now Excel workbooks are stored in an XML-based format called Open Office XML (OOXML). It is not clear how open OOXML is.

The main difference between these two file extensions is that the XLS is created on the version of Excel prior to 2007 while XLSX is created on the version of Excel 2007 and onward. XLS is a binary format while that XLSX is Open XML format.

Open Office Calc uses an XML-based standard format called Open Document Format (ODF).

Neither OOXML or ODF are good for storing data since a lot of data is stored in the spreadsheet about how to display the data, how to calculate cells, etc. This information is not relevant to the actual data. Also, the number of rows and columns is limited, so Excel is not useful for "big data."

Spreadsheet software

Spreadsheets displays a data set in a rectangular grid of cells. It has the benefits of fixed-width format text files. Spreadsheets have tools to manipulate data, but they are limited. Formulas can also be used.

The point-and-click interface of spreadsheets does not allow steps to be recorded although macros are available.

The function `read_excel` in the R package `readxl` can read both Excel 97–2004 files and Excel 2007+ files with the `read_xls` and `read_xlsx` functions, respectively. The `read_excel` auto detect the format from the file extension.

```r
library("readxl", warn.conflicts=F)
nemo_xls <- read_excel("pointnemotemp.xlsx",
                       col_names = c("date", "temp"),
                       col_types = c("date", "numeric"))
head(nemo_xls)
```

```
## # A tibble: 6 x 2
##   date                 temp
##   <dttm>              <dbl>
## 1 1994-01-16 00:00:00  279.
## 2 1994-02-16 00:00:00  280
## 3 1994-03-16 00:00:00  279.
## 4 1994-04-16 00:00:00  279.
## 5 1994-05-16 00:00:00  278.
## 6 1994-06-16 00:00:00  276.
```

Specific Excel sheets can be read with the `sheet` argument, either by sheet name or by sheet position. Cell ranges can be read with the `range` argument. See the documentation for other options.

Multiple files cannot be easily handled. Metadata and file names differentiate them and thus programming is difficult. This can be handled, but relational databases are more efficient.

2.7.2 Dates and Times (lubridate)

Date-time data can be frustrating to work with in R. R commands for date-times are generally unintuitive and change depending on the type of date-time object being used. Moreover, the methods we use with date-times must be robust to time zones, leap days, daylight savings times, and other time related quirks, and R lacks these capabilities in some situations.

The `base` R methods use `as.Date` to convert a character string to a date format.

```
nemotemp_df <- read.table("pointnemotemp.txt", skip=8,
                        colClasses=c("character", "NULL", "NULL", "NULL", "numeric"),
                        col.names=c("date", "", "", "", "temp"))
nemotemp_df$date <- as.Date(nemotemp_df$date, format = "%d-%b-%Y")
str(nemotemp_df)
```

```
## 'data.frame':    93 obs. of  2 variables:
##  $ date: Date, format: "1994-01-16" "1994-02-16" ...
##  $ temp: num  279 280 279 279 278 ...
```

The R package `lubridate` makes it easier to do the things R does with date-times and possibly to do things R does not. Specifically, lubridate provides:

- a set of intuitive date-time related functions that work the same way for all common date-time classes (including those from `chron`, `timeDate`, `zoo`, `xts`, `its`, `tis`, `timeSeries`, `fts`, and `tseries`)
- quick and easy parsing of date-times: `ymd()`, `dmy()`, `mdy()`, . . .
- simple functions to extract and modify components of a date-time, such as years, months, days, hours, minutes, and seconds: `year()`, `month()`, `day()`, . . .
- helper functions for handling time zones: `with_tz()`, `force_tz()`

```
library(readr)
library(lubridate)
nemotemp_tbl <- read_table("pointnemotemp.txt", skip=8,
                        col_types = c("c---d"),
                        col_names = c("date", "temp"))
nemotemp_tbl$date <- as_date(nemotemp_tbl$date, format = "%d-%b-%Y")
nemotemp_tbl
```

Lubridate also expands the type of mathematical operations that can be performed with date-time objects.

- durations, which measure the exact amount of time between two points
- periods, which accurately track clock times despite leap years, leap seconds, and day light savings time
- intervals, a summary of the time information between two points

2.7.3 Foreign Data Formats (Haven)

The R package haven allows you to load foreign data formats (SAS, SPSS and Stata) into R by wrapping the ReadStat C library. Haven offers similar functionality to the base `foreign` package, but it:

- reads SPSS files (`.dta` and `.por`), reads Stata 13 and 14 files (foreign only works up to Stata 12), and SAS's proprietary binary format (`sas7bdat`);
- writes SPSS, Stata, and SAS files;
- converts date-times to corresponding R classes;
- returns labelled vectors as a new labelled class.

All functions return tibbles.