A Minimal Book Example

Yihui Xie

2022-09-25

Contents

Chapter 1

Prerequisites

This is a *sample* book written in **Markdown**. You can use anything that Pandoc's Markdown supports, e.g., a math equation $a^2 + b^2 = c^2$.

The **bookdown** package can be installed from CRAN or Github:

```
install.packages("bookdown")
# or the development version
# devtools::install_github("rstudio/bookdown")
```

Remember each Rmd file contains one and only one chapter, and a chapter is defined by the first-level heading `#`.

To compile this example to PDF, you need XeLaTeX. You are recommended to install TinyTeX (which includes XeLaTeX): https://yihui.name/tinytex/.

Chapter 2

Tabular Data Representation - DataTable

2.1 DataTable Schema

Most data science programming languages derive the DataTable schema from the data source. This is easy when the data source itself has a schema, such as a SQL table, or the data source itself is consistent and complete, such as a CSV file with no missing field in it's first row.

In real world, we often work with imperfect data so a defining the schema upfront is preferred.

Example The FTX exchange provides a RESTful API for users to retrieve balance of all tokens in their wallet. When the wallet has at least one token with non-trivial balance, the response message body like the following:


```
{
    "success": true,
    "result": [
        {
            "coin": "ETH",
            "total": -0.05008361,
            "free": 0.60553512,
            "availableForWithdrawal": 0.60553512,
            "availableWithoutBorrow": 0,
            "usdValue": -65.54655549160535,
            "spotBorrow": 0.05008361
        }
```

```
    ]
}
```

We can easily convert this JSON object into a DataTable in python:

```
import requests
from pandas import DataFrame

resp = requests.get("https://ftx.com/api/wallet/balances") # ignore authentication
assert resp.status_code == 200, f"Request failed {resp.text}"
df = DataFrame(resp.json()["result"])
```

However, if the wallet is empty, the response body becomes:

```
{
    "success": true,
    "result": []
}
```

With this empty response, the python script above would generate an empty DataFrame with completely different schema. The proper way to set schema is to explicitly specify the column names and types:

```
responsJson = []  # assume we got empty data from the API response
I = lambda x: x.split(",")
df = DataFrame(
    responsJson,
    columns=I("coin,total,free,availableForWithdrawal,availableWithoutBorrow,usdValue,s
)
for col in I("total,free,availableForWithdrawal,availableWithoutBorrow,usdValue,spotBo
  df[col] = pd.to_numeric(df[col])
```

As you can see, this is verbose. Even worse, the logic is implemeted in two steps:

- step 1, set column names and load data;
- step 2, set schema for each column (and cast column types if necessary).

This is un-natural. The better way to implement this would be:

- step 1, set column names and types;
- step 2, load data.

Chapter 3

Introduction

You can label chapter and section titles using `{#label}` after them, e.g., we can reference Chapter 3. If you do not manually label them, there will be automatic labels anyway, e.g., Chapter 5.

Figures and tables with captions will be placed in `figure` and `table` environments, respectively.

```
par(mar = c(4, 4, .1, .1))
plot(pressure, type = 'b', pch = 19)
```

Reference a figure by its code chunk label with the `fig:` prefix, e.g., see Figure 3.1. Similarly, you can reference tables generated from `knitr::kable()`, e.g., see Table 3.1.

```
knitr::kable(
  head(iris, 20), caption = 'Here is a nice table!',
  booktabs = TRUE
)
```

You can write citations, too. For example, we are using the **bookdown** package (Xie, 2022) in this sample book, which was built on top of R Markdown and **knitr** (Xie, 2015).

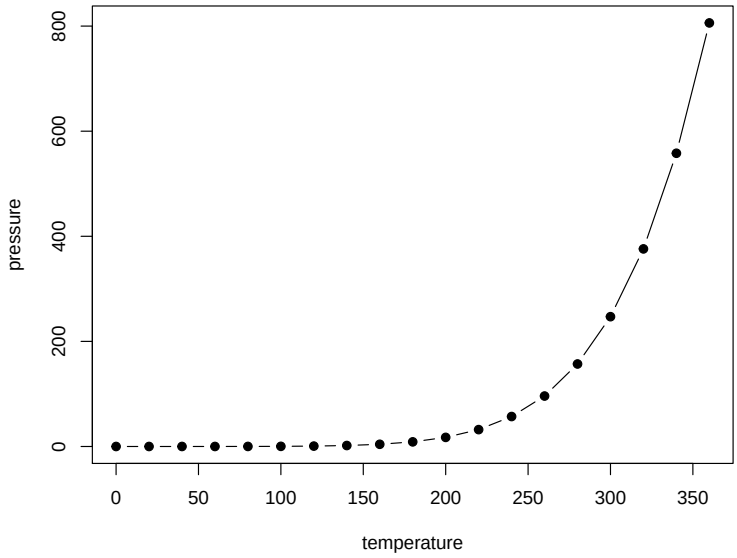

Figure 3.1: Here is a nice figure!

Table 3.1: Here is a nice table!

Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
5.1	3.5	1.4	0.2	setosa
4.9	3.0	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5.0	3.6	1.4	0.2	setosa
5.4	3.9	1.7	0.4	setosa
4.6	3.4	1.4	0.3	setosa
5.0	3.4	1.5	0.2	setosa
4.4	2.9	1.4	0.2	setosa
4.9	3.1	1.5	0.1	setosa
5.4	3.7	1.5	0.2	setosa
4.8	3.4	1.6	0.2	setosa
4.8	3.0	1.4	0.1	setosa
4.3	3.0	1.1	0.1	setosa
5.8	4.0	1.2	0.2	setosa
5.7	4.4	1.5	0.4	setosa
5.4	3.9	1.3	0.4	setosa
5.1	3.5	1.4	0.3	setosa
5.7	3.8	1.7	0.3	setosa
5.1	3.8	1.5	0.3	setosa

Chapter 4

Literature

Here is a review of existing methods.

Chapter 5

Methods

We describe our methods in this chapter.

Math can be added in body using usual syntax like this

5.1 math example

p is unknown but expected to be around 1/3. Standard error will be approximated

$$SE = \sqrt{(\frac{p(1-p)}{n})} \approx \sqrt{\frac{1/3(1-1/3)}{300}} = 0.027$$

You can also use math in footnotes like this[1].

We will approximate standard error to 0.027[2]

[1] where we mention $p = \frac{a}{b}$

[2] p is unknown but expected to be around 1/3. Standard error will be approximated

$$SE = \sqrt{(\frac{p(1-p)}{n})} \approx \sqrt{\frac{1/3(1-1/3)}{300}} = 0.027$$

Chapter 6

Applications

Some *significant* applications are demonstrated in this chapter.

6.1 Example one

6.2 Example two

Chapter 7

Final Words

We have finished a nice book.

Bibliography

Xie, Y. (2015). *Dynamic Documents with R and knitr*. Chapman and Hall/CRC, Boca Raton, Florida, 2nd edition. ISBN 978-1498716963.

Xie, Y. (2022). *bookdown: Authoring Books and Technical Documents with R Markdown*. R package version 0.28.