Introduction to Applied Science

Rex W. Douglass

11/4/22

Table of contents

Preface

This is a Quarto book.

To learn more about Quarto books visit https://quarto.org/docs/books.

```
1 + 1
```

```
[1] 2
```

1 Introduction

This is a book created from markdown and executable code.

See (**knuth84?**) for additional discussion of literate programming.

```
1 + 1
```

```
[1] 2
```

2 Modeling Literature

Bayesian Workflow Andrew Gelman, Aki Vehtari, Daniel Simpson, Charles C. Margossian, Bob Carpenter, Yuling Yao, Lauren Kennedy, Jonah Gabry, Paul-Christian Bürkner, Martin Modrák https://arxiv.org/abs/2011.01808

How to avoid machine learning pitfalls: a guide for academic researchers Michael A. Lones https://arxiv.org/abs/2108.02497

Cross-validation strategies for data with temporal, spatial, hierarchical, or phylogenetic structure David R. Roberts,Volker Bahn,Simone Ciuti,Mark S. Boyce,Jane Elith,Gurutzeta Guillera-Arroita,Severin Hauenstein,José J. Lahoz-Monfort,Boris Schröder,Wilfried Thuiller,David I. Warton,Brendan A. Wintle,Florian Hartig,Carsten F. Dormann https://onlinelibrary.wiley.com/doi/full/10.1111/ecog.02881

Effective Pandas https://store.metasnake.com/effective-pandas-book

Information geometry and divergences https://franknielsen.github.io/IG/#bookIG

Statistical Rethinking: A Bayesian Course with Examples in R and Stan (& PyMC3 & brms) https://xcelab.net/rm/statistical-rethinking/ https://www.youtube.com/playlist?list=PLDcUM9US4XdMROZOIRtIK0aOynbgZN

Part I

mathematical object

3 Set

Cites: Wikipedia; Wikidata; PlanetMath

4 List (Sequence)

AKA: Sequence, a_n where n is the nth element, (1,2,3, ….)

Distinct from: Set

Measure of:

Description: A list is a collection of objects with a specific ordering and where the same object can appear more than once. Call each object an element, and its location its index or rank. An index is a natural number counting upward from the first element in the list. Whether counting begins at 0 or 1 depends on local conventions.

Formalization:

Algorithm:

Cites: Wikipedia Wikidata Encyclopedia Of Math Wolfram PlanetMath

4.0.0.1 R

Documentation:

list: Lists – Generic and Dotted Pairs

Examples:

```
example_list = list(1,2,3)
example_list
```

```
[[1]]
[1] 1

[[2]]
[1] 2

[[3]]
[1] 3
```

4.0.0.2 Python

Documentation:

More on Lists

Examples:

```
example_list =  [1,2,3]
example_list
```

```
[1, 2, 3]
```

4.0.0.3 SQL

```
library(DBI)
# Create an ephemeral in-memory RSQLite database
con <- dbConnect(RSQLite::SQLite(), dbname = ":memory:")
dbListTables(con)
```

```
character(0)
```

```
dbWriteTable(con, "mtcars", mtcars)
dbListTables(con)
```

```
[1] "mtcars"
```

```
create table StatisticalNumbers(
 value int
)
```

```
SELECT * FROM mtcars LIMIT 5;
```

Table 4.1: 5 records

mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
21.0	6	160	110	3.90	2.620	16.46	0	1	4	4
21.0	6	160	110	3.90	2.875	17.02	0	1	4	4

mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
22.8	4	108	93	3.85	2.320	18.61	1	1	4	1
21.4	6	258	110	3.08	3.215	19.44	1	0	3	1
18.7	8	360	175	3.15	3.440	17.02	0	0	3	2

5 Tensor

Instance of: algebraic object

5.1 Frequentist

AKA:

Distinct from:

English:

Formalization:

Cites: Wikipedia ; Wikidata ; Wolfram

Code

5.1.0.1 R

Documentation: mean: Arithmetic Mean

Examples:

5.1.0.2 Python

Documentation: numpy.mean

Examples:

5.1.0.3 SQL

Documentation: PostgreSQL AVG Function

```r
library(DBI)
# Create an ephemeral in-memory RSQLite database
#con <- dbConnect(RSQLite::SQLite(), dbname = ":memory:")
#dbListTables(con)
#dbWriteTable(con, "mtcars", mtcars)
#dbListTables(con)

#Configuration failed because libpq was not found. Try installing:
#* deb: libpq-dev libssl-dev (Debian, Ubuntu, etc)
#install.packages('RPostgres')
#remotes::install_github("r-dbi/RPostgres")
#Took forever because my file permissions were broken
#pg_lsclusters
require(RPostgres)
```

```
Loading required package: RPostgres
```

```r
# Connect to the default postgres database
#I had to follow these instructions and create both a username and database that matched m
#https://www.digitalocean.com/community/tutorials/how-to-install-postgresql-on-ubuntu-20-0
con <- dbConnect(RPostgres::Postgres())
```

5.1.0.4 Torch

```r
import torch
```

5.2 Bayesian

English: **Formalization**:

Cites:

Code

Part II

operations of arithmetic

6 Addition

Instance of: operation of arithmetic

6.1 Frequentist

AKA: + ; add

Distinct from:

English:

Formalization:

Cites: Wikipedia ; Wikidata ; Wolfram

Code

6.1.0.1 R

Documentation: mean: Arithmetic Mean

Examples:

6.1.0.2 Python

Documentation: numpy.mean

Examples:

6.1.0.3 SQL

Documentation: PostgreSQL AVG Function

```r
library(DBI)
# Create an ephemeral in-memory RSQLite database
#con <- dbConnect(RSQLite::SQLite(), dbname = ":memory:")
#dbListTables(con)
#dbWriteTable(con, "mtcars", mtcars)
#dbListTables(con)

#Configuration failed because libpq was not found. Try installing:
#* deb: libpq-dev libssl-dev (Debian, Ubuntu, etc)
#install.packages('RPostgres')
#remotes::install_github("r-dbi/RPostgres")
#Took forever because my file permissions were broken
#pg_lsclusters
require(RPostgres)
```

```
Loading required package: RPostgres
```

```r
# Connect to the default postgres database
#I had to follow these instructions and create both a username and database that matched m
#https://www.digitalocean.com/community/tutorials/how-to-install-postgresql-on-ubuntu-20-0
con <- dbConnect(RPostgres::Postgres())
```

6.1.0.4 Torch

```r
import torch
```

6.2 Bayesian

English: **Formalization**:

Cites:

Code

7 Introduction

Instance of: operation of arithmetic

7.1 Frequentist

AKA: - ; minus

Distinct from:

English:

Formalization:

Cites: Wikipedia ; Wikidata ; Wolfram

Code

7.1.0.1 R

Documentation: mean: Arithmetic Mean

Examples:

7.1.0.2 Python

Documentation: numpy.mean

Examples:

7.1.0.3 SQL

Documentation: PostgreSQL AVG Function

```r
library(DBI)
# Create an ephemeral in-memory RSQLite database
#con <- dbConnect(RSQLite::SQLite(), dbname = ":memory:")
#dbListTables(con)
#dbWriteTable(con, "mtcars", mtcars)
#dbListTables(con)

#Configuration failed because libpq was not found. Try installing:
#* deb: libpq-dev libssl-dev (Debian, Ubuntu, etc)
#install.packages('RPostgres')
#remotes::install_github("r-dbi/RPostgres")
#Took forever because my file permissions were broken
#pg_lsclusters
require(RPostgres)
```

```
Loading required package: RPostgres
```

```r
# Connect to the default postgres database
#I had to follow these instructions and create both a username and database that matched m
#https://www.digitalocean.com/community/tutorials/how-to-install-postgresql-on-ubuntu-20-0
con <- dbConnect(RPostgres::Postgres())
```

7.1.0.4 Torch

```r
import torch
```

7.2 Bayesian

English: **Formalization**:

Cites:

Code

8 Multiplication

Instance of: operation of arithmetic

8.1 Frequentist

AKA: * ; × ; ; multiply

Distinct from:

English:

Formalization:

Cites: Wikipedia ; Wikidata ; Wolfram

Code

8.1.0.1 R

Documentation: mean: Arithmetic Mean

Examples:

8.1.0.2 Python

Documentation: numpy.mean

Examples:

8.1.0.3 SQL

Documentation: PostgreSQL AVG Function

```r
library(DBI)
# Create an ephemeral in-memory RSQLite database
#con <- dbConnect(RSQLite::SQLite(), dbname = ":memory:")
#dbListTables(con)
#dbWriteTable(con, "mtcars", mtcars)
#dbListTables(con)

#Configuration failed because libpq was not found. Try installing:
#* deb: libpq-dev libssl-dev (Debian, Ubuntu, etc)
#install.packages('RPostgres')
#remotes::install_github("r-dbi/RPostgres")
#Took forever because my file permissions were broken
#pg_lsclusters
require(RPostgres)
```

```
Loading required package: RPostgres
```

```r
# Connect to the default postgres database
#I had to follow these instructions and create both a username and database that matched m
#https://www.digitalocean.com/community/tutorials/how-to-install-postgresql-on-ubuntu-20-0
con <- dbConnect(RPostgres::Postgres())
```

8.1.0.4 Torch

```r
import torch
```

8.2 Bayesian

English: **Formalization**:

Cites:

Code

9 Division

Instance of: operation of arithmetic

9.1 Frequentist

AKA: / ; $\frac{numerator}{denominator}$; $\div$

Distinct from:

English:

Formalization:

Cites: Wikipedia ; Wikidata ; Wolfram

Code

9.1.0.1 R

Documentation: mean: Arithmetic Mean

Examples:

9.1.0.2 Python

Documentation: numpy.mean

Examples:

9.1.0.3 SQL

Documentation: PostgreSQL AVG Function

```r
library(DBI)
# Create an ephemeral in-memory RSQLite database
#con <- dbConnect(RSQLite::SQLite(), dbname = ":memory:")
#dbListTables(con)
#dbWriteTable(con, "mtcars", mtcars)
#dbListTables(con)

#Configuration failed because libpq was not found. Try installing:
#* deb: libpq-dev libssl-dev (Debian, Ubuntu, etc)
#install.packages('RPostgres')
#remotes::install_github("r-dbi/RPostgres")
#Took forever because my file permissions were broken
#pg_lsclusters
require(RPostgres)
```

```
Loading required package: RPostgres
```

```r
# Connect to the default postgres database
#I had to follow these instructions and create both a username and database that matched m
#https://www.digitalocean.com/community/tutorials/how-to-install-postgresql-on-ubuntu-20-0
con <- dbConnect(RPostgres::Postgres())
```

9.1.0.4 Torch

```r
import torch
```

9.2 Bayesian

English: **Formalization**:

Cites:

Code

Part III

operations of algebra

10 Dot product

Instance of: algebraic operation

AKA: scalar product; inner product ; projection product ; $ · $

Distinct from:

English:

Formalization:

$$a \cdot b$$

Cites: Wikipedia ; Wikidata ; Wolfram

Code

10.0.0.1 R

Documentation: mean: Arithmetic Mean

Examples:

10.0.0.2 Python

Documentation: numpy.mean

Examples:

10.0.0.3 SQL

Documentation: PostgreSQL AVG Function

```
library(DBI)
# Create an ephemeral in-memory RSQLite database
#con <- dbConnect(RSQLite::SQLite(), dbname = ":memory:")
#dbListTables(con)
```

```r
#dbWriteTable(con, "mtcars", mtcars)
#dbListTables(con)

#Configuration failed because libpq was not found. Try installing:
#* deb: libpq-dev libssl-dev (Debian, Ubuntu, etc)
#install.packages('RPostgres')
#remotes::install_github("r-dbi/RPostgres")
#Took forever because my file permissions were broken
#pg_lsclusters
require(RPostgres)
```

```
Loading required package: RPostgres
```

```r
# Connect to the default postgres database
#I had to follow these instructions and create both a username and database that matched m
#https://www.digitalocean.com/community/tutorials/how-to-install-postgresql-on-ubuntu-20-0
con <- dbConnect(RPostgres::Postgres())
```

10.0.0.4 Torch

```r
import torch
```

10.1 Bayesian

English: **Formalization**:

Cites:

Code

Part IV

moments of a distribution

11 Mean

Measure of: Central tendency

11.1 Frequentist

AKA: Arithmetic mean; average; $\bar{x}$ (sample mean); μ (population mean); μ_x (population mean)

Distinct from: Geometric mean (GM); Harmonic mean (HM); generalized mean/ Power mean; weighted arithmetic mean

English: Take a list of numbers, sum those numbers, and then divide by the number of numbers.

Formalization:

$$\bar{x} = \frac{1}{n}(\sum_{i=1}^{n} x_i) = \frac{x_1 + x_2 + ... + x_n}{n}$$

Cites: Wikipedia ; Wikidata ; Wolfram

Code

11.1.0.1 R

Documentation: mean: Arithmetic Mean

Examples:

```
x = c(1,2,3,4)
x
```

```
[1] 1 2 3 4
```

```r
#Algorithm
x_bar = sum(x, na.rm=T)/length(x)
x_bar
```

```
[1] 2.5
```

```r
#Base Function
x_bar = mean(x, na.rm=T)
x_bar
```

```
[1] 2.5
```

11.1.0.2 Python

Documentation: numpy.mean

Examples:

```r
x = [1,2,3,4]
print(x)
```

```
[1, 2, 3, 4]
```

```r
#Algorithm
x_bar= sum(x)/len(x)
x_bar
```

```
2.5
```

```r
#statistics Function
import statistics
x_bar = statistics.mean(x)
x_bar
```

```
2.5
```

```
#scipy Function
#<string>:1: DeprecationWarning: scipy.mean is deprecated and will be removed in SciPy 2.0
import scipy
x_bar = scipy.mean(x)
```

```
<string>:1: DeprecationWarning: scipy.mean is deprecated and will be removed in SciPy 2.0.0,
```

```
x_bar
```

```
2.5
```

```
#numpy Function
import numpy as np
x = np.array(x)
x_bar = x.mean()
x_bar
```

```
2.5
```

11.1.0.3 SQL

Documentation: PostgreSQL AVG Function

```
library(DBI)
# Create an ephemeral in-memory RSQLite database
#con <- dbConnect(RSQLite::SQLite(), dbname = ":memory:")
#dbListTables(con)
#dbWriteTable(con, "mtcars", mtcars)
#dbListTables(con)

#Configuration failed because libpq was not found. Try installing:
#* deb: libpq-dev libssl-dev (Debian, Ubuntu, etc)
#install.packages('RPostgres')
#remotes::install_github("r-dbi/RPostgres")
#Took forever because my file permissions were broken
#pg_lsclusters
require(RPostgres)
```

```
Loading required package: RPostgres
```

```
# Connect to the default postgres database
#I had to follow these instructions and create both a username and database that matched m
#https://www.digitalocean.com/community/tutorials/how-to-install-postgresql-on-ubuntu-20-0
con <- dbConnect(RPostgres::Postgres())
```

```
DROP TABLE IF EXISTS t1;
```

```
CREATE TABLE IF NOT EXISTS t1 (
    id serial PRIMARY KEY,
    amount INTEGER
);
```

```
INSERT INTO t1 (amount)
VALUES
    (10),
    (NULL),
    (30);
```

```
SELECT  * FROM  t1;
```

Table 11.1: 3 records

id	amount
1	10
2	NA
3	30

```
SELECT AVG(amount)::numeric(10,2)
FROM t1;
```

Table 11.2: 1 records

avg
20

11.1.0.4 Torch

```
import torch
```

11.2 Bayesian

Bayesian average; Solving an age-old problem using Bayesian Average; Of bayesian average and star ratings; Bayesian Average Ratings ;

English: The Bayesian average is the weighted average of a prior and the observed sample average. When would you want this? When you have strong beliefs about the true mean, or when sample size is too small to reliable calculate a mean. For example a movie rating website where a movie may have only a single 5 star rating and so would rank higher than the Godfather with over a 100 almost all 5 star ratings.

Formalization:

$$\bar{x} = \frac{C * m + (\sum_{i=1}^{n} x_i)}{c + n}$$

Where m is a prior for true mean, and C is a constant representing how many elements would be necessary to reliably estimate a sample mean.

Code

References