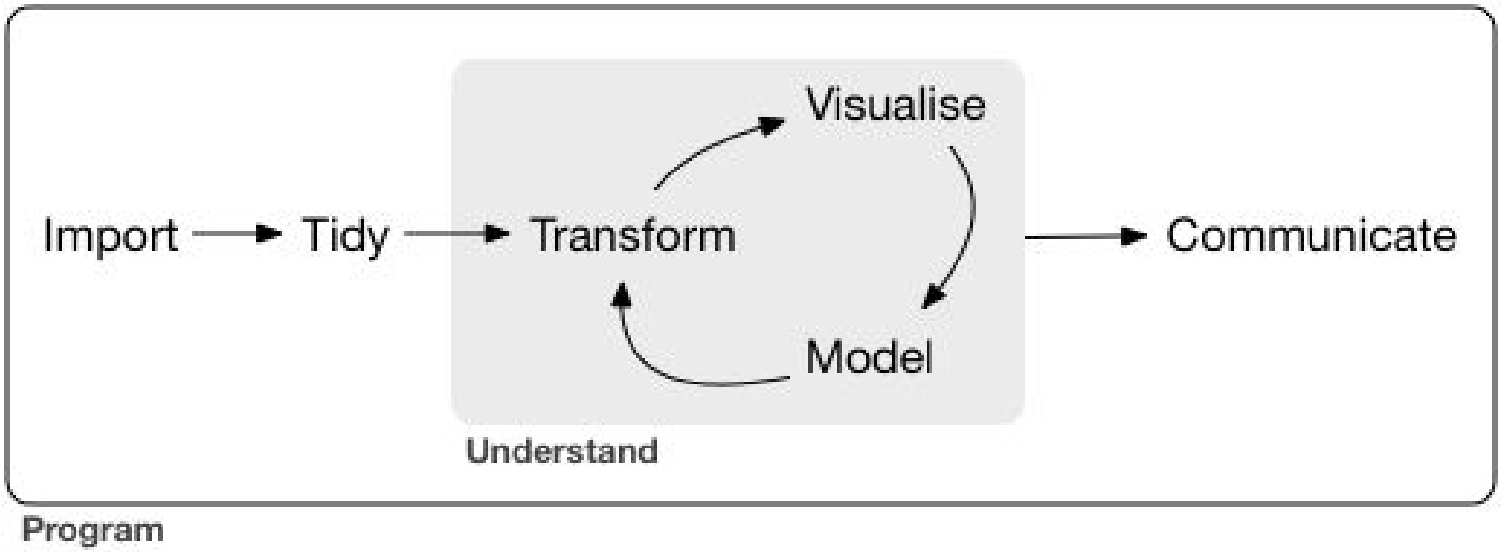
R for Data Science

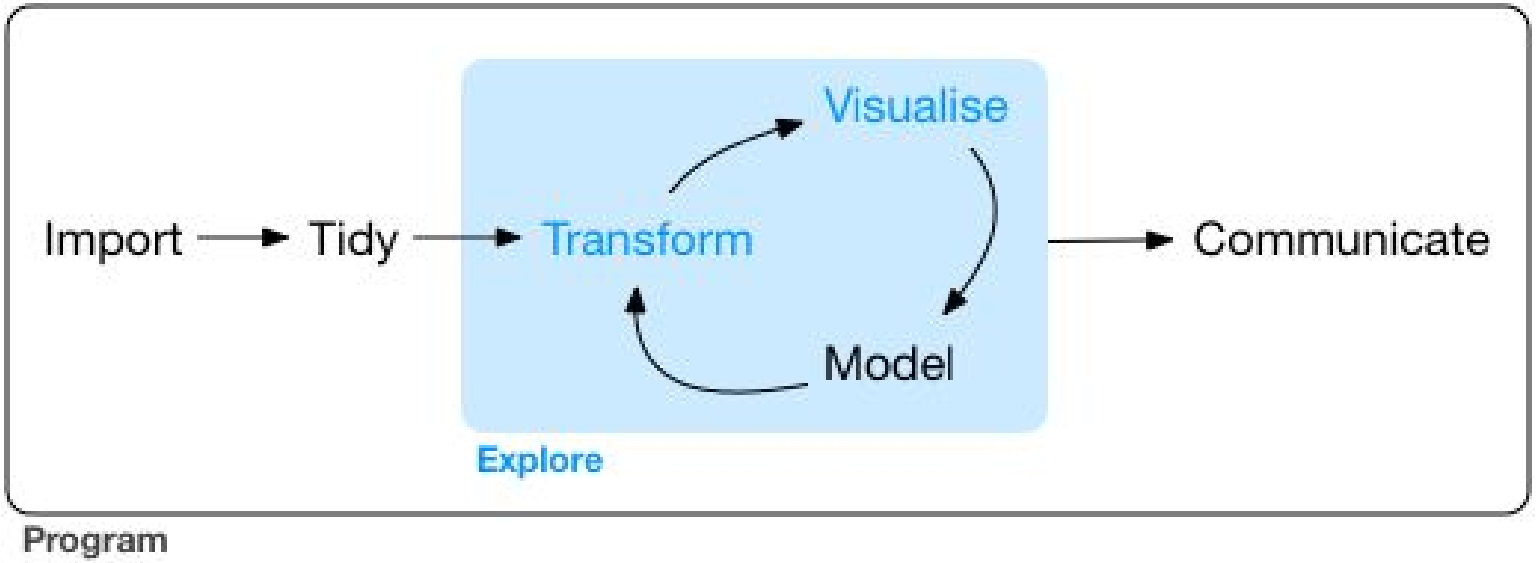
Chapters 1 - 8

Chapter 1 Introduction


Visualise
Import
Tidy
Transform
Communicate
Model
Understand
Program

I EXPLORE

Chapter 2 Introduction


Import
Tidy
Transform
Visualise
Model
Communicate
Explore
Program

Chapter 3 Data visualisation

Seven Parameters

```
ggplot(data = <DATA>) +
  <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>),
                  stat = <STAT>,
                  position = <POSITION>) +
  <COORDINATE_FUNCTION> +
  <FACET_FUNCTION>
```

Seven Parameters

```
ggplot(data = <DATA>) +
  <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>),
                  stat = <STAT>,
                  position = <POSITION>) +
  <COORDINATE_FUNCTION> +
  <FACET_FUNCTION>
```

Seven Parameters

```
ggplot(data = <DATA>) +
  <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>),
                  stat = <STAT>,
                  position = <POSITION>) +
  <COORDINATE_FUNCTION> +
  <FACET_FUNCTION>
```

Seven Parameters

```
ggplot(data = <DATA>) +
  <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>),
                  stat = <STAT>,
                  position = <POSITION>) +
  <COORDINATE_FUNCTION> +
  <FACET_FUNCTION>
```

Seven Parameters

```
ggplot(data = <DATA>) +
  <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>),
                  stat = <STAT>,
                  position = <POSITION>) +
  <COORDINATE_FUNCTION> +
  <FACET_FUNCTION>
```

Seven Parameters

```
ggplot(data = <DATA>) +
  <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>),
                  stat = <STAT>,
                  position = <POSITION>) +
  <COORDINATE_FUNCTION> +
  <FACET_FUNCTION>
```

Seven Parameters

```
ggplot(data = <DATA>) +
  <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>),
                  stat = <STAT>,
                  position = <POSITION>) +
  <COORDINATE_FUNCTION> +
  <FACET_FUNCTION>
```

Seven Parameters

```
ggplot(data = <DATA>) +
  <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>),
                  stat = <STAT>,
                  position = <POSITION>) +
  <COORDINATE_FUNCTION> +
  <FACET_FUNCTION>
```

1. Begin with the **`diamonds`** data set

2. Compute counts for each cut value with **`stat_count()`**.

carat	cut	color	clarity	depth	table	price	x	y	z
0.23	Ideal	E	SI2	61.5	55	326	3.95	3.98	2.43
0.21	Premium	E	SI1	59.8	61	326	3.89	3.84	2.31
0.23	Good	E	VS1	56.9	65	327	4.05	4.07	2.31
0.29	Premium	I	VS2	62.4	58				
0.31	Good	J	SI2	63.3	58				
…	…	…	…	…	…				

stat_count()

cut	count	prop
Fair	1610	1
Good	4906	1
Very Good	12082	1

3. Represent each observation with a bar.

4. Map the **`fill`** of each bar to the **`..count..`** variable.

carat	cut	color	clarity	depth	table	pr
0.23	Ideal	E	SI2	61.5	55	32
0.21	Premium	E	SI1	59.8	61	32
0.23	Good	E	VS1	56.9	65	32
0.29	Premium	I	VS2	62.4	58	33
0.31	Good	J	SI2	63.3	58	33
…	…	…	…	…	…	

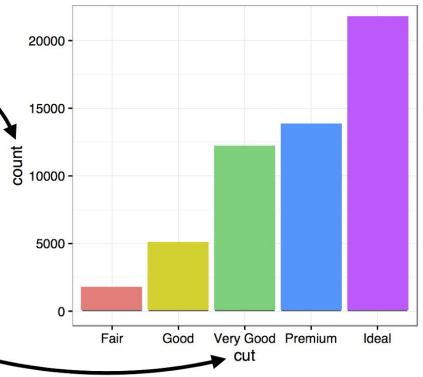
5. Place geoms in a cartesian coordinate system.

6. Map the y values to **`..count..`** and the x values to **`cut`**.

carat	cut	color	clarity	depth	table	price	x	y	z
0.23	Ideal	E	SI2	61.5	55	326	3.95	3.98	2.43
0.21	Premium	E	SI1	59.8	61	326	3.89	3.84	2.31
0.23	Good	E	VS1	56.9	65	327	4.05	4.07	2.31
0.29	Premium	I	VS2	62.4	58	334	4.20	4.23	2.63
0.31	Good	J	SI2	63.3	58	335	4.34	4.35	2.75
…	…	…	…	…	…	…	…	…	…

stat_count()

cut	count	prop
Fair	1610	1
Good	4906	1
Very Good	12082	1
Premium	13791	1
Ideal	21551	1



Chapter 4 Workflow: basics

- use `<-` for new objects
- `object_name <- value`, "object name gets value"
- start with letter, only `_` and `.` for punctuation
- snake_case recommended
- surrounding assignment with parns assigns and prints
  - `(x <- 5)`
  - `[1] 5`

Chapter 5 Data transformation

the pipe %>%

1. **`filter()`** - subset observations
2. **`arrange()`** - order observations
3. **`select()`** - choose columns
4. **`mutate()`** - add new variables
5. **`summarise()`** - produce grouped summaries

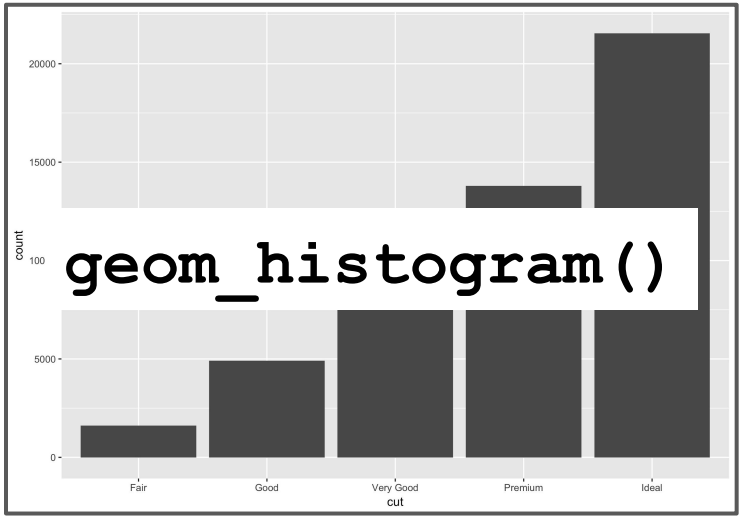
`group_by()` - add a grouping dimension to `summarise()`, `filter()`, and `mutate()`

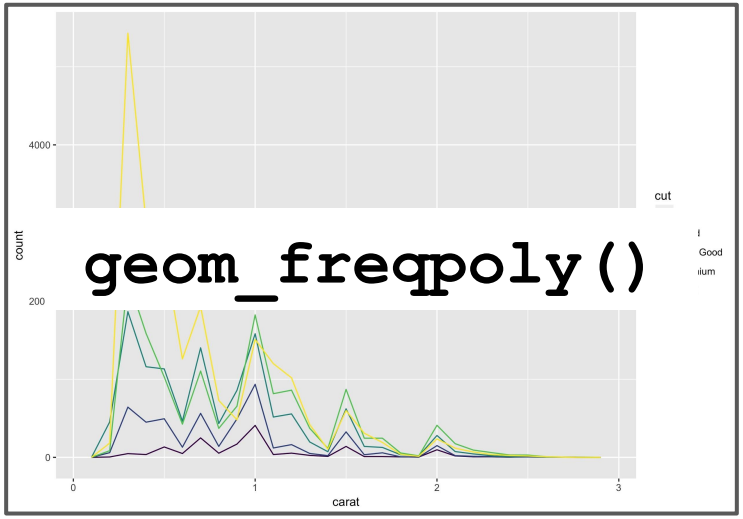
Chapter 6 Workflow: scripts

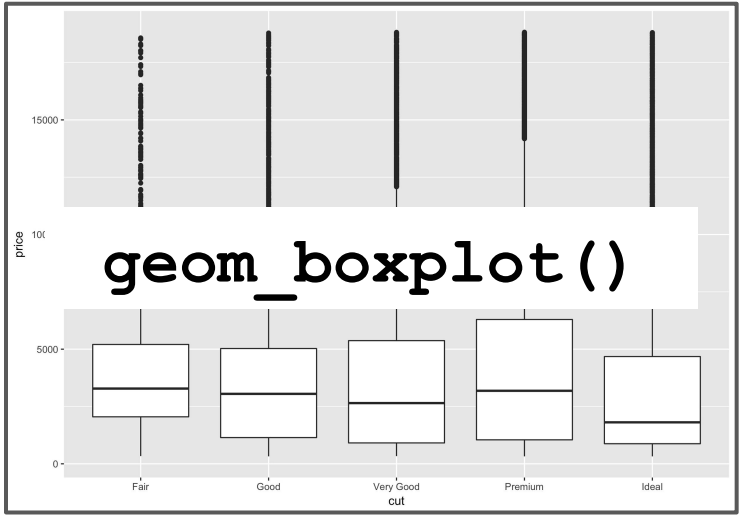
- use `Cmd/Ctrl + Enter` to run a chunk
- use `Cmd/Ctrl + Shift + S` to run the entire script, `source()`

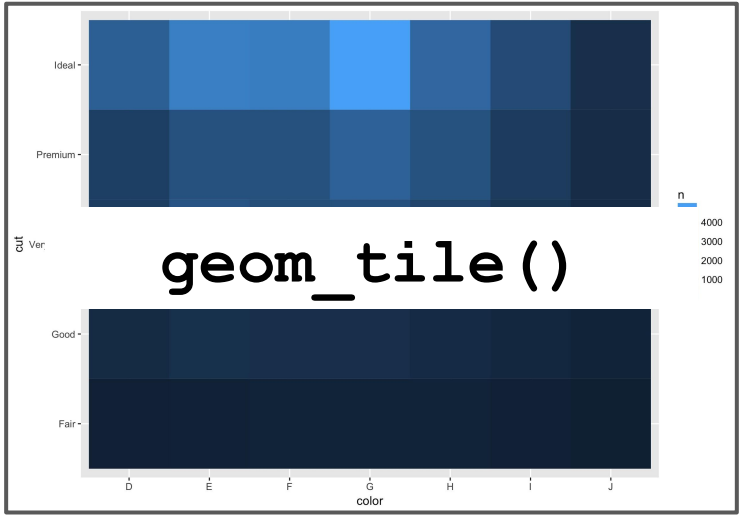
Chapter 7 Exploratory Data Analysis

- A **variable** is a quantity, quality, or property that you can measure.
- A **value** is the state of a variable when you measure it. The value of a variable may change from measurement to measurement.
- An **observation** is a set of measurements made under similar conditions (you usually make all of the measurements in an observation at the same time and on the same object). An observation will contain several values, each associated with a different variable. I'll sometimes refer to an observation as a data point.
- **Tabular data** is a set of values, each associated with a variable and an observation. Tabular data is tidy if each value is placed in its own "cell", each variable in its own column, and each observation in its own row.

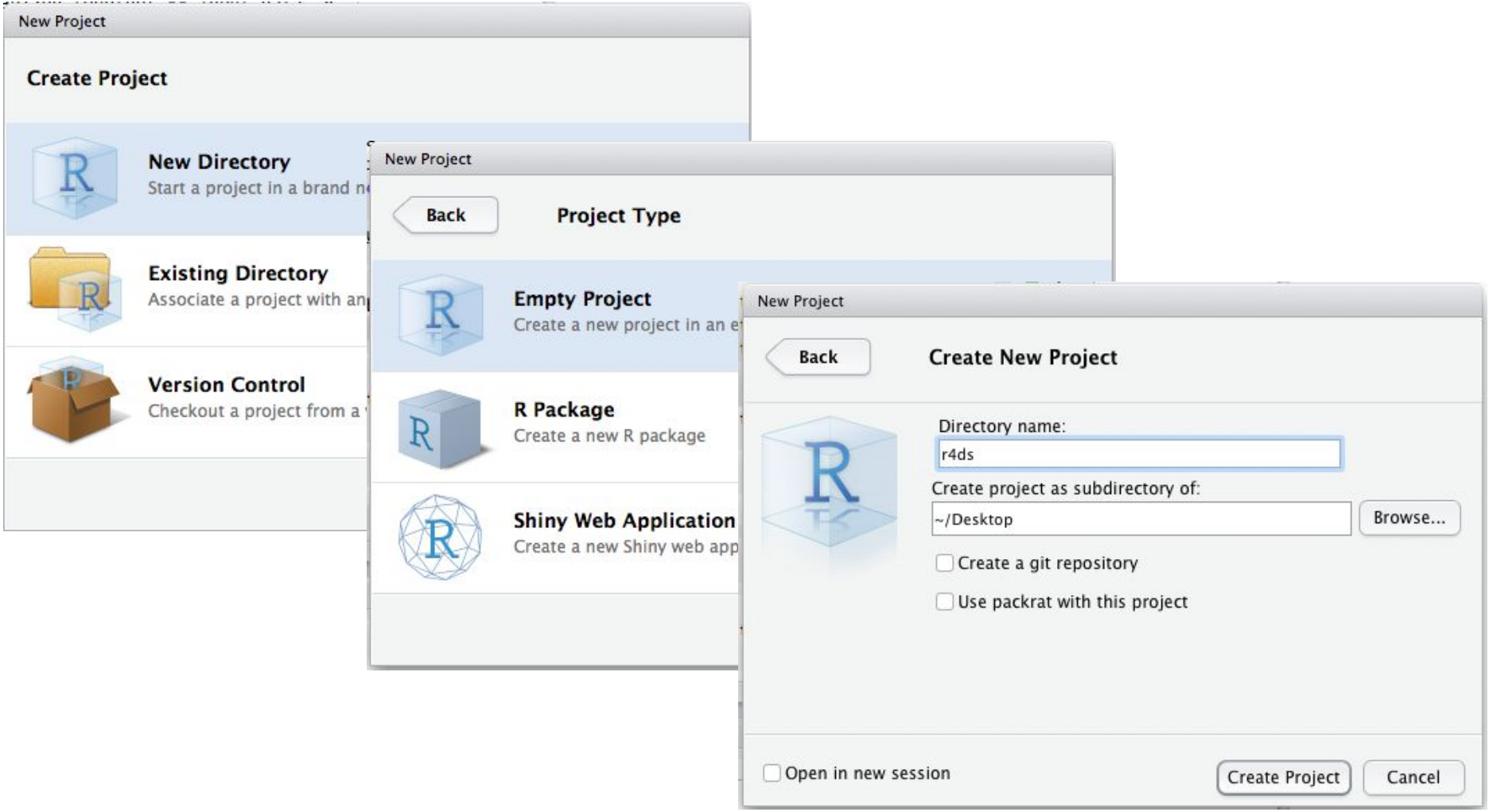

geom_histogram()
count
20000
15000
5000
0
Fair
Good
Very Good
Premium
Ideal
cut


geom_freqpoly()
count
4000
200
0
0
1
2
3
carat
cut
Good


geom_boxplot()
price
15000
5000
0
Fair
Good
Very Good
Premium
Ideal
cut


geom_tile()
Ideal
Premium
Good
Fair
cut
D
E
F
G
H
I
J
color
n
4000
3000
2000
1000

Chapter 8 Workflow: projects


New Project
Create Project
New Directory
Start a project in a brand n
Existing Directory
Associate a project with an
Version Control
Checkout a project from a
New Project
Back
Project Type
Empty Project
Create a new project in an e
R Package
Create a new R package
Shiny Web Application
Create a new Shiny web app
New Project
Back
Create New Project
Directory name:
r4ds
Create project as subdirectory of:
~/Desktop
Browse...
Create a git repository
Use packrat with this project
Open in new session
Create Project
Cancel