Data Handling: Import, Cleaning and Visualisation

Lecture 9:
Exploratory Data Analysis and Visualization, Part I

Dr. Aurélien Sallin

2024-11-29

Recap: Data Manipulation

Data preparation consists of five main steps

- **Tidy** data.
- **Reshape** datasets from wide to long (and vice versa).
- **Bind** or stack rows in datasets.
- **Join** datasets.
- **Clean** and **manipulate** data.

Reshape datasets from wide to long (and vice versa).

country	year	metric
x	1960	10
x	1970	13
x	2010	15
y	1960	20
y	1970	23
y	2010	25
z	1960	30
z	1970	33
z	2010	35

```
pivot_wider(names_from = “year”,
            names_prefix = “yr”,
            values_from = “metric”)
```

country	yr1960	yr1970	yr2010
x	10	13	15
y	20	23	25
z	30	33	35

```
pivot_longer(cols = yr1960:yr2010,
             names_to = “year”,
             names_prefix = “yr”
             values_to = “metric”)
```

Bind or stack rows in datasets.

ID	X	Y
1	a	50
2	b	10

ID	Z
3	M
4	O

ID	X	Z
5	c	P

ID	X	Y	Z
1	a	50	NA
2	b	10	NA
3	NA	NA	M
4	NA	NA	O
5	c	NA	P

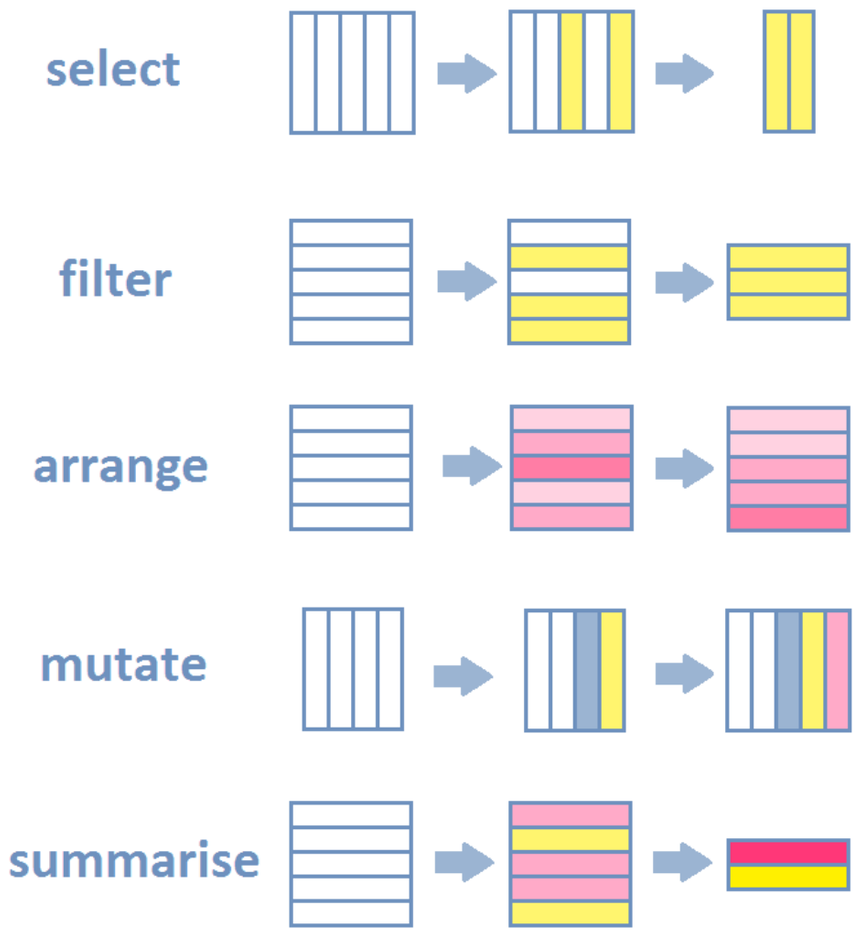
Join datasets.

Overview by R4DS:

dplyr (tidyverse)	base::merge
`inner_join(x, y)`	`merge(x, y)`
`left_join(x, y)`	`merge(x, y, all.x = TRUE)`
`right_join(x, y)`	`merge(x, y, all.y = TRUE),`
`full_join(x, y)`	`merge(x, y, all = TRUE)`

You are expected to know that `base::merge()` exists. For the exam, please focus on the `dplyr` functions.

Clean and manipulate data.



Source: Intro to R for Social Scientists

Warm up

Mutating dataframes

Consider the following data frame:

```
main_dataset
  city        temp  scale conversion_factor
1 StGallen    12 Celsius 1.0000000
2 Zürich      14 Celsius 1.0000000
3 Detroit     40 Fahrenheit 0.5555556
```

Select all the statements that are true.

```
main_dataset |>
  mutate(temp_celsius = ifelse(scale == "Fahrenheit", (temp-32) * conversion_factor, temp))
```

replaces the variable `temp` with `temp_celsius`

```
main_dataset |>
  mutate(temp_celsius = ifelse(scale == "Fahrenheit", (temp-32) * conversion_factor, temp))
```

has 3 rows and 5 columns

```
main_dataset |>
  summarize(mean_temp = mean(temp), min_temp = min(temp))
```

returns a tibble containing 2 columns and 1 row

```
main_dataset |>
  summarize(mean_scale = mean(scale), sd_scale = sd(scale))
```

Joining

Consider the following code:

```
temperature_conversions <- data.frame(conversion_factor = c(5/9, 1),
                                      scale = c("Fahrenheit", "Celsius"))

temperature <- data.frame(city = c("StGallen", "Zürich", "Detroit"),
                          temp = c(12, 14, 21),
                          scale = c("Cel", "Cel", "F"))
```

Select statements that are true:

- `inner_join(temperature, temperature_conversions, by="scale")` returns a data frame with 3 rows.
- `left_join(temperature, temperature_conversions, by="scale")` returns a data frame with 3 rows.
- `full_join(temperature, temperature_conversions, by="scale")` returns a data frame with 3 rows.

Question on merging dataframes

Consider the two following dataframes:

```
> df_c <- data.frame(id = c(1:3,1:3,5),
+                    money_spent= c(1000, 2000, 6000, 1500, 3000, 5500,3000),
+                    currency = c("CHF", "CHF", "USD", "EUR", "CHF", "USD", "CAD"),
+                    year=c(2017,2017,2017,2018,2018,2018,2018))
> df_c
  id money_spent currency year
1  1        1000      CHF 2017
2  2        2000      CHF 2017
3  3        6000      USD 2017
4  1        1500      EUR 2018
5  2        3000      CHF 2018
6  3        5500      USD 2018
7  5        3000      CAD 2018


> df_mtm <- data.frame(
+   id = rep(1:3, 2),
+   year = rep(c(2017,2017,2017,2018,2018,2018), 2),
+   first_name = rep(c("Anna", "Betty", "Claire"), 2),
+   profession = rep(c("Economist", "Data Scientist",
+                  "Data Scientist"), 2)
+ )
>
```

Now consider the following two lines of code:

```
lj_1 <- left_join(df_mtm, df_c, by = "id")
lj_2 <- left_join(df_mtm, df_c, by = c("id", "year"))
```

Today

Announcements

- Next week (05.12.2024):
  “**Understanding and reacting to customer dissatisfaction through data: A case study**” by Rachel Lund, Deloitte
- Exam and LockDown Browser: check Sharepoint on StudentWeb and test on Canvas
- Ideas for the last course
- Course evaluation: thanks for making this course a success! 🤓

Goals for the next two lectures

1. Know how to conduct exploratory data analysis (EDA).
2. Visualize data using tables.
3. Visualize data using the grammar of graphics.
4. Produce effective data visualization.

Exploratory Data Analysis and Descriptive Statistics


TERRA INCOGNITA

Exploratory Data Analysis (EDA) is the first step of data analysis

- 🔎 Get a first understanding of your dataset.
- Show key aspects of data by modelling, transforming, and visualizing your data.
  - Investigate the quality and reliability of your data.
  - Inform your own statistical analysis.
  - Inform audience (helps understand analytics parts).
- … which in turns generates new questions about your dataset (creative process 🎨).

Data exploration be like...

"To not mislead others and not embarrass yourself, know your data".*

- Understand the **number of observations**, the **units**, the **quality** of data, the **definitions** of variables, what to do with **missing values**.

What type of variation occurs within my variables?

- **Typical values**: mean, mode, range, standard deviation.
- **Surprising values**: outliers, rare values, unusual patterns.

What type of covariation occurs between my variables?

- **Covariation** and **Patterns**.

Source: R4DS, *"Statistics for Public Policy" by Jeremy G. Weber, 2024.

EDA: first steps in R

- Quick overview: `summary()`
- Cross-tabulation: `table()`
- Summarizing tools: `skimr`, `summarytools`, `janitor` (also cleaning)

EDA: first steps in R

1. Functions to compute statistics

- e.g., `mean()`

2. Functions to *apply* the statistics function to one or several columns in a tidy dataset.

- Including all values in a column.
- By group (observation categories, e.g. by location, year, etc.)

`summary()` in `base`; `summarise()` in `tidyverse`; `group_by()` in `tidyverse`; `sapply()`, `apply()`, `lapply()`, etc. in `base`; `skimr` package; etc.

EDA: an example with the `swiss` data

▶ Show code

```
swiss
```

```
   municipality Fertility Agriculture Examination Education Catholic Infant.Mortality
1    Courtelary      80.2        17.0          15        12     9.96             22.2
2      Delemont      83.1        45.1           6         9    84.84             22.2
3  Franches-Mnt      92.5        39.7           5         5    93.40             20.2
4       Moutier      85.8        36.5          12         7    33.77             20.3
5    Neuveville      76.9        43.5          17        15     5.16             20.6
6    Porrentruy      76.1        35.3           9         7    90.57             26.6
7         Broye      83.8        70.2          16         7    92.85             23.6
8         Glane      92.4        67.8          14         8    97.16             24.9
9       Gruyere      82.4          NA          12         7    97.67             21.0
10       Sarine      82.9        45.2          16        13    91.38             24.4
11      Veveyse      87.1        64.5          14         6    98.61             24.5
12        Aigle      64.1        62.0          21        12     8.52             16.5
13      Aubonne      66.9        67.5          14         7     2.27             19.1
14     Avenches      68.9        60.7          19        12     4.43             22.7
15     Cossonay      61.7        69.3          22         5     2.82             18.7
16    Echallens      68.3        72.6          18         2    24.20             21.2
17     Grandson      71.7        34.0          17         8     3.30             20.0
18     Lausanne      55.7        19.4          26        28    12.11            100.0
19    La Vallee      54.3        15.2          31        20     2.15             10.8
20       Lavaux      65.1        73.0          19         9     2.84             20.0
21       Morges      65.5        59.8          22        10     5.23             18.0
22       Moudon      65.0        55.1          14         3     4.52             22.4
```

EDA: an example with the `swiss` data

```
summary(swiss)
```

```
 municipality          Fertility      Agriculture     Examination      Education
 Length:47          Min.   :35.00   Min.   : 1.20   Min.   : 3.00   Min.   : 1.00
 Class :character   1st Qu.:64.70   1st Qu.:35.60   1st Qu.:12.00   1st Qu.: 6.00
 Mode  :character   Median :70.40   Median :54.60   Median :16.00   Median : 8.00
                    Mean   :70.14   Mean   :50.60   Mean   :16.49   Mean   :10.98
                    3rd Qu.:78.45   3rd Qu.:67.72   3rd Qu.:22.00   3rd Qu.:12.00
                    Max.   :92.50   Max.   :89.70   Max.   :37.00   Max.   :53.00
                                    NA's   :1
    Catholic       Infant.Mortality
 Min.   :  2.150   Min.   : 10.80
 1st Qu.:  5.195   1st Qu.: 18.15
 Median : 15.140   Median : 20.00
 Mean   : 41.144   Mean   : 21.64
 3rd Qu.: 93.125   3rd Qu.: 22.20
 Max.   :100.000   Max.   :100.00
```

EDA: using `summarytools()` to generate an overview (does not render well on slides)

```
summarytools::dfSummary(swiss)
```

Data Frame Summary
swiss
Dimensions: 47 x 7
Duplicates: 0

No	Variable	Stats / Values	Freqs (% of Valid)	Graph	Valid	Missing
1	municipality [character]	1. Aigle 2. Aubonne 3. Avenches 4. Boudry 5. Broye 6. Conthey 7. Cossonay 8.	1 (2.1%) 1 (2.1%) 1 (2.1%) 1 (2.1%) 1 (2.1%) 1 (2.1%) 1 (2.1%) 1 (2.1%)		47 (100.0%)	0 (0.0%)

No	Variable	Stats / Values	Freqs (% of Valid)	Graph	Valid	Missing
		Courtelary 9. Delemont 10. Echallens [37 others]	1 (2.1%) 37 (78.7%)			
2	Fertility [numeric]	Mean (sd) : 70.1 (12.5) min < med < max: 35 < 70.4 < 92.5 IQR (CV) : 13.7 (0.2)	46 distinct values	: . : : : : . . : : : . . : : : :	47 (100.0%)	0 (0.0%)
3	Agriculture [numeric]	Mean (sd) : 50.6 (23) min < med < max: 1.2 < 54.6 < 89.7 IQR (CV) : 32.1 (0.5)	46 distinct values	: : : : . . : : : : : : : : : : : : : : :	46 (97.9%)	1 (2.1%)
4	Examination [integer]	Mean (sd) : 16.5 (8) min < med < max: 3 < 16 < 37 IQR (CV) : 10 (0.5)	22 distinct values	: : : . : : : . : : : : : : : : : : .	47 (100.0%)	0 (0.0%)

No	Variable	Stats / Values	Freqs (% of Valid)	Graph	Valid	Missing
5	Education [integer]	Mean (sd) : 11 (9.6) min < med < max: 1 < 8 < 53 IQR (CV) : 6 (0.9)	19 distinct values	: : : : : : : .	47 (100.0%)	0 (0.0%)
6	Catholic [numeric]	Mean (sd) : 41.1 (41.7) min < med < max: 2.1 < 15.1 < 100 IQR (CV) : 87.9 (1)	46 distinct values	: : . : : : : : . :	47 (100.0%)	0 (0.0%)
7	Infant.Mortality [numeric]	Mean (sd) : 21.6 (12) min < med < max: 10.8 < 20 < 100 IQR (CV) : 4.1 (0.6)	38 distinct values	: . : : : : : : :	47 (100.0%)	0 (0.0%)

EDA: summaries per group

```
swiss |>
  group_by(Catholic > 50) |>
  summarize(mean(Fertility))
```

```
# A tibble: 2 × 2
  `Catholic > 50` `mean(Fertility)`
  <lgl>                       <dbl>
1 FALSE                        66.2
2 TRUE                         76.5
```

```
swiss |>
  group_by(Catholic > 50) |>
  summarize(across(.cols = c(Fertility, Education),
                   .fns = list("min" = min, "mean" = mean, "max" = max)))
```

```
# A tibble: 2 × 7
  `Catholic > 50` Fertility_min Fertility_mean Fertility_max Education_min Education_mean
  <lgl>                   <dbl>          <dbl>         <dbl>         <int>          <dbl>
1 FALSE                    35             66.2          85.8             1          12.1
2 TRUE                     42.8           76.5          92.5             2           9.11
# ℹ 1 more variable: Education_max <int>
```

The problem of missing values: potential bias

Bias!

- Missing at random
- Not missing at random

How to solve the problem of missing values

- **Complete case analysis** (listwise deletion): remove all observations with missing values.
- **Imputation**: replace missing values with a plausible value.
- **Multiple imputation**: create multiple datasets with plausible values.
- **Model-based imputation**: use a model to predict missing values.

An example of handling missing values

▶ Show example

Important commands

- `na.omit()`
- `f(..., na.rm = TRUE)` like in `mean()`, `sum()`, etc.

EDA: a challenge on summarizing categorical variables

Use what we just saw in the lecture to solve the following problem 💪. You have the following dataset:

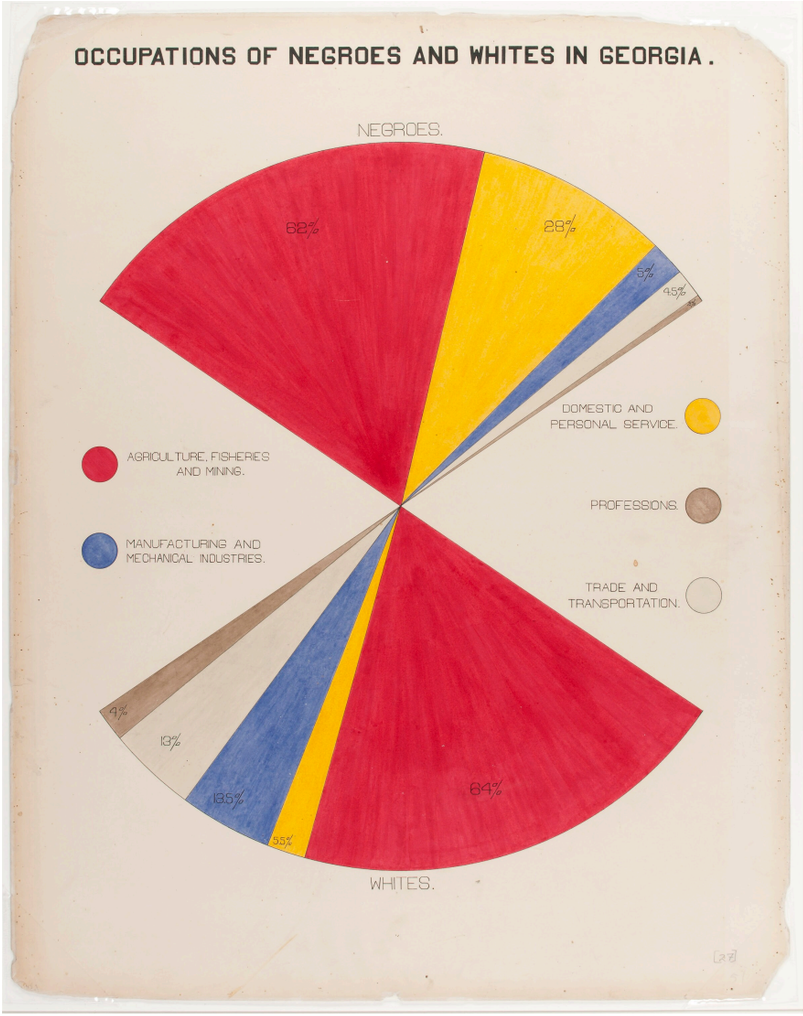
```
df_p <- data.frame(id = 1:5,
                   first_name = c("Anna", "John", "Claire", "Evan", "Brigitte"),
                   profession = c("Economist", "Data Scientist",
                                  "Data Scientist", "Economist", "Economist"),
                   salaryK = c("100 ", 120, 90, 110, 105),
                   experienceY = c(10, 10, 10, 10, 10))

df_p
```

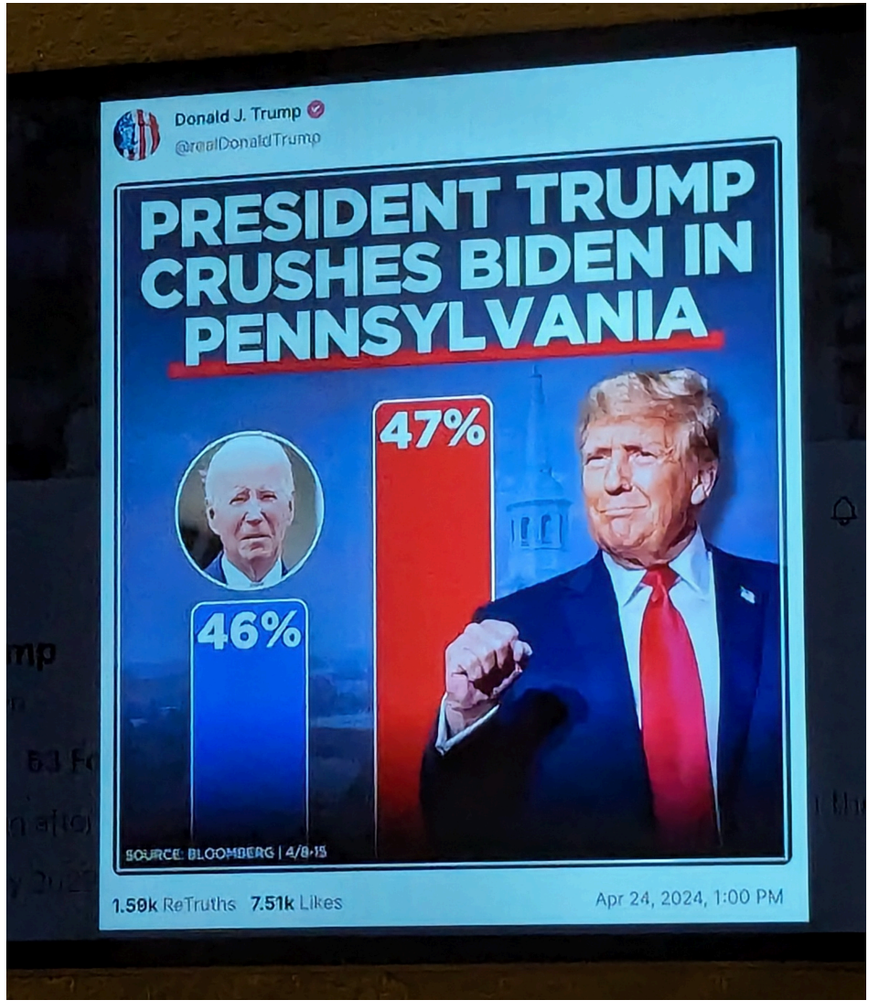
1. Clean the data
2. Summarize the data.
3. Give summary statistics on the categorical variable “profession”. What can you show, and how can you code it?
4. You are interested in quantifying the gender pay gap. Prepare the data accordingly and give an estimate of the gender pay gap.

Visualizing data

Data visualization is THE key to effectively delivering insights and to convince people



Du Bois, 1900



Trump, 2024

Data visualization through tables and graphs

- Two ways: display data through *tables* or *graphs*.
- Depends on the purpose.

Data visualization through tables and graphs

A chart typically contains at least one axis, the values are represented in terms of visual objects (dots, lines, bars) and axes typically have scales or labels.

- If we are interested in exploring, analyzing or communicating **patterns** in the data, charts are more useful than tables.

A table typically contains rows and columns, and the values are represented by text.

- If we are interested in exploring, analyzing or communicating **specific numbers** in the data, tables are more useful than graphs.

Tables

Tables

- Formatting data values for publication.
- Typical: String operations to make numbers and text look nicer.
  - Before creating a table or figure…

Tables

```
# load packages and data
library(tidyverse)
data("swiss")

# compute summary statistics
swiss_summary <- swiss |>
  summarise(avg_education = mean(Education),
            avg_fertility = mean(Fertility),
            N = n()
  )

swiss_summary
```

```
  avg_education avg_fertility  N
1      10.97872      70.14255 47
```

Problems?

Tables: round numeric values

```
swiss_summary_rounded <- round(swiss_summary, 2)
swiss_summary_rounded
```

```
  avg_education avg_fertility  N
1         10.98         70.14 47
```

Tables: detailed formatting of numbers

- Coerce to text.
- String operations.
- Decimal marks, units (e.g., currencies), other special characters for special formats (e.g. coordinates).
- `format()`-function

Tables: `format()` example

```
swiss_form <- format(swiss_summary_rounded,
                     decimal.mark = ",")
swiss_form
```

```
  avg_education avg_fertility  N
1         10,98         70,14 47
```

See also the helpful functions for formatting text-strings

- Uppercase/lowercase: `toupper()`/`tolower()`.
- Remove white spaces: `trimws()`,

```
string <- "AbCD "
toupper(string)
```

```
[1] "ABCD "
```

```
tolower(string)
```

```
[1] "abcd "
```

```
trimws(tolower(string))
```

```
[1] "abcd"
```

Get creative with tables: `gtExtras` and sparklines

```
head(USArrests, 10)
```

```
            Murder Assault UrbanPop Rape
Alabama       13.2     236       58 21.2
Alaska        10.0     263       48 44.5
Arizona        8.1     294       80 31.0
Arkansas       8.8     190       50 19.5
California     9.0     276       91 40.6
Colorado       7.9     204       78 38.7
Connecticut    3.3     110       77 11.1
Delaware       5.9     238       72 15.8
Florida       15.4     335       80 31.9
Georgia       17.4     211       60 25.8
```

Showing a raw data frame is not visualization... Problems?

Get creative with tables: `gtExtras` and sparklines

```
library(gtExtras)

USArrests_summary <- USArrests |>
  mutate(UrbanPop = case_when(UrbanPop > quantile(UrbanPop, .66) ~ "High",
                              UrbanPop > quantile(UrbanPop, .33) ~ "Middle",
                              UrbanPop > 0 ~ "Low")) |>
  group_by(UrbanPop) |>
  summarize(
    "Mean murder" = mean(Murder),
    "SD murder" = sd(Murder),
    Density = list(Murder)
  )

USArrests_summary
```

```
# A tibble: 3 × 4
  UrbanPop `Mean murder` `SD murder` Density
  <chr>            <dbl>       <dbl> <list>
1 High              8.07        3.80 <dbl [17]>
2 Low               7.41        5.20 <dbl [17]>
3 Middle            7.89        4.18 <dbl [16]>
```

Get creative with tables: `gtExtras` and sparklines

```
USArrests_summary |>
  gt() |>
  tab_header(
    title = md("Murder rates"),
    subtitle = md("Per high, middle, and low urban population ")
  ) |>
  gtExtras::gt_plt_dist(Density, type = "density", line_color = "black",
                        fill_color = "red") %>%
  fmt_number(columns = `Mean murder`:`SD murder`, decimals = 2)
```

Murder rates

Per high, middle, and low urban population

UrbanPop	Mean murder	SD murder	Density
High	8.07	3.80	
Low	7.41	5.20	
Middle	7.89	4.18	

Get creative with tables: other sources

- `kable()` for `html` / Markdown reports
- `stargazer` for your LaTeX reports or for your Office Word reports

Get creative with tables: `kable()`

```
knitr::kable(head(USArrests, 5), format = "html")
```

	Murder	Assault	UrbanPop	Rape
Alabama	13.2	236	58	21.2
Alaska	10.0	263	48	44.5
Arizona	8.1	294	80	31.0
Arkansas	8.8	190	50	19.5
California	9.0	276	91	40.6

Graphs with R (ggplot2)

Graphs with R

Three main approaches:

1. The original `graphics` package ([@r_2018]; shipped with the base R installation).

Graphs with R

Three main approaches:

1. The original `graphics` package ([@r_2018]; shipped with the base R installation).
2. The `lattice` package [@lattice_2008], an implementation of the original Bell Labs 'Trellis' system.

Graphs with R

Three main approaches:

1. The original `graphics` package ([@r_2018]; shipped with the base R installation).
2. The `lattice` package [@lattice_2008], an implementation of the original Bell Labs 'Trellis' system.
3. The *`ggplot2`* package [@wickham_2016], an implementation of Leland Wilkinson's 'Grammar of Graphics'.

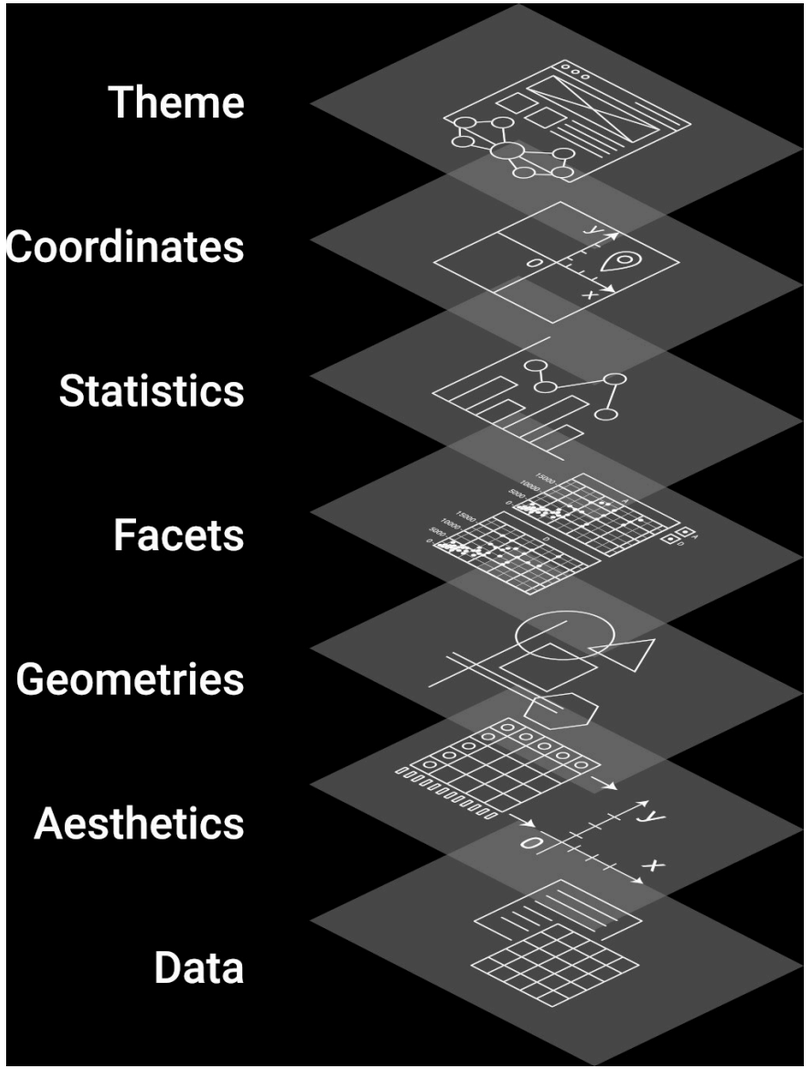
Graphs with R

Three main approaches:

1. The original `graphics` package ([@r_2018]; shipped with the base R installation).
2. The `lattice` package [@lattice_2008], an implementation of the original Bell Labs 'Trellis' system.
3. The *`ggplot2`* package [@wickham_2016], an implementation of Leland Wilkinson's 'Grammar of Graphics'.

`ggplot2` is so good that it has become *THE* reference [In python, use `plotnine` to apply the grammar of graphics.]

Grammar of graphics



ggplot2

`ggplot2` basics

Using `ggplot2` to generate a basic plot in R is quite simple. Basically, it involves three key points:

1. The data must be stored in a `data.frame`/`tibble` (in tidy format!).

`ggplot2` basics

Using `ggplot2` to generate a basic plot in R is quite simple. Basically, it involves three key points:

1. The data must be stored in a `data.frame`/`tibble` (in tidy format!).
2. The starting point of a plot is always the function `ggplot()`.

`ggplot2` basics

Using `ggplot2` to generate a basic plot in R is quite simple. Basically, it involves three key points:

1. The data must be stored in a `data.frame`/`tibble` (in tidy format!).
2. The starting point of a plot is always the function `ggplot()`.
3. The first line of plot code declares the data and the 'aesthetics' (e.g., which variables are mapped to the x-/y-axes):

`ggplot2` basics

Using `ggplot2` to generate a basic plot in R is quite simple. Basically, it involves three key points:

1. The data must be stored in a `data.frame`/`tibble` (in tidy format!).
2. The starting point of a plot is always the function `ggplot()`.
3. The first line of plot code declares the data and the 'aesthetics' (e.g., which variables are mapped to the x-/y-axes):

```
ggplot(data = my_dataframe, aes(x= xvar, y= yvar))
```

Example data set: `swiss`

```
library(tidyverse) # automatically loads ggplot2

# load the data
data(swiss)
head(swiss)
```

```
             Fertility Agriculture Examination Education Catholic Infant.Mortality
Courtelary        80.2        17.0          15        12     9.96             22.2
Delemont          83.1        45.1           6         9    84.84             22.2
Franches-Mnt      92.5        39.7           5         5    93.40             20.2
Moutier           85.8        36.5          12         7    33.77             20.3
Neuveville        76.9        43.5          17        15     5.16             20.6
Porrentruy        76.1        35.3           9         7    90.57             26.6
```

Add indicator variable

Code a province as ‘Catholic’ if more than 50% of the inhabitants are catholic:

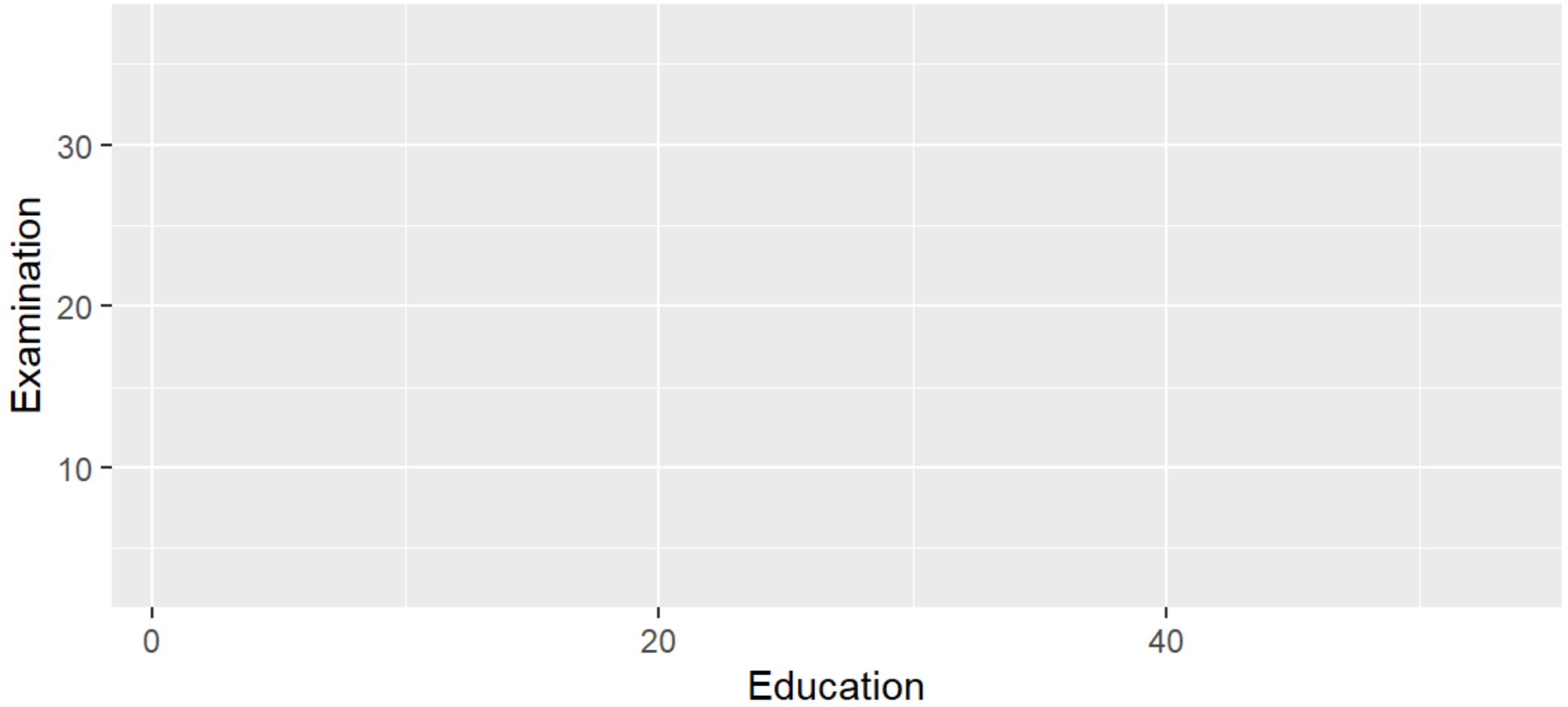
```
# via tidyverse/mutate
swiss <- mutate(swiss,
                Religion =
                  ifelse(50 < Catholic, 'Catholic', 'Protestant'))

# 'old school' alternative
swiss$Religion <- 'Protestant'
swiss$Religion[50 < swiss$Catholic] <- 'Catholic'

# set to factor
swiss$Religion <- as.factor(swiss$Religion)
```

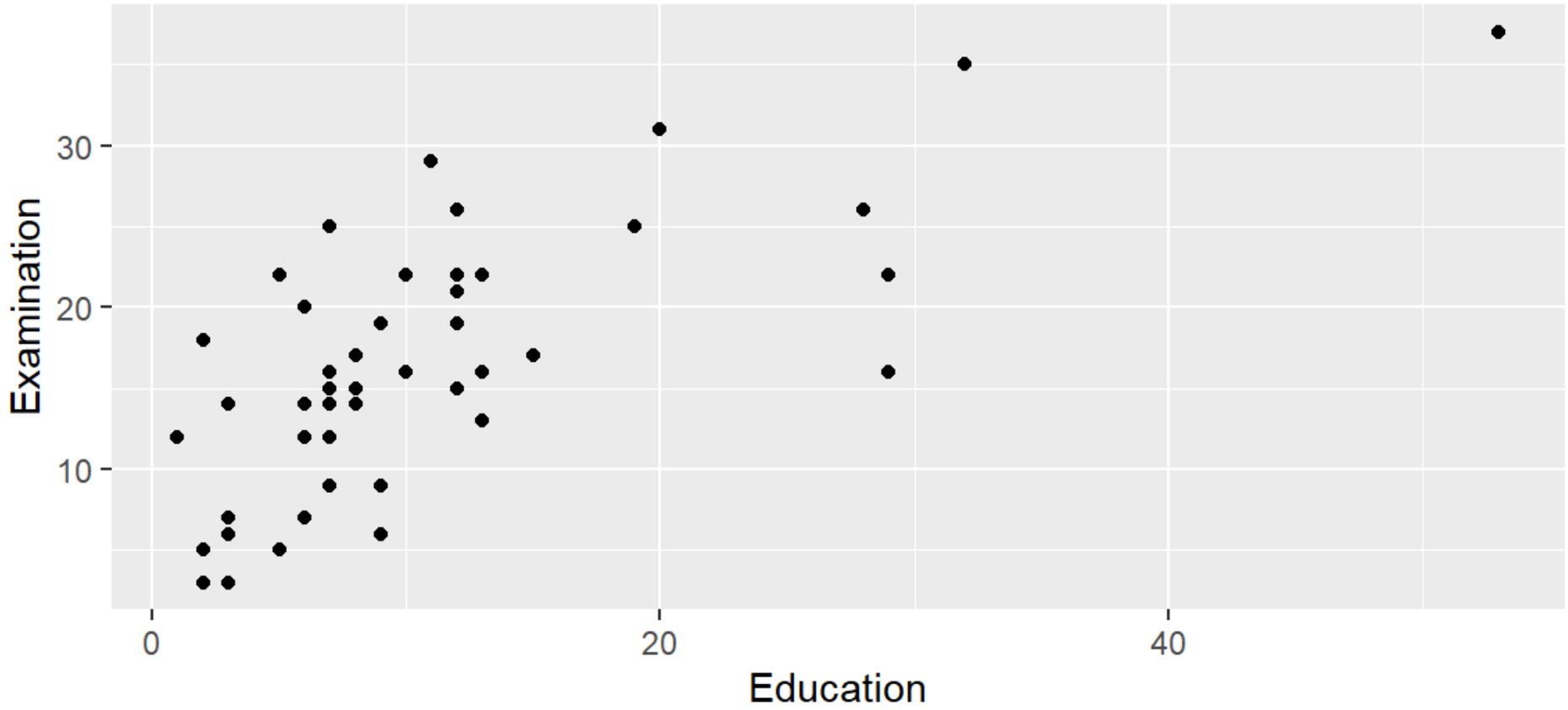
Data and aesthetics

```
ggplot(data = swiss, aes(x = Education, y = Examination))
```



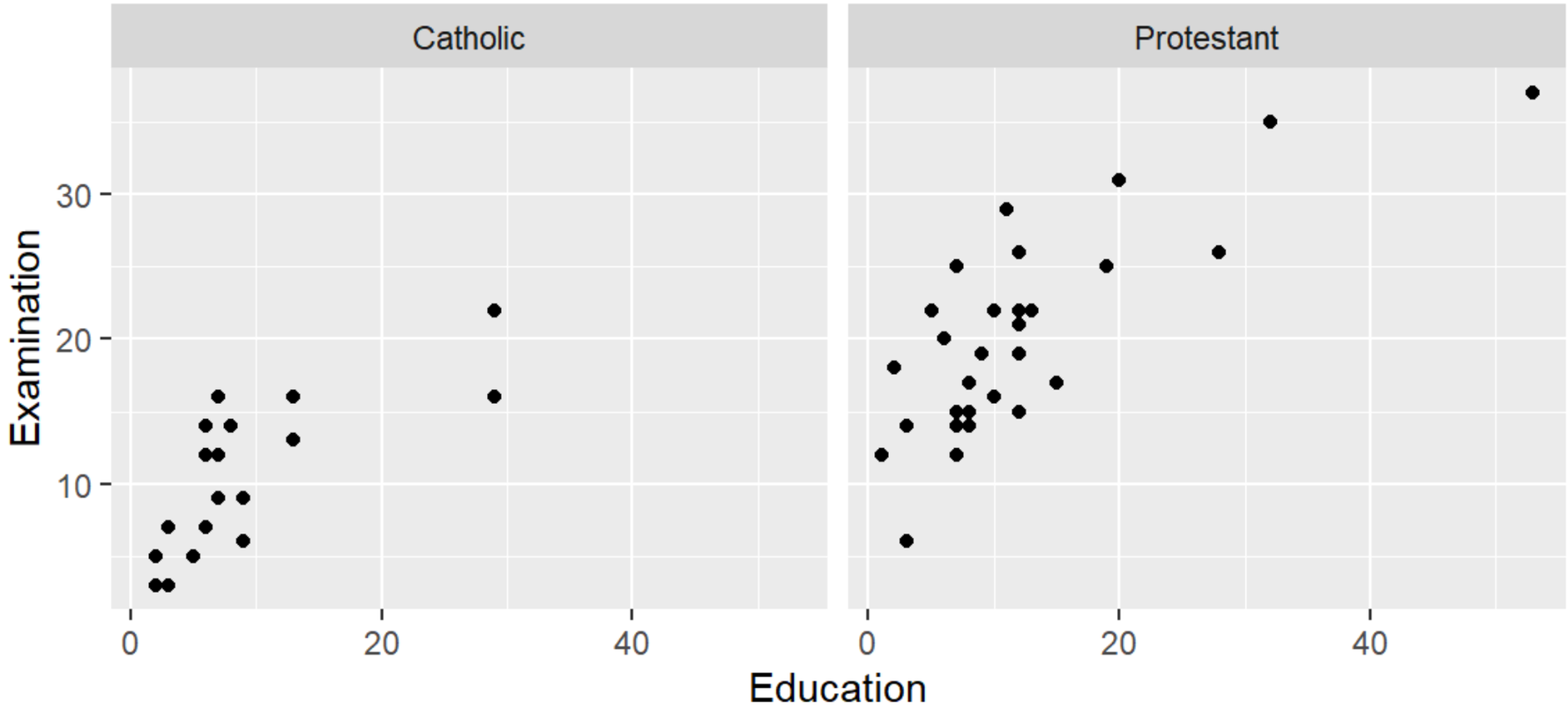
Geometries (~the type of plot)

```
ggplot(data = swiss, aes(x = Education, y = Examination)) +
    geom_point()
```



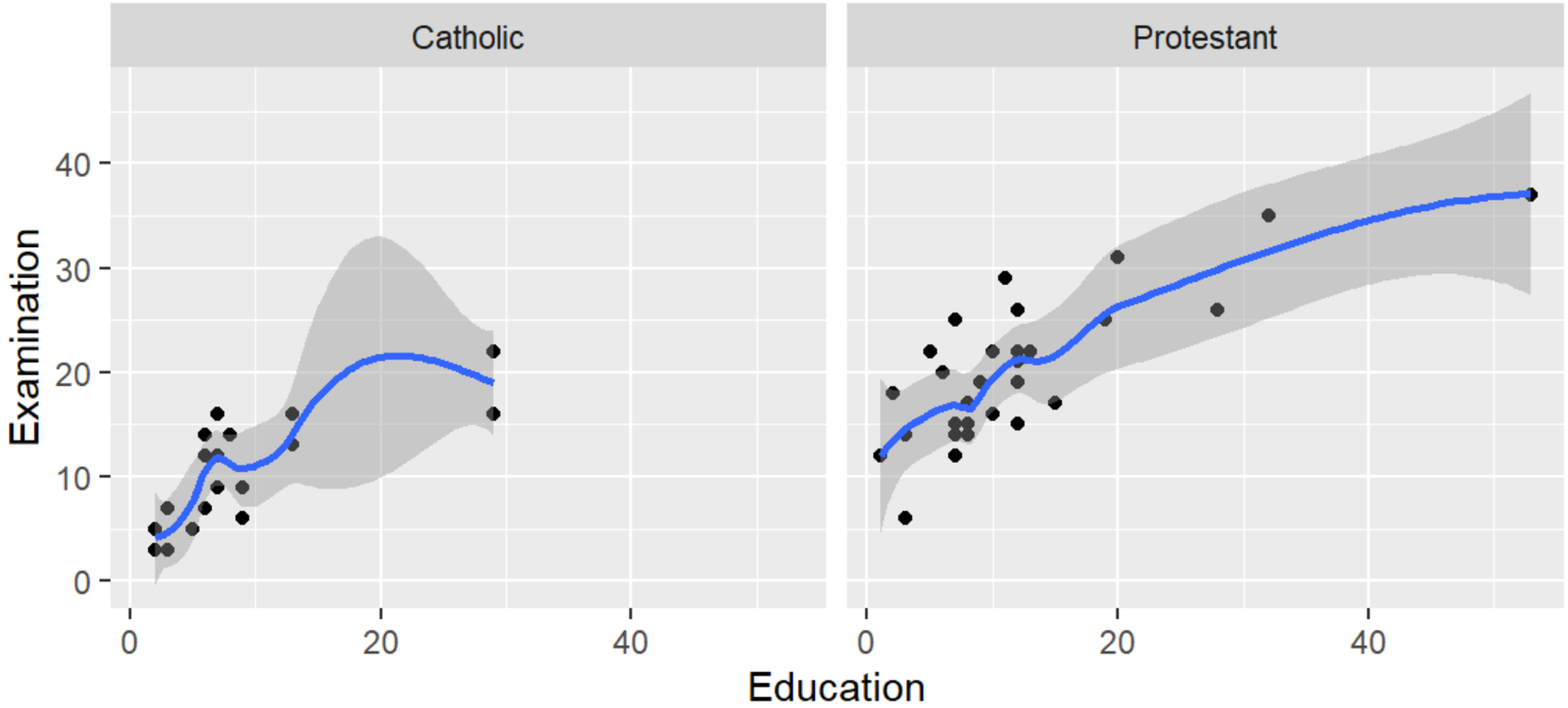
Facets

```
ggplot(data = swiss, aes(x = Education, y = Examination)) +
    geom_point() +
    facet_wrap(~Religion)
```



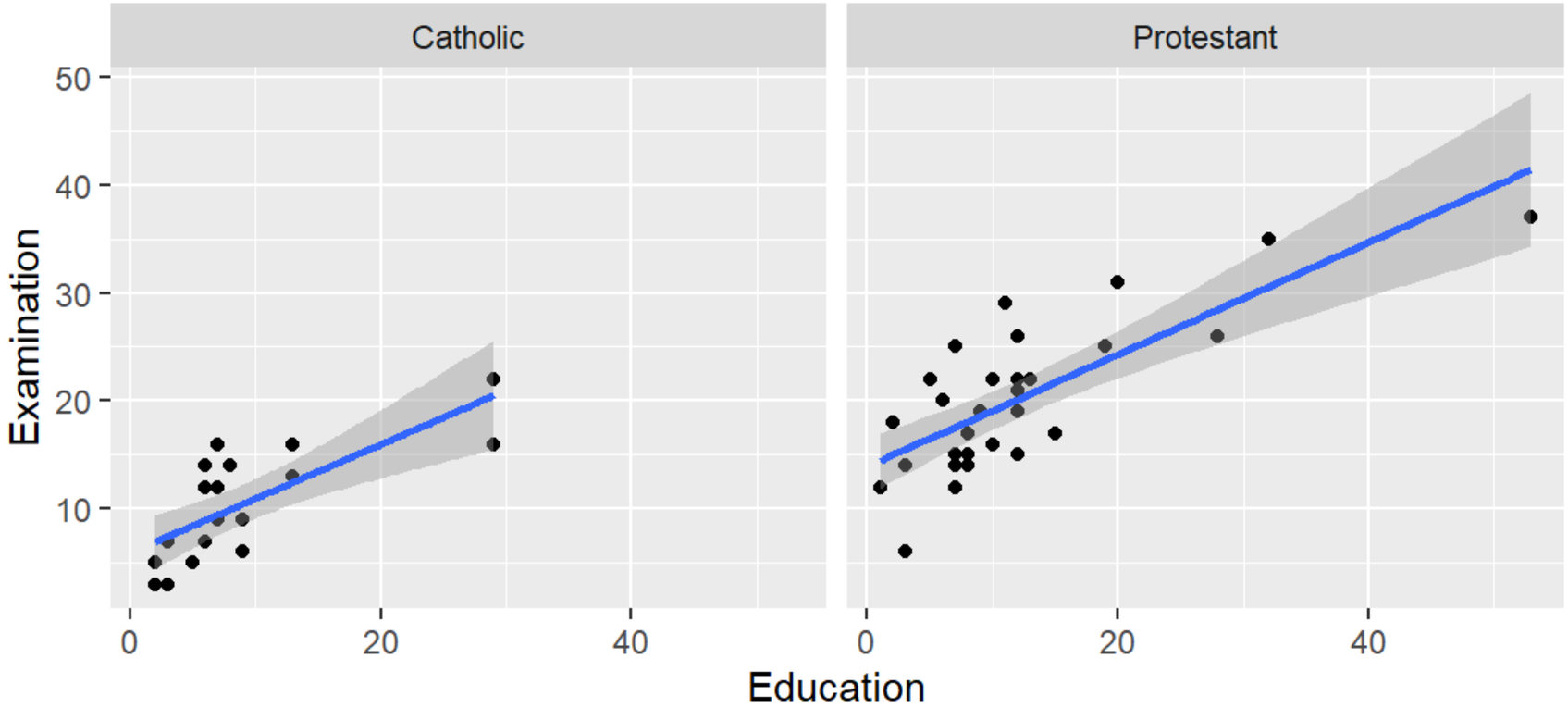
Additional layers and statistics

```
ggplot(data = swiss, aes(x = Education, y = Examination)) +
     geom_point() +
     geom_smooth(method = 'loess') +
     facet_wrap(~Religion)
```



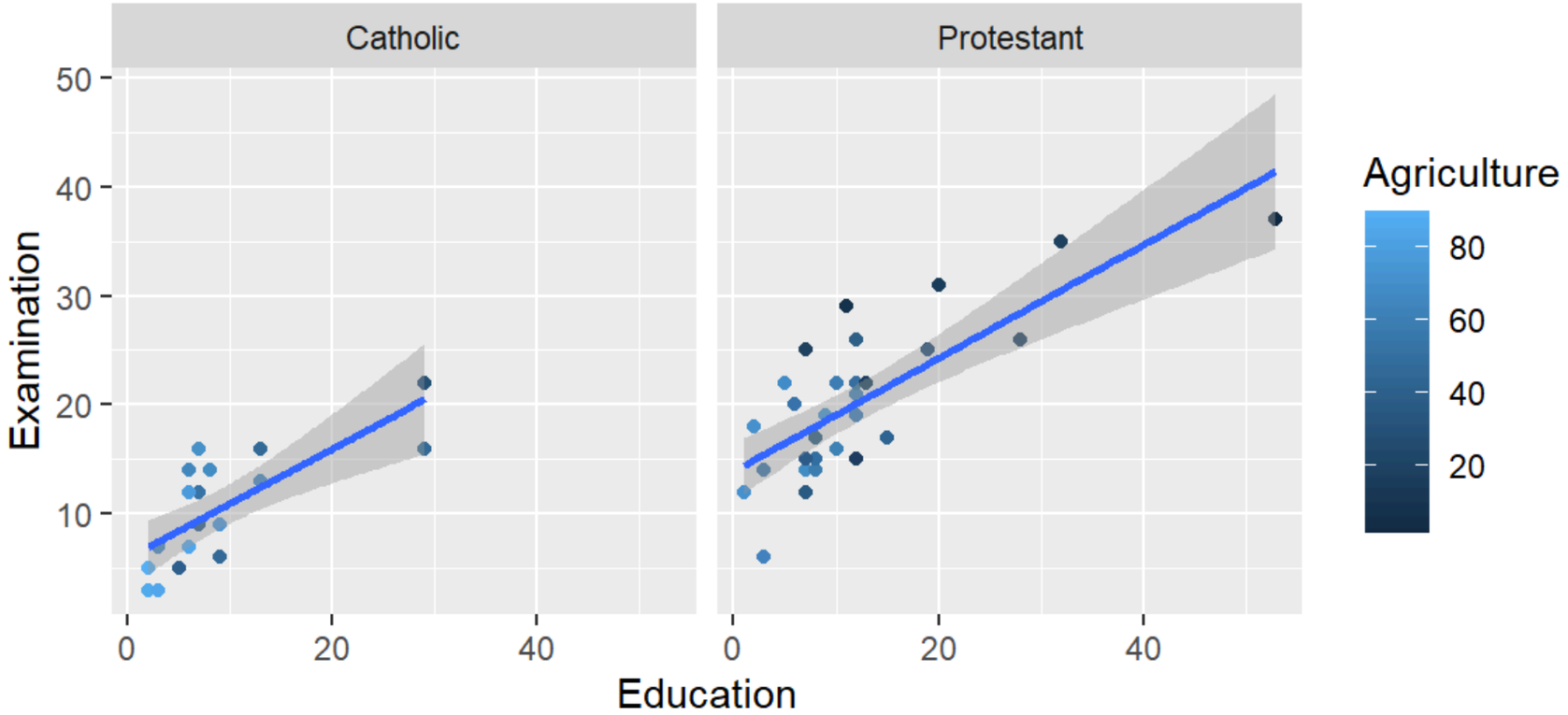
Additional layers and statistics

```
ggplot(data = swiss, aes(x = Education, y = Examination)) +
     geom_point() +
     geom_smooth(method = 'lm') +
     facet_wrap(~Religion)
```



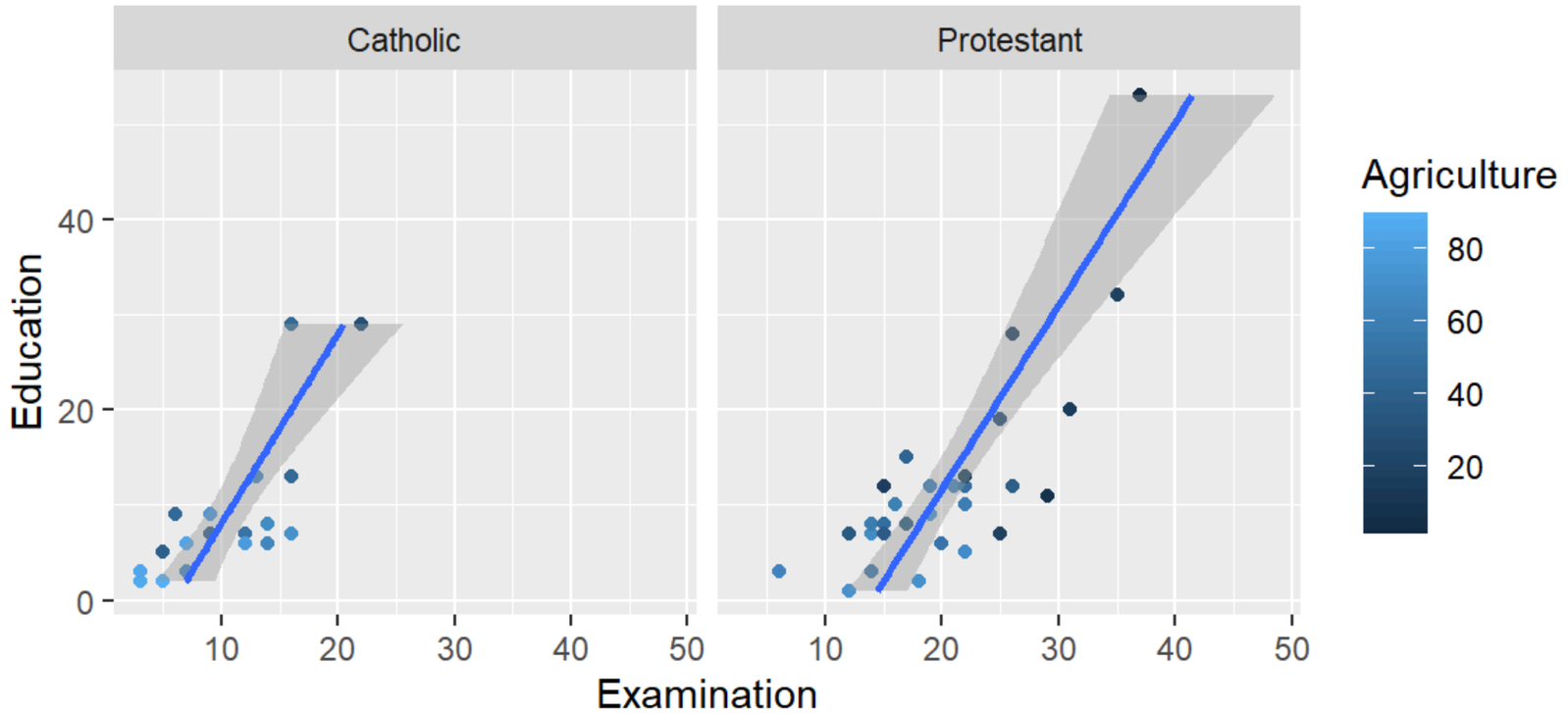
Additional aesthetics

```
ggplot(data = swiss, aes(x = Education, y = Examination)) +
    geom_point(aes(color = Agriculture)) +
    geom_smooth(method = 'lm') +
    facet_wrap(~Religion)
```



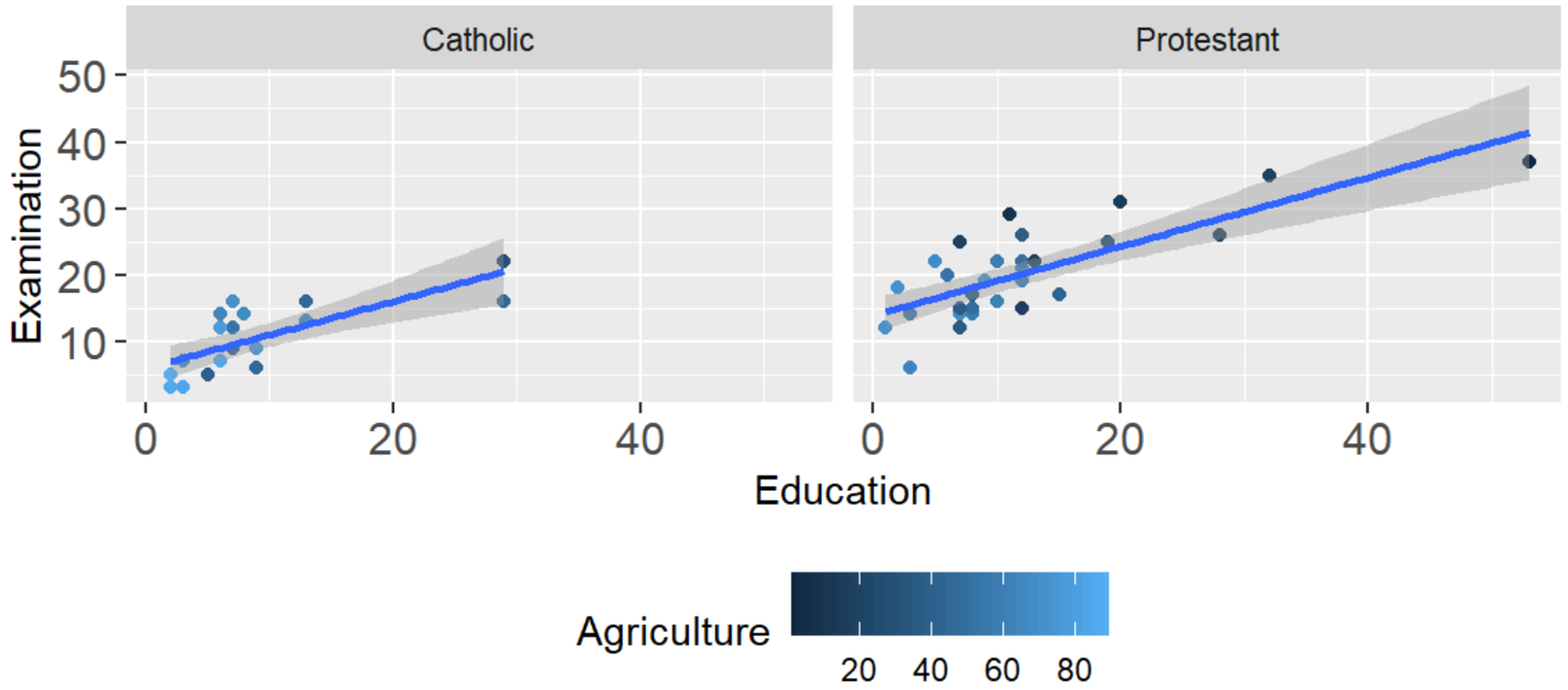
Change coordinates

```
ggplot(data = swiss, aes(x = Education, y = Examination)) +
     geom_point(aes(color = Agriculture)) +
     geom_smooth(method = 'lm') +
     facet_wrap(~Religion) +
     coord_flip()
```



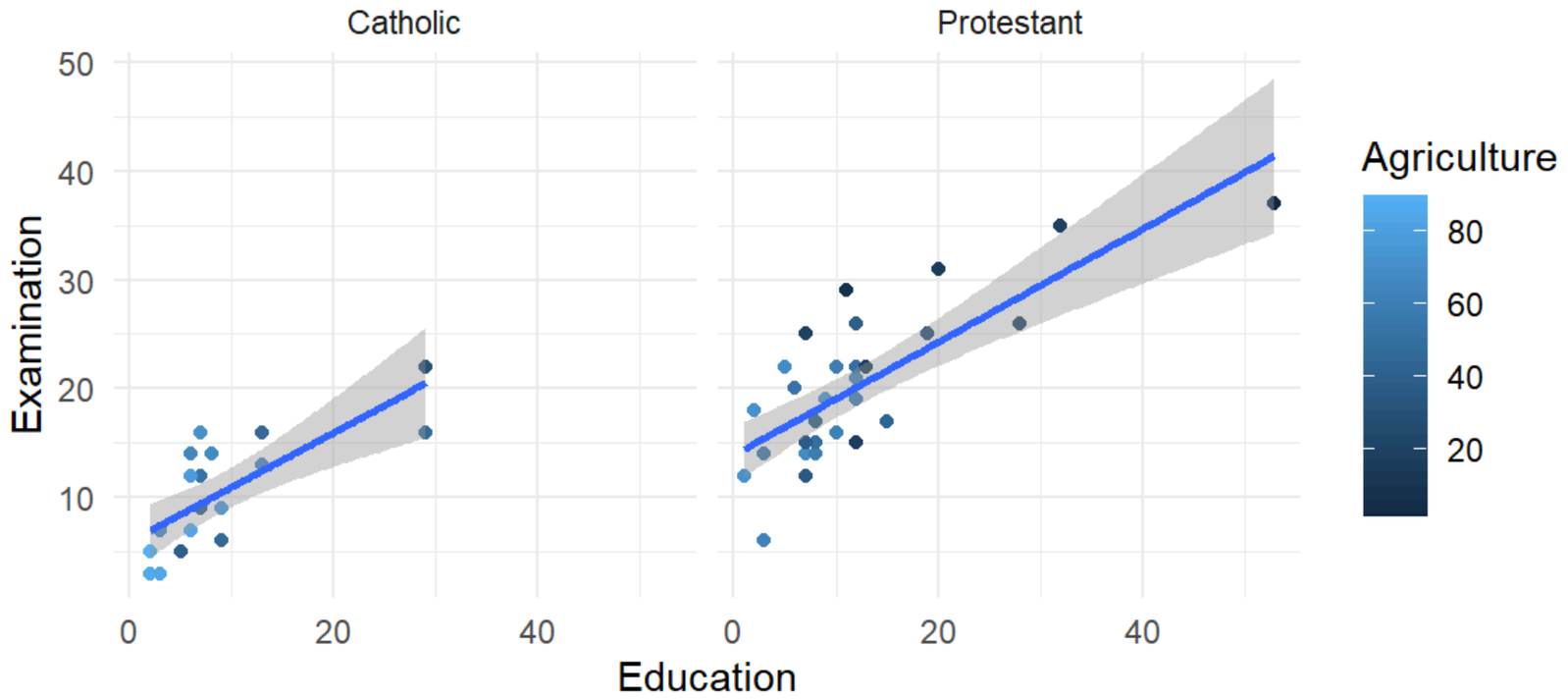
Themes

```
ggplot(data = swiss, aes(x = Education, y = Examination)) +
     geom_point(aes(color = Agriculture)) +
     geom_smooth(method = 'lm') +
     facet_wrap(~Religion) +
     theme(legend.position = "bottom", axis.text=element_text(size=12) )
```



Themes

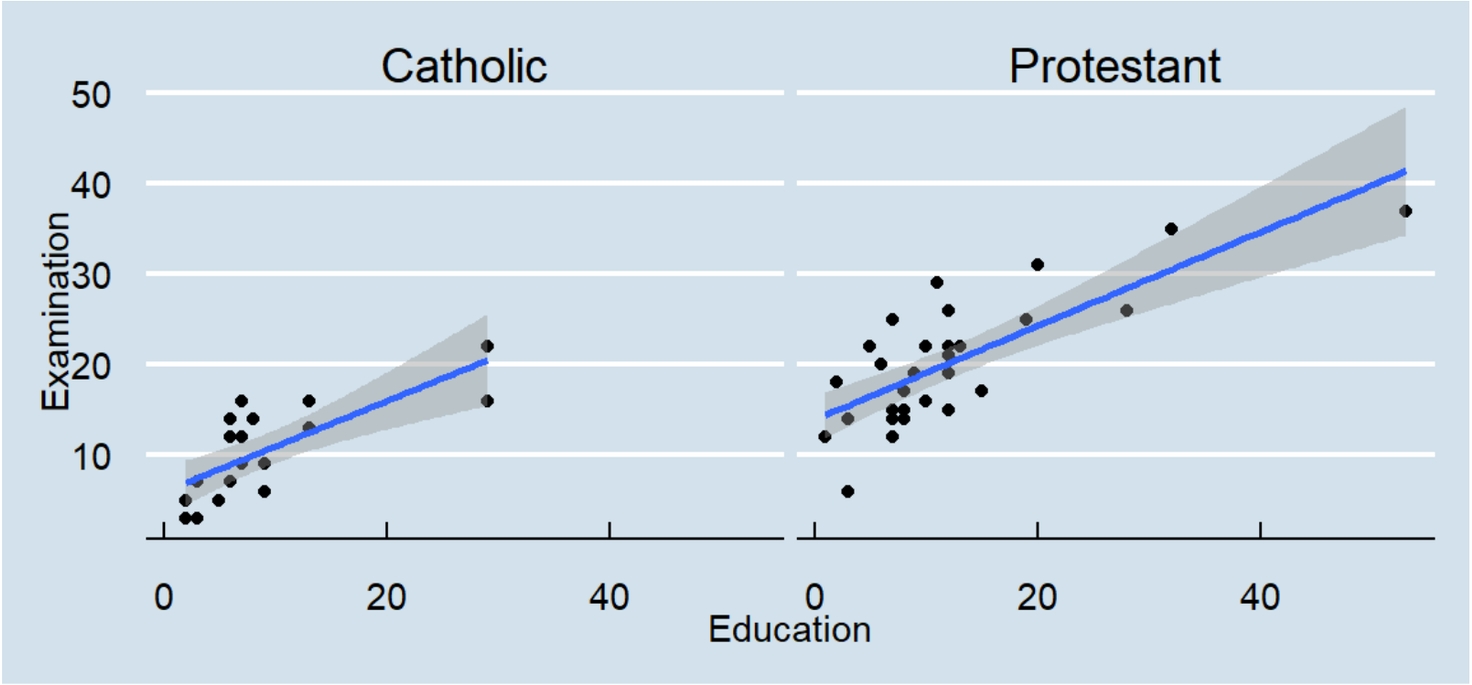
```
ggplot(data = swiss, aes(x = Education, y = Examination)) +
     geom_point(aes(color = Agriculture)) +
     geom_smooth(method = 'lm') +
     facet_wrap(~Religion) +
     theme_minimal()
```



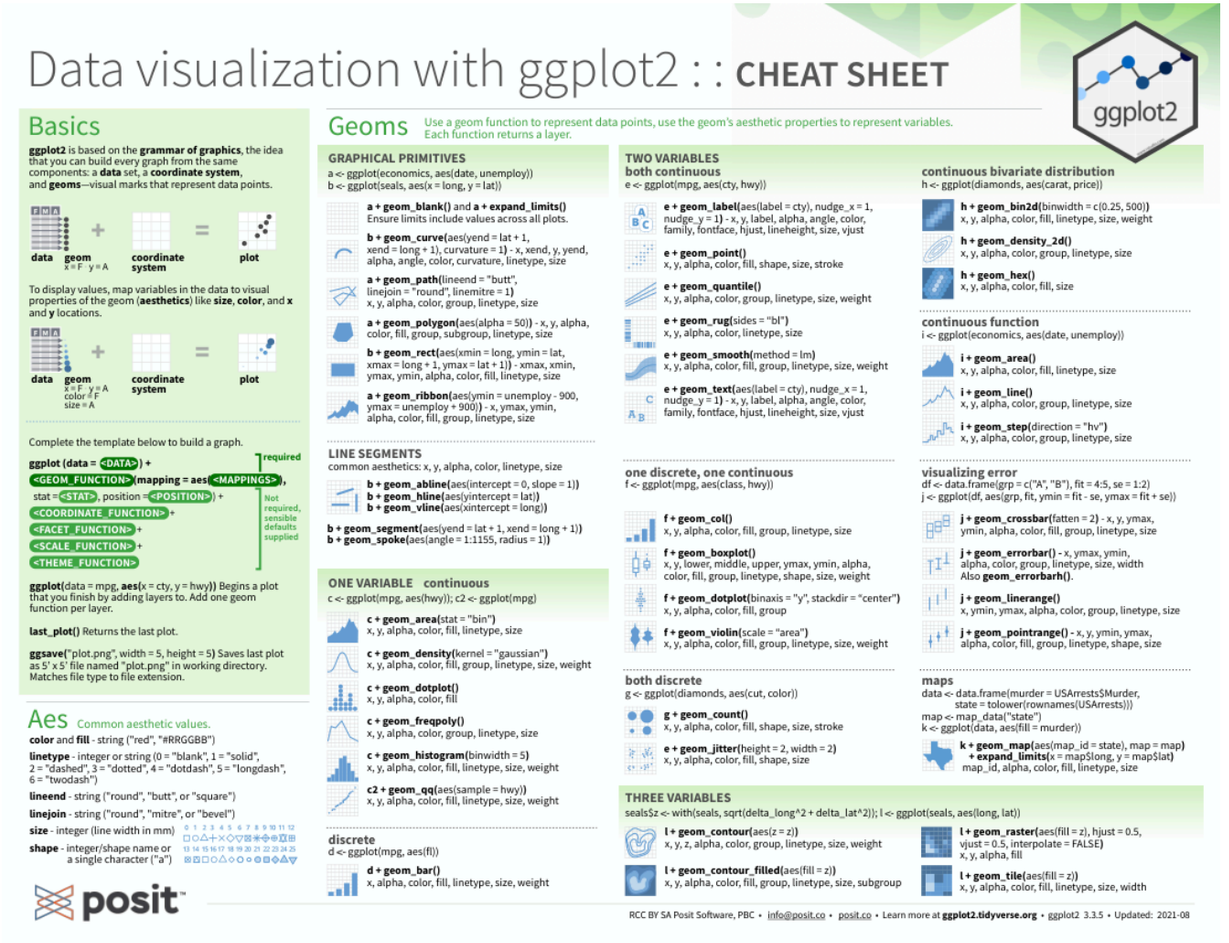
Themes

```
library(ggthemes)

ggplot(data = swiss, aes(x = Education, y = Examination)) +
     geom_point() +
     geom_smooth(method = 'lm') +
     facet_wrap(~Religion) +
     theme_economist()
```

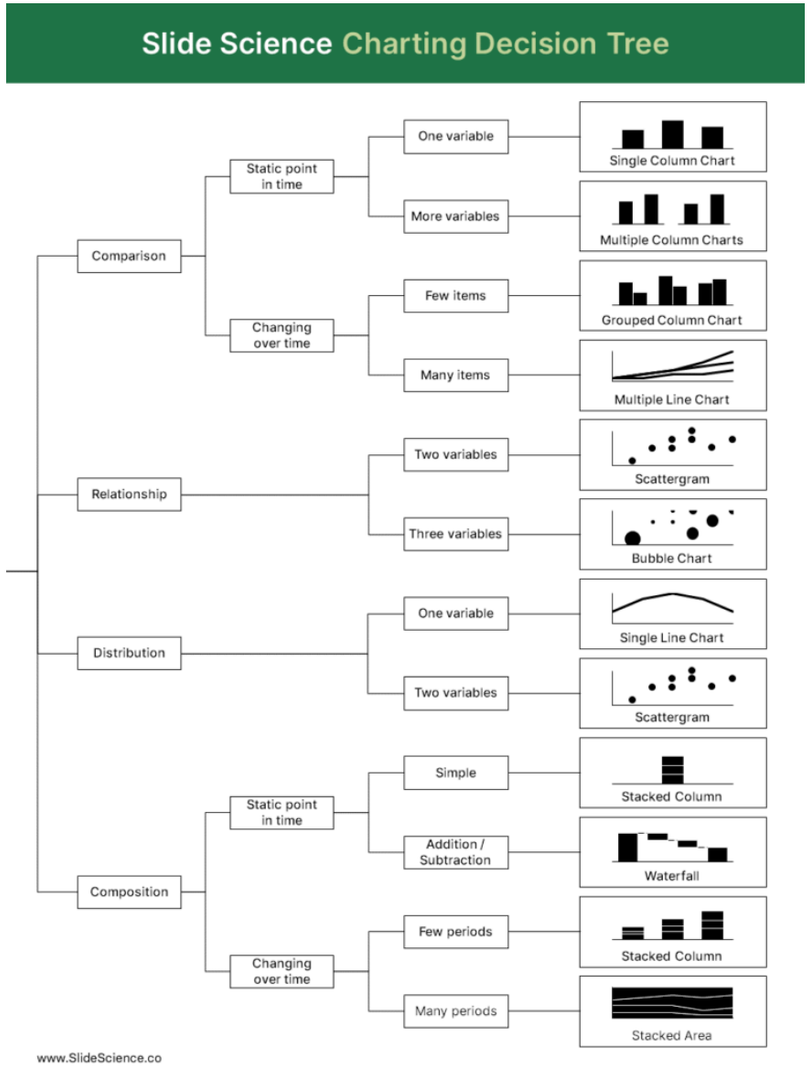


Cheat sheet for ggplot



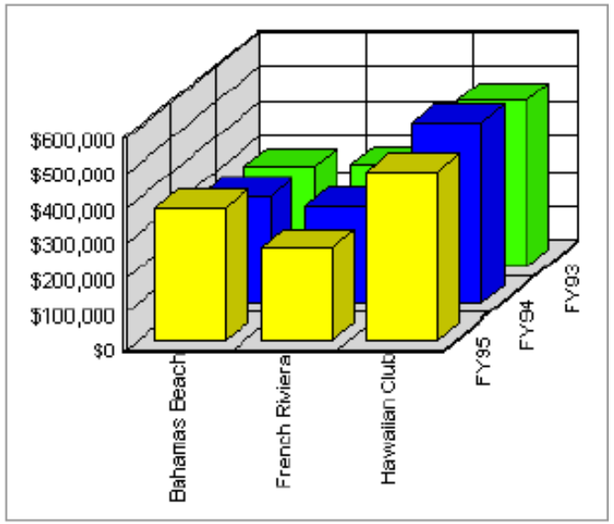
Link: https://rstudio.github.io/cheatsheets/html/data-visualization.html

What graph should I draw? A cheat sheet



Data viz: a challenge

Look at the graph below. What is wrong with it? Create your own version of the graph.



A Design Problem

Use the following data to create your own version of the graph:

▶ Show code