Tutorial 3: Week 6 PDF

Dimitrios Doudesis

21-10-2022

Contents

```r
library(tidyverse)
library(prettydoc) #for other theme options
```

```
## Warning: package 'prettydoc' was built under R version 4.0.5
```

```r
library(rmdformats) #for other theme options
library(gapminder) #for our data
```

```
## Warning: package 'gapminder' was built under R version 4.0.5
```

```r
library(knitr) #for report generation
library(kableExtra) #for pretty tables
```

```
## Warning: package 'kableExtra' was built under R version 4.0.5
```

```
#save the gapminder data into the working environment & create some variables for analysis
mydata <- gapminder %>%
  # round gdpPercap to 0 decimals:
  mutate(gdpPercap = round(gdpPercap)) %>%
  # divide pop by million, round to 1 decimal:
  mutate(pop_millions = (pop/1e6) %>% round(1))

# correlation for inline code example below
LePcor <- cor.test(mydata$lifeExp, mydata$pop)
```

Tables

When making tables, it is a good idea to use the `gt()` function from the `gt` package to produce nice looking tables rather than the usual r output.

Compare this table output:

```
mydata %>%
  select(-pop) %>%
  sample_n(10)
```

```
## # A tibble: 10 x 6
##    country                continent  year lifeExp gdpPercap pop_millions
##    <fct>                  <fct>     <int>   <dbl>     <dbl>        <dbl>
##  1 Djibouti               Africa     1977    46.5      3082          0.2
##  2 Montenegro             Europe     1987    74.9     11733          0.6
##  3 Bosnia and Herzegovina Europe     1967    64.8      2172          3.6
##  4 Slovak Republic        Europe     1957    67.4      6093          3.8
##  5 Congo, Rep.            Africa     1997    53.0      3484          2.8
##  6 Equatorial Guinea      Africa     1987    45.7       967          0.3
##  7 New Zealand            Oceania    1997    77.6     21050          3.7
##  8 Botswana               Africa     2002    46.6     11004          1.6
##  9 Portugal               Europe     2002    77.3     19971         10.4
## 10 Angola                 Africa     1962    34        4269          4.8
```

To this one:

```
mydata %>%
  select(-pop) %>%
  sample_n(10) %>%
  kbl()
```

Table 1: Any title that you want

Country	Continent	Year	Life Expectancy	GDP per Capita	Population (millions)
Belgium	Europe	2007	79.441	33693	10.4
Djibouti	Africa	1972	44.366	3694	0.2
Colombia	Americas	1977	63.837	3816	25.1
Mali	Africa	1987	46.364	684	7.6
West Bank and Gaza	Asia	1952	43.160	1516	1.0
Turkey	Europe	1982	61.036	4241	47.3
Chile	Americas	1967	60.523	5107	8.9
Mexico	Americas	2007	76.195	11978	108.7
Nepal	Asia	1952	36.157	546	9.2
Uruguay	Americas	2007	76.384	10611	3.4

country	continent	year	lifeExp	gdpPercap	pop_millions
Bulgaria	Europe	1992	71.19000	6303	8.7
Swaziland	Africa	2007	39.61300	4513	1.1
Honduras	Americas	1962	48.04100	2291	2.1
Libya	Africa	1962	47.80800	6757	1.4
Angola	Africa	1962	34.00000	4269	4.8
Tanzania	Africa	1962	44.24600	722	10.9
Argentina	Americas	1982	69.94200	8998	29.3
China	Asia	1962	44.50136	488	665.8
Nicaragua	Americas	1982	59.29800	3470	3.0
Kuwait	Asia	1962	60.47000	95458	0.4

For further styling options, the vignette and bookdown chapter can help!

Below I have grouped the data by `continent`, renamed the variables/columns from what they are called in the dataset, and added a caption to the table.

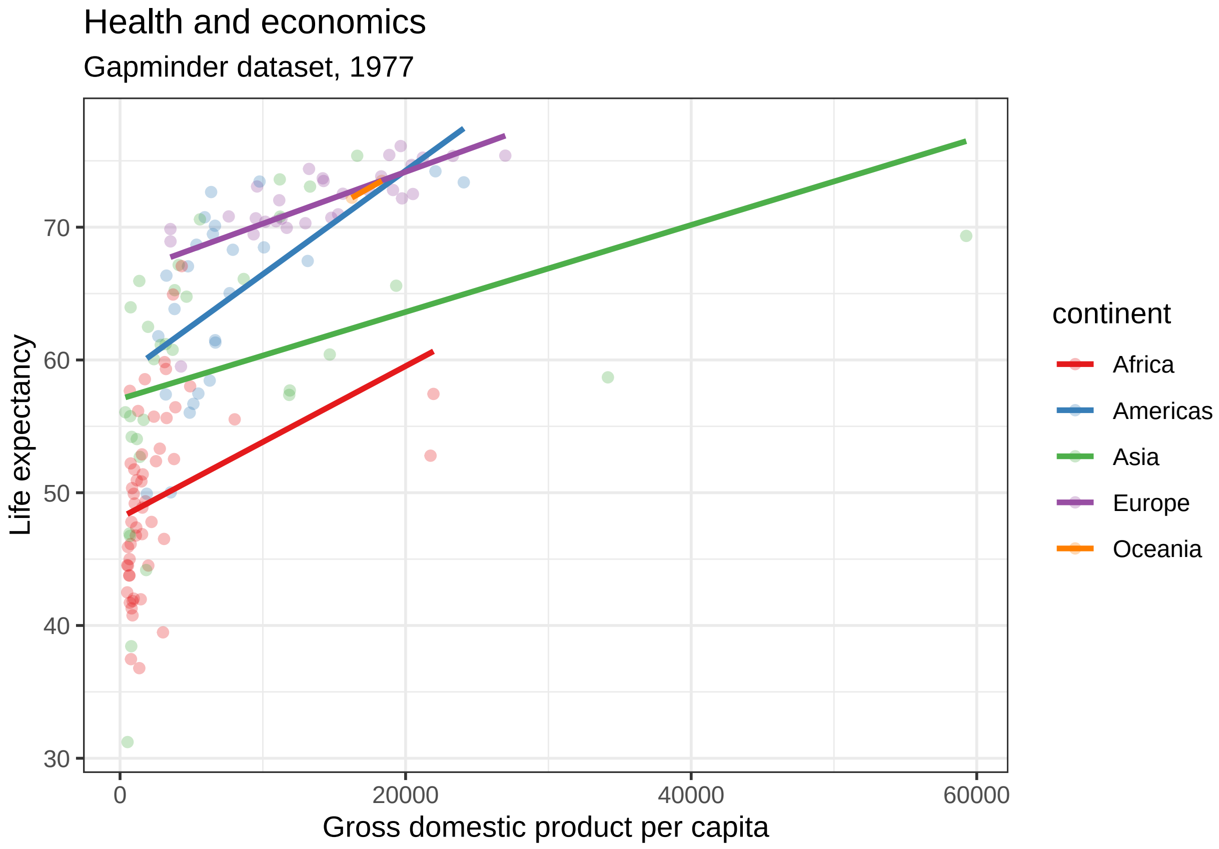
```r
mydata %>%
  select(-pop) %>%
  sample_n(10) %>% # this line of code, randomly selectes 10 countries
  kbl(
    caption = "Any title that you want",
    col.names = c(
      country = "Country",
      continent = "Continent",
      year = "Year",
      lifeExp = "Life Expectancy",
      gdpPercap = "GDP per Capita",
      pop_millions = "Population (millions)"
    )
  ) %>%
  kable_styling() # Bootstrap theme
```

```r
# Try replace "kable_styling" with:
# "kable_paper("hover", full_width = F)"
# "kable_classic(full_width = F, html_font = "Cambria")"
# "kable_minimal()" for different themes!
```

Figures

You can easily integrate figures into your report as well using a code chunk dedicated to the figure of interest.

```
mydata %>%
  filter(year == 1977) %>%
  ggplot(aes(y = lifeExp, x = gdpPercap, colour = continent)) +
  geom_point(alpha = 0.3) +
  theme_bw() +
  geom_smooth(method = "lm", se = FALSE) +
  scale_colour_brewer(palette = "Set1") +
  labs(x = "Gross domestic product per capita",
       y = "Life expectancy",
       title = "Health and economics",
       subtitle = "Gapminder dataset, 1977")
```



Images

You can easily insert images into your R Markdown file either from a webpage or from a file in the same directory:

1. From a webpage

Figure 1: Beach Chairs

To get the link to a picture from a webpage go to your internet browser of choice, Images, and right click on the image and select "Copy Image Link".

Note: When knitting to pdf, you need to add another line of code in the YAML for LaTeX `graphicx` to pull an image from the internet and then save them locally in the same directory as the RMD file.

```
`output:
   pdf_document:
     pandoc_args: ["--extract-media", "."]`
```

2. From a file

Note: to knit this document on your own device, you will need to change the file path above in number 2. "figures/Map of Health Boards.png" means go to the "figures" folder and find the file called "Map of Health Boards.png". If you do not have a figures folder within your working environment, you will get an error when trying to knit the document.

Inline Code

You can also use r code directly in text. For example:

> There are 1704 observations in the data set.

This can be an incredible time saver if you are creating a report where the data itself might still be coming in. Rather than needing to manually update descriptive information such as how many observations there are in the data set, let R do it for you!

I also will often use this when reporting in text results of analyses. For example:

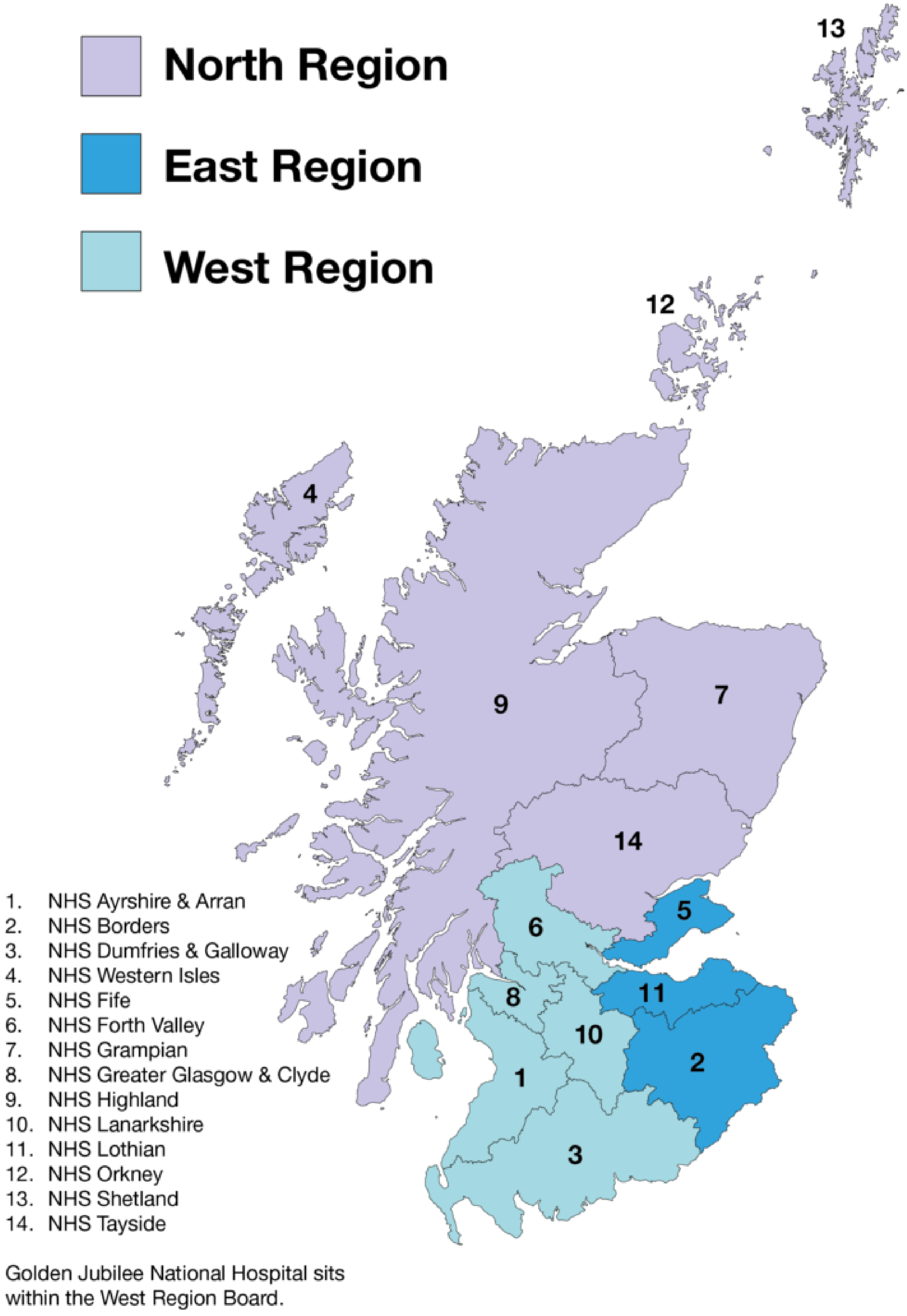


Figure 2: NHS Scotland HBs

The correlation between life expectancy and population is $r(1702) = 0.065$, $p = 0.007$

Mathematical notation

You will notice above that I used something else in combination with the inline r code - mathematical notation! This is based on LaTeX. In side a text chunk, you can use mathematical notation if you surround it by dollar signs `$` for "inline mathematics" and `$$` for "displayed equations". This post by R Pruim is my go-to resource for mathematical notation in R Markdown.

Functionalities

HTML

In general, I find that when knitting to an `html_document` there are more interesting functionalities. Indeed, Markdown was originally designed for HTML output so the HTML format has the richest features.

- floating table of contents
- code download output option
- code folding (hide or show)
- tabbed sections

You can have a look at some of the other options when knitting to HTML in the HTML document chapter in R Markdown: The Definitive Guide

Chunk names are not necessarily required to knit, but are good practice and help to support more advanced knitted approaches. Chunk labels should be

- unique
- do not use spaces, rather use dashes (-)
- use alphanumeric characters (a-z, A-Z and 0-9), because they are not special characters and will surely work for all output formats
- spaces and underscores in particular, may cause trouble in certain packages, such as `bookdown`.

We do not cover the `bookdown` package as part of this course, but it extends the functionality of rmarkdown to allow for figures and tables to be easily cross-referenced within your text (among other things).

PDF

PDFs in R Markdown also have a variety of features, but not nearly as many as HTML documents. To find out more, check out the PDF document chapter in R Markdown: The Definitive Guide

Word

The most notable feature of Word documents in R Markdown is the ability to create a "style reference document". To find out more, including a short video on how to create and use a reference document, check out the Word document chapter in R Markdown: The Defintive Guide

Themes

In general, there are a variety of theme options available to use with R Markdown HTML documents

`prettydoc`

The `prettydoc` package includes a variety of other theme options when knitting to HTML: https://prettydoc.statr.me/index.html

Once you have installed the package, you can open a prettydoc formatted document from "From Template" tab when choosing to create a new R Markdown file.

To use a `prettydoc` theme not from a template, you need to edit the YAML accordingly:

```
`output:
    prettydoc::html_pretty:
    theme: cayman`
```

Note: When using `html_pretty` engine for themes, `code_folding`, `code_download`, and `toc_float` are not applicable.

`rmdformats`

The `rmdformats` package includes a variety of other theme options when knitting to HTML: https://github.com/juba/rmdformats. Some themes allow for things like a dynamic table of contents, but not all of them. See the "Features matrix" table on the above webpage for more info.

Similar to above, once you have installed the package, you can open a rmdformts formatted document from "From Template" tab when choosing to create a new R Markdown file.

To use a `rmdformats` theme not from template, you need to edit the YAML accordingly:

```
`output:
    rmdformats::robobook`
```

Reproducibility!

In practice, depending on your audience, you will need to decide to show your code or not. It is unlikely that you will want to show the code used to produce your analysis, tables, or figures to an audience unfamiliar with R and would therefore set `echo = FALSE` in the set-up chunk. For this course, though, and in particular for the programming assignment, you will need to set `echo=TRUE` so that we can see your code and the product of that code.

While it can take up space, it is good practice to finish a document calling the `sessionInfo` function, which lists all of the packages you used, their versions, and more.

```
## R version 4.0.4 (2021-02-15)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 22000)
##
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=English_United Kingdom.1252
## [2] LC_CTYPE=English_United Kingdom.1252
## [3] LC_MONETARY=English_United Kingdom.1252
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United Kingdom.1252
## 
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
## 
## other attached packages:
##  [1] kableExtra_1.3.4 knitr_1.40       gapminder_0.3.0  rmdformats_1.0.4
##  [5] prettydoc_0.4.1  forcats_0.5.1    stringr_1.4.0    dplyr_1.0.10
##  [9] purrr_0.3.4      readr_2.1.3      tidyr_1.2.1      tibble_3.1.8
## [13] ggplot2_3.3.6    tidyverse_1.3.2
## 
## loaded via a namespace (and not attached):
##  [1] Rcpp_1.0.6          lattice_0.20-41     svglite_2.0.0
##  [4] lubridate_1.8.0     assertthat_0.2.1    digest_0.6.27
##  [7] utf8_1.2.1          R6_2.5.0            cellranger_1.1.0
## [10] backports_1.2.1     reprex_2.0.2        evaluate_0.17
## [13] highr_0.8           httr_1.4.2          pillar_1.8.1
## [16] rlang_1.0.5         googlesheets4_1.0.1 readxl_1.3.1
## [19] rstudioapi_0.13     Matrix_1.3-2        rmarkdown_2.17
## [22] labeling_0.4.2      splines_4.0.4       webshot_0.5.2
## [25] googledrive_2.0.0   munsell_0.5.0       broom_1.0.1
## [28] compiler_4.0.4      modelr_0.1.8        xfun_0.34
## [31] systemfonts_1.0.1   pkgconfig_2.0.3     mgcv_1.8-33
## [34] htmltools_0.5.2     tidyselect_1.1.2    bookdown_0.21
## [37] viridisLite_0.3.0   fansi_0.4.2         crayon_1.5.2
## [40] tzdb_0.3.0          dbplyr_2.2.1        withr_2.4.1
## [43] grid_4.0.4          nlme_3.1-152        jsonlite_1.7.2
## [46] gtable_0.3.0        lifecycle_1.0.2     DBI_1.1.1
## [49] magrittr_2.0.3      scales_1.2.1        cli_3.4.1
## [52] stringi_1.5.3       farver_2.1.0        fs_1.5.2
## [55] xml2_1.3.3          ellipsis_0.3.2      generics_0.1.0
## [58] vctrs_0.4.2         RColorBrewer_1.1-2  tools_4.0.4
## [61] glue_1.6.2          hms_1.1.2           fastmap_1.1.0
## [64] yaml_2.2.1          colorspace_2.0-0    gargle_1.2.1
## [67] rvest_1.0.3         haven_2.5.1
```