Data Analysis
Week 4: Report writing using R Markdown

1 Introduction

This week we will take what we have learned in previous weeks and produce a report using R Markdown. The package `rmarkdown` allows reports to be created within R, thus allowing for R code and output to be easily embedded within a report. Hence, all of the R code and plots obtained from an analysis are contained within a single file.

Under Week 4 on Moodle there is an Example Report produced using R Markdown relating to fitting a regression model using one numerical explanatory variable that was done in Week 3. The corresponding R Markdown file can also be downloaded and opened in R to see how the document was produced (press the `Knit` button to create the PDF version). It is advised to have this document open within RStudio while working through the remaining sections in order to see examples put into practice.

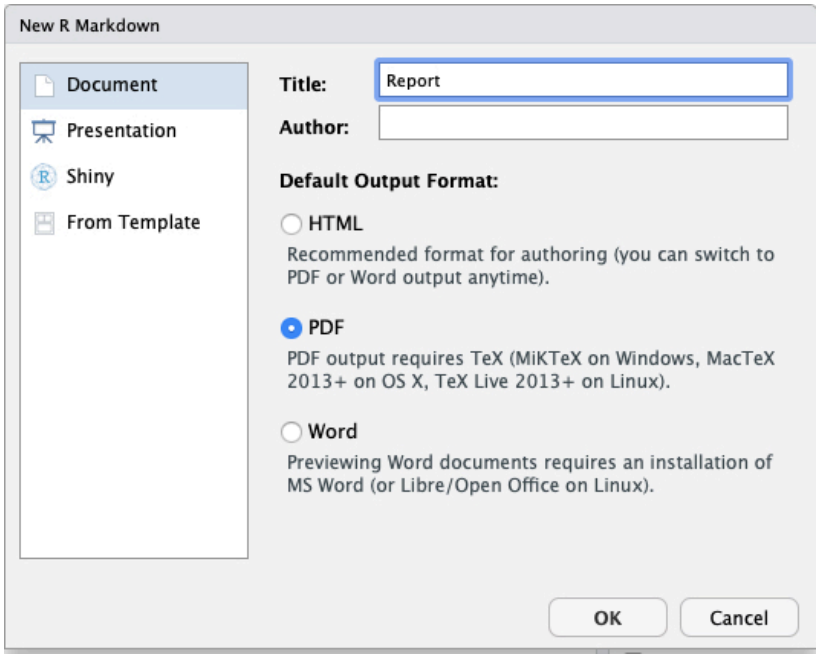
The following sections will take you through the different steps required to produce the Example Report on Moodle. For creating a new R Markdown file from scratch see Section 1.1, otherwise move onto Section 2.

1.1 Creating a new R Markdown document from scratch

If you want to create a new R Markdown document from scratch within RStudio you can go to:

`File -> New File -> R Markdown...`

This will open the following window within RStudio:



From here select **Document** and the **Default Output Format** as **PDF**. Give your document a title and select OK. This will open within RStudio a `.Rmd` file containing simple instructions on how to do some basic stuff using R Markdown. To see what the PDF of the default document looks like click on the `Knit` button at the top of the document window.

2 Title

The title of the document can be found at the top of the `.Rmd` file within its preamble, which is shown below. When titling your document, ensure the title is within inverted commas.

```
---
title: "Example Report"
output:
  pdf_document:
          latex_engine: xelatex
          number_sections: yes
fig_caption: yes
---
```

3 Sections

Sections within an R Markdown document are created using `#`. For example, `# Introduction` will create a section titled **Introduction**:

1 Introduction

Note: the section is numbered as we have set `number_sections: yes` within the preamble of the document. If you do not wish to have numbered sections, then set `number_sections: no` in the preamble. I would recommend using numbered sections as it makes it easier to refer to them within the text.

Each section can be assigned labels so that they can be referred to within the text. For example, to give our **Introduction** section a label we simply add the label `{#sec:intro}` to the section title as follows:

```
# Introduction {#sec:intro}
```

where `sec:intro` is the name chosen for this particular section. It is a good idea to label your sections appropriately so that it is easy to refer to them later. The section can now be referred to within the text of the document using the `\ref` command. That is

```
Section \ref{sec:intro} ...
```

will produce

```
Section 1 ...
```

where the 1 is a clickable hyperlink that will take you to the beginning of that section within the document.

3.1 Subsections

Subsections can be added to a document in a similar fashion using `##` such that `## Subsection {#sec:sub}` will create a subsection with the label `sec:sub` and title **Subsection**:

1.1 Subsection

3.2 Code chunks

Code chunks allow for R code to be embedded within a document. Not only can the code be easily included within a document, the code can also be evaluated. Hence, you can produce an entire report based on an analysis that is contained within a single file instead of having separate files containing your R code, plot images and comments.

Using R code within an R Markdown document is done within the following environment:

```{r label, echo=FALSE, eval=TRUE, warning=FALSE, message=FALSE}

```

where each code chunk is given its own `label`. The additional arguments are:

- **echo**: include the R code within the code chunk in the document (TRUE/FALSE);
- **eval**: evaluate the R code within the code chunk (TRUE/FALSE);
- **warning**: suppress warnings from R (TRUE/FALSE); and
- **message**: suppress messages from R (TRUE/FALSE).

For example, let's say we wanted to select the `score` and `bty_avg` variables from the `evals` data set to be used later, we can do that using the following code chunk:

```{r evals, echo = FALSE, eval = TRUE, warning = FALSE}
evals.scores <- evals %>%
  select(score, bty_avg)
```

This will evaluate the R code and store the subsetted data set as the object `evals.scores` so that it can be used later. If you want to embed the code within the Markdown document then you would simply set `echo = TRUE` and it will include the R code as follows:

```
evals.scores <- evals %>%
  select(score, bty_avg)
```

Additional arguments can be passed to code chunks other than those displayed above. The most useful ones other than those relate to figure sizing and positioning and are discussed in Section 5.

3.3 Inline code

R code can be included within text by enclosing the code with `` `r ` ``. This allows for expressions to be evaluated by R and not be hardwired by the user. For example, if you wanted to convey the number of observations within `evals.scores` then we can enclose `nrow(evals.scores)` within `` `r ` `` to obtain the number of observations, rather than hardwiring 463 into the text. This can help to prevent potential human error when presenting information. It can also help with consistency and ease-of-use, since `n = nrow(evals.scores)` could be stored as an R object and referred to whenever necessary within the text using inline R code.

4 Tables

There are several ways to produce tables in Markdown. Here, a couple of different approaches will be presented. The first approach uses the `kable` function from the `knitr` package (see also the `kableExtra` package) and essentially puts a wrapper around the table produced in R in order to make it more visually appealing within the R Markdown document.

Let's say we wanted to create a table of the first 5 rows of the `iris` data from the `datasets` library. We can create the table using the `kable` function as follows:

```{r table}
kable(head(iris, n = 5), caption = '\\label{tab:iris} The first 5 rows of the iris data.')
```

This produces a similar table to the one below:

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

Note: The above table may be different from the one you produce. If you want to remove the vertical lines from the table include the argument `booktabs = TRUE` within the `kable` function. See also the Example Report Markdown file for an example of using the `kable_styling` function for changing the size and positioning of a table.

Notice that within the `caption` argument of the `kable` function there is `\\label{tab:iris}`. This is how you label tables in order to refer to the within the text. For example,

```
Table \ref{tab:iris} displays the first 5 rows of the iris data...
```

will produce

```
Table 1 displays the first 5 rows of the iris data...
```

where, like for sections, the 1 will be a hyperlink directed to the table.

4.1 Tables 'by hand'

Tables can also be produced 'by hand' in Markdown. For example, the table above corresponding to the first 5 rows of the `iris` data can be produced by hand as follows:

```
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

Table: The fist 5 rows of the iris data.
```

This produces the following table:

Table 2: The fist 5 rows of the iris data.

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

The vertical separators `|` are used between columns, while `---` is placed below table/column headings. Alignment of the columns is done using colons, that is, for left alignment put `:---`, for right alignment put `---:`, and for centred alignment put `:---:`. A limitation of creating tables by hand is that you cannot give them a label to refer to them later within the text (which will also need to be done by hand).

5 Figures

Including plots within an R Markdown document is straightforward. The R code for the plot is simply included within a code chunk including additional arguments for plot size and positioning. For example, to include the scatterplot of teaching and beauty scores shown below:

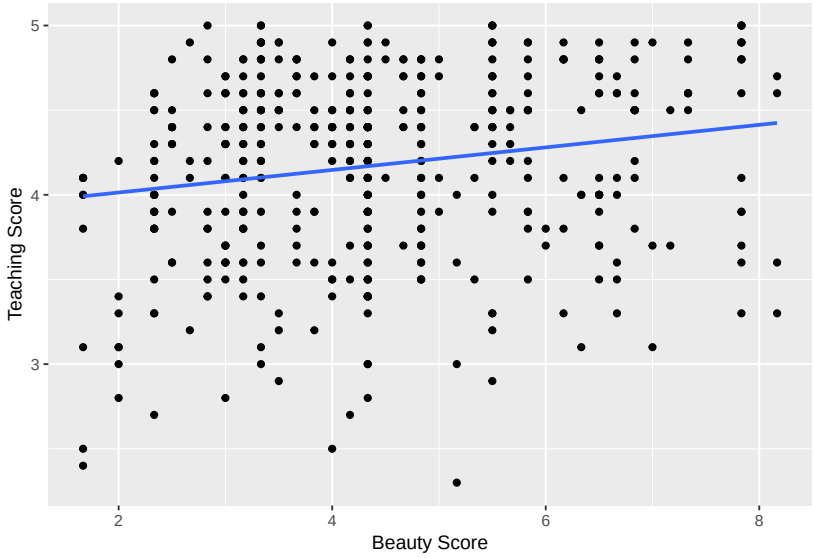


Figure 1: Relationship between teaching and beauty scores. The best-fitting line has been superimposed.

we use the following code chunk:

```
```{r scatplot, echo = FALSE, eval = TRUE, out.width = '80%', fig.align = "center", warning = FALSE, fig.cap
= "\\label{fig:scat} Relationship between teaching and beauty scores. The best-fitting line has been
superimposed."}
ggplot(evals.scores, aes(x = bty_avg, y = score)) +
  geom_point() +
  labs(x = "Beauty Score", y = "Teaching Score") +
  geom_smooth(method = "lm", se = FALSE)
```
```

Here, we have `echo = FALSE` and `eval = TRUE` as we only want to put the plot into the document and not the accompanying R code. Labelling is done the same way as with tables, that is, the label `\\label{fig:scat}` is placed within the figure caption, which is given by the argument `fig.cap`. For size and positioning of the figure we can include:

- `out.width`: a percentage of the actual size of the produced plot;
- `fig.width`: an integer value denoting the width of the figure;
- `fig.align`: the alignment of the figure within the body of the document; and
- `fig.pos`: can be used to force the positioning of the figure within the document.

For examples of using each of the above arguments see the Example Report Markdown file.

6 Mathematics

Mathematics and statistical equations can be presented nicely within an R Markdown document. For example, the following equation referring to a linear regression model:

$$y_i = \alpha + \beta x_i + \epsilon_i, \quad \epsilon_i \sim N(0, \sigma^2),$$

is done using the following:

```
$$y_i = \alpha + \beta x_i + \epsilon_i, ~~~~ \epsilon_i \sim N(0, \sigma^2),$$
```

That is, we use:

- `$$` signs to produce mathematics which is centred, and a single `$` to include mathematics within a sentence or paragraph;
- `_` and `^` are used for subscripts and superscripts, respectively;
- Greek letters are obtained using `\` and the letters name, i.e. `\alpha` gives α; and
- tildes (`~`) are used to put spacing between notation.

If you have a categorical variable then you would write it using an indicator function. For example, if we have gender as a categorical variable, where females are the baseline category, then we could write our model as follows:

$$y_i = \alpha + \beta_{\text{Male}} \cdot \mathbb{I}_{\text{Male}}(x),$$

```
$$y_i = \alpha + \beta_{\mbox{Male}} \cdot \mathbb{I}_{\mbox{Male}}(x),$$
```

where $\mathbb{I}_{\text{Male}}(x)$ `$\mathbb{I}_{\mbox{Male}}(x)$` is an indicator function such that

$$\mathbb{I}_{\text{Male}}(x) = \begin{cases} 1 & \text{if the gender of } x\text{th observation is Male,} \\ 0 & \text{Otherwise.} \end{cases}$$

```
$$\mathbb{I}_{\mbox{Male}}(x)=\left\{
            \begin{array}{ll}
              1 ~~~ \mbox{if Gender of} ~ x \mbox{th observation is Male},\\
              0 ~~~ \mbox{Otherwise}.\\
            \end{array}
          \right.$$
```

For additional tricks inserting mathematics into documents see the Example Report Markdown file.

7 Further Tasks

The following tasks are here to help you prepare for the class test in Week 5. For the class test you will be given a data set to analyse and produce a report using R Markdown given a basic template. Using the *Basic Template* from Moodle, write a report based on the following data sets.

1. Return to Week 3 and the section on simple linear regression with one categorical explanatory variable. Produce a report based on life expectancy across various countries around the world from the `gapminder` data set.
2. Digitalis is a heart medicine similar to toxins found in a genus of plants commonly known as foxglove. These plants can be poisonous towards humans and pets. As part of digitalis experiments, 144 domestic adult cats had their heart (in grams) and body weight (in kilograms) measured. The data set `cats` can be found in the `MASS` library. Ignoring their gender, is there a relationship between heart weight and body weight in domestic cats?