FIT3152 Data analytics – Lecture 3

Graphics...

- Quick follow up of last week's lecture

R Tips

- R Markdown, Scripts, User-defined functions

Assignment 1

Data Manipulation

- Making tables and summaries, Working with factors
- Transforming data, Dates and times

From faculty marketing

Deloitte.

德勤

Deloitte Connect

Connect with us, boost your future

MAKING AN IMPACT THAT MATTERS since 1845

Date

31 March 2021

Time

20:00 – 21:30 (China time)

Registration deadline

21 March 2021

Venue

Zoom

Scan the QR code for registration and more information.

About Deloitte

As the largest professional services organization globally, our purpose is to make an impact that matters. 175 years ago, our founders pioneered the practice of professional services. Today, we employ approximately 330,000 professionals across 150+ countries and territories. Making an impact is a commitment we have always adhered to. Deloitte reported FY2020 revenue of USD47.6 billion, showing our ambition to become the most innovative professional services provider and undisputed leader in the market.

In the era of digital transformation, we have released a series of innovative talent management and incentive tools through digital. Our innovative digital solutions bring value to clients.

Our commitment to diversity and inclusion ensures that Deloitte is committed to creating a better working environment and development space for employees, and to empower Deloitte talent to adapt to different challenges.

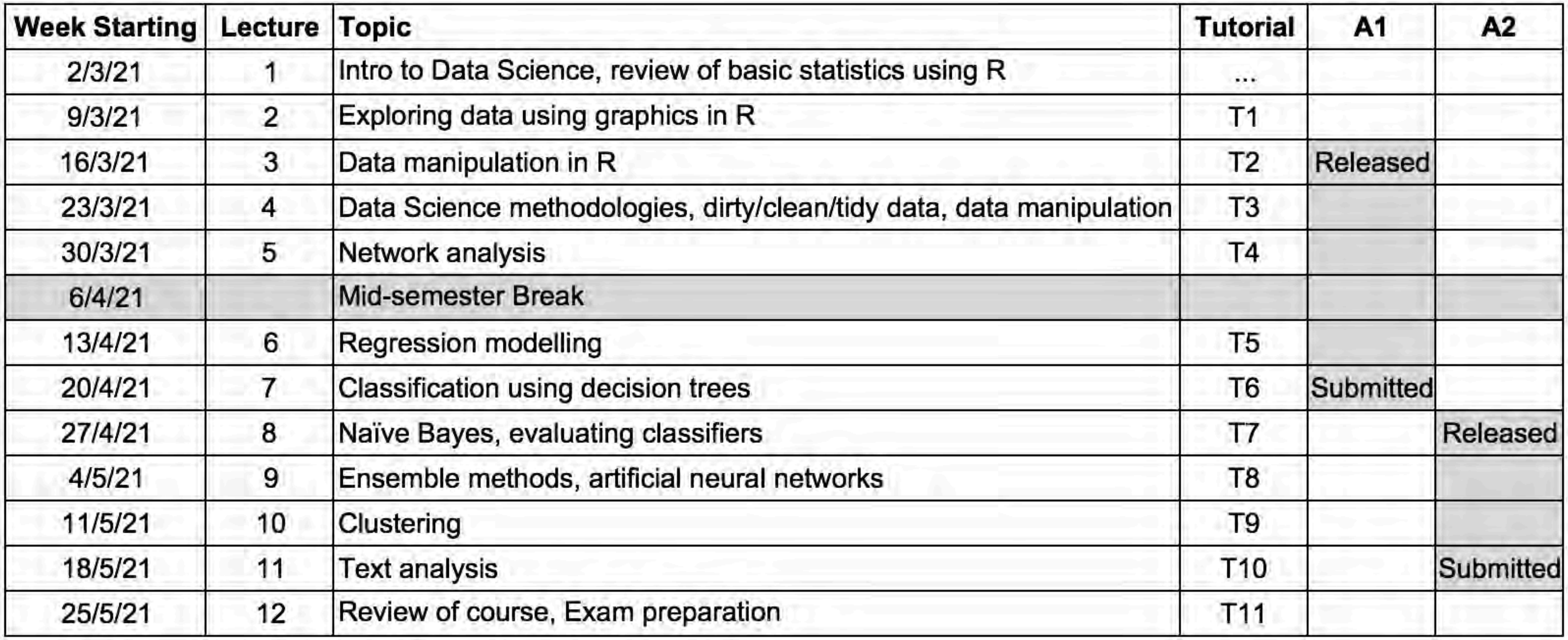
www.deloitte.com/cn

Consultations

Clayton consultations have commenced. Any student can attend any consultation. You can view the schedule on Moodle. We will update the schedule and and offer more Zoom consultations depending on demand.

- Monday 1:00 - 2:00PM G21, 14 Rainforest Walk, Abishek.
- Tuesday 1:00 - 2:00PM G21, 14 Rainforest Walk, Heshan.
- Wednesday 11:00 -12:00PM G21, 14 Rainforest Walk, Michael.
- Wednesday 5:00 - 6:00PM Online Zoom Consultation, Anil. https://monash.zoom.us/j/86328038790?pwd=cE9kMXczdFpQQXZ1Njd5RnFZUXJpZz09
- Friday 2:00 -3:00PM G20, 14 Rainforest Walk, Karina.

Week-by-week



Week Starting	Lecture	Topic	Tutorial	A1	A2
2/3/21	1	Intro to Data Science, review of basic statistics using R	...		
9/3/21	2	Exploring data using graphics in R	T1		
16/3/21	3	Data manipulation in R	T2	Released	
23/3/21	4	Data Science methodologies, dirty/clean/tidy data, data manipulation	T3	▨	
30/3/21	5	Network analysis	T4	▨	
6/4/21		Mid-semester Break		▨	
13/4/21	6	Regression modelling	T5	▨	
20/4/21	7	Classification using decision trees	T6	Submitted	
27/4/21	8	Naïve Bayes, evaluating classifiers	T7		Released
4/5/21	9	Ensemble methods, artificial neural networks	T8		▨
11/5/21	10	Clustering	T9		▨
18/5/21	11	Text analysis	T10		Submitted
25/5/21	12	Review of course, Exam preparation	T11		

Brief review of Lecture 2…

Visualising data

- The number of dimensions in a data set
- Major families of graph types: time series, statistical distributions, maps, hierarchies, networks.
- Plotting the Iris data. Basic scatterplot, increasing the number of dimensions shown.
- Ggplot2 package, and the grammar of graphics approach.

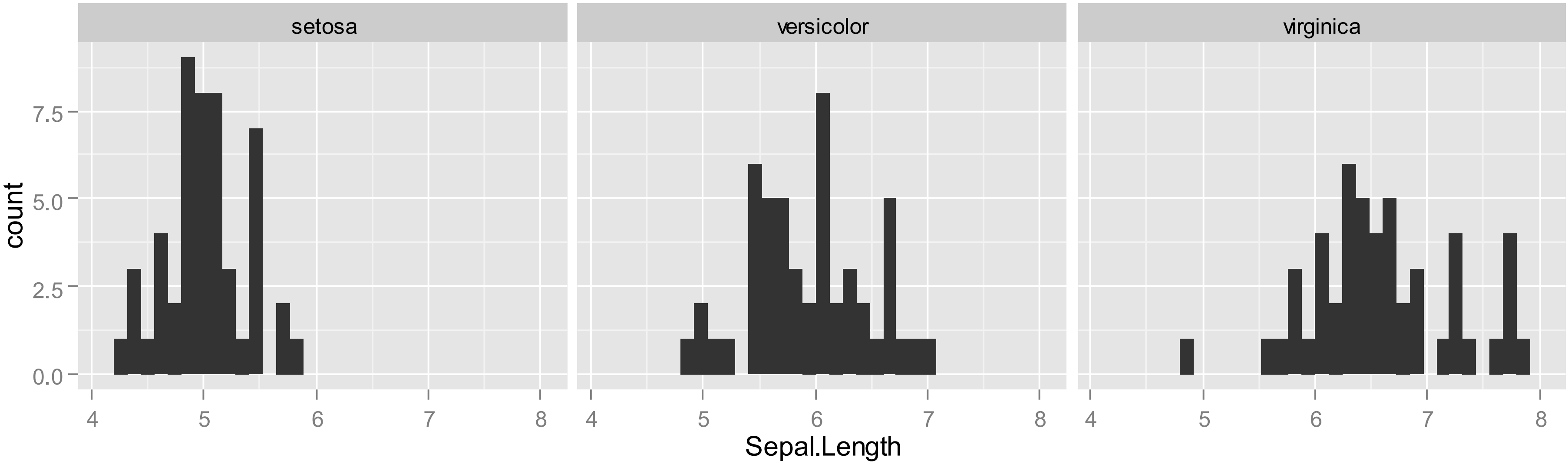
R for Data Science

- A physical and web-based book by the author of ggplot2, Hadley Wickham, and Garrett Grolemund: http://r4ds.had.co.nz/
- The book takes you through all aspects of the data science workflow (more later)
- A good chapter on ggplot2, including the syntax underpinning all ggplots, for example:

```
>   ggplot(data = <DATA>) +
    <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>))
```

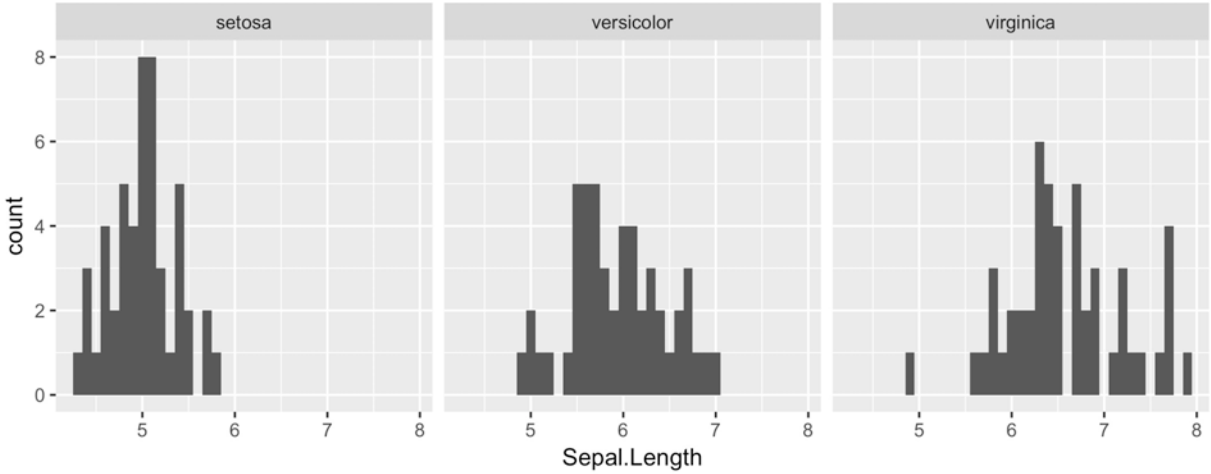
Syntax: histogram + facet_wrap

```
> qplot(Sepal.Length, data = iris, geom = "histogram",
  facets = Species ~ .) + facet_wrap(~ Species, ncol = 3)
```



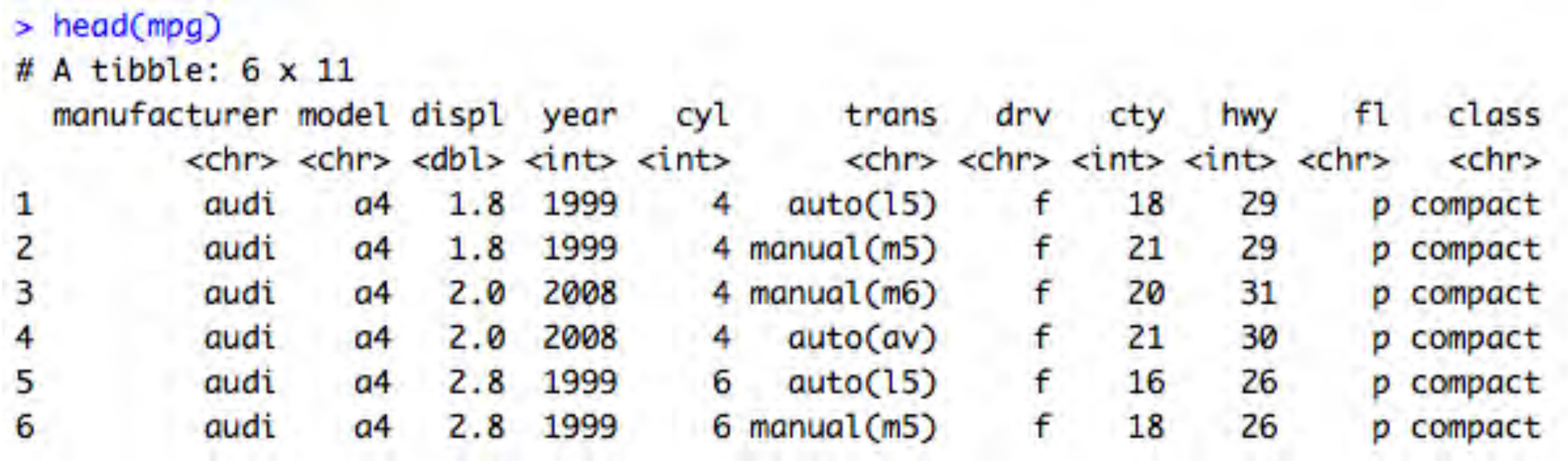
Using the grammar approach...

```
>  m = ggplot(iris, aes(x = Sepal.Length))
>  m = m + geom_histogram(binwidth = 0.1)
>  m = m + facet_wrap(~Species, ncol = 3)
>  ggsave("irissepallen.jpg", m, width = 20, height = 8,
   units = "cm")
```



MPG example

Recall the mpg data set.



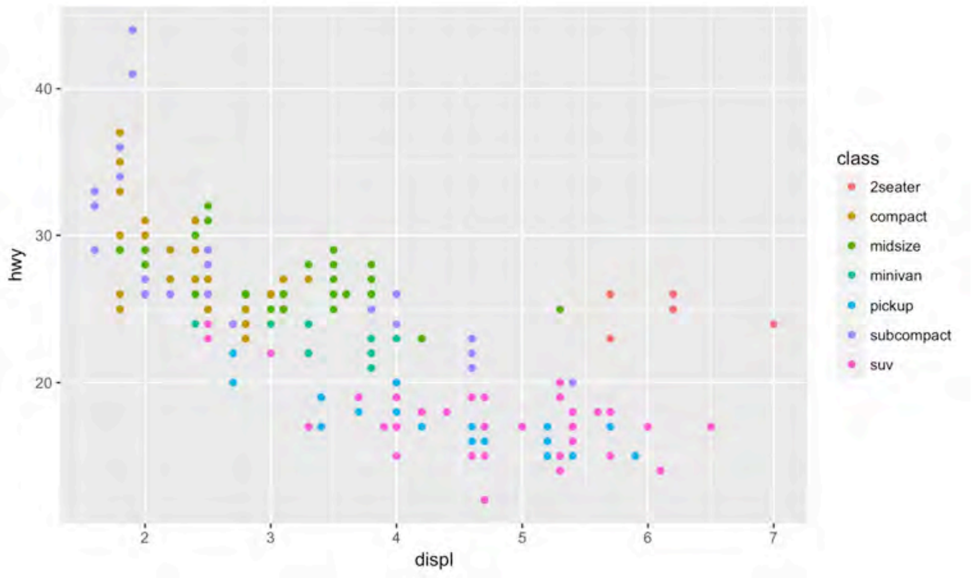
```
> head(mpg)
# A tibble: 6 x 11
  manufacturer model displ  year   cyl      trans   drv   cty   hwy    fl   class
         <chr> <chr> <dbl> <int> <int>      <chr> <chr> <int> <int> <chr>   <chr>
1         audi    a4   1.8  1999     4   auto(l5)     f    18    29     p compact
2         audi    a4   1.8  1999     4 manual(m5)     f    21    29     p compact
3         audi    a4   2.0  2008     4 manual(m6)     f    20    31     p compact
4         audi    a4   2.0  2008     4   auto(av)     f    21    30     p compact
5         audi    a4   2.8  1999     6   auto(l5)     f    16    26     p compact
6         audi    a4   2.8  1999     6 manual(m5)     f    18    26     p compact
```

Investigate the relationship between fuel consumption and engine displacement.

Grammar of graphics

Grammar of graphics approach (from R4DS)

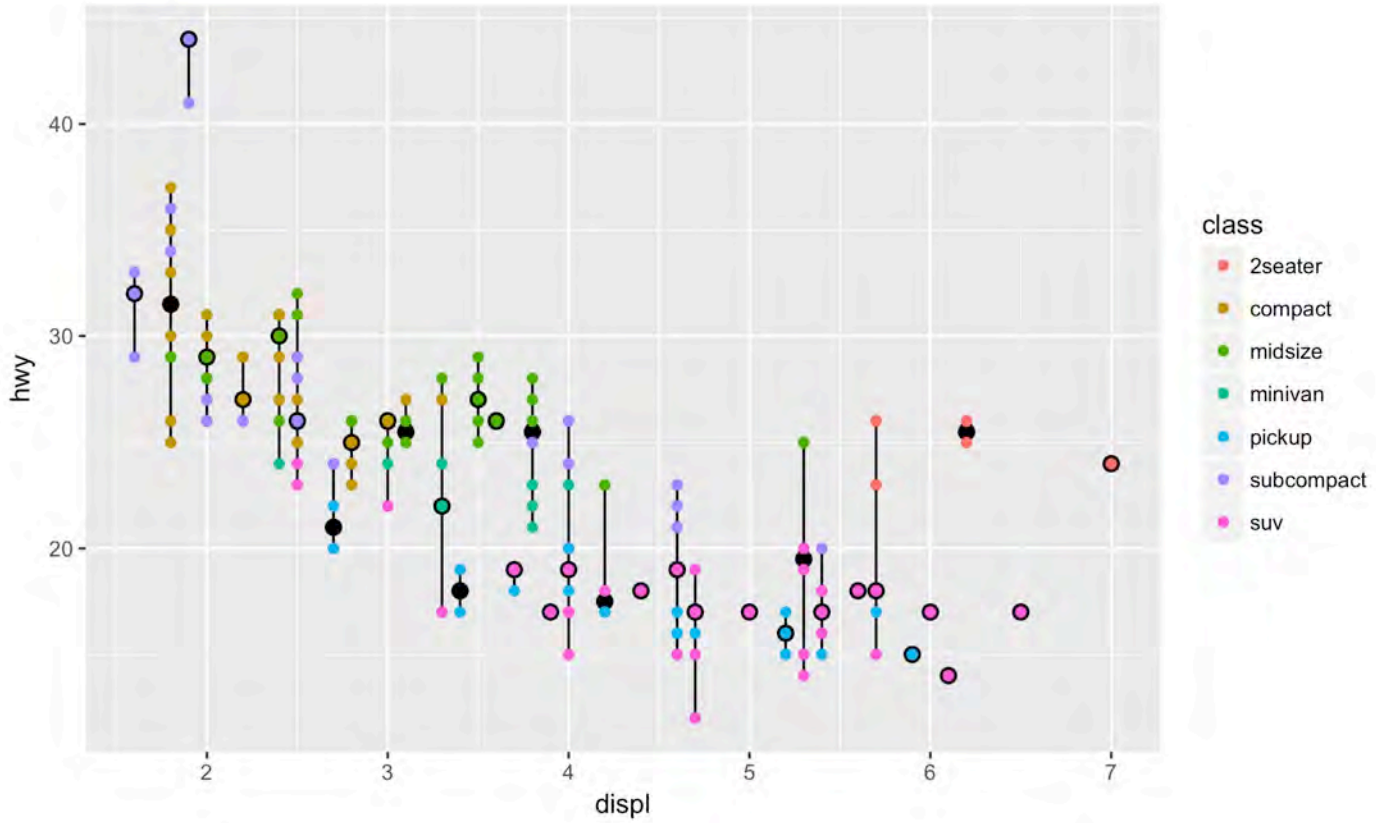
```
>   g = ggplot(data = mpg)
>   g = g + geom_point(mapping = aes(x = displ, y = hwy,
    color = class))
>   g
```



Underplotting (min, median, max)

```
> d <- ggplot(mpg, aes(displ, hwy, color = class)) + geom_point()
> d = d + stat_summary(mapping = aes(x = displ, y = hwy), fun.min = min, fun.max = max, fun = median, orientation = "x", colour = "black")
> d = d + geom_point(mapping = aes(x = displ, y = hwy, color = class)) # overplots original points
> d
> ggsave("hwyvdispl.jpg", d, width = 20, height = 12, units = "cm")
```

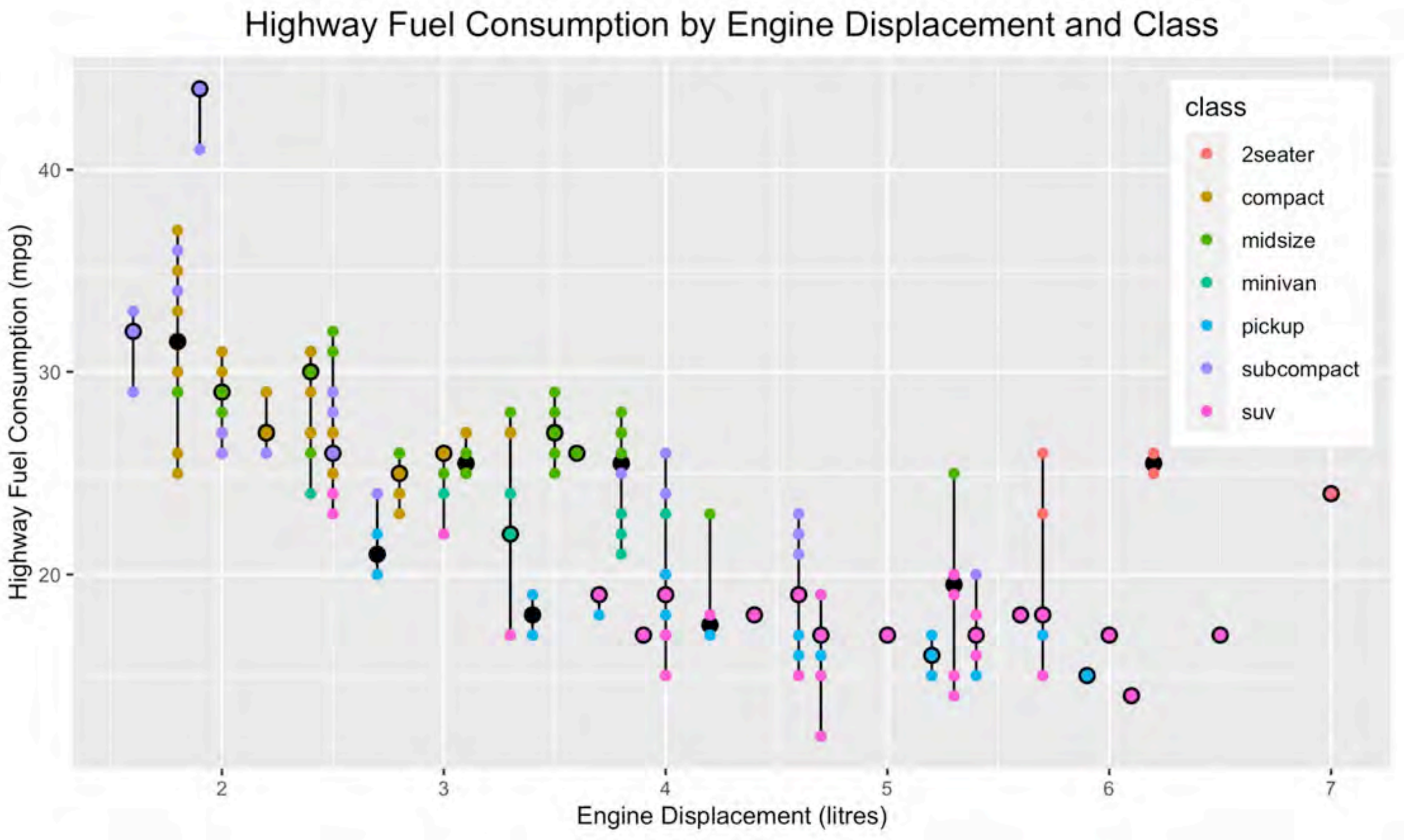
The plot. What improvements would you make?



Improving: axes, title, legend

```
> d = d + theme(axis.text = element_text(size = 8))
> d = d + theme(axis.title = element_text(size = 10))
> # could by axis.text.x or .y etc. to adjust separately
> d = d + xlab("Engine Displacement (litres)")
> d = d + ylab("Highway Fuel Consumption (mpg)")
> d = d + theme(plot.title = element_text(size = 14))
> d = d + theme(plot.title = element_text(hjust = 0.5))
> d = d + ggtitle("Highway … and Class")
> d = d + theme(legend.position = c(0.91, 0.71))
```

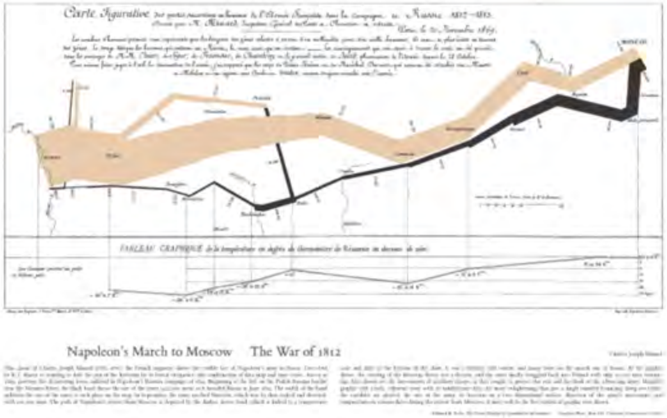
Making incremental improvements by trial and error

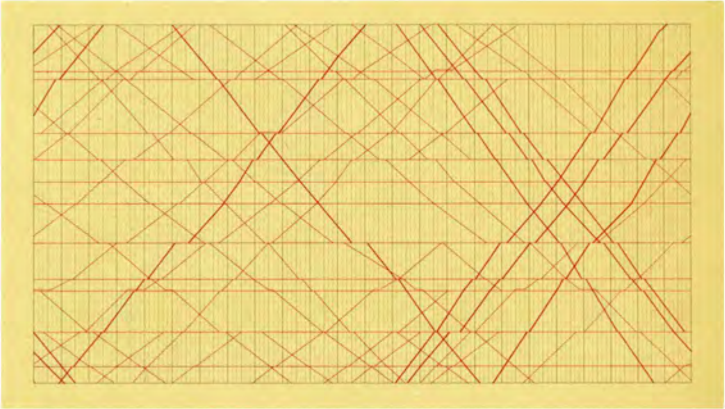


Better graphics

One source of inspiration is Edward Tufte:

- Read: Tufte, E. The visual display of quantitative information, Graphics Press. https://monash.hosted...
- A strong advocate for good information design.





https://www.edwardtufte.com/tufte/

https://medium.com/

Review questions

- Answer in the Zoom chat if you like.

Question 1

The figure is from the ________ graph family?

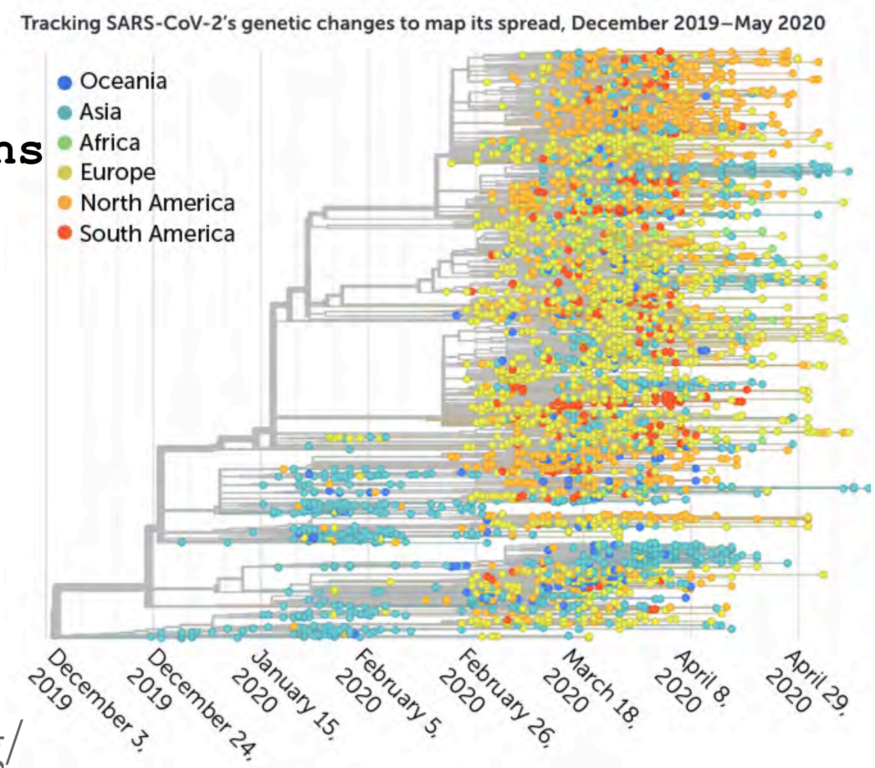
A. **Time Series**

B. **Statistical Distributions**

C. **Maps**

D. **Hierarchies**

E. **Networks**



Source: https://www.sciencenews.org/

Question 2

The figure is from the ________ graph family?

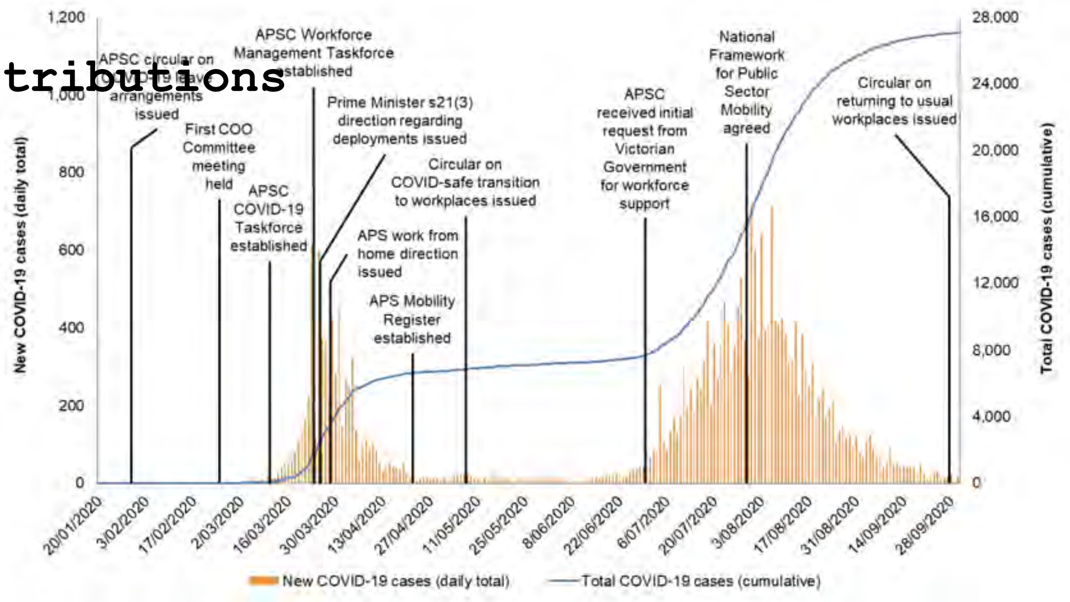
A. Time Series

B. Statistical Distributions

C. Maps

D. Hierarchies

E. Networks



Source: https://www.anao.gov.au/

Question 3

The figure is from the ________ graph family?

A. Time Series

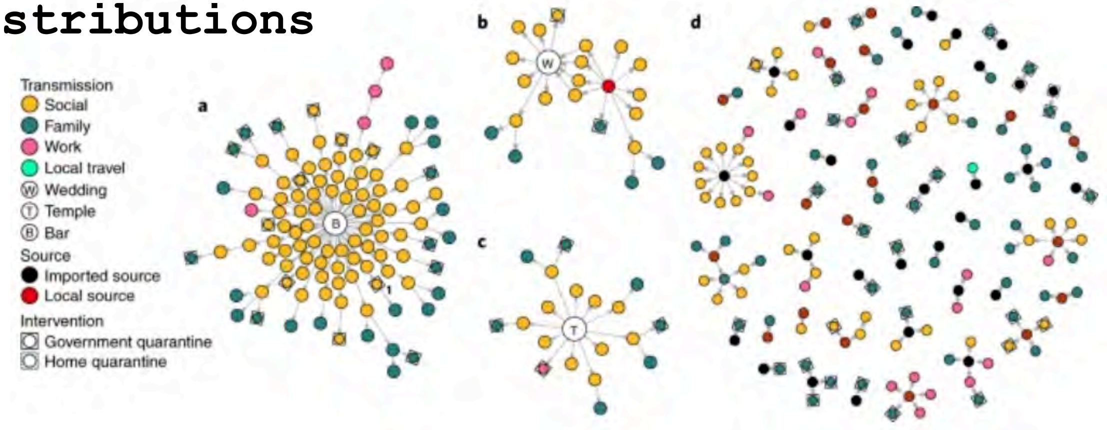
B. Statistical Distributions

C. Maps

D. Hierarchies

E. Networks

Fig. 2: Chains of SARS-CoV-2 transmission in Hong Kong initiated by local or imported cases.



Source: https://www.nature.com/

Question 4

The figure is from the ________ graph family?

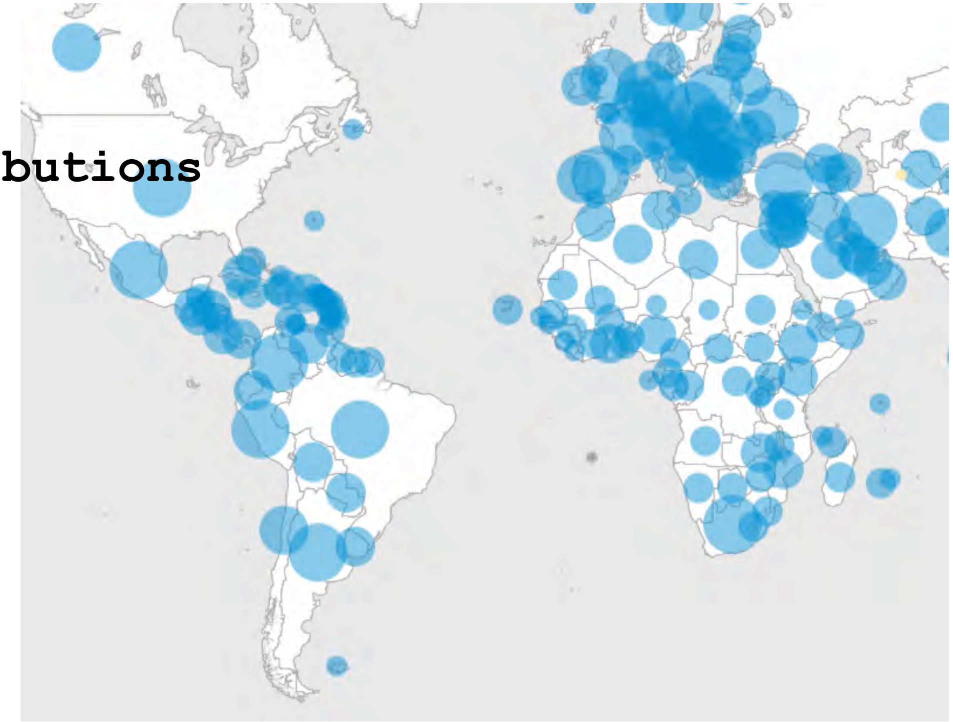
A. Time Series

B. Statistical Distributions

C. Maps

D. Hierarchies

E. Networks

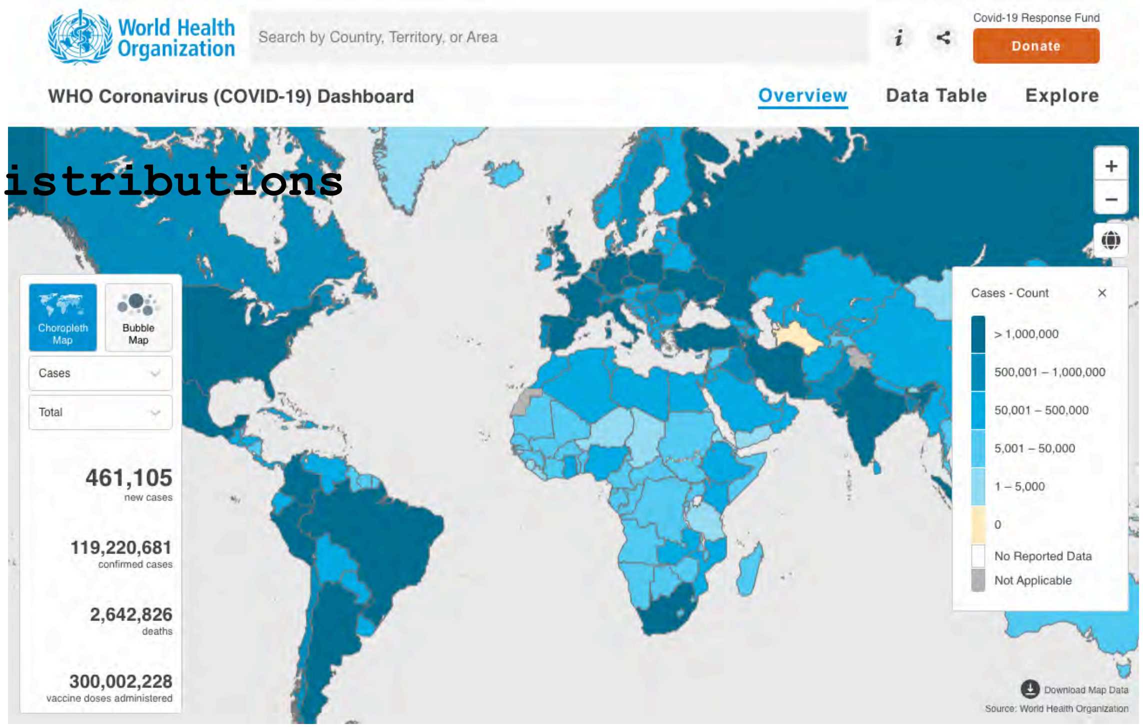


Source: https://covid19.who.int/

Question 5

The figure is from the ________ graph family?

- A. Time Series
- B. Statistical Distributions
- C. Maps
- D. Hierarchies
- E. Networks



Source: https://covid19.who.int/

Some R tips

Scripts:

- Very important, learn how to use these now if you've not done so already.

RMarkdown:

- Useful if you're doing a job that requires a lot of routine reporting, but not essential.

User-Defined Functions

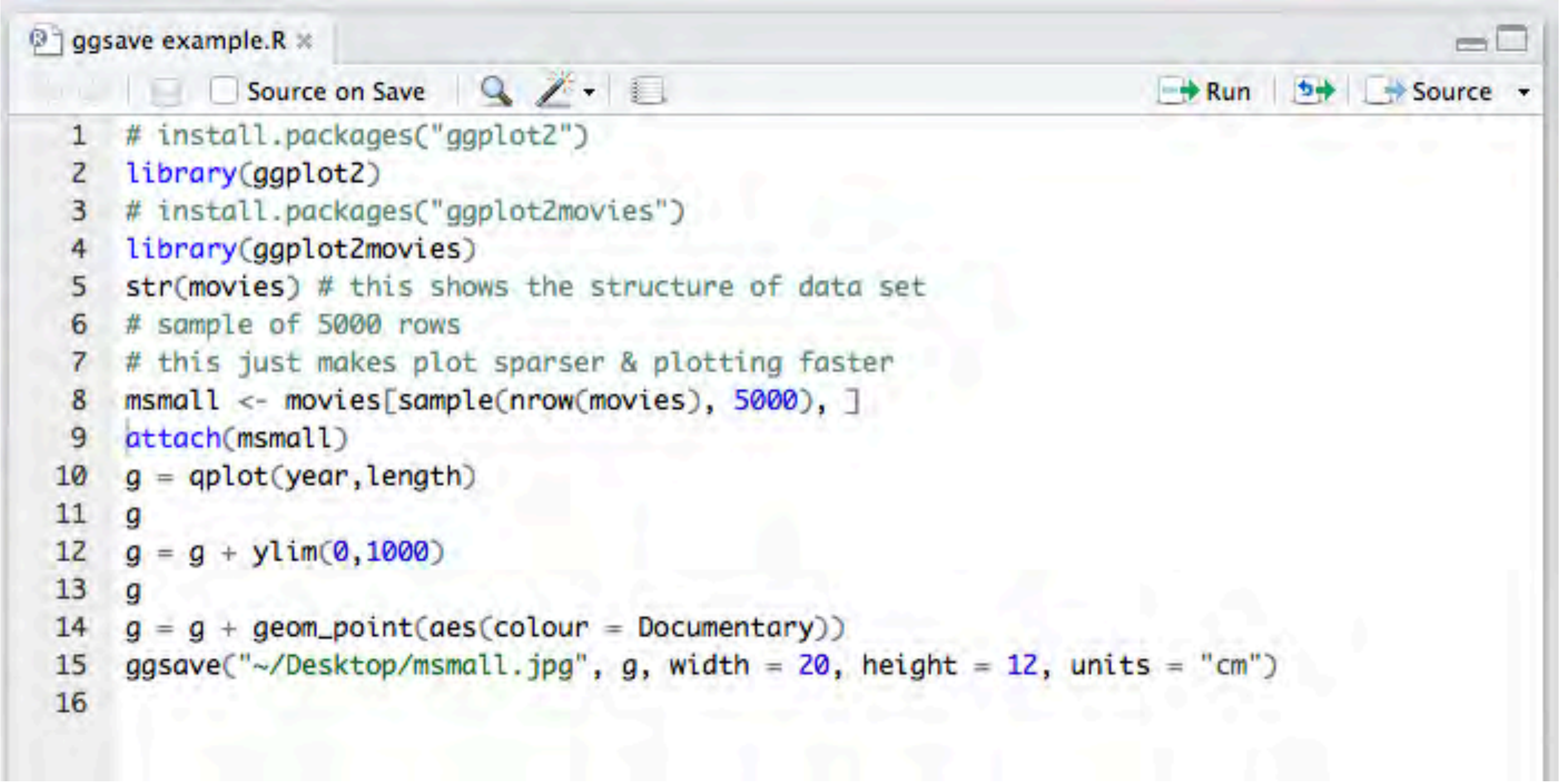
- Useful, and they improve your R code, but not essential. We will learn functions defined on the fly.

Scripts

Scripts allow you to save your working from session to session.

- Use them to automate environment settings etc.
- Create a new script: File > New File > R Script
- Save with a filename
- Use “Source” to evaluate on the fly
- Note: # comments, pre-emptive text
- Next slide shows movies example as a script…

Scripts



```
# install.packages("ggplot2")
library(ggplot2)
# install.packages("ggplot2movies")
library(ggplot2movies)
str(movies) # this shows the structure of data set
# sample of 5000 rows
# this just makes plot sparser & plotting faster
msmall <- movies[sample(nrow(movies), 5000), ]
attach(msmall)
g = qplot(year,length)
g
g = g + ylim(0,1000)
g
g = g + geom_point(aes(colour = Documentary))
ggsave("~/Desktop/msmall.jpg", g, width = 20, height = 12, units = "cm")

```

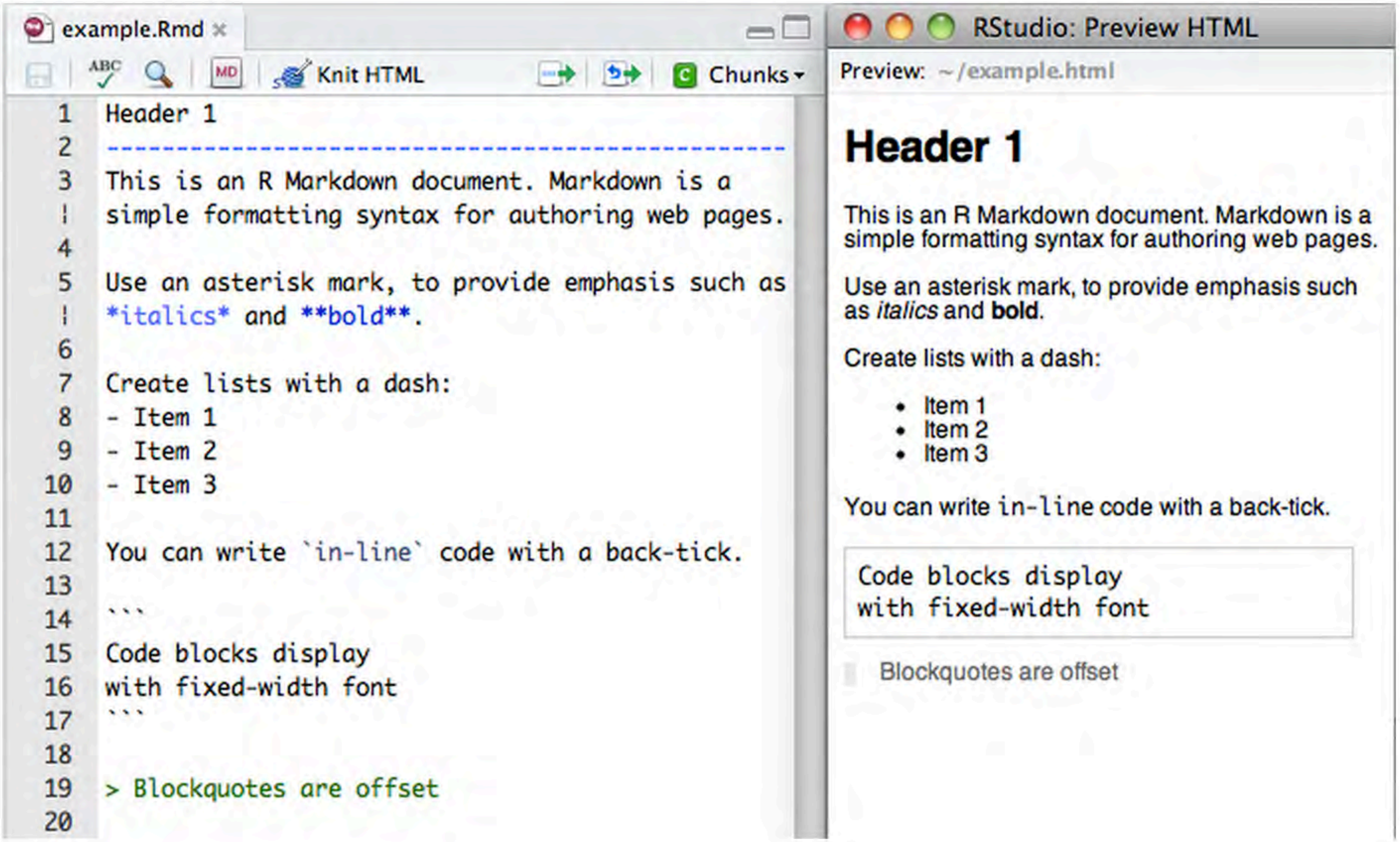
R Markdown

Is a package that enables the creation of HTML and PDF documents etc. based on your R session. You may choose to use it but it is optional.

- Has core syntax of markdown.
- You can embed R code and graphics.
- You can get started with R Markdown by creating a new R Markdown file in R Studio (the required files will be automatically installed).

http://rmarkdown.rstudio.com/

R Markdown



```
Header 1
--------------------------------------------------
This is an R Markdown document. Markdown is a simple formatting syntax for authoring web pages.

Use an asterisk mark, to provide emphasis such as *italics* and **bold**.

Create lists with a dash:
- Item 1
- Item 2
- Item 3

You can write `in-line` code with a back-tick.

```
Code blocks display
with fixed-width font
```

> Blockquotes are offset
```

Header 1

This is an R Markdown document. Markdown is a simple formatting syntax for authoring web pages.

Use an asterisk mark, to provide emphasis such as *italics* and **bold**.

Create lists with a dash:

- Item 1
- Item 2
- Item 3

You can write `in-line` code with a back-tick.

```
Code blocks display
with fixed-width font
```

> Blockquotes are offset

http://rmarkdown.rstudio.com/

Creating user-defined functions

It is possible to create named, user-defined, functions that can be saved between sessions using a script (see ATHR pp. 40 – 41).

Syntax:

```
>   my_function <- function(arg1, arg2, …) {
>   object <- Calculations(arg1, arg2, …)
>   Return(object)
>   }
```

Creating user-defined functions

Example:

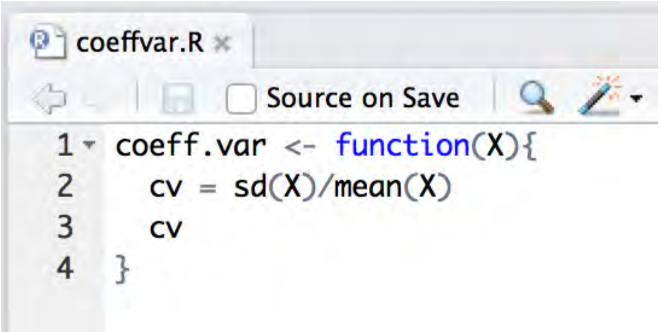
```
>   coeff.var <- function(X){
>   cv = sd(X)/mean(X)
>   cv}


>   Y = c(1, 2, 3, 4, 5, 6)
>   coeff.var(Y)
    [1] 0.5345225
```

Saving and re-using functions

In Rstudio:

- Create a new R script,
- Write function in script editor,
- Save as (filename.R)

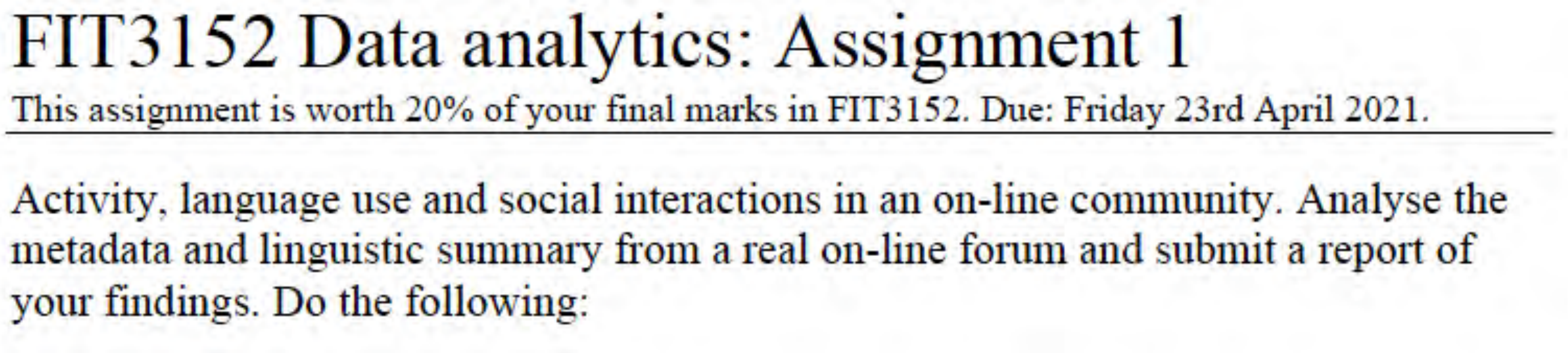


```
coeff.var <- function(X){
  cv = sd(X)/mean(X)
  cv
}
```

To run function in a new session of R studio:

- Open and run script: code > source file (filename.R)
- See Solutions to Tutorial 1 (Tutorial01.R) as an example of multiple functions in a single script.

Assignment 1

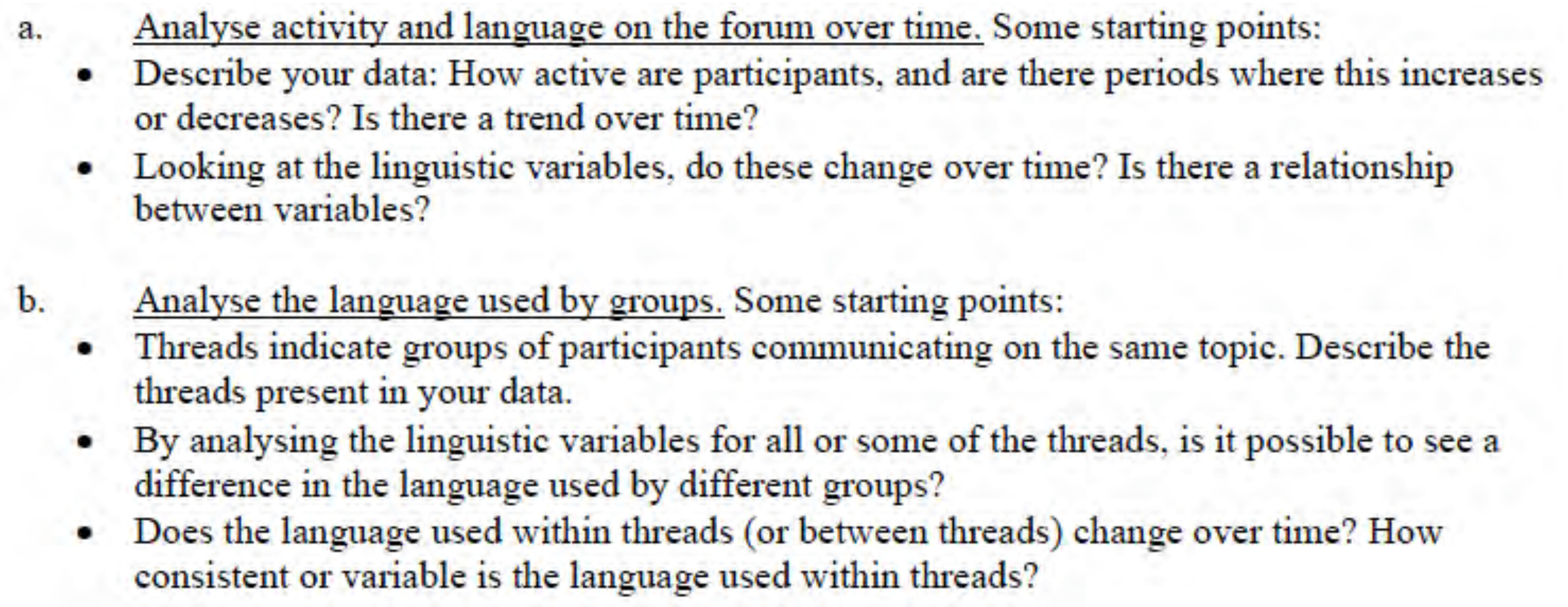


FIT3152 Data analytics: Assignment 1

This assignment is worth 20% of your final marks in FIT3152. Due: Friday 23rd April 2021.

Activity, language use and social interactions in an on-line community. Analyse the metadata and linguistic summary from a real on-line forum and submit a report of your findings. Do the following:

Assignment 1



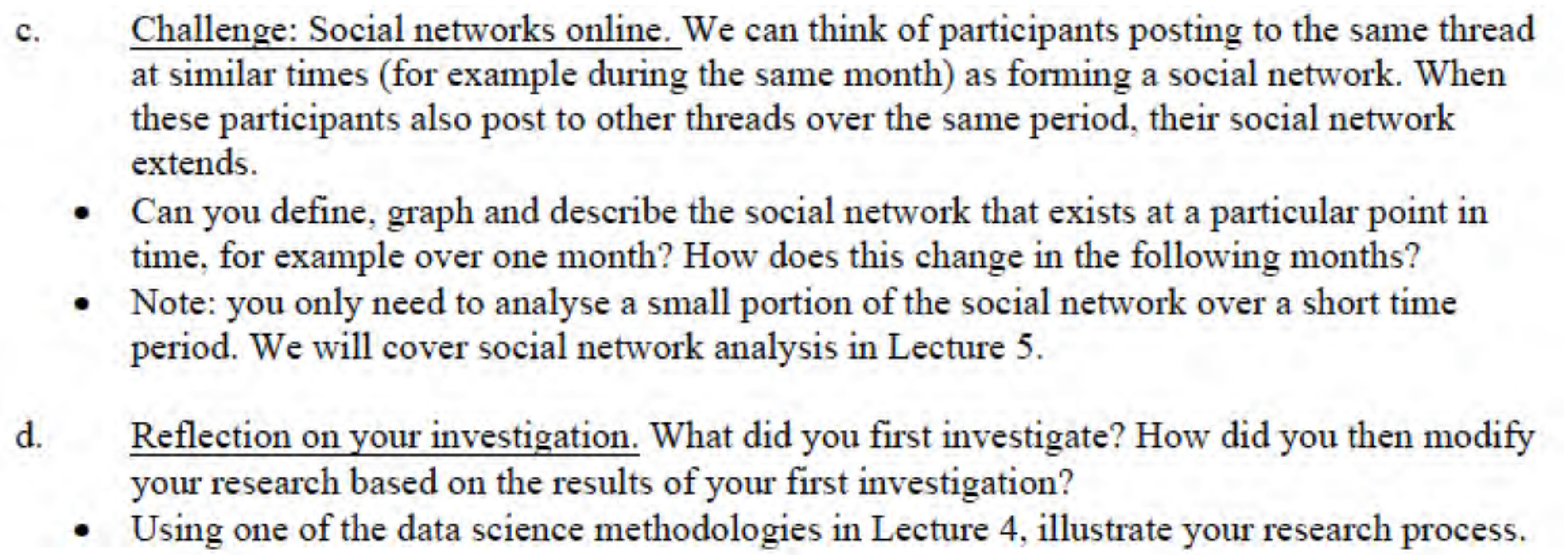
a. Analyse activity and language on the forum over time. Some starting points:

- Describe your data: How active are participants, and are there periods where this increases or decreases? Is there a trend over time?
- Looking at the linguistic variables, do these change over time? Is there a relationship between variables?

b. Analyse the language used by groups. Some starting points:

- Threads indicate groups of participants communicating on the same topic. Describe the threads present in your data.
- By analysing the linguistic variables for all or some of the threads, is it possible to see a difference in the language used by different groups?
- Does the language used within threads (or between threads) change over time? How consistent or variable is the language used within threads?

Assignment 1



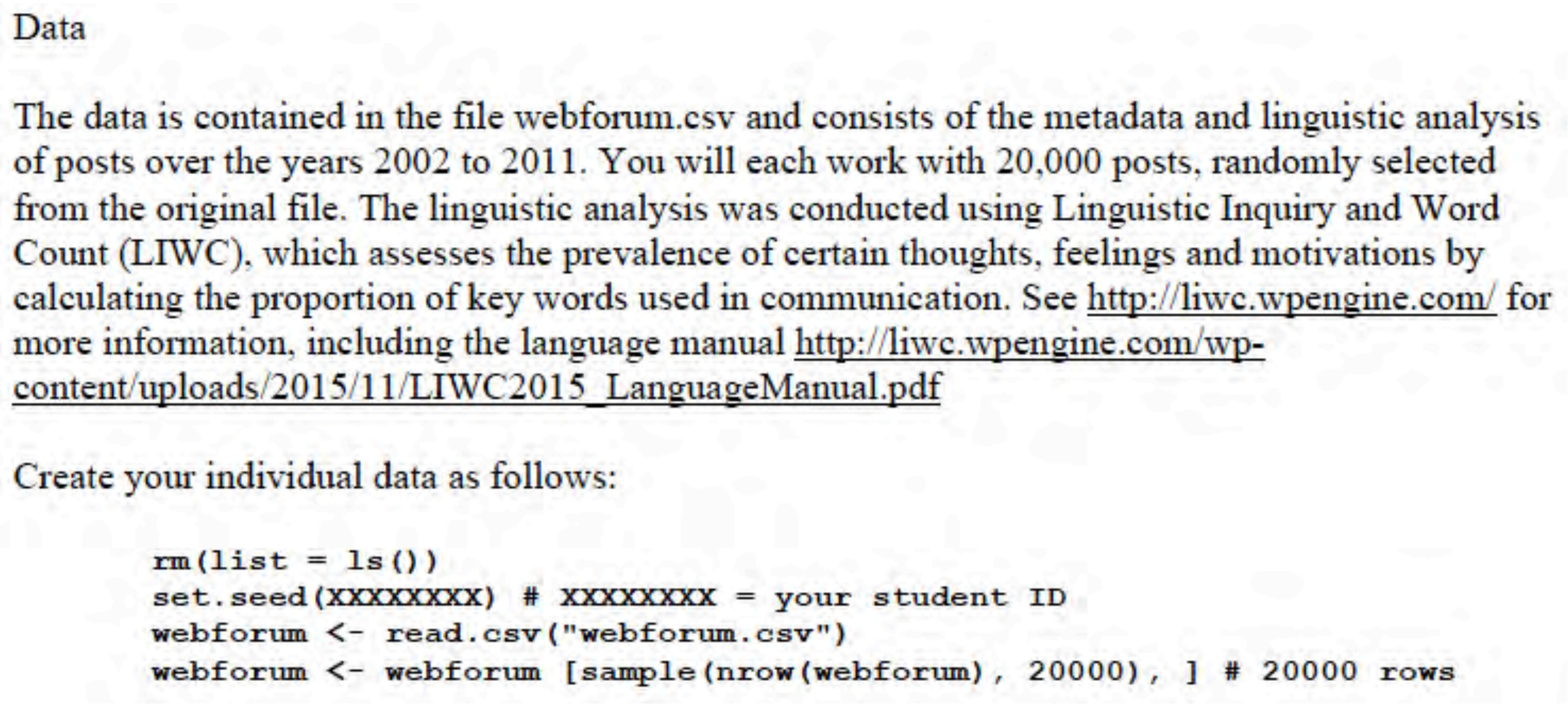
c. Challenge: Social networks online. We can think of participants posting to the same thread at similar times (for example during the same month) as forming a social network. When these participants also post to other threads over the same period, their social network extends.

- Can you define, graph and describe the social network that exists at a particular point in time, for example over one month? How does this change in the following months?
- Note: you only need to analyse a small portion of the social network over a short time period. We will cover social network analysis in Lecture 5.

d. Reflection on your investigation. What did you first investigate? How did you then modify your research based on the results of your first investigation?

- Using one of the data science methodologies in Lecture 4, illustrate your research process.

Assignment 1



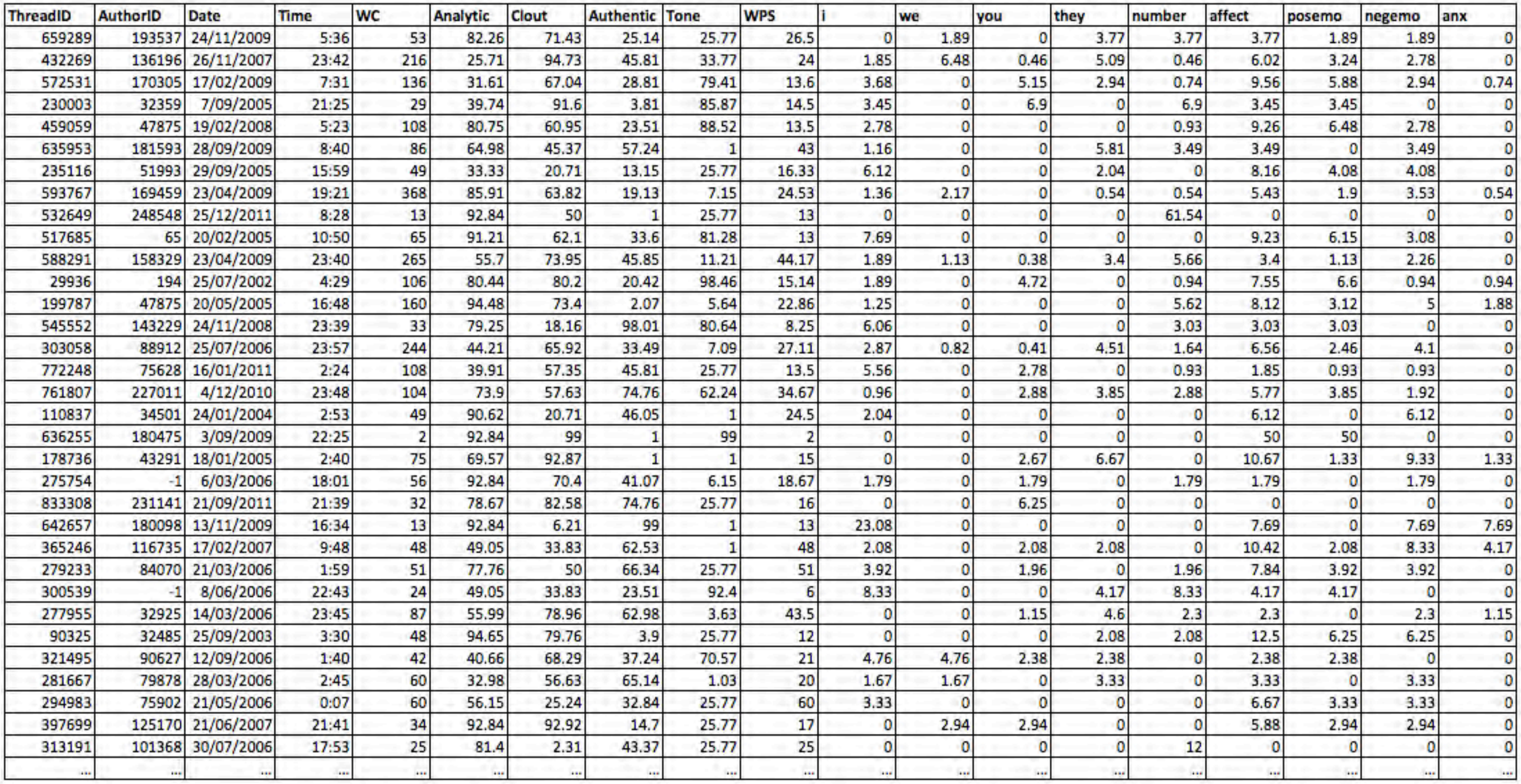
Data

The data is contained in the file webforum.csv and consists of the metadata and linguistic analysis of posts over the years 2002 to 2011. You will each work with 20,000 posts, randomly selected from the original file. The linguistic analysis was conducted using Linguistic Inquiry and Word Count (LIWC), which assesses the prevalence of certain thoughts, feelings and motivations by calculating the proportion of key words used in communication. See http://liwc.wpengine.com/ for more information, including the language manual http://liwc.wpengine.com/wp-content/uploads/2015/11/LIWC2015_LanguageManual.pdf

Create your individual data as follows:

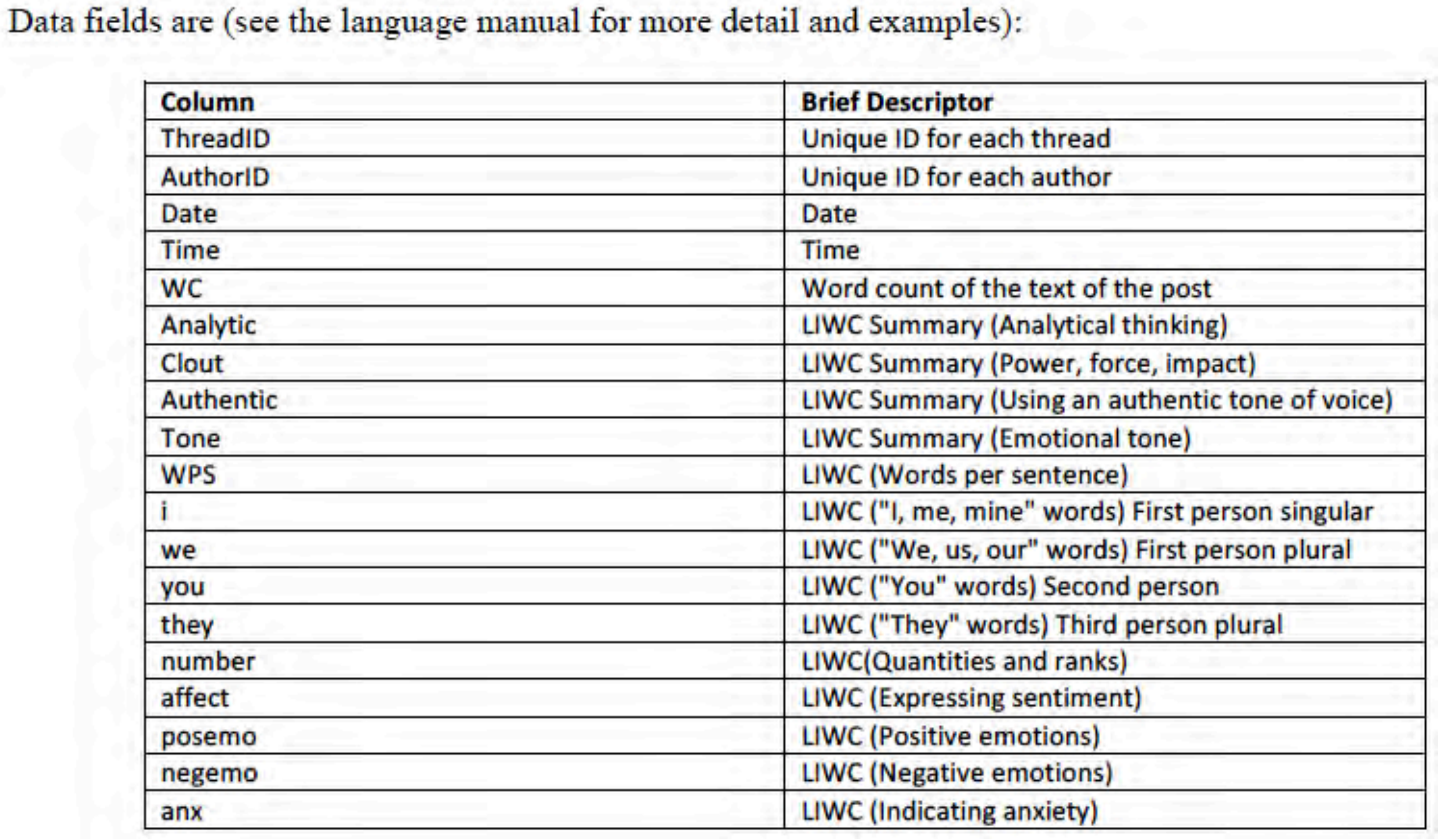
```
rm(list = ls())
set.seed(XXXXXXXX) # XXXXXXXX = your student ID
webforum <- read.csv("webforum.csv")
webforum <- webforum [sample(nrow(webforum), 20000), ] # 20000 rows
```

Assignment 1



ThreadID	AuthorID	Date	Time	WC	Analytic	Clout	Authentic	Tone	WPS	i	we	you	they	number	affect	posemo	negemo	anx
659289	193537	24/11/2009	5:36	53	82.26	71.43	25.14	25.77	26.5	0	1.89	0	3.77	3.77	3.77	1.89	1.89	0
432269	136196	26/11/2007	23:42	216	25.71	94.73	45.81	33.77	24	1.85	6.48	0.46	5.09	0.46	6.02	3.24	2.78	0
572531	170305	17/02/2009	7:31	136	31.61	67.04	28.81	79.41	13.6	3.68	0	5.15	2.94	0.74	9.56	5.88	2.94	0.74
230003	32359	7/09/2005	21:25	29	39.74	91.6	3.81	85.87	14.5	3.45	0	6.9	0	6.9	3.45	3.45	0	0
459059	47875	19/02/2008	5:23	108	80.75	60.95	23.51	88.52	13.5	2.78	0	0	0	0.93	9.26	6.48	2.78	0
635953	181593	28/09/2009	8:40	86	64.98	45.37	57.24	1	43	1.16	0	0	5.81	3.49	3.49	0	3.49	0
235116	51993	29/09/2005	15:59	49	33.33	20.71	13.15	25.77	16.33	6.12	0	0	2.04	0	8.16	4.08	4.08	0
593767	169459	23/04/2009	19:21	368	85.91	63.82	19.13	7.15	24.53	1.36	2.17	0	0.54	0.54	5.43	1.9	3.53	0.54
532649	248548	25/12/2011	8:28	13	92.84	50	1	25.77	13	0	0	0	0	61.54	0	0	0	0
517685	65	20/02/2005	10:50	65	91.21	62.1	33.6	81.28	13	7.69	0	0	0	0	9.23	6.15	3.08	0
588291	158329	23/04/2009	23:40	265	55.7	73.95	45.85	11.21	44.17	1.89	1.13	0.38	3.4	5.66	3.4	1.13	2.26	0
29936	194	25/07/2002	4:29	106	80.44	80.2	20.42	98.46	15.14	1.89	0	4.72	0	0.94	7.55	6.6	0.94	0.94
199787	47875	20/05/2005	16:48	160	94.48	73.4	2.07	5.64	22.86	1.25	0	0	0	5.62	8.12	3.12	5	1.88
545552	143229	24/11/2008	23:39	33	79.25	18.16	98.01	80.64	8.25	6.06	0	0	0	3.03	3.03	3.03	0	0
303058	88912	25/07/2006	23:57	244	44.21	65.92	33.49	7.09	27.11	2.87	0.82	0.41	4.51	1.64	6.56	2.46	4.1	0
772248	75628	16/01/2011	2:24	108	39.91	57.35	45.81	25.77	13.5	5.56	0	2.78	0	0.93	1.85	0.93	0.93	0
761807	227011	4/12/2010	23:48	104	73.9	57.63	74.76	62.24	34.67	0.96	0	2.88	3.85	2.88	5.77	3.85	1.92	0
110837	34501	24/01/2004	2:53	49	90.62	20.71	46.05	1	24.5	2.04	0	0	0	0	6.12	0	6.12	0
636255	180475	3/09/2009	22:25	2	92.84	99	1	99	2	0	0	0	0	0	50	50	0	0
178736	43291	18/01/2005	2:40	75	69.57	92.87	1	1	15	0	0	2.67	6.67	0	10.67	1.33	9.33	1.33
275754	-1	6/03/2006	18:01	56	92.84	70.4	41.07	6.15	18.67	1.79	0	1.79	0	1.79	1.79	0	1.79	0
833308	231141	21/09/2011	21:39	32	78.67	82.58	74.76	25.77	16	0	0	6.25	0	0	0	0	0	0
642657	180098	13/11/2009	16:34	13	92.84	6.21	99	1	13	23.08	0	0	0	0	7.69	0	7.69	7.69
365246	116735	17/02/2007	9:48	48	49.05	33.83	62.53	1	48	2.08	0	2.08	2.08	0	10.42	2.08	8.33	4.17
279233	84070	21/03/2006	1:59	51	77.76	50	66.34	25.77	51	3.92	0	1.96	0	1.96	7.84	3.92	3.92	0
300539	-1	8/06/2006	22:43	24	49.05	33.83	23.51	92.4	6	8.33	0	0	4.17	8.33	4.17	4.17	0	0
277955	32925	14/03/2006	23:45	87	55.99	78.96	62.98	3.63	43.5	0	0	1.15	4.6	2.3	2.3	0	2.3	1.15
90325	32485	25/09/2003	3:30	48	94.65	79.76	3.9	25.77	12	0	0	0	2.08	2.08	12.5	6.25	6.25	0
321495	90627	12/09/2006	1:40	42	40.66	68.29	37.24	70.57	21	4.76	4.76	2.38	2.38	0	2.38	2.38	0	0
281667	79878	28/03/2006	2:45	60	32.98	56.63	65.14	1.03	20	1.67	1.67	0	3.33	0	3.33	0	3.33	0
294983	75902	21/05/2006	0:07	60	56.15	25.24	32.84	25.77	60	3.33	0	0	0	0	6.67	3.33	3.33	0
397699	125170	21/06/2007	21:41	34	92.84	92.92	14.7	25.77	17	0	2.94	2.94	0	0	5.88	2.94	2.94	0
313191	101368	30/07/2006	17:53	25	81.4	2.31	43.37	25.77	25	0	0	0	0	12	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...

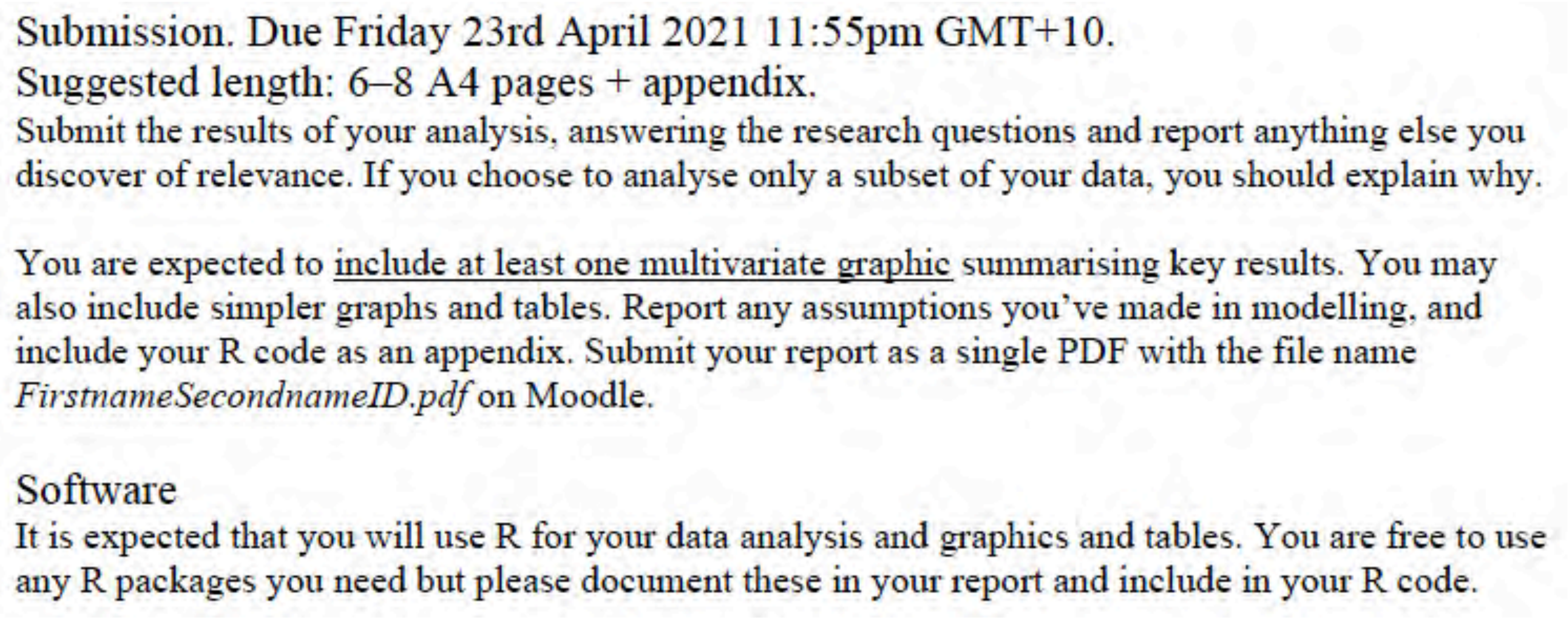
Assignment 1



Data fields are (see the language manual for more detail and examples):

Column	Brief Descriptor
ThreadID	Unique ID for each thread
AuthorID	Unique ID for each author
Date	Date
Time	Time
WC	Word count of the text of the post
Analytic	LIWC Summary (Analytical thinking)
Clout	LIWC Summary (Power, force, impact)
Authentic	LIWC Summary (Using an authentic tone of voice)
Tone	LIWC Summary (Emotional tone)
WPS	LIWC (Words per sentence)
i	LIWC ("I, me, mine" words) First person singular
we	LIWC ("We, us, our" words) First person plural
you	LIWC ("You" words) Second person
they	LIWC ("They" words) Third person plural
number	LIWC(Quantities and ranks)
affect	LIWC (Expressing sentiment)
posemo	LIWC (Positive emotions)
negemo	LIWC (Negative emotions)
anx	LIWC (Indicating anxiety)

Assignment 1



Submission. Due Friday 23rd April 2021 11:55pm GMT+10.

Suggested length: 6–8 A4 pages + appendix.

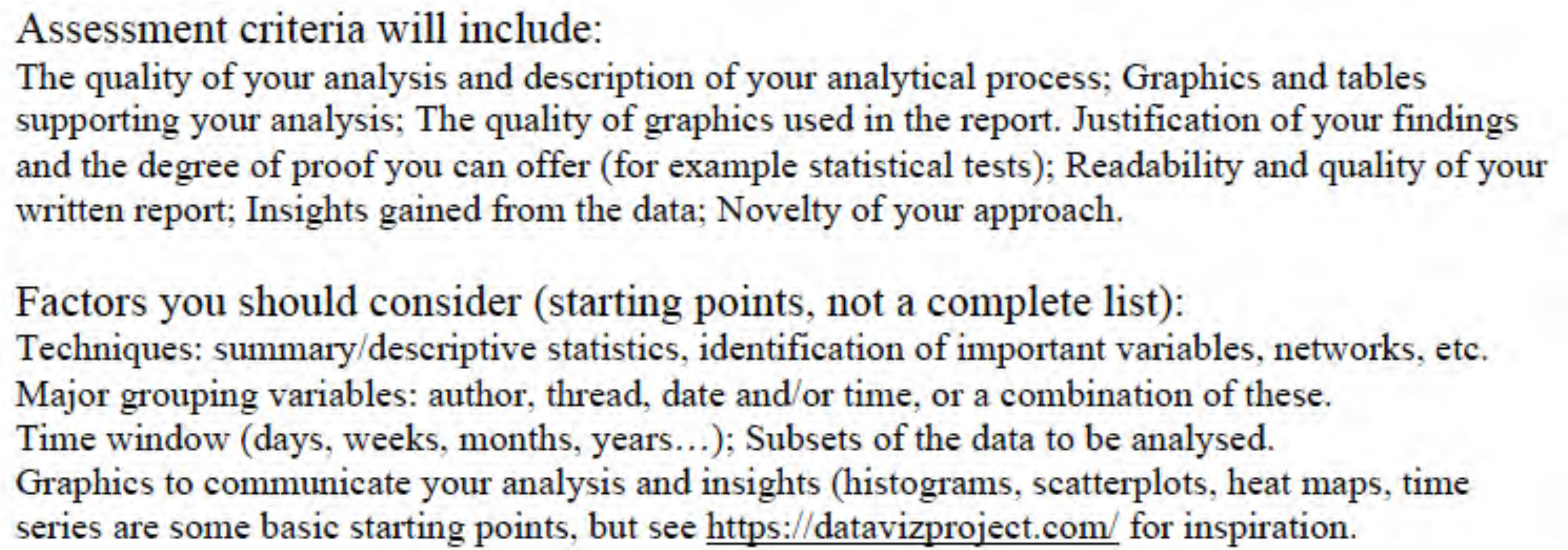
Submit the results of your analysis, answering the research questions and report anything else you discover of relevance. If you choose to analyse only a subset of your data, you should explain why.

You are expected to include at least one multivariate graphic summarising key results. You may also include simpler graphs and tables. Report any assumptions you've made in modelling, and include your R code as an appendix. Submit your report as a single PDF with the file name *FirstnameSecondnameID.pdf* on Moodle.

Software

It is expected that you will use R for your data analysis and graphics and tables. You are free to use any R packages you need but please document these in your report and include in your R code.

Assignment 1



Assessment criteria will include:

The quality of your analysis and description of your analytical process; Graphics and tables supporting your analysis; The quality of graphics used in the report. Justification of your findings and the degree of proof you can offer (for example statistical tests); Readability and quality of your written report; Insights gained from the data; Novelty of your approach.

Factors you should consider (starting points, not a complete list):

Techniques: summary/descriptive statistics, identification of important variables, networks, etc.
Major grouping variables: author, thread, date and/or time, or a combination of these.
Time window (days, weeks, months, years…); Subsets of the data to be analysed.
Graphics to communicate your analysis and insights (histograms, scatterplots, heat maps, time series are some basic starting points, but see https://datavizproject.com/ for inspiration.

Data manipulation

Summarizing data by groups

Data grouped by factors:

- Applying a function to a single column
- Applying a function to a group of columns

Why do we need to do this?

- To simplify the data, making comparisons easier
- Reduce data complexity, enabling further analysis

Edgar Anderson's Iris data

50 samples from 3 species:

- Iris setosa, – virginica, – versicolor

Four features measured:

- Sepal width and length
- Petal width and length

Is it possible to distinguish species using physical measurements?

- Data is packaged with R: "iris"

http://en.wikipedia.org/wiki/Iris_flower_data_set

Print

```
>   iris # = print(iris)
    Sepal.Length Sepal.Width Petal.Length Petal.Width    Species
1            5.1         3.5          1.4         0.2     setosa
2            4.9         3.0          1.4         0.2     setosa
3            4.7         3.2          1.3         0.2     setosa
4            4.6         3.1          1.5         0.2     setosa
5            5.0         3.6          1.4         0.2     setosa
6            5.4         3.9          1.7         0.4     setosa
7            4.6         3.4          1.4         0.3     setosa
8            5.0         3.4          1.5         0.2     setosa
9            4.4         2.9          1.4         0.2     setosa
10           4.9         3.1          1.5         0.1     setosa
11           5.4         3.7          1.5         0.2     setosa
12           4.8         3.4          1.6         0.2     setosa
13           4.8         3.0          1.4         0.1     setosa
14           4.3         3.0          1.1         0.1     setosa
15           5.8         4.0          1.2         0.2     setosa
...
```

Two challenges

(1) Easy!

- Create a table of column means grouped by species.

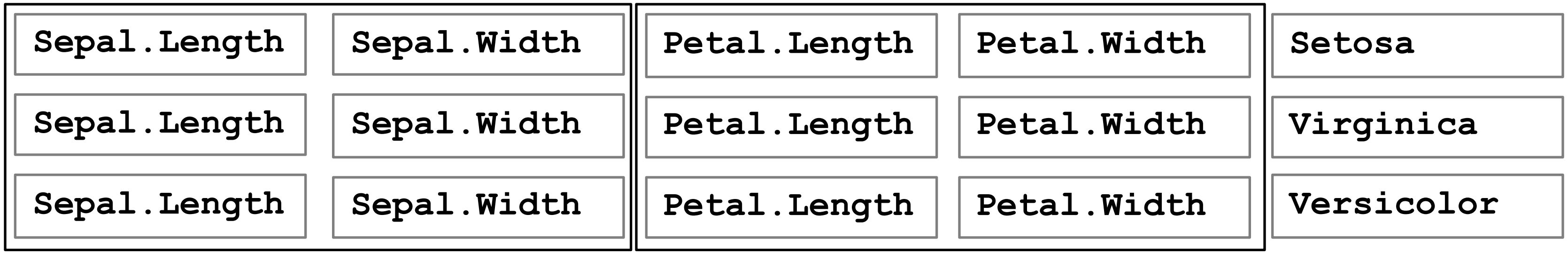
(2) Harder!

- Create a CSV file containing the correlation between sepal length and sepal width as well as petal length and petal width for each species.

High level view

Data analysis is easier if you have a high level view of the data:

- 4 columns + 1 factor (Species)
- Two pairs of related columns: sepals & petals



Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Setosa
Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Virginica
Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Versicolor

Challenge 1. Function: aggregate

The ‘aggregate’ function creates a table by applying a function to data in individual columns grouped by a factor (or factors). To calculate averages:

- Note: columns referred to their index (number) [ ] for compactness

```
> aggregate(iris[1:4], iris[5], mean)
     Species Sepal.Length Sepal.Width Petal.Length Petal.Width
1     setosa         5.01        3.43         1.46       0.246
2 versicolor         5.94        2.77         4.26       1.326
3  virginica         6.59        2.97         5.55       2.026
```

?aggregate

- Description

  aggregate(x, ...) : Splits the data into subsets, computes summary statistics for each, and returns the result in a convenient form.

- Usage

  aggregate(x, by, FUN, ..., simplify = TRUE)

- Arguments

  X : An R object.

  By : List of grouping elements

  FUN : Function to compute the summary statistics

  Simplify : Indicates whether results should be simplified to a vector or matrix if possible.

Challenge 2. Function: by

‘by’ enables a function to be applied across individual or multiple columns of a data frame grouped by a factor or factors.

- To calculate the correlation of sepal length and width

```
>   by(iris, iris[5], function(df) cor(df$Sepal.Length,
    df$Sepal.Width))
```

```
Species: setosa
[1] 0.743
Species: versicolor
[1] 0.526
Species: virginica
[1] 0.457
```

?by

- Description

  Apply a Function to a Data Frame Split by Factors

- Usage

  ```
  by(data, INDICES, FUN, ..., simplify = TRUE)
  ```

- Arguments

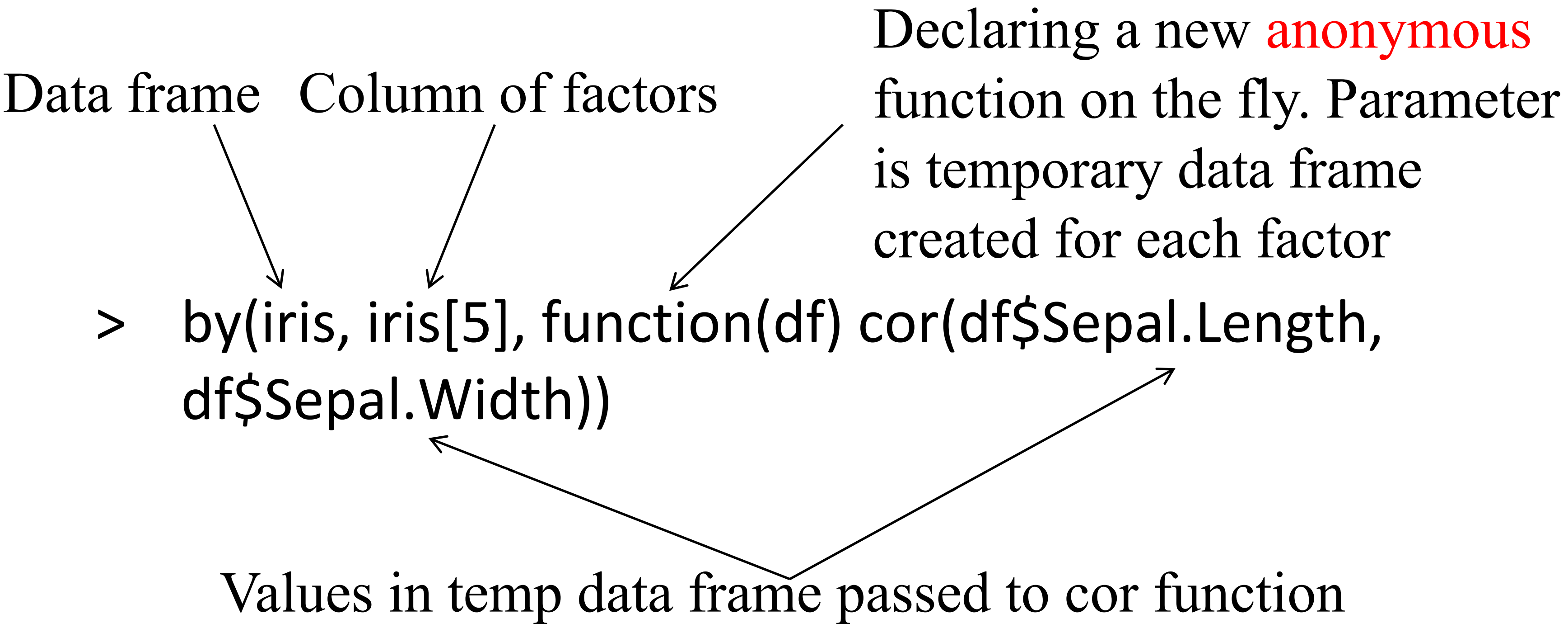
  Data : an R object, normally a data frame, possibly a matrix.

  INDICES : a factor or a list of factors, each of length nrow(data).

  FUN : a function to be applied to data frame subsets of data...

?by: applying the cor function

Looking more closely at the way correlation is calculated:



Data frame

Column of factors

Declaring a new anonymous function on the fly. Parameter is temporary data frame created for each factor

```
>   by(iris, iris[5], function(df) cor(df$Sepal.Length,
    df$Sepal.Width))
```

Values in temp data frame passed to cor function

Note: anonymous functions

If a function is only to be used once, it can be defined when it is used. These are anonymous functions (having no name) see ATHR p.41.

Example (sum the sepal length by species)

```
>       by(iris, iris[5], function(df) sum(df$Sepal.Length))
        Species: setosa
        [1] 250.3
        Species: versicolor
        [1] 296.8
        Species: virginica
        [1] 329.4
```

…

Changing earlier example to a more compact notation, using column indexes.

From:

```
>   by(iris, iris[5], function(df) cor(df$Sepal.Length,
    df$Sepal.Width))
```

To:

```
>   by(iris, iris[5], function(df) cor(df[1], df[2]))
```

Function: as.table

This function converts the output format of a function from a list to a table

```
>   as.table(by(iris, iris[5], function(df) cor(df[1], df[2])))
    Species
        setosa versicolor  virginica
         0.743      0.526      0.457
```

Function: as.data.frame

This function converts “coerces” the output of a table into a data frame

```
>   Sepal.cor <- as.data.frame(as.table(by(iris, iris[5],
    function(df) cor(df[1], df[2]))))
>   Sepal.cor
         Species  Freq
    1     setosa 0.743
    2 versicolor 0.526
    3  virginica 0.457
```

Function: colnames

This function assigns new column names to a data frame.

```
> colnames(Sepal.cor) <- c("Species", "Sepal.cor")
> Sepal.cor
     Species Sepal.cor
1     setosa     0.743
2 versicolor     0.526
3  virginica     0.457
```

Now for petals…

Repeating the previous code for petals…

```
>   Petal.cor <- as.data.frame(as.table(by(iris, iris[5],
    function(df) cor(df[3], df[4]))))
>   colnames(Petal.cor) <- c("Species", "Petal.cor")
>   Petal.cor
          Species Petal.cor
    1      setosa     0.332
    2 versicolor      0.787
    3  virginica      0.322
```

Merging data frames (and saving)

Using a common column – “Species” – and rounding data. *Note: we could have used cbind – since the two dataframes are in alignment.*

```
>   iris.cor <- merge(Sepal.cor, Petal.cor, by = "Species")
>   iris.cor[,2] = round(iris.cor[,2], digits = 3)
>   iris.cor[,3] = round(iris.cor[,3], digits = 3)
>   write.csv(iris.cor, file = "Iris.cor.csv",
    row.names=FALSE)
```

The saved file

SepalPetalcor.csv

Species	Sepal.cor	Petal.cor
setosa	0.743	0.332
versicolor	0.526	0.787
virginica	0.457	0.322

This is a much more efficient way of calculating and saving only the required correlations than the method shown last lecture. See following slide…

Correlation matrix – by… factor

From last week: Pairwise correlation by species

```
> by(iris[1:4], factor(iris$Species), cor)
```

```
factor(iris$Species): setosa
             Sepal.Length Sepal.Width Petal.Length Petal.Width
Sepal.Length    1.0000000   0.7425467    0.2671758   0.2780984
Sepal.Width     0.7425467   1.0000000    0.1777000   0.2327520
Petal.Length    0.2671758   0.1777000    1.0000000   0.3316300
Petal.Width     0.2780984   0.2327520    0.3316300   1.0000000
---------------------------------------------------------------
factor(iris$Species): versicolor
             Sepal.Length Sepal.Width Petal.Length Petal.Width
Sepal.Length    1.0000000   0.5259107    0.7540490   0.5464611
Sepal.Width     0.5259107   1.0000000    0.5605221   0.6639987
Petal.Length    0.7540490   0.5605221    1.0000000   0.7866681
Petal.Width     0.5464611   0.6639987    0.7866681   1.0000000
---------------------------------------------------------------
factor(iris$Species): virginica
             Sepal.Length Sepal.Width Petal.Length Petal.Width
Sepal.Length    1.0000000   0.4572278    0.8642247   0.2811077
Sepal.Width     0.4572278   1.0000000    0.4010446   0.5377280
Petal.Length    0.8642247   0.4010446    1.0000000   0.3221082
Petal.Width     0.2811077   0.5377280    0.3221082   1.0000000
```

Two more challenges

(3) Easy!

- Examine the difference between the aspect ratios (Length / Width) for sepals and petals between the different species.

(4) Harder!

- Report the data for the flower having the longest petal in each species.

Adding (and removing) columns

By default R will add a new column to a data frame if the output of a column operation is specified as a new column.

Alternatively the cbind function can be used to append a vector or data frame by columns.

This lets us store the results of row operations, including factor generation.

Making new columns

Add two columns containing the aspect ratio (length/width) for sepals and petals:

```
> niris <- iris # creating a new data frame
> niris$Sepal.ar <- niris$Sepal.Length/niris$Sepal.Width # add new column
> niris$Petal.ar <- niris$Petal.Length/niris$Petal.Width # add new column
> head(niris)
```

The augmented data frame: niris

```
> head(niris)
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species Sepal.ar Petal.ar
1          5.1         3.5          1.4         0.2  setosa     1.46     7.00
2          4.9         3.0          1.4         0.2  setosa     1.63     7.00
3          4.7         3.2          1.3         0.2  setosa     1.47     6.50
4          4.6         3.1          1.5         0.2  setosa     1.48     7.50
5          5.0         3.6          1.4         0.2  setosa     1.39     7.00
6          5.4         3.9          1.7         0.4  setosa     1.38     4.25
```

Deleting columns

This is easy – but cannot be undone!

To remove a single column, do it by name.

To remove the first column:

```
>   niris$Sepal.Length <- NULL
```

Tedious for multiple columns. A quicker but potentially dangerous way to remove first 4 columns:

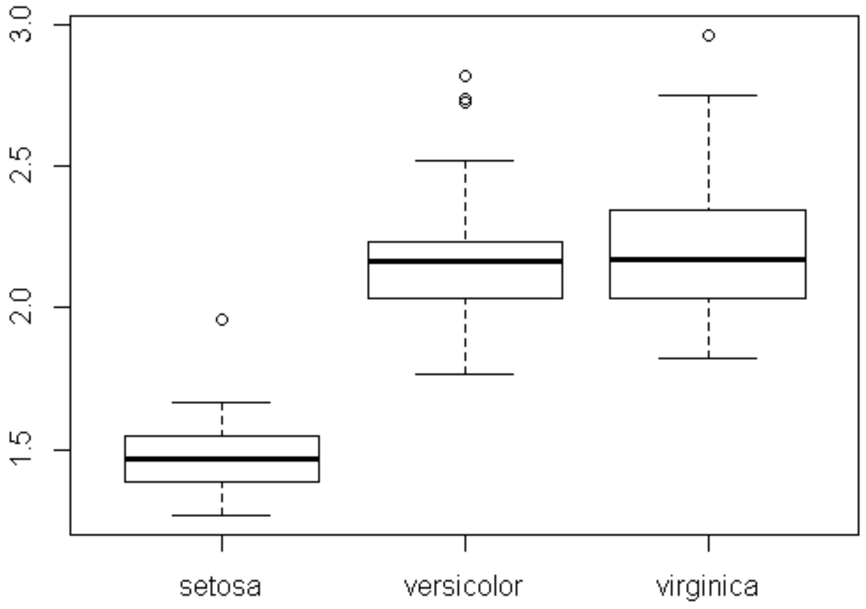
```
>   niris <- niris[,c(5:7)] # reassign cols 5:7 on to self!
```

After removing columns:

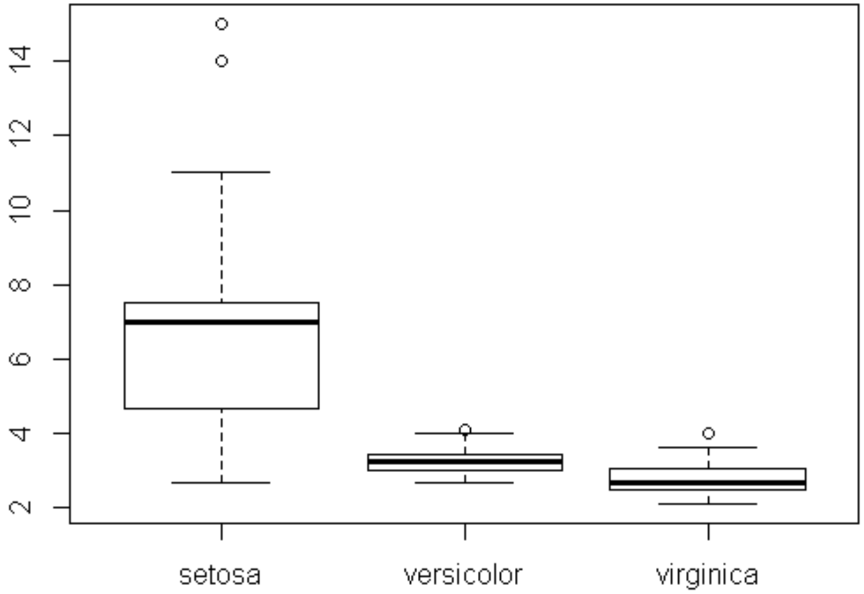
> head(niris)

```
 Species Sepal.ar Petal.ar
1  setosa     1.46     7.00
2  setosa     1.63     7.00
3  setosa     1.47     6.50
4  setosa     1.48     7.50
5  setosa     1.39     7.00
6  setosa     1.38     4.25
```

```
>    boxplot(Sepal.ar~Species, data = niris)
```

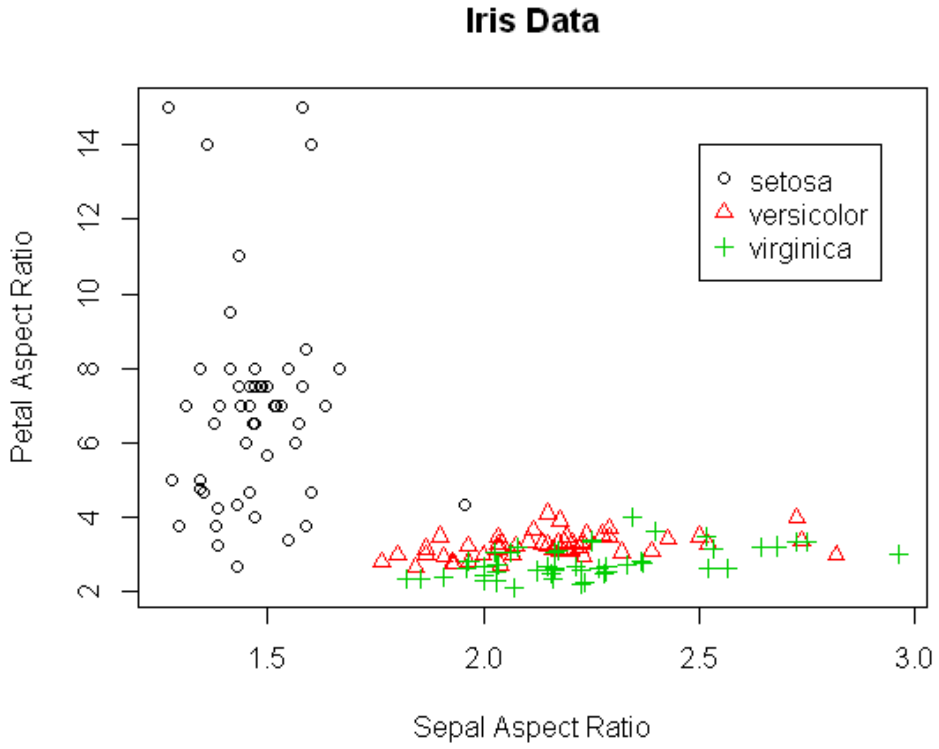


```
>    boxplot(Petal.ar~Species, data = niris)
```



Scatterplot

Petal vs Sepal aspect ratio (Length / Width)



Scatterplot code:

```
>   with(niris, plot(Sepal.ar, Petal.ar, col = Species,
    pch=as.numeric(Species), main = ("Iris Data"), xlab =
    "Sepal Aspect Ratio", ylab = ("Petal Aspect Ratio")))
>   with(niris, legend(2.5, 14, as.vector(unique(Species)),
    pch=unique(Species), col = unique(Species)))
```

Find longest petal for each species

One way to find the longest petal in each species is to apply the ‘aggregate’ function to the Petal.Length column of the dataframe:

```
> aggregate(iris[3], iris[5], max)
     Species Petal.Length
1     setosa          1.9
2 versicolor          5.1
3  virginica          6.9
```

Suppose we want find the flower having the longest petal and report all its measurements.

?which.max

- Description

  Determines the location, i.e., index of the (first) minimum or maximum of a numeric vector.

- Usage

```
which.min(x)
which.max(x)
```

- Arguments

  x : numeric (integer or double) vector, whose min or max is searched for.

Printing the row with the longest petal

To find the row containing the longest petal (ignoring species), use:

```
>   which.max(iris[,3])
    [1] 119
```

To print this row use:

```
>   iris[which.max(iris[,3]),]
        Sepal.Length Sepal.Width Petal.Length Petal.Width   Species
    119          7.7         2.6          6.9         2.3 virginica
```

To find the maximum for each species use ‘by’ function and a temporary data frame.

Longest petal in each group

Putting together gives:

```
>   by(iris, iris[5], function(df) df[which.max(df[,3]),])
    Species: setosa
       Sepal.Length Sepal.Width Petal.Length Petal.Width Species
    25          4.8         3.4          1.9         0.2  setosa
    ------------------------------------------------------------
    Species: versicolor
       Sepal.Length Sepal.Width Petal.Length Petal.Width    Species
    84            6         2.7          5.1         1.6 versicolor
    ------------------------------------------------------------
    Species: virginica
        Sepal.Length Sepal.Width Petal.Length Petal.Width   Species
    119          7.7         2.6          6.9         2.3 virginica
```

We will now tidy up the output...

?do.call

- Description

  do.call constructs and executes a function call from a name or a function and a list of arguments to be passed to it.

- Usage

```
do.call(what, args, quote = FALSE, envir =
parent.frame())
```

- Arguments

  what : function or string naming the function

  args : a list of arguments to the function call.

  quote : indicating whether to quote the arguments.

  envir : an environment within to evaluate the call.

?rbind

- Description

  Take a sequence of vector, matrix or data frames arguments and combine by columns or rows.

- Usage

  ```
  cbind(..., deparse.level = 1)
  rbind(..., deparse.level = 1)
  ```

- Arguments

  ... : vectors or matrices. These can be given as named arguments.

  deparse.level : (ignore at this stage)

Tidying the output

First, assign a variable name (for clarity):

```
>   max.type <- by(iris, iris[5], function(df)
    df[which.max(df[,3]),])

>   do.call(rbind, max.type)
```

```
           Sepal.Length Sepal.Width Petal.Length Petal.Width    Species
setosa              4.8         3.4          1.9         0.2     setosa
versicolor          6.0         2.7          5.1         1.6 versicolor
virginica           7.7         2.6          6.9         2.3  virginica
```

This can be converted to a data frame using:

```
>   XX <- as.data.frame(do.call(rbind, max.type))
```

Working with dates and times

To work with dates in R, you first need to convert the character representation of date into a ‘date’ object using the ‘as.Date’ function so that data is read and interpreted correctly.

- We will do this before applying the ‘which.min’ and ‘which.max’ functions to a date.
- The Dunnhumby data (Tutorial 2) records the sale date and amount spent by 20 customers.
- Find the earliest sale date for each customer.

Dunnhumby : data

customer_id	visit_date	visit_delta	visit_spend
40	4/04/10	NA	44.83
40	6/04/10	2	69.68
40	19/04/10	13	44.61
40	1/05/10	12	30.39
40	2/05/10	1	60.73
40	12/05/10	10	50
40	15/05/10	3	3
40	18/05/10	3	36.89
40	19/05/10	1	9.07
40	23/05/10	4	14.01
40	26/05/10	3	16.97
40	31/05/10	5	8.69
…	…	…	…

Without date conversion

Calculating minimums without date conversion:

```
>   min.type <- by(DH, DH[1], function(df)
    df[which.min(df[,2]),])
>   do.call(rbind,min.type)
```

```
.    customer_id visit_date visit_delta visit_spend
40            40   01-05-10          12       30.39
79            79   01-01-11           9       81.70
119          119   01-03-11           3       10.69
123          123   01-02-11           4       35.20
134          134   01-02-11           1       54.77
```

With date conversion

Calculating minimums with date conversion:

```
>   min.type <- by(DH, DH[1], function(df)
    df[which.min(as.Date(df[,2],"%d-%m-%y")),])
>   do.call(rbind,min.type)
```

```
.    customer_id visit_date visit_delta visit_spend
40            40   04-04-10          NA       44.83
79            79   07-04-10          NA      150.87
119          119   01-04-10          NA       20.00
123          123   02-04-10          NA       66.94
134          134   01-04-10          NA       50.32
```

Summary

Summarizing data using factors

- Functions:
  - > aggregate, by

Creating and removing columns

Searching, indexing and combining rows

- Functions:
  - > as.table, as.data.frame, colnames, which.max, do.call, rbind, cbind

Answers to the review questions

1. D: Hierarchy (Phylogenetic tree)
2. A: Time Series (Daily infections)
3. E: Network (Transmission network)
4. C: Map (Cases: bubble graph)
5. C: Map (Cases: Choropleth, from the WHO COVID-19 dashboard)

References

Books – online from the Monash Library

- Spector, P., Data manipulation with R. (pp 113 – 118 used as a reference for last part of today's lecture)
- Wickham, H., ggplot2: elegant graphics for data analysis.

R Reference card (Tom Short) available from contributed documentation on CRAN site.

http://cran.r-project.org/