Data Science Specialization

Coursera Course

Author: Shivani Nandani

Institute: Johns Hopkins University

Date: January 2022

Version: 1.0

Bio: A ten-course introduction to data science.

Victory won't come to us unless we go to it.

Contents

Chapter 1 The Data Scientist's Toolbox

Course[3] outcomes:

- an introduction to the main tools and ideas in the data scientist's toolbox
- an overview of the data, questions, and tools that data analysts and data scientists work with
- two components:
  1. a conceptual introduction to the ideas behind turning data into actionable knowledge
  2. a practical introduction to the tools that will be used in the program like version control, markdown, git, GitHub, R, and RStudio

1.1 What is data science?

- can mean different things to different people
- at it's core, data science is using data to answer questions
- involves
  - statistics, computer science, mathematics
  - data cleaning and formatting
  - data visualization
- An Economist Special Report sums up this melange of skills well - they state that a data scientist is broadly defined as someone "who combines the skills of software programmer, statistician and storyteller slash artist to extract the nuggets of gold hidden under mountains of data"

1.2 What is big data?

- vast amount of data collected in currently
- rise of inexpensive computing
- **volume** - big data involves large datasets - and these large datasets are becoming more and more routine
- **velocity** - generated and collected faster than ever before
- **variety** - different types of data

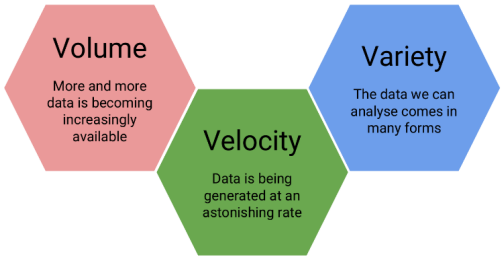

Figure 1.1: A summary of three qualities that characterize big data

1.3 What is a data scientist?

- the most basic of definitions would be that a data scientist is somebody who uses data to answer questions

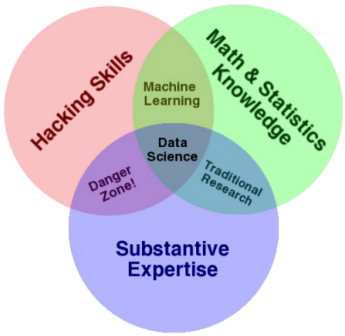

Figure 1.2: Drew Conway's Venn diagram of data science

- according to the illustrative Venn diagram[1], data science is the intersection of three sectors - substantive expertise, hacking skills, and math and statistic

1.4 What is data science?

- Cambridge English Dictionary definition - *Information, especially facts or numbers, collected to be examined and considered and used to help decision-making.*
- Wikipedia definition - *A set of values of qualitative or quantitative variables.*
- both these definitions say that data is values of numbers or facts but the Cambridge definition focuses more on the *what* it is used for while the wikipedia definition talks about what data entails. here, "a set of values" talks about the population, the set of a whole that you are trying to discover something about. The "variables" are measurements or characteristics of an item (example - height of a person). It can be either qualitative or quantitative.
- common types of messy data:
  - sequencing data
  - population census data
  - electronic media records (EMR) and other large datasets
  - geographic information system (GIS) data (mapping)
  - image analysis and image extrapolation
  - language and translations
  - website traffic
  - personal/ad data (e.g.: facebook, netflix predictions etc.)
- Sample project - Hilary: the most poisoned baby name in US history

1.5 R programming language

- download from Comprehensive R Archive Network (CRAN)
- focused on statistical analysis and graphics
- A package is not to be confused with a library (these two terms are often conflated in colloquial speech about R). A library is the place where the package is located on your computer. To think of an analogy, a library is, well, a library... and a package is a book within the library. The library is where the

[1] http://drewconway.com/zia/2013/3/26/the-data-science-venn-diagram

books/packages are located.

1.6 Types of Data Science

There are, broadly speaking, six categories in which data analyses fall.

1. Descriptive
2. Exploratory
3. Inferential
4. Predictive
5. Causal
6. Mechanistic

1.6.1 Descriptive analysis

The goal of descriptive analysis is to describe or summarize a set of data. Whenever you get a new dataset to examine, this is usually the first kind of analysis you will perform. Descriptive analysis will generate simple summaries about the samples and their measurements.

This type of analysis is aimed at summarizing your sample - not for generalizing the results of the analysis to a larger population or trying to make conclusions. Description of data is separated from making interpretations; generalizations and interpretations require additional statistical steps.

Some examples of purely descriptive analysis can be seen in censuses. Here, the government collects a series of measurements on all of the country's citizens, which can then be summarized. Here, you are being shown the age distribution in the US, stratified by sex. The goal of this is just to describe the distribution. There is no inferences about what this means or predictions on how the data might trend in the future. It is just to show you a summary of the data collected.

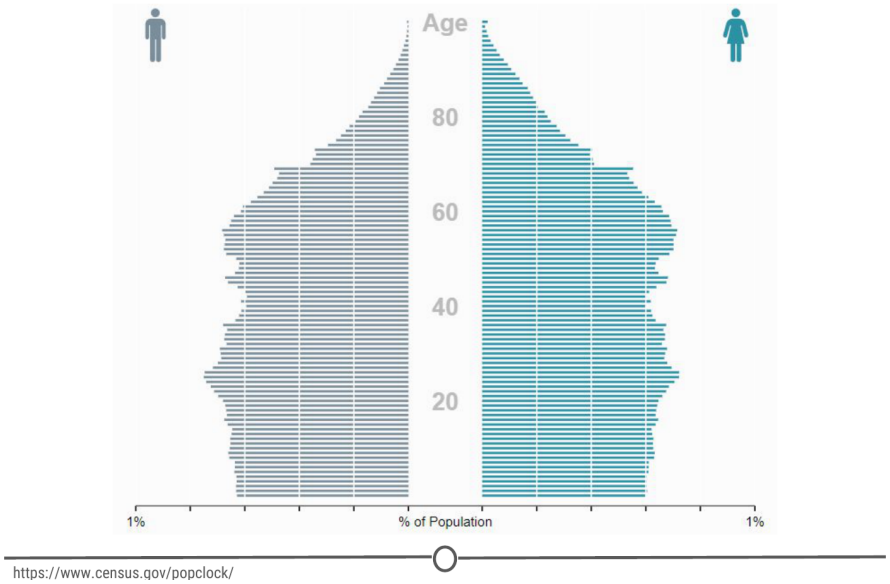

Figure 1.3: A population pyramid describing the population distribution in the US

1.6.2 Exploratory analysis

The goal of exploratory analysis is to examine or explore the data and find relationships that weren't previously known. Exploratory analyses explore how different measures might be related to each other but do not confirm that relationship as causative. "Correlation does not imply causation" - just because you observe a relationship between two variables during exploratory analysis, it does not mean that one necessarily causes the other.

Because of this, exploratory analysis, while useful for discovering new connections, should not be the final say in answering a question. It can allow you to formulate hypotheses and drive the design of future studies and data collection, but exploratory analysis alone should never be used as the final say on why or how data might be related to each other.

1.6.3 Inferential analysis

The goal of inferential analyses is to use a relatively small sample of data to infer or say something about the population at large. Inferential analysis is commonly the goal of statistical modelling, where you have a small amount of information to extrapolate and generalize that information to a larger group.

Inferential analysis typically involves using the data you have to estimate that value in the population and then give a measure of your uncertainty about your estimate. Since you are moving from a small amount of data and trying to generalize to a larger population, your ability to accurately infer information about the larger population depends heavily on your sampling scheme - if the data you collect is not from a representative sample of the population, the generalizations you infer won't be accurate for the population.

An example of inferential analysis is a study in which a subset of the US population was assayed for their life expectancy given the level of air pollution they experienced. This study uses the data they collected from a sample of the US population to infer how air pollution might be impacting life expectancy in the entire US.

1.6.4 Predictive analysis

The goal of predictive analysis is to use current data to make predictions about future data. Essentially, you are using current and historical data to find patterns and predict the likelihood of future outcomes.

Like in inferential analysis, your accuracy in predictions is dependent on measuring the right variables. If you aren't measuring the right variables to predict an outcome, your predictions aren't going to be accurate. Additionally, there are many ways to build up prediction models with some being better or worse for specific cases, but in general, having more data and a simple model generally performs well at predicting future outcomes.

All this being said, much like in exploratory analysis, just because one variable may predict another, it does not mean that one causes the other; you are just capitalizing on this observed relationship to predict the second variable.

A common saying is that prediction is hard, especially about the future. There aren't easy ways to gauge how well you are going to predict an event until that event has come to pass; so evaluating different approaches or models is a challenge.

We spend a lot of time trying to predict things - the upcoming weather, the outcomes of sports events, and in the example we'll explore here, the outcomes of elections. Nate Silver of FiveThirtyEight tries and predicts the outcomes of U.S. elections (and sports matches) using historical polling data and trends and current polling. FiveThirtyEight's models accurately predicted the 2008 and 2012 elections and was widely considered

an outlier in the 2016 US elections, as it was one of the few models to suggest Donald Trump at having a chance of winning.

1.6.5 Causal analysis

The caveat to a lot of the analyses we've looked at so far is that we can only see correlations and can't get at the cause of the relationships we observe. Causal analysis fills that gap; the goal of causal analysis is to see what happens to one variable when we manipulate another variable - looking at the cause and effect of a relationship.

Generally, causal analyses are fairly complicated to do with observed data alone; there will always be questions as to whether it is correlation driving your conclusions or that the assumptions underlying your analysis are valid. More often, causal analyses are applied to the results of randomized studies that were designed to identify causation. Causal analysis is often considered the gold standard in data analysis, and is seen frequently in scientific studies where scientists are trying to identify the cause of a phenomenon, but often getting appropriate data for doing a causal analysis is a challenge.

One thing to note about causal analysis is that the data is usually analysed in aggregate and observed relationships are usually average effects; so, while on average giving a certain population a drug may alleviate the symptoms of a disease, this causal relationship may not hold true for every single affected individual.

As we've said, many scientific studies allow for causal analyses. Randomized control trials for drugs are a prime example of this. For example, one randomized control trial examined the effects of a new drug on treating infants with spinal muscular atrophy. Comparing a sample of infants receiving the drug versus a sample receiving a mock control, they measure various clinical outcomes in the babies and look at how the drug affects the outcomes.

1.6.6 Mechanistic analysis

Mechanistic analyses are not nearly as commonly used as the previous analyses - the goal of mechanistic analysis is to understand the exact changes in variables that lead to exact changes in other variables. These analyses are exceedingly hard to use to infer much, except in simple situations or in those that are nicely modeled by deterministic equations. Given this description, it might be clear to see how mechanistic analyses are most commonly applied to physical or engineering sciences; biological sciences, for example, are far too noisy of data sets to use mechanistic analysis. Often, when these analyses are applied, the only noise in the data is measurement error, which can be accounted for.

You can generally find examples of mechanistic analysis in material science experiments. Here, we have a study on biocomposites (essentially, making biodegradable plastics) that was examining how biocarbon particle size, functional polymer type and concentration affected mechanical properties of the resulting "plastic." They are able to do mechanistic analyses through a careful balance of controlling and manipulating variables with very accurate measures of both those variables and the desired outcome.

1.7 Experimental Design

- Experimental design is organizing an experiment so that you have the correct data (and enough of it!) to clearly and effectively answer your data science question. This process involves clearly formulating your question in advance of any data collection, designing the best set-up possible to gather the data to

answer your question, identifying problems or sources of error in your design, and only then, collecting the appropriate data.

- Going into an analysis, you need to have a plan in advance of what you are going to do and how you are going to analyse the data. If you do the wrong analysis, you can come to the wrong conclusions.

1.7.1 Principles of experimental design

- **Independent variable (AKA factor):** The variable that the experimenter manipulates; it does not depend on other variables being measured. Often displayed on the x-axis.
- **Dependent variable:** The variable that is expected to change as a result of changes in the independent variable. Often displayed on the y-axis, so that changes in X, the independent variable, effect changes in Y.
- So when you are designing an experiment, you have to decide what variables you will measure, and which you will manipulate to effect changes in other measured variables. Additionally, you must develop your hypothesis, essentially an educated guess as to the relationship between your variables and the outcome of your experiment.
- **Confounder:** An extraneous variable that may affect the relationship between the dependent and independent variables.
- In some experimental design paradigms, a control group may be appropriate. This is when you have a group of experimental subjects that are not manipulated. So if you were studying the effect of a drug on survival, you would have a group that received the drug (treatment) and a group that did not (control). This way, you can compare the effects of the drug in the treatment versus control group.
- In these study designs, there are other strategies we can use to control for confounding effects. One, we can blind the subjects to their assigned treatment group. Sometimes, when a subject knows that they are in the treatment group (eg: receiving the experimental drug), they can feel better, not from the drug itself, but from knowing they are receiving treatment. This is known as the placebo effect. To combat this, often participants are blinded to the treatment group they are in; this is usually achieved by giving the control group a mock treatment (eg: given a sugar pill they are told is the drug). In this way, if the placebo effect is causing a problem with your experiment, both groups should experience it equally.
- Blinding your study means that your subjects don't know what group they belong to - all participants receive a "treatment". And this strategy is at the heart of many of these studies; spreading any possible confounding effects equally across the groups being compared. For example, if you think age is a possible confounding effect, making sure that both groups have similar ages and age ranges will help to mitigate any effect age may be having on your dependent variable - the effect of age is equal between your two groups.
- This "balancing" of confounders is often achieved by randomization. Generally, we don't know what will be a confounder beforehand; to help lessen the risk of accidentally biasing one group to be enriched for a confounder, you can randomly assign individuals to each of your groups. This means that any potential confounding variables should be distributed between each group roughly equally, to help eliminate/reduce systematic errors.
- Replication is pretty much what it sounds like, repeating an experiment with different experimental subjects. A single experiment's results may have occured by chance; a confounder was unevenly distributed across your groups, there was a systematic error in the data collection, there were some outliers, etc.

However, if you can repeat the experiment and collect a whole new set of data and still come to the same conclusion, your study is much stronger. Also at the heart of replication is that it allows you to measure the variability of your data more accurately, which allows you to better assess whether any differences you see in your data are significant.

1.7.2 Sharing data

Share your data and code using GitHub.
Guide - https://github.com/jtleek/datasharing

1.7.3 p-hacking

One of the many things often reported in experiments is a value called the p-value[2]. This is a value that tells you the probability that the results of your experiment were observed by chance.

What you need to look out for is when you manipulate p-values towards your own end. Often, when your p-value is less than 0.05 (in other words, there is a 5 percent chance that the differences you saw were observed by chance), a result is considered significant. But if you do 20 tests, by chance, you would expect one of the twenty (5%) to be significant. In the age of big data, testing twenty hypotheses is a very easy proposition. And this is where the term p-hacking comes from: This is when you exhaustively search a data set to find patterns and correlations that appear statistically significant by virtue of the sheer number of tests you have performed. These spurious correlations can be reported as significant and if you perform enough tests, you can find a data set and analysis that will show you what you wanted to see.

Check out this FiveThirtyEight[3] activity where you can manipulate and filter data and perform a series of tests such that you can get the data to find whatever relationship you want.

1.8 Big Data

- three qualities commonly attributed to big data sets: Volume, Velocity, Variety.
- even though these qualities are not new, big data became very popular recently due to the technology and data storage has evolved to be able to hold larger and larger data steps
- our ability to collect and record data has improved with time such that the speed with which data is collected is unprecedented

1.8.1 Structured and unstructured data

Structured data is what you traditionally might think of data; long tables, spreadsheets, or databases with columns and rows of information that you can sum or average or analyse however you like within those confines. Unfortunately, this is rarely how data is presented to you in this day and age. The data sets we commonly encounter are much messier, and it is our job to extract the information we want and corral it into something tidy and structured.

With the digital age and the advance of the internet, many pieces of information that weren't traditionally collected were suddenly able to be translated into a format that a computer could record, store, search, and

[2] https://www.youtube.com/watch?v=UsU-02Z1rAs

[3] https://projects.fivethirtyeight.com/p-hacking/

analyse. And once this was appreciated, there was a proliferation of this unstructured data being collected from all of our digital interactions: emails, Facebook and other social media interactions, text messages, shopping habits, smartphones (and their GPS tracking), websites you visit, how long you are on that website and what you look at, CCTV cameras and other video sources, etc. The amount of data and the various sources that can record and transmit data has exploded.

It is because of this explosion in the volume, velocity, and variety of data that "big data" has become so salient a concept; these data sets are now so large and complex that we need new tools and approaches to make the most of them. As you can guess given the variety of data types and sources, very rarely is the data stored in a neat, ordered spreadsheet, that traditional methods for cleaning and analysis can be applied to!

1.8.2 Challenges of working with big data

Given some of the qualities of big data above, you can already start seeing some of the challenges that may be associated with working with big data.

- It is big: there is a lot of raw data that you need to be able to store and analyse;
- It is constantly changing and updating: By the time you finish your analysis, there is even more new data you could incorporate into your analysis! Every second you are analyzing, is another second of data you haven't used!
- The variety can be overwhelming: There are so many sources of information that it can sometimes be difficult to determine what source of data may be best suited to answer your data science question! And finally,
- It is messy: You don't have neat data tables to quickly analyse - you have messy data. Before you can start looking for answers, you need to turn your unstructured data into a format that you can analyse!

1.8.3 Benefits to working with big data

So with all of these challenges, why don't we just stick to analyzing smaller, more manageable, curated datasets and arriving at our answers that way?

Sometimes questions are best addressed using these smaller datasets, but many questions benefit from having lots and lots of data, and if there is some messiness or inaccuracies in this data, the sheer volume of it negates the effect of these small errors. So we are able to get closer to the truth even with these messier datasets.

Additionally, when you have data that is constantly updating, while this can be a challenge to analyse, the ability to have real time, up to date information allows you to do analyses that are accurate to the current state and make on the spot, rapid, informed predictions and decisions.

One of the benefits of having all these new sources of information is that questions that weren't previously able to be answered due to lack of information, suddenly have many more sources to glean information from and new connections and discoveries are now able to be made! Questions that previously were inaccessible now have newer, unconventional data sources that may allow you to answer these formerly unfeasible questions.

Another benefit to using big data is that it can identify hidden correlations. Since we can collect data on a myriad of qualities on any one subject, we can look for qualities that may not be obviously related to our outcome variable, but the big data can identify a correlation there - instead of trying to understand precisely why an engine breaks down or why a drug's side effect disappears, researchers can instead collect and analyze massive quantities of information about such events and everything that is associated with them, looking for

patterns that might help predict future occurrences. Big data helps answer what, not why, and often that's good enough.

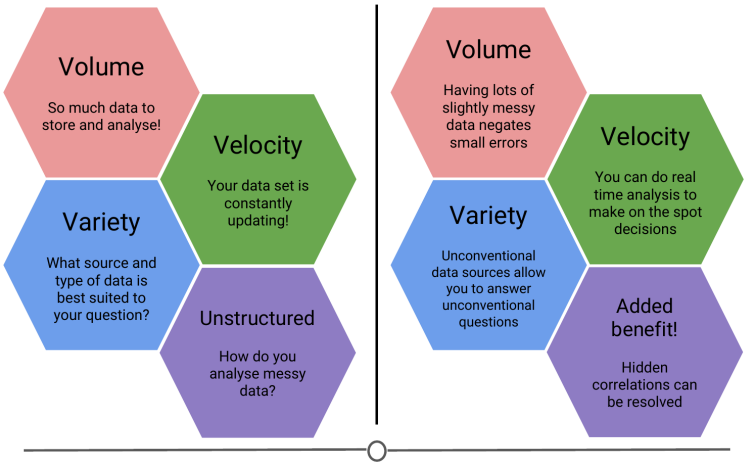

Figure 1.4: Comparing the challenges and benefits to working with big data

Chapter 2 R programming

Course[2] outcomes:

- how to program in R
- how to use R for effective data analysis
- how to install and configure software necessary for a statistical programming environment and describe generic programming language concepts as they are implemented in a high-level statistical language
- programming in R
- reading data into R
- accessing R packages
- writing R functions
- debugging, profiling R code
- organizing and commenting R code

2.1 References

- https://www.rstudio.com/resources/cheatsheets/
- https://iqss.github.io/dss-workshops/R/Rintro/base-r-cheat-sheet.pdf
- https://www.codecademy.com/learn/learn-r/modules/learn-r-introduction/cheatsheet
- http://datasciencefree.com/basicR.pdf

2.2 What is R?

- R is a dialect of S.
- S is a language that was developed by John Chambers and others at Bell Labs.
- S philosophy - wanted to create an interactive environment where the users did not consciously think of themselves as programming. Then as their needs becomes clearer and their sophistication increases, they should be able to slide gradually into programming, when the language and system aspects become more important.
- R was created in New Zealand by Ross Ihaka and Robert Gentleman.
- syntax is very similar to S.

2.3 R data types

- "<-" is the assignment operator
- "#" indicates comment
- 1:20 creates a sequence 1 to 20
- everything in R is an object
  five basic/atomic classes of objects:
  - character
  - numeric (real numbers)
  - integer

  - complex
  - logical (true/false)
- most basic object is a vector
  - a vector can only contain objects of the same class
  - exception - list (represented as a vector) can contain objects of different classes
  - empty vectors can be created with `vector()` function
  - the function has two arguments - class type and length
- numbers in R
  - numbers are generally treated as numeric objects (i.e. double precision real numbers)
  - specify `L` suffix if integer type required
  - `Inf` represents infinity
  - `NaN` represents an undefined value ("not a number") (e.g. 0/0 or a missing value)
- R objects can have the following attributes (`attributes()` allows you to access the attributes of an object)
  - names, dimnames
  - dimensions (matrices, arrays)
  - class
  - length
  - other user-defined attributes/metadata
- creating vectors
  - the function `c()` can be used to create vectors of objects
    e.g. `x <- c(0.5, 0.6)` gives a numeric vector of length two with first element as 0.5 and second element as 0.6
  - example using the function `vector()` – `x <- vector("numeric", length = 10)` – creates a numeric type vector of length 10 with all values initialized to 0
- when different objects are mixed in a vector, *coercion* occurs so that every element in the vector is of the same class
- explicit coercion can be done using the `as.*` functions (e.g. `as.numeric(vector_name)` etc.)
- when coercion fails, you get `NA` values as a result
- lists
  - creating using `list()` function
  - example: `x <- list(1, "a", TRUE, 1+4i)` – creates a list of number, character, logical and complex
- matrices
  - matrices are vectors with a dimension attribute
  - matrix is not a class of objects
  - the dimension attribute is itself an integer vector of length 2 (nrow, ncol)
  - example: `m <- matrix(nrow = 2, ncol = 3)`
  - `dim(m)` gives the dimension of the matrix
  - the matrices are constructed in a column-wise manner
  - cbind-ing and rbind-ing - column binding and row binding respectively
- factors
  - used to represent categorical data

  - can be ordered or unordered
  - a factor can be thought of as an integer vector where each integer has a label
  - treated specially by modeling functions like `lm()` and `glm()`
  - function - `factor()`
  - also has level
  - `unclass(x)` strips the levels and gives integer
  - the order of the levels can be set using the `levels` argument to `factor()`. this is important because the first level is used using the baseline level (which is determined in an alphabetical order by default)
- missing values
  - missing values are denoted by `NA` or `NaN` (`NaN` is used for undefined mathematical operations)
  - `is.na()` is used to test objects if they are `NA`
  - `is.nan()` is used to test objects if they are `NaN`
  - `is.na()` is used to test objects if they are `NA`
  - `NA` can have class - integer, character etc.
  - A `NaN` is also a `NA` but the converse is not true
- data frames
  - used to store tabular data
  - represented as a special type of list where every element of the list has to have the same length
  - each element of the list can be thought of as a column and the length of each element of the list is the number of rows
  - can store different classes
  - also have a special attribute called `row.names`
  - usually created by calling `read.table()` or `read.csv()` or `data.frame()`
  - can be converted to a matrix by calling `data.matrix()`
  - example: `x <- data.frame(foo = 1:4, bar = c(T, T, F, F))` creates a data frame of the following kind

```
  foo   bar
1   1  TRUE
2   2  TRUE
3   3 FALSE
4   4 FALSE
```

- names attribute
  - R objects can also have names, which is very useful for writing readable code and self-describing objects
  - set using function `names()`
  - example: `names(x) <- c("foo", "bar", "norf")`
  - list can also have names
  - matrices have dimnames

2.4 Dealing with data

- reading data
  - `read.table, read.csv` for reading tabular data
  - `readLines` for reading lines of a text file
  - `source` for reading R code files (`inverse of dump`)
  - `dget` for reading in R code files (`inverse of dout`)
  - `load` for reading in saved workspaces
  - `unserialize` for reading single R objects in binary form
- writing data
  - `write.table`
  - `writeLines`
  - `dump`
  - `dput`
  - `save`
  - `serialize`
- `read.csv` has comma as separator while `read.table` has space as separator
- reading lager datasets with `read.table`
  - read the help page for `read.table`
  - have a rough idea of the amount of data that will be read - to understand if you have enough memory to store the dataset
  - how to calculate this?

Bibliography

[1] Ethan Deng and Liam Huang. "ElegantBook Template". In: *ElegantLaTeX Program* (2021). URL: https://www.overleaf.com/latex/templates/elegantbook-template/kdfqycvyydcn.

[2] Brian Caffo Jeff Leek Roger D. Peng. "R Programming". In: *Coursera - Johns Hopkins University* (2022). URL: https://www.coursera.org/learn/r-programming.

[3] Brian Caffo Jeff Leek Roger D. Peng. "The Data Scientist's Toolbox". In: *Coursera - Johns Hopkins University* (2022). URL: https://www.coursera.org/learn/data-scientists-tools.