STAT1378

CODING AND COMMUNICATION IN STATISTICS

Assignment 2 2021

## Instructions

- Due date: Friday, 15 October 2021, 11:55 PM
- Total points: 100, scaled to 15% of the unit marks
- Submission of a `zip` file, named `Assignment2.zip`, which may include
  - a single `Rmd` file and a `bib` file for the references.
  - a `zip` file named `assign_q2.zip` containing your package for Q2.
    - a separate submission of your package via `Github Classroom` is needed for achieving a higher grade
- Remember to comment your code, markers have limited time allocated to mark and won't be able to debug your code if it gets too time-consuming. You are not writing code for yourself or even your present self. Comments are an integral part of your assignment.
- There is no minimum and maximum page requirement–try and express your ideas clearly and concisely.
- The skills tested in this assignment will be the basis for the project. It is a good idea to avoid copying from other students. Discussing strategies is good, as it favours your learning, while copying code is going to damage you in the long run, both in this unit and in the continuation of your studies.

**Exercise 1** [**20 points**] As the output of this exercise, you are required to submit your code and its output under a section in your markdown file.

The data is created by Benedikt Claus. It's about Mario Kart World Records and contains world records for the classic racing game on the Nintendo 64.

The records are saved in the `records.csv` file and contain the following variables:

| variable | class | description |
|---|---|---|
| track | character | Track name |
| type | factor | Single or three lap record |
| shortcut | factor | Shortcut or non-shortcut record |
| player | character | Player's name |
| system_played | character | Used system (NTSC or PAL) |
| date | date | World record date |
| time_period | period | Time as 'hms' period |
| time | double | Time in seconds |
| record_duration | double | Record duration in days |

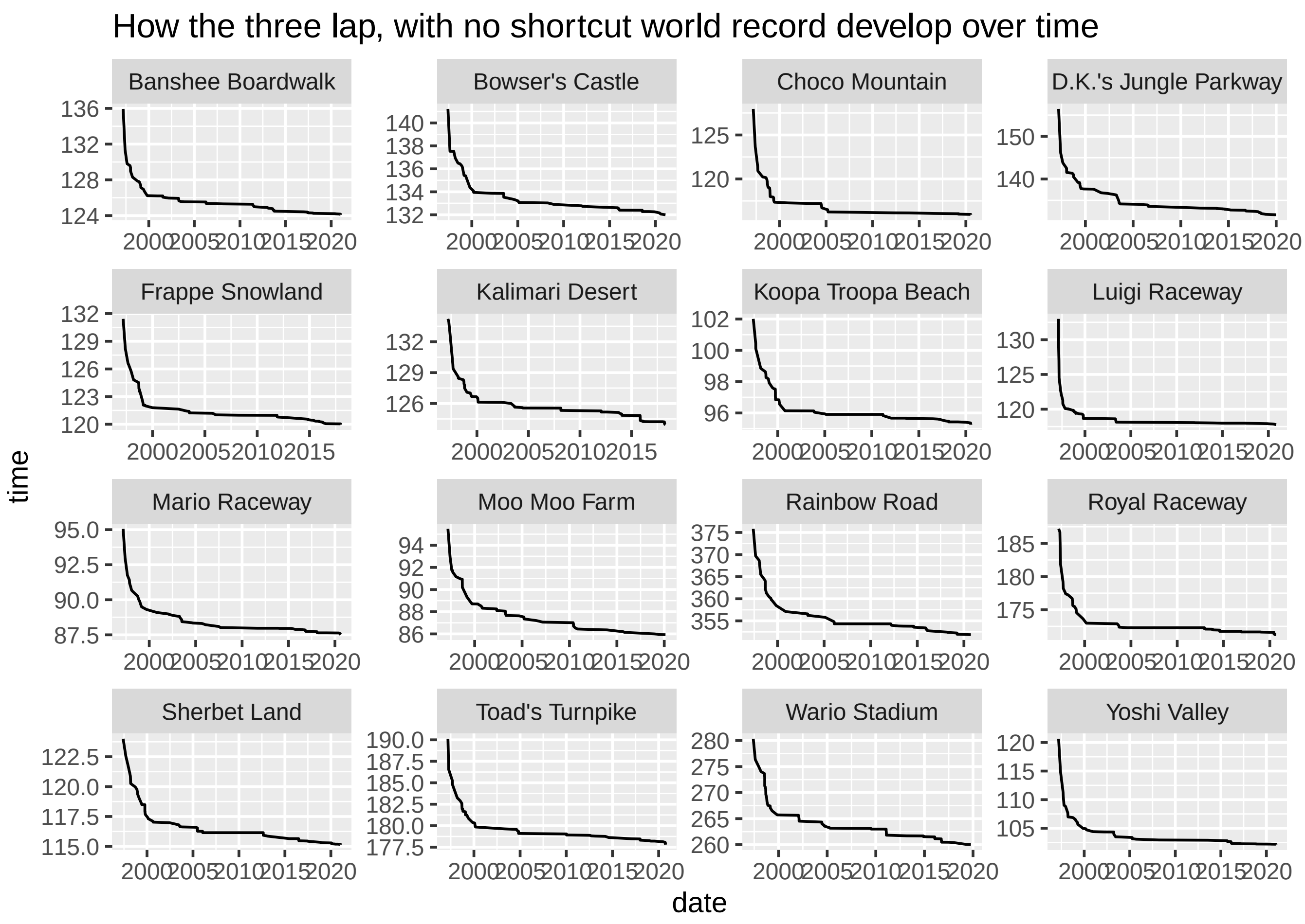
This Video talks about the history of Mario Kart 64 World Records in greater detail. Despite it's first release back in 1996, it is still actively played by many and latest world records was achieved in Feb 2021 when the dataset was created in May 2021.

The game consists of 16 individual tracks, and world records can be achieved for the fastest single lap or the fastest completed race (for three laps). Also, through the years, players discovered shortcuts in many of the tracks. Fortunately, shortcut and non-shortcut world records are listed separately.

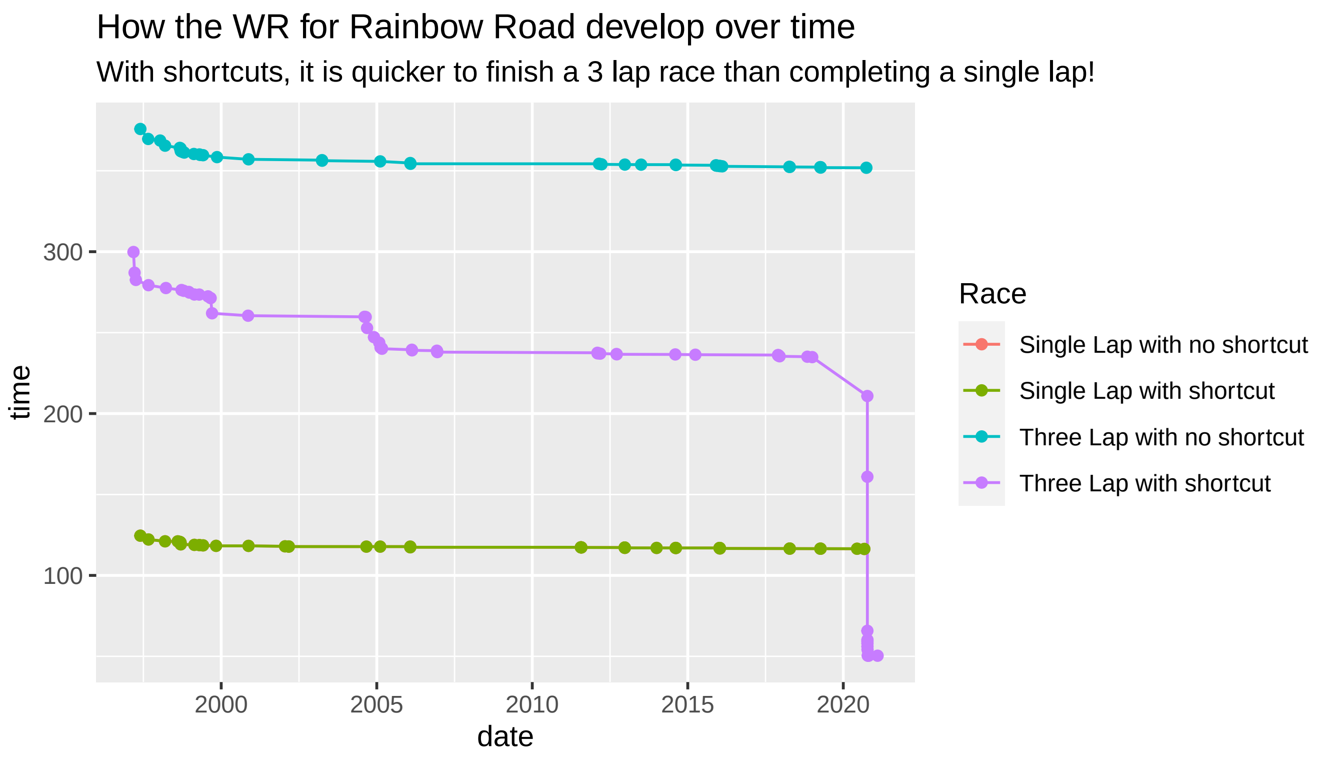
Furthermore, the Nintendo 64 was released for NTSC- and PAL-systems. On PAL-systems, the game runs a little slower at 25 frames per second. All times in this dataset are PAL-times, but they can be converted back to NTSC-times.

For this question, you have to reproduce the following four visualisations. You may also have to perform some data wrangling. Each plot is worth 5 marks.

**Plot 1**

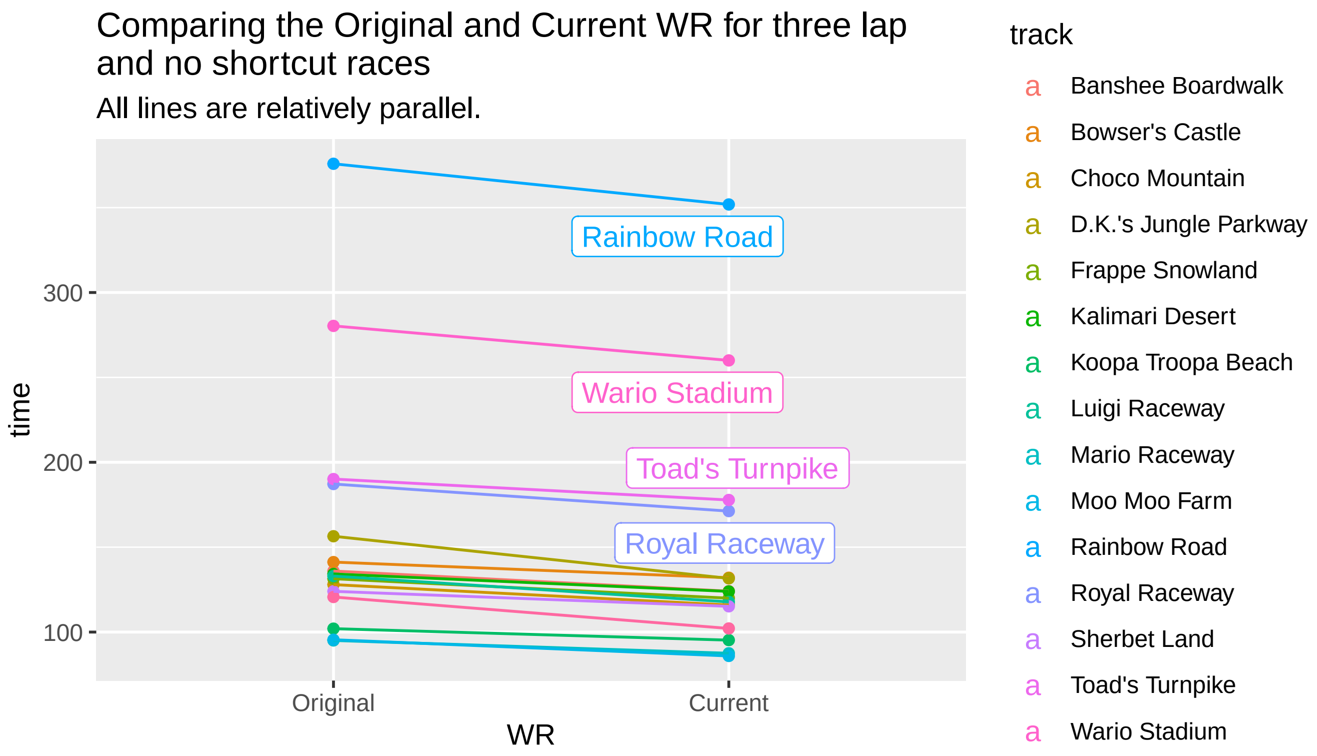


**Plot 2**



How the WR for Rainbow Road develop over time
With shortcuts, it is quicker to finish a 3 lap race than completing a single lap!
time
300
200
100
2000
2005
2010
2015
2020
date
Race
Single Lap with no shortcut
Single Lap with shortcut
Three Lap with no shortcut
Three Lap with shortcut

**Plot 3**

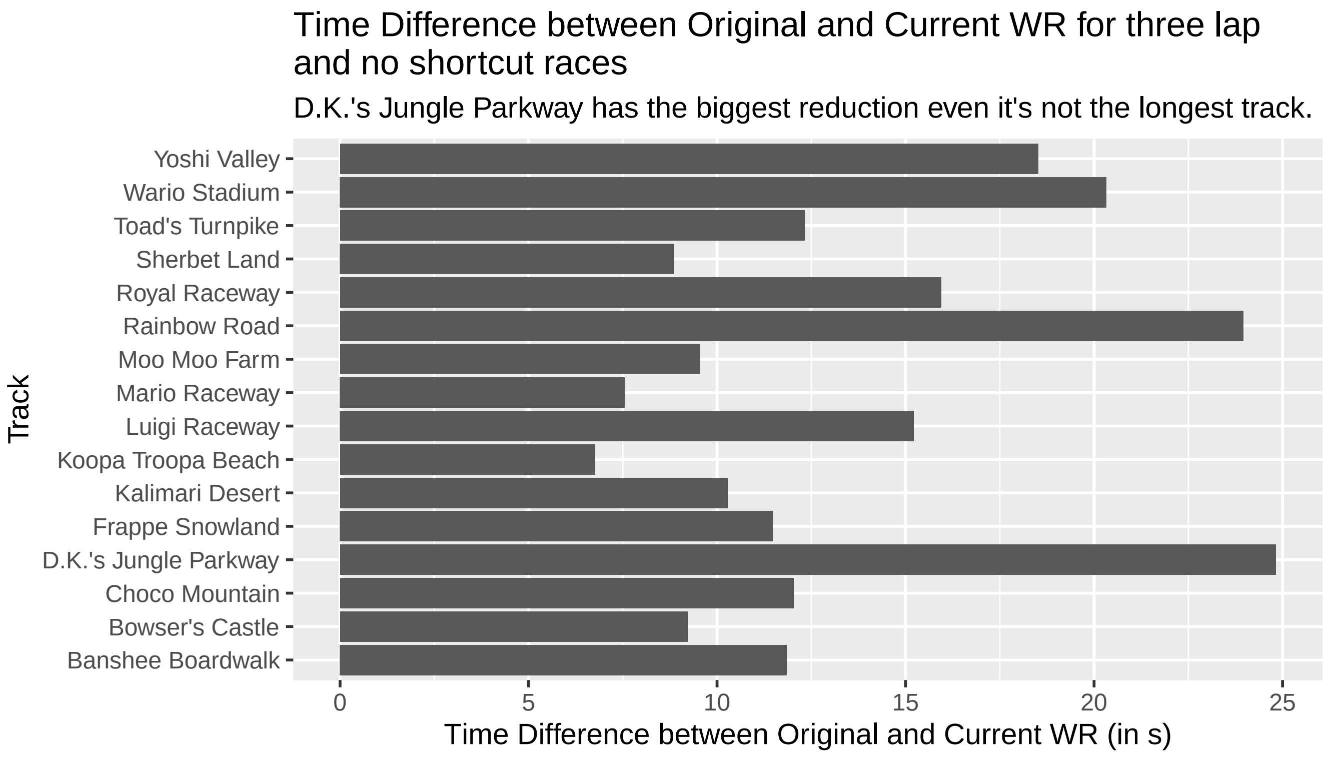


Comparing the Original and Current WR for three lap and no shortcut races
All lines are relatively parallel.
time
300
200
100
Rainbow Road
Wario Stadium
Toad's Turnpike
Royal Raceway
Original
Current
WR
track
Banshee Boardwalk
Bowser's Castle
Choco Mountain
D.K.'s Jungle Parkway
Frappe Snowland
Kalimari Desert
Koopa Troopa Beach
Luigi Raceway
Mario Raceway
Moo Moo Farm
Rainbow Road
Royal Raceway
Sherbet Land
Toad's Turnpike
Wario Stadium

Hint:

- I used `ggrepel::geom_label_repel` to generate the label with the default `max.overlaps` setting. Depends on the size of your plot/window, you may display more or less of the labels.

**Plot 4**



**Exercise 2** [**20 points**] As the output of this exercise, you are required to submit an R package either as a zip file or push it to `Github`.

If you intend to submit your assignment via `Github`, you will have to use this URL: https://classroom.github.com/a/oBJL4po- to create a repository that is linked to your account. This will allow our marker(s) to gain access to your repository. We won't accept your `Github` submission if your repository is not created this way.

Your repository URL would look something like this:

https://github.com/MQ-STAT1378/assignment2-question2-yournamehere

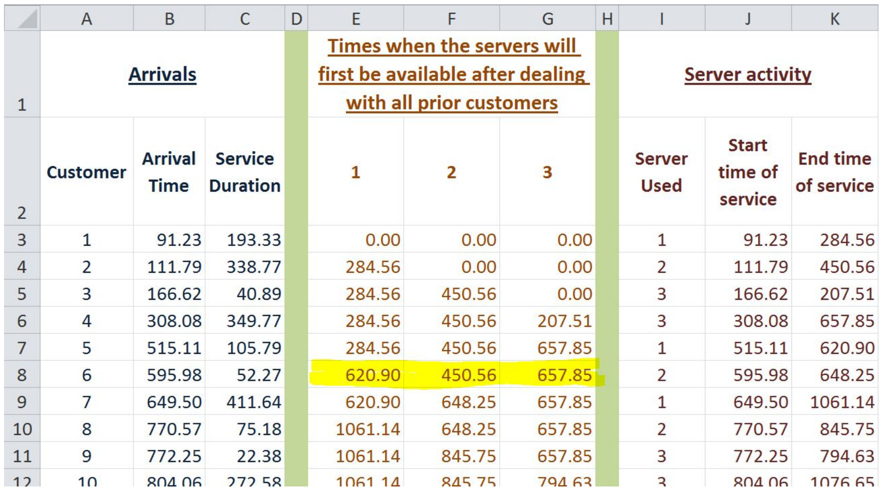
You can then clone the (blank) repo to your computer, and it can then be used just like any other repositories.

If you are using RStudio cloud for this, you will have to link your `Github` account to the Rstudio cloud signin. This can be done by changing the setting in `Authentication`. It is imperative to click the option `Private repo access also enabled`.

Your task for this question is to create an R package using the script `Multiserver.R` (available on iLearn), which is used to simulate how customers will go through a first come, first serve queuing system as long as "fate" has already decided when each customer will arrive and their service times.

For those that interested in the logic behind `Multiserver.R`, we can use `bankmodel.xlsx` (available on iLearn) to illustrate a queuing system in a bank. Notice that those times can be regenerated, so the numbers on the spreadsheet may differ from those shown below.

We want to calculate the times at which each customer's service time starts and end. Consider Customer 6. The highlighted times show when the three servers will become available after dealing with Customers 1 through 5.



| | A | B | C | D | E | F | G | H | I | J | K |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | | Arrivals | | | | Times when the servers will first be available after dealing with all prior customers | | | | Server activity | |
| 2 | Customer | Arrival Time | Service Duration | | 1 | 2 | 3 | | Server Used | Start time of service | End time of service |
| 3 | 1 | 91.23 | 193.33 | | 0.00 | 0.00 | 0.00 | | 1 | 91.23 | 284.56 |
| 4 | 2 | 111.79 | 338.77 | | 284.56 | 0.00 | 0.00 | | 2 | 111.79 | 450.56 |
| 5 | 3 | 166.62 | 40.89 | | 284.56 | 450.56 | 0.00 | | 3 | 166.62 | 207.51 |
| 6 | 4 | 308.08 | 349.77 | | 284.56 | 450.56 | 207.51 | | 3 | 308.08 | 657.85 |
| 7 | 5 | 515.11 | 105.79 | | 284.56 | 450.56 | 657.85 | | 1 | 515.11 | 620.90 |
| 8 | 6 | 595.98 | 52.27 | | 620.90 | 450.56 | 657.85 | | 2 | 595.98 | 648.25 |
| 9 | 7 | 649.50 | 411.64 | | 620.90 | 648.25 | 657.85 | | 1 | 649.50 | 1061.14 |
| 10 | 8 | 770.57 | 75.18 | | 1061.14 | 648.25 | 657.85 | | 2 | 770.57 | 845.75 |
| 11 | 9 | 772.25 | 22.38 | | 1061.14 | 845.75 | 657.85 | | 3 | 772.25 | 794.63 |
| 12 | 10 | 804.06 | 272.58 | | 1061.14 | 845.75 | 794.63 | | 3 | 804.06 | 1076.65 |

Server 2 becomes available at time 450.56, earlier than the other two servers, so we determine that Customer 6 will go to Server 2 (and that is shown in cell `I8`).

However, Customer 6's service does not start at time 450.56–because, at that time, Customer 6 hasn't arrived yet! Cell `J8` is later of cells `B8` and `F8`, i.e., the transaction begins when the customer and server are *both* ready.

Adding the transaction duration (from cell `C8`) gives the time shown in `K8`, when the transaction finishes and Customer 6 leaves the bank. Lastly, that time, 648.25, becomes the new next-available time for Server 2 and is copied down to cell `F9`.

The tabular simulation fully describes the comings and goings of customers through the system. We could then use this simulation to answer various questions about the system. For instance, how would we calculate the average time that each customer had to wait in the queue? Also, adding another server to the system would reduce the average waiting time, but by how much? Feel free to explore the answers to these questions, but they are not part of this assignment.

All we need at this stage is some arrival and service times. To get you started, you can find `bank.csv` on iLearn, which was generated by using the following code.

```r
# to generate 100 customer's arrival time and their service times
set.seed(2048)
# customers arrived according to a Poisson Process
# = inter arrival time has an exponential distribution
# Set arrival rate to one per 60 seconds = mean inter-arrival time is 60s
arrival_time <- cumsum(rexp(100, 1/60))
# service time is exponential distributed with mean 150s + 20s greeting
service_time <- rexp(length(arrival_time), 1/150) + 20
bank <- data.frame(arrival_time, service_time)
```

For this question, you have to

- Create an R package with all the appropriate documentation.
- Add the bank data set to your package.
- Implement `testthat` with **at least 2 expectations**.

- Used `Git` & `Github` appropriately with **at least 2 commits** together with an appropriate `readme` file.
- Used `Github Pages` for your package and `Github Action` for Continuous Integration (CI).
- Currently we used `data.frame()` on line 34 in `Multiserver()`. Change that to `tibble()` and update the package accordingly.

A marking rubric/checklist for this question can be found below. Please add a section on your `Rmd` file to indicate which tasks were completed/attempted.

**Notice that you can't activate "Pages" while your repo is private. You will receive the full marks for that part as long as those website related files are included in your repo.**

**Marking Check List for Q2**

- Had the correct R package structure; files were put in appropriate locations; the package can be installed [2 marks]
- Had all the appropriate documentation for the package [4 marks]
- Added the provided data set with proper documentation to the package [3 marks]
- Implemented `testthat` correctly [4 marks]
- Had used `Git` & `Github` correctly [2 marks]
- Created a basic webpage for the package and deployed it to `Github` [2 marks]
- Implemented `tibble` [2 marks]
- Implemented `Github Action` for CRAN checks [1 mark]

**Exercise 3 [60 points]** In this question, we are going to compare two methods used in the analysis of 2-by-2 contingency tables. One is the `chi-squared test`, and the other is the `Fisher's exact test`.

We will use this example based on the one from the `Fisher's exact test` Wiki page. A sample of teenagers might be divided into male and female on the one hand and those that are and are not currently studying for a statistics exam on the other. For simplicity, we will be conducting a two-sided test, i.e. testing whether the proportion of studying individuals is the same for both genders or not. The observed frequencies are

| | Men | Women |
|---|---|---|
| Studying | 3 | 27 |
| Not-studying | 33 | 9 |

As output, you need to include your report/write up as part of your `Rmd` file with the appropriate sections/dividers.

You can solve this exercise at different levels.Please state in your report which level you are aiming for.

- Pass [24 points]: Produce an `Rmd` file that will generate a reproducible **pdf** report describing/summarising the `chi-squared test` and its application. You can use the example above as an illustration. You are welcome to base this on the notes you had from your 1000 level Statistics units and the Wikipedia page about the $\chi^2$ test, but please do not copy the content from the Wikipedia page directly. Make sure you use the function that matches the method described in your report.

  You should also include multiple citations, including at least one journal article, at least one book and also at least one package (if you used a package, you can cite it even they are not $\chi^2$ test related).

  To achieve full mark for this part, we expect

  - an `Rmd` file that formatted correctly with no compiling errors
  - all R code will be run in the background and not be visible on your final report;
  - correct usage of the section/ordered list/unordered list environment;

  - all mathematical expressions & tables will be typeset appropriately in your `Rmd` file;
  - at least 3 correctly formatted citations, including at least one book, one journal & one package;
  - an appropriate summary of the statistical concepts underpinning a $\chi^2$ test.

- Credit [10 points]: As at the pass level, but more tasks are handled by `R`

  - All calculations/results will be handled/populated by `R`;
  - All citations & in-line references are generated from a BibTex file.

- Distinction [10 points:] Added an additional (sub-)section based on the `Fisher's exact test`. Once again, you have to describe/summarise the method and use the provided dataset to illustrate. You can base this on its Wikipedia page https://en.wikipedia.org/wiki/Fisher%27s_exact_test but please do not copy the content from the Wikipedia page directly.

- High Distinction [16 points] As at the distinction level, but add another (sub-)section and investigate this statement from the Fisher's exact test's Wikipedia page: "*Despite the fact that Fisher's test gives exact p-values, some authors have argued that it is conservative, i.e. that its actual rejection rate is below the nominal significance level.*"

  Your task is to investigate the above statement by conducting a simulation study. You have to generate 10,000 2-by-2 contingency tables in 4 different settings (so 40,000 in total). Using the terminology as the example we had above, and the 4 different settings are:

  - Setting 1 (Small sample size, small $p$): There will be a fixed 20 Men and 20 Women at each iteration and then a 20% success rate of those to be randomly allocated to the `Studying` group.
  - Setting 2 (Small sample size, normal $p$): There will be a fixed 20 Men and 20 Women at each iteration and then a 50% success rate of those to be randomly allocated to the `Studying` group.
  - Setting 3 (Large sample size, small $p$): There will be a fixed 100 Men and 100 Women at each iteration and then a 20% success rate of those to be randomly allocated to the `Studying` group.
  - Setting 4 (Large sample size, normal $p$): There will be a fixed 100 Men and 100 Women at each iteration and then a 50% success rate of those to be randomly allocated to the `Studying` group.

  For each contingency table, run both the $\chi^2$ test and Fisher's exact test and collect their $P$-values. You can then compare the rejection proportion based on three significance levels: 10%, 5% and 1%. Summarise your findings.

  **Important:**

  - In Setting 1, you may end up with no student in the "Studying" group by chance. Your code will have to check and replace such a sample if it happens.
  - In Settings 1 & 2, you can ignore any warning messages when the assumption for the $\chi^2$ test is not satisfied.