Assignment 2 – CSC2062

Worth 25% of the module assessment. Assignment is marked out of 100 marks.

Deadline: 11pm Friday, 18th March 2022.

This version: 2022-01-22.

Changelog:

2022-01-22: corrected typo: it is 140 images in total.

Introduction

In this assignment, you will:

(a) Create a dataset of handwritten letters and symbols (which you will use for your analyses and experiments in the rest of Assignment 2, and in Assignment 3).

(b) Perform feature engineering, i.e. calculate features (variables) from the handwritten letters and symbols which may be useful for identifying the handwritten symbols automatically.

(c) Perform statistical analysis of the datasets, using methods of statistical inference.

(d) Implement and evaluate introductory machine learning models that perform classification on the dataset.

When you use a procedure that has an element of randomness, please use the seed value 42 (your code should give the same results each time it runs). This assignment must be completed in **R**. You may not use Microsoft Excel to complete any part of this assignment.

Please read carefully the information about the assessment criteria and marking process at the end of this document.

Section 1 (10 marks): Creating a dataset

This section asks you to build a dataset of images composed of written letters and face doodles. Each image is represented by a black & white matrix with size 18 rows by 18 columns. In the matrix, the number "1" represents black pixels and "0" represents white pixels. As such, one image can be stored in a plaintext ".csv" file containing the matrix (and no headers), as in these examples:

```
0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0
0,0,0,0,0,0,1,1,1,0,0,0,0,0,0,0
0,0,0,1,1,1,1,0,1,1,1,0,0,0,0,0
0,0,1,1,1,1,0,0,0,0,1,1,0,0,0,0
0,0,1,1,0,0,0,0,0,0,0,1,0,0,0,0
0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0
1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0
1,1,0,0,0,1,1,1,1,1,0,0,0,0,0,0
0,1,1,0,1,1,1,0,1,1,1,0,0,0,0,0
0,0,1,1,0,0,0,0,0,0,1,1,1,0,0,0
0,0,1,1,0,0,0,0,0,0,0,1,1,0,0,0
0,0,0,1,1,0,0,0,0,0,0,1,1,0,0,0
0,0,0,0,1,1,1,1,1,1,1,1,0,0,0,0
0,0,0,0,0,0,1,1,1,1,1,0,0,0,0,0
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
```

Figure 1: Matrix representation of the handwritten numeral "6".

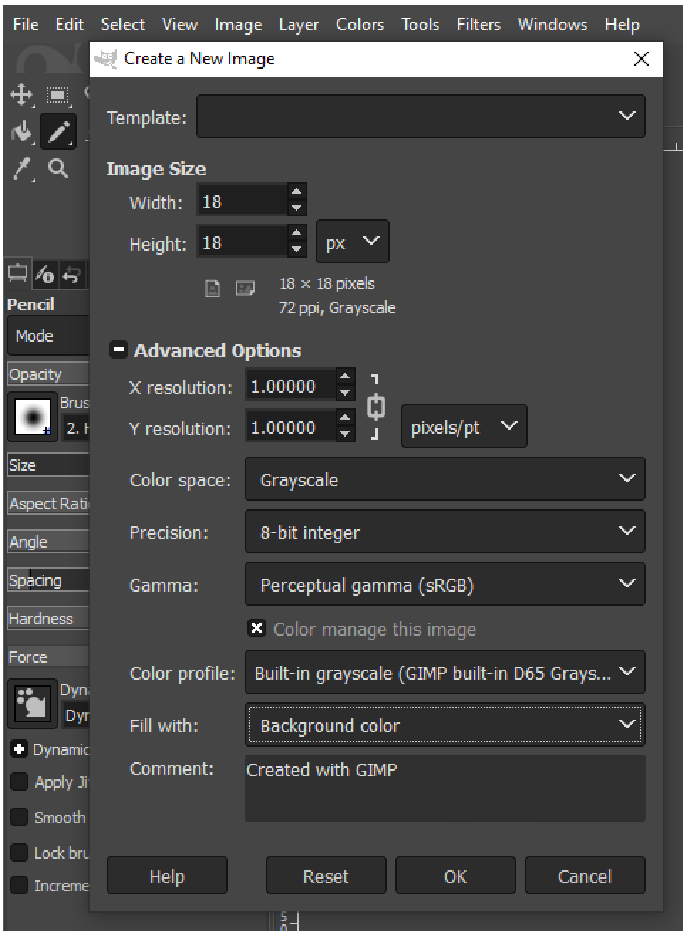
```
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
0,0,0,1,1,0,0,0,0,0,0,1,1,0,0,0
0,0,1,1,1,1,0,0,0,0,1,1,1,0,0,0
0,0,1,1,1,1,0,0,0,0,1,1,1,0,0,0
0,0,0,1,1,0,0,0,0,0,1,1,0,0,0,0
0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0
1,1,0,0,0,0,0,1,1,0,0,0,0,0,0,0
0,1,1,0,0,0,0,1,1,0,0,0,0,1,0,0
0,0,1,1,0,0,0,0,0,0,0,0,1,1,0,0
0,0,1,1,0,0,0,0,0,0,0,0,1,1,0,0
0,0,0,1,1,0,0,0,0,0,0,1,1,0,0,0
0,0,0,0,1,1,1,1,1,1,1,1,0,0,0,0
0,0,0,0,0,0,1,1,1,1,1,0,0,0,0,0
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
```

Figure 2: Matrix representation of a handwritten "happy face".

The goal here is to create a dataset containing eight handwritten images of each of the 10 letters in the set {**a,b,c,d,e,f,g,h,i,j**}, as well as 20 images of a happy face **:-)** , 20 images of sad face **:-(** , and 20 images of exclamation mark **!**. The faces should consist of two eyes, a nose and mouth, as shown (although they should be the right way up, as in figure 2, not on their side as depicted in the emoticon). Each image should be obtained by writing/drawing the hand-written symbol yourself (as described below). The quality of the drawing is not essential, as long as the symbol in the image can easily be identified by a human. The images will vary from sample to sample, due to your handwriting; however, each character should fit reasonably well in the 18 x 18 box (i.e. do not draw a tiny character in one corner of the 18 x 18 box; this will make your life easier when it comes to doing analyses!). In total, you are creating 140 images.

Each image is a black & white square grid with size 18 rows by 18 columns. In the matrix of this grid, the number “1” represents black pixels and “0” represents white pixels. As such, one image can be stored in a comma-delimited plaintext “.csv” file containing the matrix (and no header row); see Figs. 1 & 2 above.

You may use whatever means you prefer to obtain the 140 .csv files, provided they are handwritten and are in the comma-delimited .csv format specified above. However, it is strongly recommended that you use the software GIMP (http://www.gimp.org). GIMP is available for free for all PC OSs, and is also installed on the lab machines and the EEECS virtual machines. Using GIMP, you can create a new image with 18 by 18 points (px), advanced options 1 pixel/pt, color space grayscale, fill with background colour. This will give you a small white square, which you can magnify to e.g. 2000% in order to make it easier to draw on (see Fig 3).



Create a New Image

Template:

Image Size

Width: 18

Height: 18 px

18 × 18 pixels
72 ppi, Grayscale

Advanced Options

X resolution: 1.00000

Y resolution: 1.00000 pixels/pt

Color space: Grayscale

Precision: 8-bit integer

Gamma: Perceptual gamma (sRGB)

☒ Color manage this image

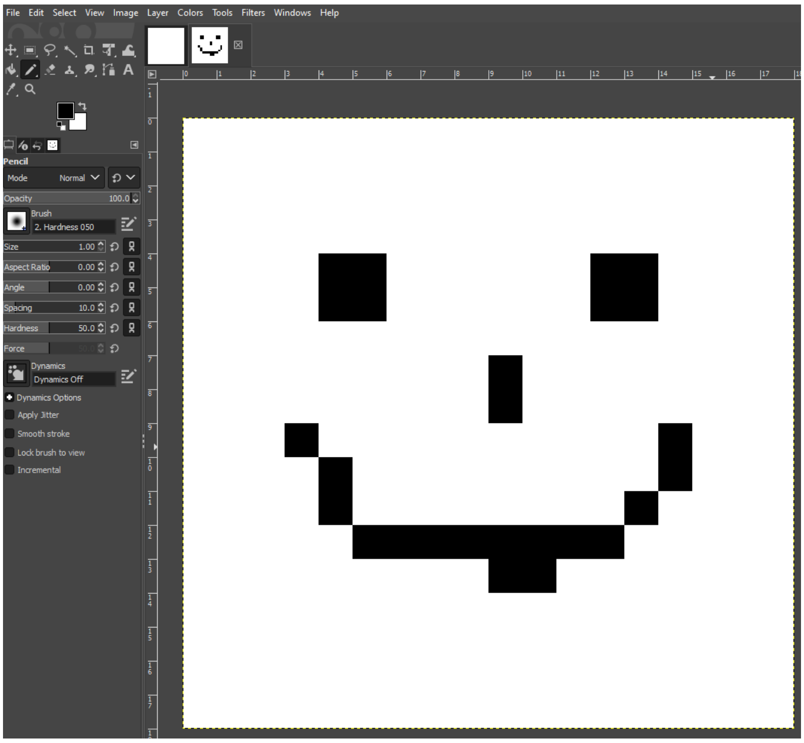
Color profile: Built-in grayscale (GIMP built-in D65 Grays...

Fill with: Background color

Comment: Created with GIMP

Figure 3: Creating the blank canvas of size 18 x 18 pixels in the GIMP interface.

To draw on the image, you can select the pencil tool and adjust the brush size to 1 pixel (see Fig.4).


File Edit Select View Image Layer Colors Tools Filters Windows Help

Pencil
Mode Normal
Opacity 100.0
Brush 2. Hardness 050
Size 1.00
Aspect Ratio 0.00
Angle 0.00
Spacing 10.0
Hardness 50.0
Force
Dynamics Dynamics Off
Dynamics Options
□ Apply Jitter
□ Smooth stroke
□ Lock brush to view
□ Incremental

Figure 4. A smiley face drawn on the 18x18 grid with pencil size of 1 pixel.

The standard file formats of GIMP are useful to save the images, but we need a more easily readable format. One good option is to export as PGM, type ASCII. This PGM file can be opened in GIMP, but it is also simply a text file that can be opened in a text editor (or read as text file by R code). The PGM text file has a header consisting of the following four lines:

```
P2
# CREATOR: ...
18 18
255
```

The third and fourth lines of the header above specify the pixel array size and the maximum allowed pixel value, respectively. (The images are greyscale, with 0 representing fully black and 255 representing fully white).[1]

The remaining lines of the file specify the pixel values, with one value on each line; the total number of pixel values should correspond to the specified array size (i.e. 18*18=324).

For our purposes, a number < 128 will represent a black pixel, while a number >= 128 represents a white one. Such a format can be easily converted into a matrix containing ones and zeros, as presented in Figure 1 & 2 above (you must write some R code to do this; reading in the PGM file and writing the .csv file). You shall save each image matrix as a csv file following the specification above, and using the filename STUDENTNR_LABEL_INDEX.csv, where STUDENTNR is your student number (e.g. 402345), INDEX is a two numeral code from ‘01’ to ‘20’, indexing the set of images you must create for each symbol, and LABEL is the name of the symbol in the image, as specified in Fig 5 below.

[1] For further information about this image format, see https://en.wikipedia.org/wiki/Netpbm_format

Symbol	Label
a	a
b	b
c	c
d	d
e	e
f	f
g	g
h	h
i	i
j	j

Symbol	Label
Sad face	sad
Smiley face	smiley
!	xclaim

Figure 5: Labels to use for the created images

For example, if your student number is 4012345, then 40**12345_f_08.csv** would be the eighth image you created for the *f* symbol. (As well as creating the square csv files, you may also want to keep the PGM files, in case you need to inspect the data later on).

As part of your submission, upload the csv files that you create in a directory called “images”. Any code you wrote to create the csv files should be presented in the assignment R markdown notebook (see submission instructions at the end of this document).

It is very important to upload the images in the correct csv format as these files will be used to verify your calculations in the next section. The .csv files should be comma delimited, not tab-delimited or anything else. File Encoding should be UTF8 (not UTF8-BOM or anything else). You can check the encoding in Notepad++.

In your report notebook, very briefly (2-3 sentences) explain in your own words how you created the images and obtained the matrices from them.

Section 2 (30 marks): Feature Engineering

Using each 18x18 matrix obtained from an image as described above, you must create an array of characteristics that describe some features of the image. Each feature will be a number (i.e. each feature is a numeric variable). There are 16 features in total. In the feature definitions that follow, a pixel has 8 neighbours, which will be referred to as follows:

upper-left	upper	upper-right
left	[pixel]	right
lower-left	lower	lower-right

Figure 6: The eight neighbours of a pixel.

Features to be calculated (corresponding to columns of your features output file):

Feature Index	Short Feature Name	Feature Description
	label	The true name of the symbol in the image (i.e. one of the 13 symbol names given in Fig. 5). The label is not a true feature, and should not be used as a feature for statistical tests or during model training.
	index	The index of this image instance (a number from 01 to 08 for the letters and 01 to 20 for the non-letters; note the zero padding). The index is not a true feature, and should not be used as a feature for statistical tests or during model training.
1	**nr_pix**	The number of black pixels in the image.
2	**rows_with_1**	Number of rows with exactly 1 black pixel
3	**cols_with_1**	Number of columns with exactly 1 black pixel
4	**rows_with_3p**	Number of rows with 3 or more black pixels
5	**cols_with_3p**	Number of columns with 3 or more black pixels
6	**aspect_ratio**	The vertical distance, in pixels, between the topmost and bottommost black pixels in the image (measuring from the pixel centre) is the height. The horizontal distance, in pixels, between the leftmost and rightmost black pixels in the image (measuring from the pixel centre) is the width. This feature is width/height.
7	**neigh_1**	Number of black pixels with only 1 black pixel neighbour
8	**no_neigh_above**	Number of black pixels with no black pixel neighbours in the "upper left", "upper" or "upper right" positions
9	**no_neigh_below**	Number of black pixels with no black pixel neighbours in the "lower left" "lower" and "lower right" positions
10	**no_neigh_left**	Number of black pixels with no black pixel neighbours in the "upper left", "left" or "lower left" positions
11	**no_neigh_right**	Number of black pixels with no black pixel neighbours in the "upper right" "right" and "lower right" positions
12	**no_neigh_horiz**	Number of black pixels with no black pixel neighbours in the "left" and "right" positions
13	**no_neigh_vert**	Number of black pixels with no black pixel neighbours in the "upper" and "lower" positions
14	**connected_areas**	Two black pixels A and B are connected if they are neighbours of each other, or if a black pixel neighbour of A is connected to B (this definition is actually symmetric); a connected region is a maximal set of black pixels which are connected to each other; this feature has the number of connected regions in the image. *Hint: research the `raster` and `clump` functions in the raster R library.*
15	**eyes**	An "eye" is a region of whitespace that is completely surrounded by lines of the character. For example, capital "A" contains one eye, capital "B" contains two eyes, and capital "C" contains no eyes. A region of whitespace is an eye if there is a ring of black pixels surrounding it which are all connected (i.e. they form a chain of neighbours). This feature is the number of eyes in the image. *Hint: this is similar to the previous feature, except now you need the number of separate whitespace regions. Two whitespace pixels are only connected if they are orthogonally connected. Investigate the `directions` parameter of the `clump` function in the raster library.*
16	**custom**	Design any other feature you like, which you think may be useful for distinguishing between symbols. Justify your choice, and try to avoid capturing information already captured by other features.

Your task in this section is to write code to calculate each of the features above. In calculating pixel neighbours, you can assume that the images are padded on each side with white pixels.

Save your calculated features in a file called **`STUDENTNR_features.csv`**, where STUDENTNR is your student number. This file will consist of 161 rows. The first row gives the column names (i.e. the strings in the **Feature Short Name** column in the table above, comma-delimited). The remaining 140 rows list the comma-separated feature values for each of your 140 images. The first entry in the row will be the LABEL word, the second will be the image INDEX, and the remaining 18 entries will be the calculated features.

For example, the features for your second "!" image may be as follows (values are purely hypothetical):

```
xclaim,2,13,13,0,0,1,0.0769,2,2,3,13,13,13,1,2,0,42
```

The rows that correspond to the instances of a particular symbol should be grouped together in the features file, and the order of those rows should correspond to the INDEX used in the image filenames. In other words, the 140 data rows of **`STUDENTNR_features.csv`** should be sorted first by the label (alphabetical order) and secondly by the index.

If you cannot calculate a particular feature, you may use a random number for the feature values instead. (You will lose marks for not calculating the feature, but you can use the random values in the analyses that follow in the subsequent section. You should report that you have done this in the assignment report text).

In your report, very briefly describe and explain the code you have written to calculate the features above. If you ran into difficulties, you should still explain your thought processes and attempts to calculate the features. In the case of the custom feature, you should explain your rationale for choosing the feature you did, as well as how it is calculated (i.e. you should give a justification for why you think this feature should be useful).

You should put the file **`STUDETNR_features.csv`** in the main assignment directory (see "Deliverables" section below). Put code for this section in the R Markdown Notebook, and, optionally, in a file called **section2_code.r**. The working directory for the code should be the main assignment directory (i.e. the directory containing the notebook, the feature csv file, and the code files). Your code should use relative paths; i.e. it should read the image matrixes from "./images".

Section 3: Statistical analyses of feature data (35 marks)

In this section, you will perform statistical analyses of the feature data, in order to explore which features are important for distinguishing between different kinds of handwritten symbols.

You shall use descriptive statistics (mean, variance, etc.), hypothesis testing, and suitable visualisation to perform your analysis of the data. You are encouraged to provide tables, figures, and/or graphs in the report to support your discussions and findings.

It is your responsibility to define the appropriate assumptions to run the tests, and to choose an appropriate test according to the data characteristics and the question that you are studying. In general, you will not be told what tests or R functions to use; it is up to you to explain and justify your choices. You are not necessarily restricted to the hypothesis tests that were discussed in the lectures. You may assume a significance level of 0.05 for the analyses when running hypothesis testing.

In particular, in the report you should address each of the following subtasks, using appropriate statistical tests, tables, graphs, etc.

1. Construct suitable histograms for the first six features, over the full set of 168 items. Briefly describe the shape of the distributions and comment on any interesting patterns across the datasets.
2. Present useful summary statistics (e.g. mean, median, and standard deviation) about all the features, for (a) the full set of letters, and (b) the full set of non-letters. Briefly discuss the summary statistics, and whether they already suggest which features may be useful for discriminating letters and digits. For three features you feel may be interesting for discrimination between groups, consider suitable visualisations (e.g. histogram of feature values for the groups[2])
3. Perform statistical analyses to test whether there is a significant difference between letters and non-letters. According to your analysis, which features are most useful to discriminate whether an image is a letter or a non-letter? Consider suitable visualisations to illustrate the degree of discrimination between these features. Briefly interpret your findings (i.e. why might these features be useful) and the validity of any assumptions.
4. Investigate the degree of linear association between features. Describe and conduct a suitable statistical test to measure the degree of linear association between the features. What features have the highest degree of linear association? Consider suitable visualisations. For features that have high association, comment on why this may be the case, given the feature definitions.

For the questions above, you shall explain your reasoning, assumptions and steps of the procedure (including the statistical analysis) when preparing the report. Justify your reasoning. If you are generating p-values for analysing the statistical significance of some features, make sure to explain how they were obtained. It is your task to decide and justify what the most appropriate inference to be performed in each case is, and to discuss the results you obtained.

Section 4 Regression and Machine Learning (25 marks)

1. Suppose that instead of calculating the **aspect_ratio** feature in Section 2, you instead would like to predict the aspect ratio value from the other features you calculated. Fit a multiple regression model to predict **aspect_ratio** as parsimoniously as you can from a subset of these variables (consider approaches to feature selection). Give the results table of the regression model.
2. Using the most useful discriminatory feature (i.e. on the basis of results and visualisations from section 3.3 above), fit a logistic regression model and report the results. Visualise the fitted logistic regression model.
3. For the three features **nr_pixels**, **aspect_ratio** and **neigh_1**, create three new categorical features **split1, split2 and split3.** These features will be based on a median split of the data: when the feature value is above the median the value will be 1 and when it is below the median value the value will be 0. Create a table like the one below that gives proportion of "1"s for each of the three classes "letter", "face", "exclamation mark":

	Split1	Split2	Split3
Letters			
Faces			
Exclamation Marks			

[2] A nice example: https://stackoverflow.com/questions/36049729/r-ggplot2-get-histogram-of-difference-between-two-groups

4. How will a naïve bayes classifier classify a new item that has above median values for the three features? I.e. use the data to estimate **P(class|(split1=1,split2=1,split3=1))**, for each of the three classes. Do the same for **P(class|(split1=0,split2=0,split3=0)).** Briefly comment on and interpret the results.

Assessment criteria and marking process

The most important criteria in marking is the completeness, accuracy, professionalism and clarity of your report (approximately 65% weighting, across the full submission). In your report, you should **clearly demonstrate that you understand the methods used** in each sub-task. Explain your chosen approach, your reasoning, and the assumptions and steps of the procedures used. You should **explain and interpret your results**, demonstrating understanding and independent thinking. What are your results telling you? Are the results what you would expect? If you ran into difficulties, explain what they were and the efforts you made to try to overcome them.

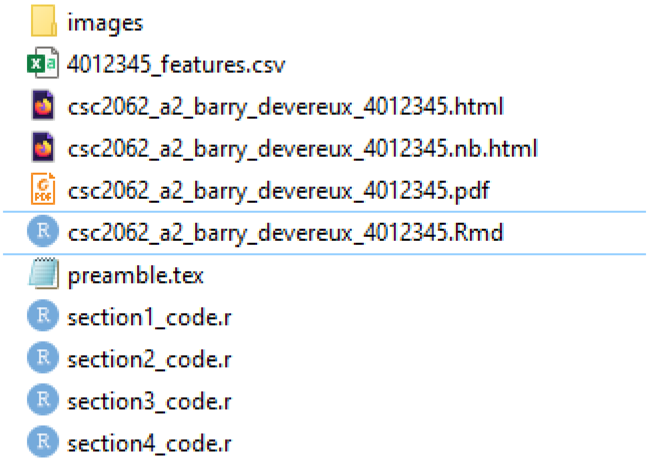
Code has a weighting in marking of approximately 30%. Your code should be clear and logically organised, and do what is required, but code efficiency and code sophistication is not important (this assignment does not require complex programming). However, you should use loops and variables (rather than hard-coded values) where appropriate. If you use freely licenced code, packages, or libraries (which is encouraged), these should be appropriately referenced (e.g. by citing a URL in a comment). For example, using StackOverflow code snippits is fine, provide you acknowledge the use and provide the URL to the code snippits in the comments, and follow the MIT licence. The code must be easy to use and the comments must include information about the required steps to replicate the results that you have obtained and are presenting in your report (transparency and replicability are essential in data analysis).

Attention to detail and following the assignment instructions accurately will also be considered in marking (approximately 5% weighting). Each sub-task has a precise specification. Make sure you carefully follow the instructions, and use the features specified for each task, the specified procedures (seed value, data file specifications and file names, directory structure and names, etc). Make sure you upload your deliverable files precisely in the specified formats.

It should be straightforward for the assessor to rerun your code to produce the same results as presented in your report. Ensure that the different subsections are clearly labelled in any source code files, and in the report. Each of these subtasks should be addressed in a separate subsection of your report, following the report template.

Deliverables

You must submit your assignment online, using the module webpage, by 11pm Friday, 18th March 2022. The online uploaded file must be a ZIP file called **a2_STUDENT_NAME_STUDENTNR.zip**, containing multiple files and directories. The contents and structure of the zip file are specified below:



images
4012345_features.csv
csc2062_a2_barry_devereux_4012345.html
csc2062_a2_barry_devereux_4012345.nb.html
csc2062_a2_barry_devereux_4012345.pdf
csc2062_a2_barry_devereux_4012345.Rmd
preamble.tex
section1_code.r
section2_code.r
section3_code.r
section4_code.r

Figure 7. The uploaded zip file structure.

Please use the provided zip file and the report template for preparing your report. The word limit for the report is 4000 words (excluding code, tables and figures; you can include as many tables and figures as you feel is appropriate).

A RAR file is not a ZIP file. A broken or corrupt ZIP file is not a ZIP file. **Do not include `.Rdata` files or other log /history files in your upload**; these may make your zip file very large.

It is your responsibility to ensure the assignment is uploaded and double-checked in good time before the deadline. Standard university penalties apply for late submission.

By submitting this assignment, you acknowledge that it is your own work and that you are aware of university regulations regarding academic offences, including (but not restricted to) plagiarism and collusion. Collusion/plagarism will be manually and algorithmically checked for.