# Data engineering project

Report on the MongoDB + OULAD dataset

**Put together by** **Kory Frankee** (fran0618)

**Class:** COMP 2031

**Word count: ~** 1,300of 3,000 words

**Page count:** 5 of 10

(Excludes references, figures, title page, and appendices)

## What I worked with

**R version** 4.3.3

**R Studio version** 2023.12.1

**Packages I used in one’s code and for what:**

* mongolite {for connecting to a Mongo DB}
* tidyverse
  + ggplot2 {for plotting}
  + tibble {for storing data}
  + dplyr {for data manipulation}
  + tidyr {for rectangling}
  + readr {for reading .csv files}
  + stringr {for working with strings}
  + forcats {for factoring data}
* R6 {for classes}
* sqldf {for SQL}. Note that dplyr *“aims to translate the most common R functions to their SQL equivalents”* and with it, you can even see the SQL query behind code written in dbplyr[[1]](#endnote-1). I, however, opted to write SQL in some cases even though I could have written it in dbplyr
* colourvalues {for converting from an RGB value to a HEX value}

## What I’ve done

#### I. Loading the libraries in

Wrote code to check if the packages I want to use are installed and if not, I install them. I then loaded them

#### II. Loading the data in

To help put the data into R, I developed a *“data loader”* class. **Within it, I wrote functions for:**

* Putting data retrieved from the account of a MongoDB user > from a cluster > from a MongoDB collection into a tibble
* Putting data from a .csv file into a tibble

#### III. Data observations

To help understand the data, I wrote a *“data checker”* class. Within it, I wrote functions for:

* Checking for duplicate primary key IDs
* Checking that the combination of multiple foreign key IDs was unique in each row
* Checking the data types of the columns
* Telling me basic facts about the dataset like the number of rows, columns, and duplicate, missing, and unique values

After loading in the data from the MongoDB dataset, I discovered that the scores were nested, which is a problem I’ll have to address

A screenshot of a graph

Description automatically generated

Of the 400,000 rows, there were 10,000 students and 501 classes with a unique ID, 3 assignment types (‘exam’, ‘homework’ and ‘quiz’), and 400,000 unique scores in the range of 0 to 100.

(Conclusion) The Mongo DB scores show that the mock data is unrealistic. (P1) Markers don’t tend to provide scores with upwards of 7 decimal places. (P2) Marks don’t tend to be all different to one another when the number of assignments marked is high, which they are. (P3) Marks tend to have a minimum of 0 as some students receive a 0 for not handing in their work. Here, the minimum score is ~ 0.0002. (P4) Marks tend to have a maximum of 100 as some students receive full marks for their work. Here, the maximum is ~ 99.99. (P5) Scores tend to have a normal distribution. I’ve provided a visualization of the distribution of scores for all the assignments to show how uniform it is **(Appendix 0)**

With that in mind, **I can still work with this data**

After looking at the data from the OULAD dataset, I had many problems with it.

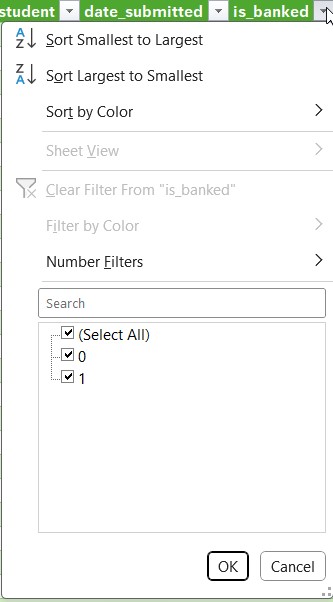
1. Missing student scores for *{student\_assessment.csv}*

A screenshot of a search box

Description automatically generatedA screenshot of a search box

Description automatically generated

1. Use of quantitative (‘0’/’1’ values) for if result is transferred from a previous presentation. I think nominal (‘no/’yes’ values) would be clearer and acceptable as no calculations are done on this *{student\_assessment.csv}*



A screenshot of a computer

Description automatically generated

1. Column names that aren’t clear. They require the reader to look up their meaning on the OULAD website[[2]](#endnote-2) {all .csv files}
2. Wrongly entered age values (e.g. ‘55<=’ should be ‘>55’) *{student\_info.csv}*

A screenshot of a search box

Description automatically generated

A screenshot of a search box

Description automatically generated

1. Missing IMD band values and a wrongly entered IMD band value ’10-20’ with no ‘%’ sign at the end of it) *{student\_info.csv}*

A screenshot of a screenshot of a computer

Description automatically generated

A screenshot of a screenshot of a computer

Description automatically generated

1. Missing when student unregistered values *{student\_registration.csv}*

A screenshot of a search box

Description automatically generated

A screenshot of a search box

Description automatically generated

1. Missing when student registered values *{student\_registration.csv}*

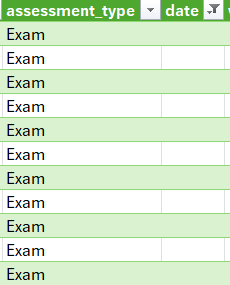
A screenshot of a computer

Description automatically generated

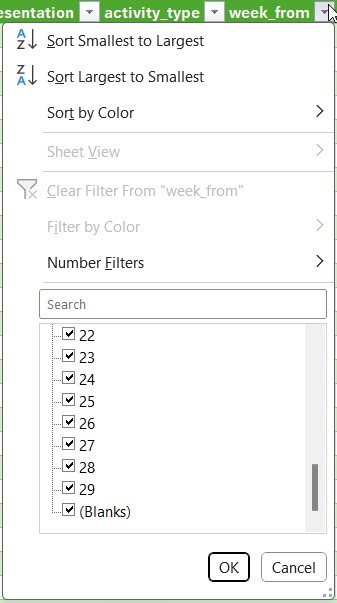
A screenshot of a computer

Description automatically generated

* Missing exam due dates (all missing values should be the replaced with the value for the last week of the course {*assessments.csv}*



1. Missing values for when students use VLE materials *{vle.csv}*



A screenshot of a search box

Description automatically generated

1. Missing values for when students stop using VLE materials *{vle.csv}*

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

1. The sum of all assessments for a module + presentation combination should add up to 100% and it doesn’t. **There are two reasons it doesn’t add up to 100%:**
   1. The assessments are weighted too low.

* As it is, a student could do well on every single assignment for a module + presentation combination and fail due to all their assessments being weighted very lowly, which is very counterintuitive

E.g. For the student with an ID of *‘465730’* doing module *‘BBB’* and presentation *‘2013J’*, I observed that this student had high scores for all their assessments but failed because the assessment weightings add up to ~ 47.91%, not 100%

A screenshot of a graph

Description automatically generated

* 1. Both the exams and all other assessments cannot be both be weighted at 100% in the case of a student doing an exam and other assessments for a module + presentation combination
* As it is, a student can obtain a weighted score ≥ 101 and ≤ 200. No university grading system is designed for grades being out of 200% as opposed to 100%.

#### IV. Data cleanup

I wanted to address the one problem I had with the MongoDB dataset and the 11 problems I had with the OULAD dataset. To help with this, I developed a *“data cleanup”* class. **Within it, I wrote functions for:**

* Unnesting a column
* Finding and replacing values
* Removing junk columns
* Putting values in a column into a vector of length 1 with comma-separated terms and quotation marks between each term. I can use such vectors in conjunction with a SQL IN
* Reordering columns (arrange())
* Setting the names of columns

I also wrote a function to work out the most common IMD value for each of the regions with missing values. I then filled in the missing IMD values for a particular region with the most common IMD value for that region.

How I made it so assessment weightings for a module + presentation combo added up to 100%

* Case 1) Some of the assessment weightings for the module + presentation combo
* Case 2) All of the assessment weightings for the module + presentation combo

As a result of this change, students that do well for all their assessments won’t fail unless they withdrew and obtained a WNF **(Appendix 13)**

#### V. Data additions

*Everything I do in this section is based on what Australian universities do*[[3]](#endnote-3) [[4]](#endnote-4) [[5]](#endnote-5) [[6]](#endnote-6) [[7]](#endnote-7)

I added a grades column to the MongoDB dataset, which I filled in with tertiary grades (‘F’, ‘P’, ‘Cr’, ‘D’, and ‘HD’) based on what score students received for their assessments

For the OULAD dataset, I addressed the issue with the assessment weightings for the **problematic 2nd case**

**Cases 1) and 2):**

1. Module has assessments but no exams.

* For this case, assessment weights should add up to 100%

1. Module has assessments, including exams.

* For this case, I added a second assessment weight column that scales all the assessment values to a new weighting. I set exams to be worth 40% (can change this value depending on the grading decisions of the university) and other assessments to be worth 100% - exam % so that they both add up to 100%

For both cases, I needed to add columns for the weighted score for each module, which will be equivalent to a grade (e.g. > 85% is a ‘HD’/’7’)

The weighted score (and grade) I decide to give a student is based on if their module had exams or not and if the student withdrew or not

**(Appendix 05)**

From here, I decided to calculate students’ cumulative GPA

#### VI. Data subsets and supersets

To help obtain a subset of the data, I wrote a “data subsetter” class with functions for:

* Getting a few rows of interest (filter())
* Getting a few columns of interest (select())

To help join data together, I wrote a “data supersetter” class with functions for:

* Merging tables

I joined the OULAD tables that I was able to join (without R Studio freezing) into one big table since I found that easier to work with

**(Appendix 04)**

Using a combination of subsetting and supersetting was helpful when, for example, I was working with the *“student\_vle.csv”* file. It was so large that R Studio and one’s laptop couldn’t even view the table for it. I had to split it up into 6 parts, perform the operations I needed on it, and then put the pieces of it back together again

#### VII. Dataset querying

Sometimes I needed the dataset to be in a particular form and to help with this, I wrote a *“dataset querier”* class with functions for:

* Querying the data using SQL

#### VIII. Data analysis

To help analyse the data, I wrote a “data analysis” class with functions for:

* Working out statistical information (mean, mean of each group, median, maximum, minimum, quartile value, quartile value for each group, standard deviation, z-score, & range of values). **Here’s an example of what I’m calling the “mean of each group” where each group is a different class:**

A screenshot of a table

Description automatically generated

**To do:**

* Work out what the easiest class is
* Work out what the most difficult class is
* Work out the number of students with a high enough cumulative GPA to proceed with postgraduate studies. Set the minimum to 4.5[[8]](#endnote-8)
* Speak more about trends I found

#### IX. Data visualization

To help visualize the data, I wrote a “data visualization” class with functions for:

* Graphing a scatter plot
* Graphing a complicated scatter plot that makes use of two datasets

**(Appendix 01)**

* Graphing a bar chart

**(Appendix 02, 07)**

* Graphing multiple bar charts side by side

(**Appendix 03**, **06)**

* Getting the inverted colours for a list of colours (used for distinguishing the colours of labels from colours of the plot objects)
* Graphing a violin plot

**(Appendix 08, 09, 10, 11)**

* Graphing a histogram plot with a density plot

**(Appendix 0)**

* Graphing a density plot

**(Appendix 12)**

**To do:**

* Utilize other graphs

#### X. Data modelling

**To do:**

* Simple linear model
* General linear model w/
  + Categorical predictors
  + Categorical and continuous predictors
  + Continuous predictors
* Evaluate the performance of the models
* Interpret the models

simple linear regression model: 1 outcome (y) and 1 explanatory (independent) variable (x)

multiple linear regression model: 1 outcome and 2 or more explanatory variables

Linear model:

y has to be continuous variable (weight, height, age...), range of values from negative to positive values

if y is a discrete variable or counts, then you'll use a different family of models (poisson model, negative binomial model)

if y is a binary categorical model, then you'll fit a logistic regression models

if the relationship is not linear then you may fit a non-linear model

the predictors can be of any type: numeric, categorical (character string or factor)

after fitting your model you have to evaluate the model performance (R squared, root mean square error) and also the model assumptions

# assumptions

# linearity or linear relationship

# independence of observations: theoretical analysis, how the data was collected, samples are collected independently of each other

# time series data, or time correlated data, spatially correlated data

# height, age

# normality of residuals: residuals should follow a normal distribution, use a diagnostic plot for checking normality

# homogeneity of variance (homoscedasticity): the variance of residuals is constant (no change or pattern in variance of residuals)

# diagnostic plots

# linearity, normality of residuals and homogeneity of variance

# first plot: linearity assumption, the red line should be fairly flat

# second plot QQ-plot: x axis: quantiles for normal distribution, y axis residual quantiles for your data

# data points should overlay the dashed line

# third plot: homogeneity of variance, the red line should be fairly flat

# there are formal statistical tests for checking normality and homogeneity of variance

# evaluate the model performance: root mean square error RMSE

# error (residual): difference between the actual value and the value the model predicts for a given observation

#### Appendices

##### MongoDB dataset visualizations

###### Appendix 00

|  |
| --- |
| Visualization of the distribution of student scores |
|  |

###### Appendix 01

|  |
| --- |
| Here, I’ve how the selected student has performed compared to other students in the same classes using an average of their scores for all their assessment items as the basis of comparison. |
| Showing it in action...  With student ID of 0    With student ID of 1 |

###### Appendix 02

|  |
| --- |
| Visualization of the least to most common grades students receive for their assessment items. This reveals that more students succeed with their assignments than fail at them by a small margin. This means that students are underperforming or that the markers are quite harsh. |
|  |

###### Appendix 03

|  |
| --- |
| Visualization of student scores across two different classes. **Useful insights could be gathered from this side by side analysis if:**   * We could compare the grades for a given class with previous years to see if there has been an improvement in student performance + teaching quality |
|  |

##### OULAD dataset schema

###### Appendix 04

|  |
| --- |
| Shows the original schema |
|  |

|  |
| --- |
| Shows the new schema |
|  |

##### OULAD dataset changes made to the data

###### Appendix 05

|  |
| --- |
| Showing how if student has (Orange) or has not (Blue) taken an exam affects how I calculate the grade  Showing how if student has not (Yellow) finished the course affects how I calculate the grade |
|  |

##### OULAD dataset visualizations

###### Appendix 06

|  |
| --- |
| Visualization of student grades for the module and presentation combinations |
| Before change to assessment weightings so that the assessments for a module and presentation pairing add up to 100%  A screenshot of a graph  Description automatically generated  After change to assessment weightings so that the assessments for a module and presentation pairing add up to 100% |

###### Appendix 07

|  |
| --- |
| Visualization of cumulative GPA |
| Before change to assessment weightings so that the assessments for a module and presentation pairing add up to 100%    After change to assessment weightings so that the assessments for a module and presentation pairing add up to 100% |

###### Appendix 08

|  |
| --- |
| Visualization of what affect poverty has on student scores |
|  |

###### Appendix 09

|  |
| --- |
| Visualization of what affect the age of a student has on student scores |
|  |

###### Appendix 10

|  |
| --- |
| Visualization of what affect being disabled has on student scores |
|  |

###### Appendix 11

|  |
| --- |
| Visualization of what affect gender has on student scores |
|  |

###### Appendix 12

|  |
| --- |
| Visualization of how fast students are at handing in their assignments |
| A graph of a person  Description automatically generated with medium confidence |

###### Appendix 13

Before change to assessment weightings so that the assessments for a module and presentation pairing add up to 100%

A screenshot of a group

Description automatically generated

A bar code with text

Description automatically generated

After change to assessment weightings so that the assessments for a module and presentation pairing add up to 100%

A screenshot of a computer

Description automatically generated

A screen shot of a bar code

Description automatically generated

#### References

1. https://dbplyr.tidyverse.org/articles/sql-translation.html [↑](#endnote-ref-1)
2. https://analyse.kmi.open.ac.uk/open\_dataset [↑](#endnote-ref-2)
3. Australian Education Info 2024, *“Grading system in Australia”,* <https://www.australiaeducation.info/Education-System/Grading-System.html>, viewed March 2024 [↑](#endnote-ref-3)
4. University of Sydney, *“How do I work out my final grade?”,* <https://usqassist.custhelp.com/app/answers/detail/a\_id/2615/~/how-do-i-work-out-my-final-grade%3F>, viewed March 2024 [↑](#endnote-ref-4)
5. https://student.ask.adelaide.edu.au/app/answers/detail/a\_id/1684/~/how-do-withdraw-no-fails-%28wnf%29-and-withdraw-fails-%28wf%29-affect-my-grade-point [↑](#endnote-ref-5)
6. https://usqassist.custhelp.com/app/answers/detail/a\_id/2615/~/how-do-i-work-out-my-final-grade%3F [↑](#endnote-ref-6)
7. https://policy.usq.edu.au/documents/20592PL [↑](#endnote-ref-7)
8. https://www.adelaide.edu.au/degree-finder/maiml\_maiml.html [↑](#endnote-ref-8)