Trybe Technical Test - OULAD

Lucas Cardoso de Menezes

07/08/2022

Purpose and context:

The learning team at the University wants to understand what the profile of people students is, what are the factors related to people's performance and what recommendations/initiatives you suggest for people students' performance to improve.

The job was to explore the data available and drive business decision making.

Database: *OULAD - Open University Learning Analytics Dataset*

Description:

A dataset containing demographic information about students, their courses taken, and the final outcomes of each course.

Roadmap

1. More about the dataset;
2. Database schema;
3. Preparing the environment for data analysis;
4. Understanding the profile of the students(I);
5. Performance factors (II);
6. Conclusion(III).

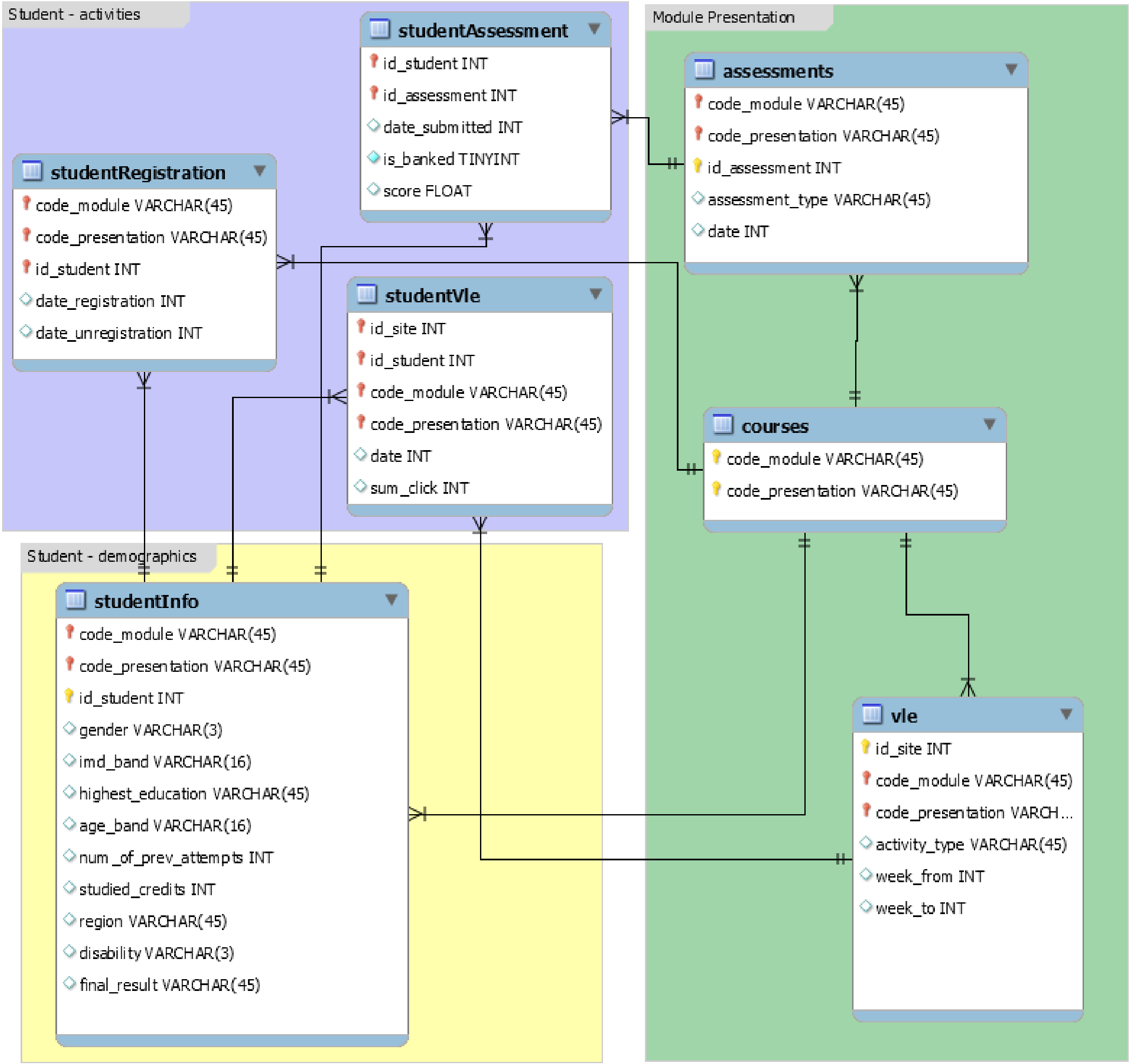
1. More about the dataset

The anonymized *Open University Learning Analytics Dataset (OULAD)*, contains data on courses, students, and their interactions with the *Virtual Learning Environment (VLE)* for seven selected courses. The courses - starting in February and October - are marked as "B" and "J" respectively. The dataset consists of tables connected using unique identifiers. All tables are stored in csv format.

Kuzilek J., Hlosta M., Zdrahal Z. Open University Learning Analytics dataset Sci.

2. database schema

Here we have a schema (https://analyse.kmi.open.ac.uk/open_dataset (https://analyse.kmi.open.ac.uk/open_dataset)) to illustrate the data structure of the dataset.



As you can see, there are many different types of data involved, but since we want to understand the profile of the people students and the performance factors we will use:

- Demographic data of the sample;
- A measure of the students' commitment to the course over the term;
- A measure of their performance over the period.

Going to the indicated website, we can see that this information is contained in the following tables:

- studentInfo;
- studentAssessment;
- assessments;
- studentVle;
- vle.

These tables will be our data sources for meeting the objectives.

Preparing the environment for data analysis

To perform this analysis we will use two R packages, *dplyr* and *plotly*. One to assist in manipulating the tables and the other in generating the graphs that will be presented.

The packages are available at:

- dplyr: https://cran.r-project.org/web/packages/dplyr/index.html (https://cran.r-project.org/web/packages/dplyr/index.html)
- plotly: https://cran.r-project.org/web/packages/plotly/index.html (https://cran.r-project.org/web/packages/plotly/index.html)

Or using the commands:

```
# install.packages("dplyr")
# install.packages("plotly")
```

We will use *libary* to call the packages after installation:

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.2.2
```

```
library(plotly)
```

```
## Warning: package 'plotly' was built under R version 4.2.2
```

```
## Warning: package 'ggplot2' was built under R version 4.2.2
```

To finalize the environment preparation we must inform the location where the input data is located:

```
setwd(dir = "C:/Users/luqui/Documents/Trybe")
```

4. understanding the profile of the students(I)

To be able to understand the profile of the students present in the database we will use the **studentInfo** table as it contains demographic information about the students along with their results. The file contains the following columns:

```
student_info <-
  read.csv(
    file = paste0(getwd(),"/Input/studentInfo.csv")
  )

colnames(student_info)
```

```
##  [1] "code_module"          "code_presentation"    "id_student"
##  [4] "gender"               "region"               "highest_education"
##  [7] "imd_band"             "age_band"             "num_of_prev_attempts"
## [10] "studied_credits"      "disability"           "final_result"
```

We will use only the columns that contain information linked to the student’s demographic profile, and we will also leave only the values without repetition, so that we have unique data for each student:

```
student_info_profile <-
  student_info[,c(2:6,8,11)] %>%
  distinct(id_student, .keep_all = TRUE)

colnames(student_info_profile)
```

```
## [1] "code_presentation" "id_student"        "gender"
## [4] "region"            "highest_education" "age_band"
## [7] "disability"
```

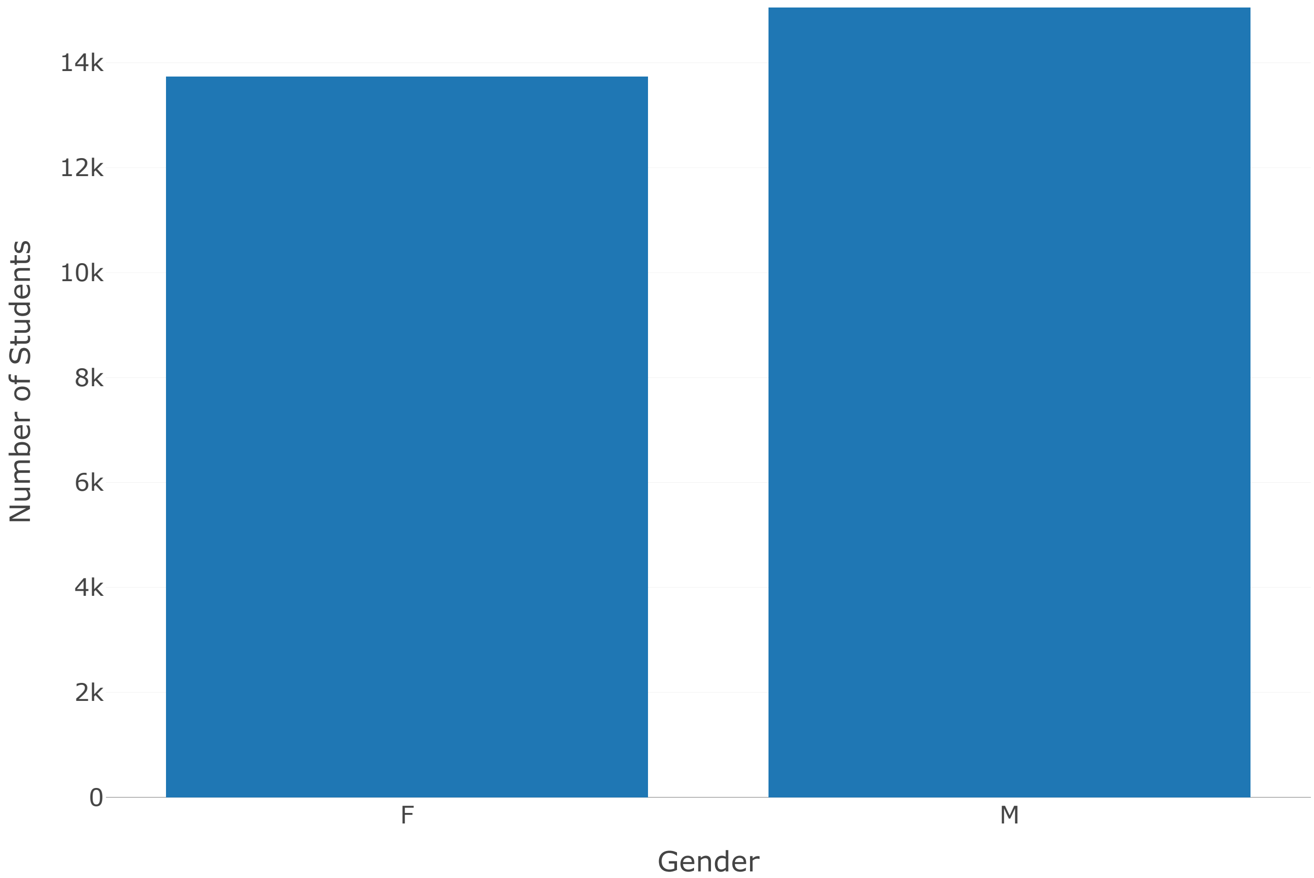
Generating the quantites:

```
#Types of presentation and the number of students
student_info_presentation <-
  student_info_profile %>%
  mutate(
    type_presentation = substr(code_presentation, nchar(code_presentation), nchar(code_presen
tation))
  ) %>%
  group_by(type_presentation) %>%
  count() %>%
  ungroup()

student_info_presentation
```

```
## # A tibble: 2 × 2
##   type_presentation     n
##   <chr>             <int>
## 1 B                 11190
## 2 J                 17595
```


```r
#Students per Gender
student_info_gender <-
  student_info_profile %>%
  group_by(gender) %>%
  count() %>%
  ungroup()

student_info_gender
```

```
## # A tibble: 2 × 2
##   gender     n
##   <chr>  <int>
## 1 F      13739
## 2 M      15046
```

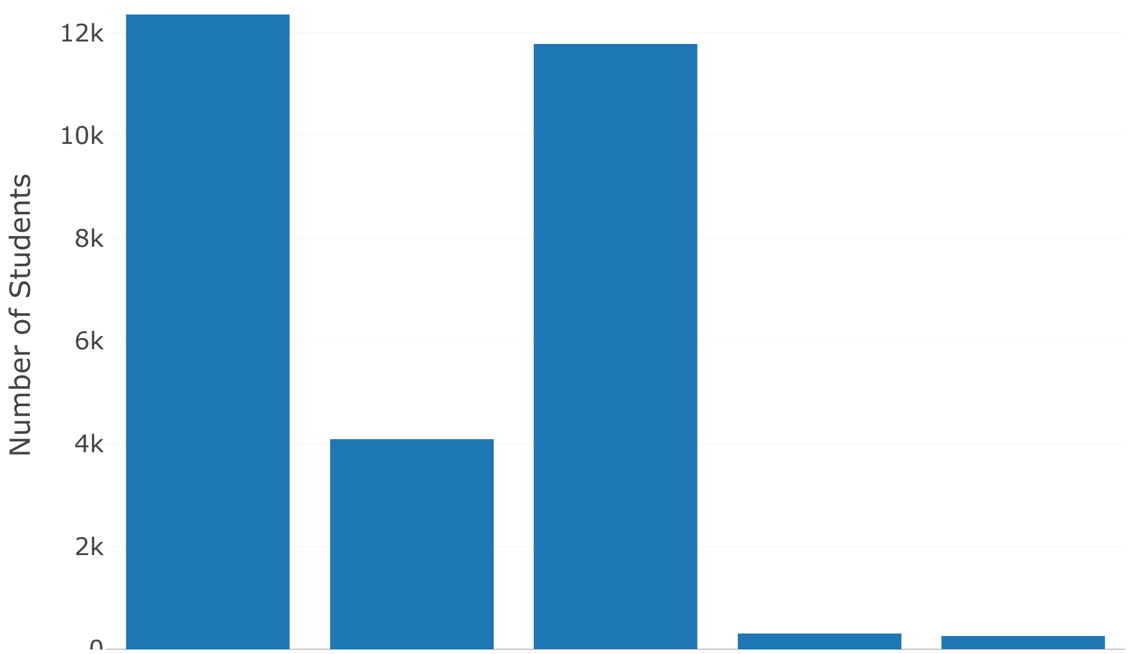


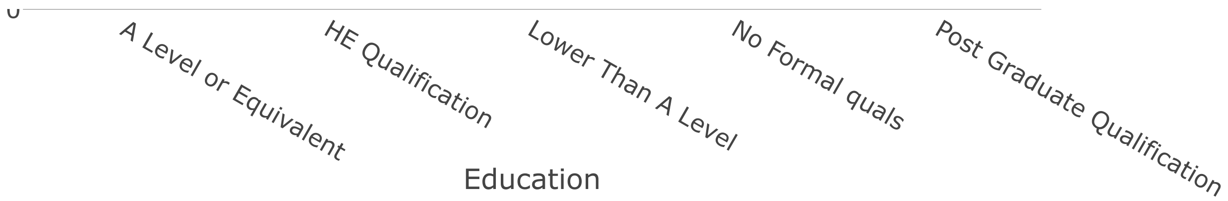

```r
#Students per region
student_info_region <-
  student_info_profile %>%
  group_by(region) %>%
  count() %>%
  ungroup()

student_info_region
```

```
## # A tibble: 13 × 2
##    region                   n
##    <chr>                <int>
##  1 East Anglian Region   3000
##  2 East Midlands Region  2095
##  3 Ireland               1072
##  4 London Region         2845
##  5 North Region          1588
##  6 North Western Region  2548
##  7 Scotland              2934
##  8 South East Region     1875
##  9 South Region          2737
## 10 South West Region     2154
## 11 Wales                 1876
## 12 West Midlands Region  2269
## 13 Yorkshire Region      1792
```

```
#Education
student_info_education <-
  student_info_profile %>%
  group_by(highest_education) %>%
  count() %>%
  ungroup()

student_info_education
```

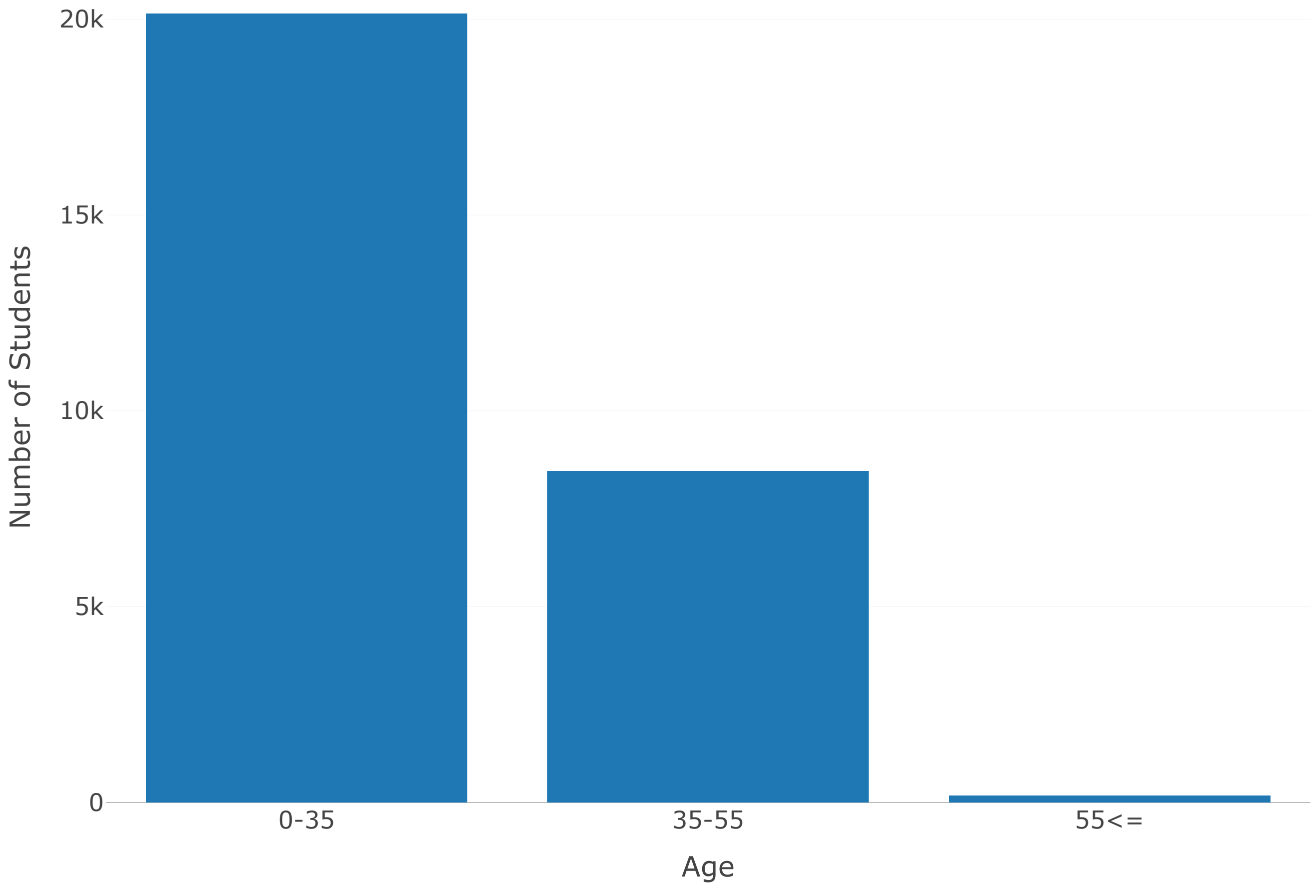
```
## # A tibble: 5 × 2
##   highest_education               n
##   <chr>                       <int>
## 1 A Level or Equivalent       12355
## 2 HE Qualification             4092
## 3 Lower Than A Level          11780
## 4 No Formal quals               306
## 5 Post Graduate Qualification   252
```





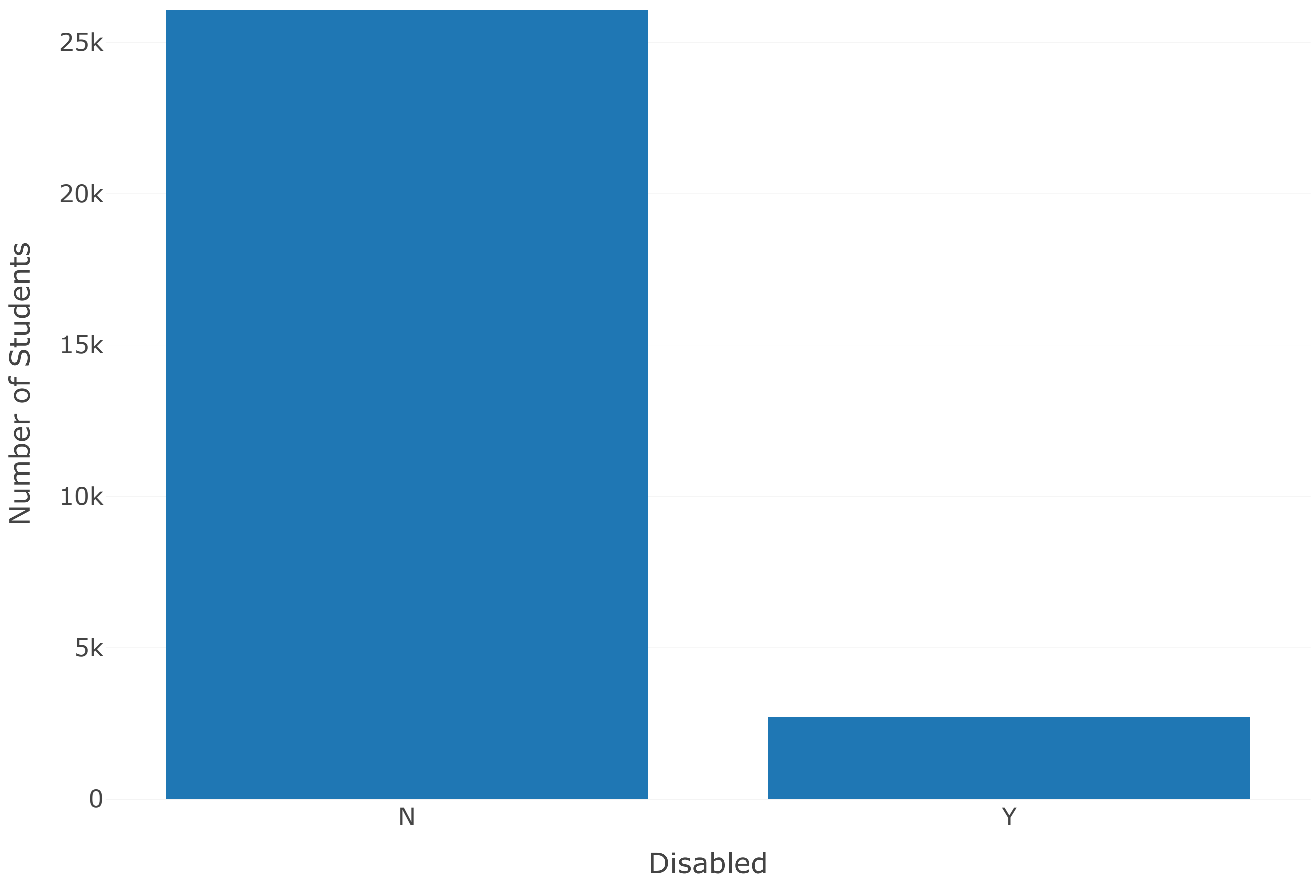

```
#Age
student_info_age <-
  student_info_profile %>%
  group_by(age_band) %>%
  count() %>%
  ungroup()

student_info_age
```

```
## # A tibble: 3 × 2
##   age_band     n
##   <chr>    <int>
## 1 0-35     20145
## 2 35-55     8462
## 3 55<=       178
```



```
#Disabled
student_info_disability <-
  student_info_profile %>%
  group_by(disability) %>%
  count() %>%
  ungroup()

student_info_disability
```

```
## # A tibble: 2 × 2
##   disability     n
##   <chr>      <int>
## 1 N          26068
## 2 Y           2717
```




```
#Demographic information compiled
student_demographic_data <-
  student_info_profile %>%
  group_by(gender, region, highest_education, age_band, disability) %>%
  count() %>%
  ungroup()

student_demographic_data
```

```
## # A tibble: 336 × 6
##    gender region               highest_education     age_band disability     n
##    <chr>  <chr>                <chr>                 <chr>    <chr>      <int>
##  1 F      East Anglian Region A Level or Equivalent 0-35     N            392
##  2 F      East Anglian Region A Level or Equivalent 0-35     Y             70
##  3 F      East Anglian Region A Level or Equivalent 35-55    N            169
##  4 F      East Anglian Region A Level or Equivalent 35-55    Y             29
##  5 F      East Anglian Region HE Qualification      0-35     N             65
##  6 F      East Anglian Region HE Qualification      35-55    N             66
##  7 F      East Anglian Region Lower Than A Level    0-35     N            399
##  8 F      East Anglian Region Lower Than A Level    0-35     Y             60
##  9 F      East Anglian Region Lower Than A Level    35-55    N            200
## 10 F      East Anglian Region Lower Than A Level    35-55    Y             41
## # … with 326 more rows
```

Observing the graphs, we conclude that we have a multicultural profile of students, coming from different regions, with different levels of knowledge, with the majority being up to 35 years old. However, we decided to correlate the data we consider most important (age, education, and region) to better understand the profile of the sample:

```
#Schooling by student's region
student_info_region_education <-
  student_info_profile %>%
  group_by(region, highest_education) %>%
  count() %>%
  ungroup()

student_info_region_education
```

```
## # A tibble: 60 × 3
##    region               highest_education               n
##    <chr>                <chr>                       <int>
##  1 East Anglian Region  A Level or Equivalent        1305
##  2 East Anglian Region  HE Qualification              313
##  3 East Anglian Region  Lower Than A Level           1324
##  4 East Anglian Region  No Formal quals                49
##  5 East Anglian Region  Post Graduate Qualification     9
##  6 East Midlands Region A Level or Equivalent         944
##  7 East Midlands Region HE Qualification              176
##  8 East Midlands Region Lower Than A Level            960
##  9 East Midlands Region No Formal quals                11
## 10 East Midlands Region Post Graduate Qualification     4
## # … with 50 more rows
```

When we correlate schooling by region, we see a still homogeneous picture.

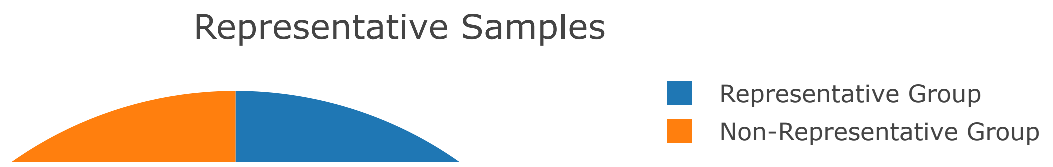
```
#Schooling by student age
student_info_age_education <-
  student_info_profile %>%
  group_by(age_band, highest_education) %>%
  count() %>%
  ungroup()

student_info_age_education
```

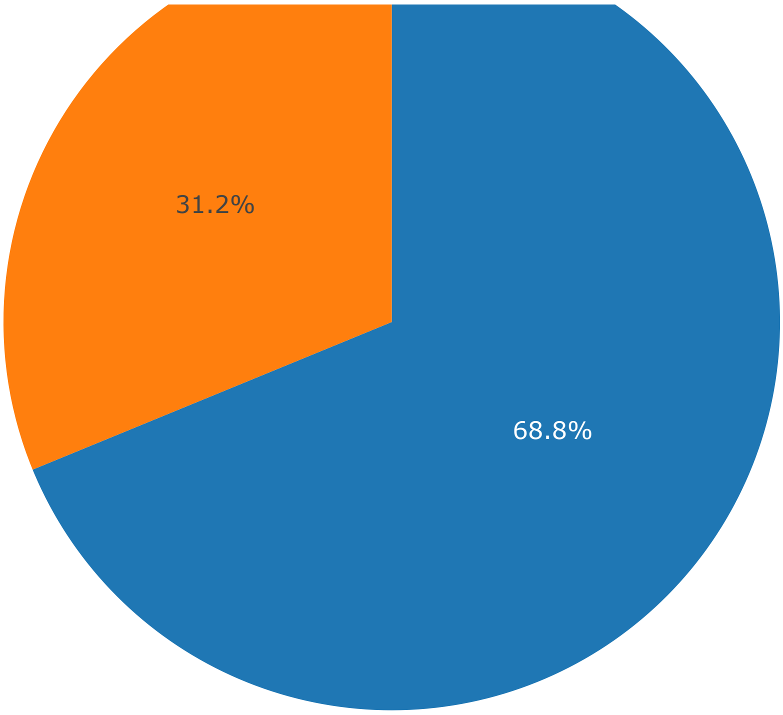
```
## # A tibble: 14 × 3
##    age_band highest_education               n
##    <chr>    <chr>                       <int>
##  1 0-35     A Level or Equivalent        9290
##  2 0-35     HE Qualification             2228
##  3 0-35     Lower Than A Level           8284
##  4 0-35     No Formal quals               258
##  5 0-35     Post Graduate Qualification    85
##  6 35-55    A Level or Equivalent        3032
##  7 35-55    HE Qualification             1756
##  8 35-55    Lower Than A Level           3469
##  9 35-55    No Formal quals                48
## 10 35-55    Post Graduate Qualification   157
## 11 55<=     A Level or Equivalent          33
## 12 55<=     HE Qualification              108
## 13 55<=     Lower Than A Level             27
## 14 55<=     Post Graduate Qualification    10
```

Now, correlating *Age x Education*, we can identify points where there is a larger sample size, so we decided to create a table of only the students that contain this profile:

```
#Schooling by student age
representative_student_group_info <-
  student_info_profile %>%
  subset(
    age_band == "0-35" &
      (
        highest_education == "A Level or Equivalent"  |
          highest_education == "Lower Than A Level"  |
          highest_education == "HE Qualification"
      )
    )

#How much this group represents
group_percentage <-
  (nrow(representative_student_group_info)/nrow(student_info_profile))*100

group_percentage
```

```
## [1] 68.79277
```





After these correlations, it can be seen that those students aged ‘0-35’ who have College Level, or High School complete/studying, represent the general profile of the students, as they are 68.8% of the samples.

5. performance factors (II)

Performance on each assessment is a good indicator of students’ knowledge of the course. We will separate the final exams from the other assessments, as their status and participation in the final assessment are different from the others.

Reading the assessment data

```
  #Information from the tests per student
student_assessment <-
  read.csv(
    file = paste0(getwd(),"/Input/studentAssessment.csv")
  )

  #Tasting information
assessments <-
  read.csv(
    file = paste0(getwd(),"/Input/assessments.csv")
  )
```

Separating the Exams

```
final_exams <-
  assessments %>%
  subset(assessment_type == "Exam")

head(final_exams)
```

```
##    code_module code_presentation id_assessment assessment_type date weight
## 6          AAA             2013J          1757            Exam   NA    100
## 12         AAA             2014J          1763            Exam   NA    100
## 24         BBB             2013B         14990            Exam   NA    100
## 36         BBB             2013J         15002            Exam   NA    100
## 48         BBB             2014B         15014            Exam   NA    100
## 54         BBB             2014J         15025            Exam   NA    100
```

```
others_exams <-
  assessments %>%
  subset(assessment_type != "Exam")

head(others_exams)
```

```
##   code_module code_presentation id_assessment assessment_type date weight
## 1         AAA             2013J          1752             TMA   19     10
## 2         AAA             2013J          1753             TMA   54     20
## 3         AAA             2013J          1754             TMA  117     20
## 4         AAA             2013J          1755             TMA  166     20
## 5         AAA             2013J          1756             TMA  215     30
## 7         AAA             2014J          1758             TMA   19     10
```

Let's identify the average rating per student per module, and identify the activities of those with the highest and lowest average ratings.

```
#Creating the data frame 'student_group_kpis

student_group_kpis <-
  student_assessment %>%
  mutate(pass = ifelse(score>=40, TRUE, FALSE))

#Putting together the exam information and creating the columns of who passed the exam and th
e grid weight

student_group_others_exams <-
  student_group_kpis %>%
  inner_join(others_exams, by = "id_assessment")

student_group_others_exams <-
  student_group_others_exams %>%
  mutate(weight_grade = score*weight/100)

head(student_group_others_exams[,c(1,6,7,11)])
```

```
##   id_assessment pass code_module weight
## 1          1752 TRUE         AAA     10
## 2          1752 TRUE         AAA     10
## 3          1752 TRUE         AAA     10
## 4          1752 TRUE         AAA     10
## 5          1752 TRUE         AAA     10
## 6          1752 TRUE         AAA     10
```

```
#Final assessment average per student per module

avg_grade_others_exams <-
  student_group_others_exams %>%
  dplyr::group_by(id_student, code_module, code_presentation) %>%
  mutate(avg_grade = sum(weight_grade)) %>%
  select("id_student","code_module","code_presentation", "avg_grade")

head(avg_grade_others_exams)
```

```
## # A tibble: 6 × 4
## # Groups:   id_student, code_module, code_presentation [6]
##   id_student code_module code_presentation avg_grade
##        <int> <chr>       <chr>                 <dbl>
## 1      11391 AAA         2013J                  82.4
## 2      28400 AAA         2013J                  65.4
## 3      31604 AAA         2013J                  76.3
## 4      32885 AAA         2013J                  55
## 5      38053 AAA         2013J                  66.9
## 6      45462 AAA         2013J                  67.8
```

```
#Final exams scores

student_group_final_exams <-
  student_group_kpis %>%
  inner_join(final_exams, by = "id_assessment")  %>%
  dplyr::rename("exams_score" = "score")%>%
  select("id_student","code_module","code_presentation", "exams_score")

head(student_group_final_exams)
```

```
##   id_student code_module code_presentation exams_score
## 1     558914         CCC             2014B          32
## 2     559706         CCC             2014B          78
## 3     559770         CCC             2014B          54
## 4     560114         CCC             2014B          64
## 5     560311         CCC             2014B         100
## 6     560494         CCC             2014B          92
```

Having gathered the data from the assessments, let’s check the data on student interactions with the university’s virtual environment

Checking interactions:

The datasets pertaining to the university's virtual environment contain the student interaction feed with the available content. From this data, we can infer how a student was in touch with his subjects, whether he studied it solidly, and how he used the content.

```
#Reading the tables of interactions

student_vle <-
  read.csv(
    file = paste0(getwd(),"/Input/studentVle.csv")
  )

head(student_vle)
```

```
##   code_module code_presentation id_student id_site date sum_click
## 1         AAA             2013J      28400  546652  -10         4
## 2         AAA             2013J      28400  546652  -10         1
## 3         AAA             2013J      28400  546652  -10         1
## 4         AAA             2013J      28400  546614  -10        11
## 5         AAA             2013J      28400  546714  -10         1
## 6         AAA             2013J      28400  546652  -10         8
```

```
vle <-
  read.csv(
    file = paste0(getwd(),"/Input/vle.csv")
  )

head(vle)
```

```
##   id_site code_module code_presentation activity_type week_from week_to
## 1  546943         AAA             2013J      resource        NA      NA
## 2  546712         AAA             2013J     oucontent        NA      NA
## 3  546998         AAA             2013J      resource        NA      NA
## 4  546888         AAA             2013J           url        NA      NA
## 5  547035         AAA             2013J      resource        NA      NA
## 6  546614         AAA             2013J      homepage        NA      NA
```

If we look at the VLE table, we can indentify that there are some data without reference to the period of use, so to make the analysis more feasible we will filter them out.

```
#Clearing the ELV data, because some samples do not have the reference week for the materials

vle <-
  vle %>%
  subset(!is.na(week_from))

head(vle)
```

```
##     id_site code_module code_presentation activity_type week_from week_to
## 114  546732         AAA             2013J     oucontent         2       2
## 199  546719         AAA             2013J     oucontent         1       1
## 211  546681         AAA             2013J     oucontent         1       1
## 265  877040         AAA             2014J     oucontent         2       2
## 324  877045         AAA             2014J     oucontent         1       1
## 392  877044         AAA             2014J     oucontent         1       1
```

Here we can track the average time after the start of the course that the student has taken to use the materials and the average number of clicks per material:

```
#Overall average per student per module

avg_per_student <-
  student_vle %>%
  dplyr::group_by(id_student, code_module, code_presentation) %>%
  mutate(
    date_mean = mean(date),
    sum_click_mean = mean(sum_click)) %>%
  select("id_student","code_module","code_presentation", "date_mean", "sum_click_mean")

head(avg_per_student)
```

```
## # A tibble: 6 × 5
## # Groups:   id_student, code_module, code_presentation [1]
##   id_student code_module code_presentation date_mean sum_click_mean
##        <int> <chr>       <chr>                 <dbl>          <dbl>
## 1      28400 AAA         2013J                  87.0           3.34
## 2      28400 AAA         2013J                  87.0           3.34
## 3      28400 AAA         2013J                  87.0           3.34
## 4      28400 AAA         2013J                  87.0           3.34
## 5      28400 AAA         2013J                  87.0           3.34
## 6      28400 AAA         2013J                  87.0           3.34
```

Since we cannot identify performance faotres in the students who dropped out, we will take them out of the representative samples:

```
#Filtering only representative samples (According to the students' profile analysis)

representative_student_group_info <-
  student_info %>%
  subset(
    age_band == "0-35" &
      (
        highest_education == "A Level or Equivalent"  |
          highest_education == "Lower Than A Level"  |
          highest_education == "HE Qualification"
      ) &
      final_result != "Withdrawn"
  ) %>%
  distinct(id_student, .keep_all = TRUE)
```

```
#Compiling the relevant tables

df_1 <-
  inner_join(avg_grade_others_exams, student_group_final_exams,
             by = c("id_student", "code_module", "code_presentation"))

df_2 <-
  inner_join(representative_student_group_info, df_1,
             by = c("id_student", "code_module", "code_presentation"))
```

```
## Warning in inner_join(representative_student_group_info, df_1, by = c("id_student", : Each
row in `x` is expected to match at most 1 row in `y`.
## ℹ Row 3831 of `x` matches multiple rows.
## ℹ If multiple matches are expected, set `multiple = "all"` to silence this
##   warning.
```

```
final_df <-
  inner_join(df_2, avg_per_student,
             by = c("id_student", "code_module", "code_presentation")) %>%
  select(num_of_prev_attempts, final_result, avg_grade, exams_score, date_mean, sum_click_mea
n)
```

```
## Warning in inner_join(df_2, avg_per_student, by = c("id_student", "code_module", : Each ro
w in `x` is expected to match at most 1 row in `y`.
## ℹ Row 1 of `x` matches multiple rows.
## ℹ If multiple matches are expected, set `multiple = "all"` to silence this
##   warning.
```

```
head(final_df[,-2])
```

```
##   num_of_prev_attempts avg_grade exams_score date_mean sum_click_mean
## 1                    0     89.65          94  119.3379       4.343939
## 2                    0     89.65          94  119.3379       4.343939
## 3                    0     89.65          94  119.3379       4.343939
## 4                    0     89.65          94  119.3379       4.343939
## 5                    0     89.65          94  119.3379       4.343939
## 6                    0     89.65          94  119.3379       4.343939
```

```
summary(final_df[,-2])
```

```
##  num_of_prev_attempts   avg_grade      exams_score       date_mean
##  Min.   :0.0000       Min.   : 3.72   Min.   :  0.00   Min.   : 27.01
##  1st Qu.:0.0000       1st Qu.:58.88   1st Qu.: 51.00   1st Qu.: 89.75
##  Median :0.0000       Median :74.25   Median : 67.00   Median :103.28
##  Mean   :0.1059       Mean   :70.98   Mean   : 65.98   Mean   :103.55
##  3rd Qu.:0.0000       3rd Qu.:86.15   3rd Qu.: 82.00   3rd Qu.:115.64
##  Max.   :5.0000       Max.   :99.80   Max.   :100.00   Max.   :230.00
##                       NA's   :20653
##  sum_click_mean
##  Min.   : 1.077
##  1st Qu.: 2.273
##  Median : 2.681
##  Mean   : 2.956
##  3rd Qu.: 3.331
##  Max.   :16.242
##
```

```
nrow(final_df[final_df$final_result == "Pass",])
```

```
## [1] 8043978
```

```
nrow(final_df[final_df$final_result == "Distinction",])
```

```
## [1] 2145211
```

```
nrow(final_df[final_df$final_result == "Fail",])
```

```
## [1] 1282021
```

With a much higher “Pass” count than the other labels, we should be on the lookout. Two outliers were detected: One with average clicks well above the standard values and another with a single occurrence from a number of previous attempts. To keep our data as consistent as possible, these cases will be removed.

```
final_df <-
  final_df %>%
  subset(sum_click_mean<10)
```

```
final_df <-
  final_df %>%
  subset(num_of_prev_attempts<4)
```

```
nrow(final_df)
```

```
## [1] 11435514
```

Separating the data to understand the profile of the students

who passed and those who failed

```
pass_student <-
  final_df %>%
  subset(final_result != "Fail")

head(pass_student[,-2])
```

```
##   num_of_prev_attempts avg_grade exams_score date_mean sum_click_mean
## 1                    0     89.65          94  119.3379       4.343939
## 2                    0     89.65          94  119.3379       4.343939
## 3                    0     89.65          94  119.3379       4.343939
## 4                    0     89.65          94  119.3379       4.343939
## 5                    0     89.65          94  119.3379       4.343939
## 6                    0     89.65          94  119.3379       4.343939
```

```
nrow(pass_student)
```

```
## [1] 10155306
```

```
fail_student <-
  final_df %>%
  subset(final_result == "Fail")

head(fail_student[,-2])
```

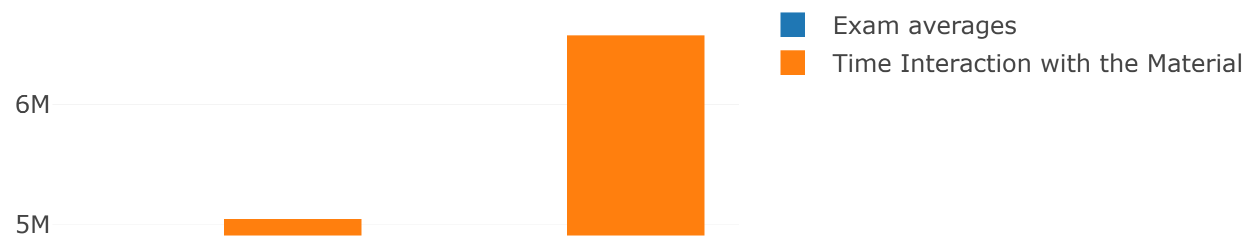
```
##       num_of_prev_attempts avg_grade exams_score date_mean sum_click_mean
## 42369                    0     40.96          24  96.93333       5.114286
## 42370                    0     40.96          24  96.93333       5.114286
## 42371                    0     40.96          24  96.93333       5.114286
## 42372                    0     40.96          24  96.93333       5.114286
## 42373                    0     40.96          24  96.93333       5.114286
## 42374                    0     40.96          24  96.93333       5.114286
```

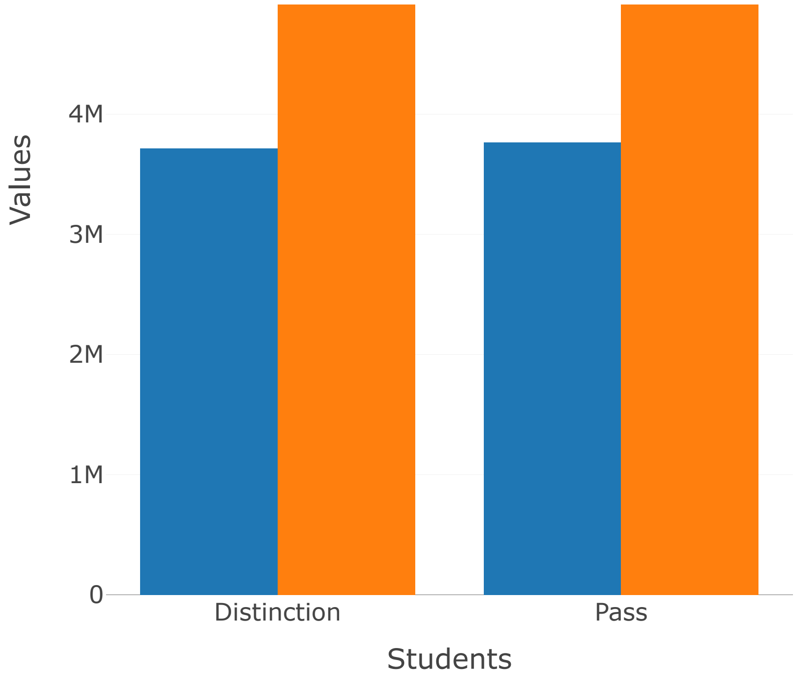
```
nrow(fail_student)
```

```
## [1] 1280208
```

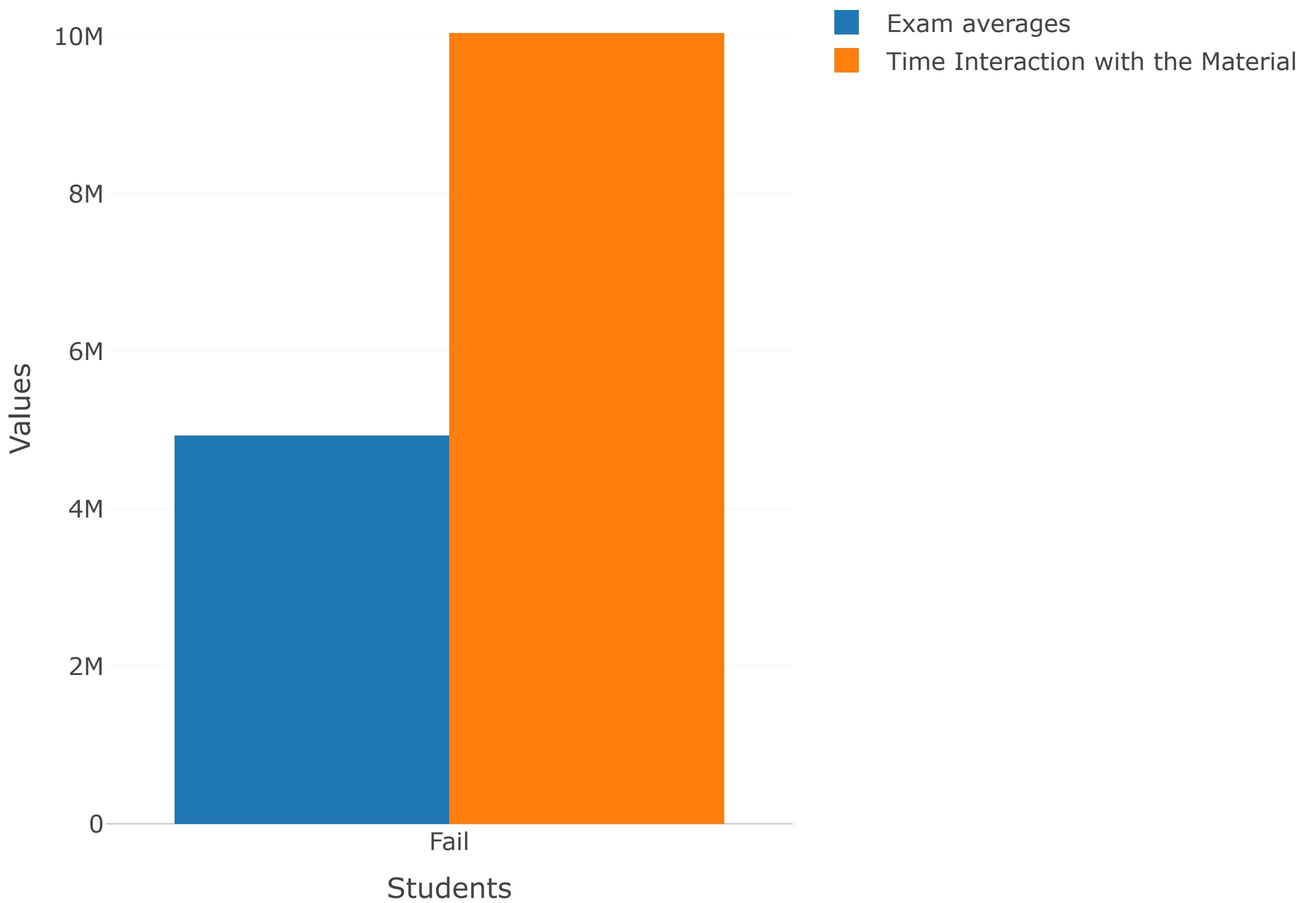
Since we have a large amount of samples, I will use below only the first one hundred thousand samples (100,000) to build the graphs:

Data from those who passed





Data of those who failed



6. Conclusion(III).

After the analysis performed, it can be seen that students who passed and performed better, overall had more interaction time with the material provided, as well as higher scores on the *Tutor Marked Assessment (TMA)* and *Computer Marked Assessment (CMA)* exams compared to those who did not pass. This may indicate that by focusing on initiatives to increase student interactivity with the online platform by increasing engagement with the non-final exam papers, it is likely that the chance of success for students who did not pass would increase. For this analysis to become more assertive it would be necessary to follow up with some regression modeling and identify whether this hypothesis is valid.