Volume 6(2), 140—152. http://dx.doi.org/10.18608/jla.2019.62.10

crsra: A Learning Analytics Tool for Understanding Student Behaviour in Massive Open Online Courses

Aboozar Hadavand[1], John Muschelli[2], Jeffrey Leek[3]

Abstract
Due to the fundamental differences between traditional education and massive open online courses (MOOCs), and because of the ever-increasing popularity of the latter, more research is needed to understand current and future trends in MOOCs. Although research in the field has grown rapidly in recent years, one of the main challenges facing researchers remains the complexity and messiness of the data. Therefore, it is imperative to provide tools that pave the way for more research on this new subject. This paper introduces a package called `crsra` based on the statistical software R to help tidy and perform preliminary analysis on massive loads of data provided by Coursera. The advantages of the package are as follows: (a) faster loading and organizing of data for analysis, (b) an efficient method for combining data from multiple courses and even from across institutions, and (c) provision of a set of functions for analyzing student behaviours.

Notes for Practice

- The size and complexity of massive open online course (MOOC) data present an overwhelming challenge to many researchers.
- The package `crsra` tidies and performs preliminary analysis on Coursera data.
- The advantages of the package are faster loading of data, an efficient method for combining data from multiple courses and institutions, and provision of a set of functions for analyzing student behaviours.

Keywords
MOOCs, learning analytics tools, R package, big data in education, student progress

Submitted: 07.11.2018 — **Accepted:** 01.02.2019 — **Published:** 05.08.2019

[1] *Address: Department of Biostatistics, Johns Hopkins University, 615 N Wolfe Street, Baltimore, MD 21205, USA*
[2] *Address: Department of Biostatistics, Johns Hopkins University, 615 N Wolfe Street, Baltimore, MD 21205, USA*
Corresponding author: [3] *Email: jtleek@gmail.com Address: Department of Biostatistics, Johns Hopkins University, 615 N Wolfe Street, Baltimore, MD 21205, USA*

1. Introduction

Research on massive open online courses (MOOCs) is young. Bozkurt, Akgün-Özbek, & Zawacki-Richter (2017) study literature published on MOOCs through 2015 and find that the number of articles published on the subject increased from 1 in 2008 to 170 in 2015. More research is needed to fully understand the effectiveness, reach, limits, and potential of MOOCs (Maiz Olazabalaga, Castaño Garrido, & Garay Ruiz, 2016; Gašević, Kovanović, Joksimović, & Siemens, 2014). However, the main challenge in studying MOOCs is data. Data on MOOCs is not usually publicly available since it is owned by private providers and there are concerns about the privacy of students. More important, as Lopez, Seaton, Ang, Tingley, & Chuang (2017) point out, the size and complexity of MOOC data present an overwhelming challenge to many researchers. Therefore, it is imperative to provide tools that pave the way for more research on the new subject of MOOCs.

There has been some effort to provide tools for facilitating big data in MOOCs. Pardos & Kao (2015) integrate data request/authorization and distribution workflows and provide a basic tool for replicating research in learning analytics. The MOOC Learner Project[1] is another project that provides open-source software that enables researchers to extract learning insights from the data collected on edX or open edX platforms. Neither of these tools is designed to analyze Coursera data. In this paper, we introduce `crsra`, a software package for the R programming language that helps clean and analyze massive loads of data from Coursera MOOCs. Coursera, launched in January 2012, is one of the leading providers of MOOCs. With

[1]This tool requires elementary Python programming and elementary familiarity with creating MySQL databases and importing data from SQL dumps into MySQL databases. It is available at http://mooclearnerproject.csail.mit.edu/.

over 25 million learners, Coursera is the most popular provider in the world, followed by edX, the MOOC provider that was a result of a collaboration between Harvard University and MIT, with over 10 million users. Coursera has more than 150 university partners from 29 countries and offers more than 2,000 courses, from computer science to philosophy[2]. Furthermore, Coursera offers 180+ specializations, its own credential system, and four fully online Masters degrees. A typical course on Coursera includes recorded video lectures, graded assignments, quizzes, and discussion forums.

Since its early years, Coursera has encouraged researchers to analyze students' data and has facilitated the use of the data and the platform for A/B testing. In November 2015, Coursera introduced a dashboard for self-service data exports. Through this tool, partner institutions and instructors could download data for a single course or all courses associated with the institution. Research data exports are sets of .csv files and are designed for use in relational database systems. One of the advantages of the data is the existence of a single *hashed user ID* for each student. This user ID is consistent for learners across all courses offered by an individual institution and allows learner grades and progress to be connected across courses.

The advantages of the package are as follows:

1. Data for analysis is loaded faster.
2. Methods of combining data from multiple courses and even across institutions are efficient. This is important since although MOOC researchers have access to thousands of students in their sample, few studies benefit from data across multiple courses and institutions. Such analysis helps draw more robust conclusions about student behaviours (Reich, 2015).
3. A set of functions for analyzing student behaviours is provided.

Section 2 summarizes the main features and functionalities of the package. In section 3, we will use the package to analyze student progress across Johns Hopkins University (JHU) Data Science Specialization courses on Coursera. Section 4 concludes the paper.

2. The `crsra` Package

2.1 The Coursera Data

Instructors of courses offered on Coursera have access to data on students who have interacted with their courses on the platform. This interaction can be just a click on the course link or enrolling in and completing a course. While instructors previously had to request data export by contacting Coursera, nowadays, downloading the data can easily be done through the instructor's dashboard. Universities that offer multiple courses on Coursera can download data on all courses, with the option of tracking students across courses using hashed user IDs.

All universities are provided with the same data structure. Depending on the enrolment for each course, however, the exported data files can be as large as several gigabytes, which can potentially be an issue for analyzing the data. There are five types of research data export for each course. Table 1 summarizes these five types. The data is written in roughly 100 tables, containing information such as course content; students' demographic data, progress, and outcomes; and forum data. Figure 1 shows different types of research data exports provided by Coursera.

While Coursera provides tools for creating PostgreSQL databases in a docker container[3], as we mentioned earlier, importing data for analysis remains a challenge for researchers with limited experience with relational databases. Moreover, such tools are usually not platform independent[4].

2.2 Installing the `crsra` Package

The `crsra` package helps import and organize Coursera's research data exports into R. It also runs some preliminary analysis on the data. In the following section, we introduce the package and provide instructions on how to import Coursera research data exports. You can either directly install the package from CRAN using `install.packages("crsra")` or install the development version through `devtools`. Then execute the following commands:

```
library("devtools")
devtools::install_github("jhudsl/crsra", build_vignettes = TRUE)
```

[2]`https://blog.coursera.org/about/`

[3]The tool is called `courseraresearchexports` and can be found at `https://github.com/coursera/courseraresearchexports`.

[4]In an initial version of `crsra` based on PostgreSQL, we had the problem of some team members not being able to set up the database correctly on their PCs.

Table 1. Types of Research Data for Export

Data Type	Description
Assessment submission data	Contains assessment submissions of quizzes, peer review, and programming assignments by learners
Course grade data	Contains the highest grade achieved by each learner on each required assessment, the time stamp of the learner's highest-scoring submission, and each learner's overall grade in the course
Course progress data	Contains data documenting the time stamp for when the learner interacted with each piece of course content and the time stamps for when items were opened, completed, reopened, reattempted, etc.
Demographic data	Contains the following information for all enrolled learners: general geographical data (based on IP address), browser language preference, and information for learners who completed their learner profile responses or participated in Coursera's platform-wide demographic survey (including age, gender, education level, and employment status)
Discussion data	Contains forum activity data such as posts, responses, upvotes/downvotes, flags, and questions and answers associated with course content items

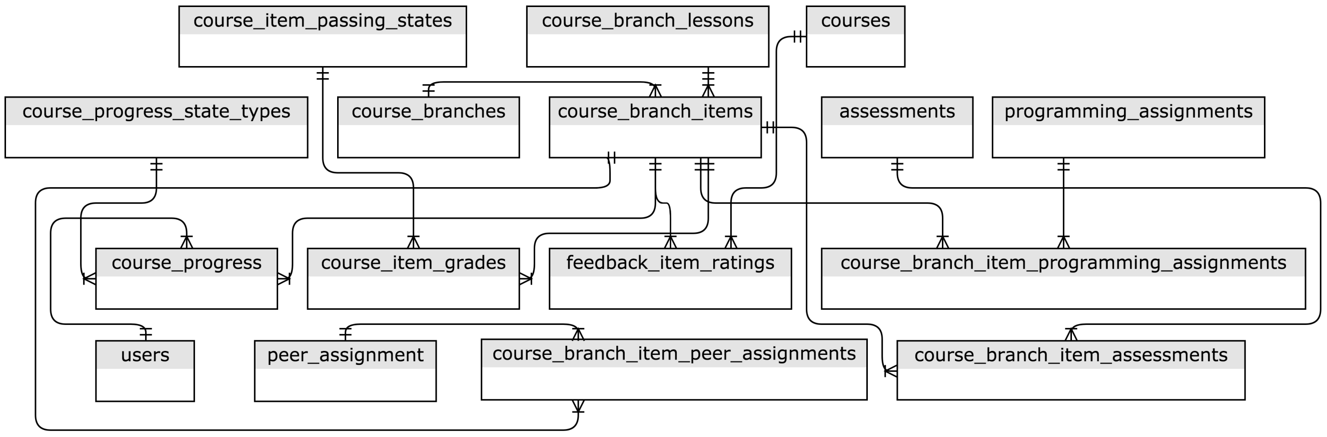

Figure 1. Major relationships between table groups, with minor connections omitted (Source: Coursera)

2.3 Importing Data

We have provided a dummy dataset for those who do not have immediate access to the data or would like to practise with a dummy set before using the package on their main data. This dummy data set can be called using the function `data()` as follows:

```
data(example_course_data)
```

This example course data is meant to replicate the original Coursera data exports and, therefore, includes all 100 tables. The following instruction can be used to import your own data exports. To import your data dump into R, you will first need to unzip it into one folder. There are two ways to import the data. You can import all your courses at once by pointing your working directory to the folder that contains all the unzipped folders. Then execute the command `crsra_import`. Setting the `check_problems` attribute to FALSE will turn off warnings when importing the .csv files.

```
my_import <- crsra_import(workdir = ".", check_problems = TRUE)
```

A specific table within a course can be called as follows:

```
user_table <- my_import[["Regression Models"]][["users"]]
```

A second method is to import an individual course. For this type of import, unzip the data export for the course and set your working directory to the folder that contains all the .csv tables.

```
my_course_import <- crsra_import_course(workdir = ".", check_problems = TRUE)
```

Calling a table is a little different in this method since the imported data only contains one course. To call a table, execute

```
user_table_2 <- my_course_import[["users"]]
```

Note that the data import may take some time if the course data is large and if there are several courses in your working directory. To see the data import in use, we use the package on data from the JHU Data Science Specialization on Coursera. This specialization, developed by Jeffrey Leek, Roger Peng, and Brian Caffo, consists of 10 courses. There have been more than two million enrollments since the launch of this program in April 2014. The size of data on the students who took these 10 courses since 2015 is around 18 gigabytes. In the following example, we use the `crsra` package to import a Coursera data dump at our disposal on all the courses and to find the number of students who passed a specific course item (course item `67c1O`) in the course "Regression Models."

```
## Imports the table course_item_grades for the course Regression Models.
## We only look at individuals who have finished course item 67c1O.
## Value 2 for the column course_item_passing_state_id indicates that the person
## has passed the course item.

my_import_passed <- subset(my_import[["Regression Models"]][["course_item_grades"]],
    course_item_id == "67c1O" & course_item_passing_state_id == 2)
nrow(my_import_passed)

# 1  8640
```

The package also includes a few other functions in addition to the main `crsra_import()` function. A list of functions and their descriptions is provided in Table 2. While instructors can see some course statistics through the dashboard that Coursera provides, a lot of the details about the courses are not provided. The functions provided in the package `crsra` are meant to fill this gap.

2.4 Summary of Student Grades

Similar to the example above, we can use the function `crsra_gradesummary()` to calculate the average student grade for the courses in the data import[5]. By using the argument `groupby`, we can calculate average grades for different learner subgroups based on gender, education, student status, employment status, and country. For instance, the following analysis returns the average overall course grade for male and female learners in the course The Data Scientist's Toolbox. The results show that female learners' grades are on average 6 points lower on a 100-point scale than male learners' grades.

```
## Calculates the grade summary for each course in the data:
grade_sum <- crsra_gradesummary(my_import, groupby = "gender")
```

[5]The Coursera instructor's dashboard provides the grade summary for each course; however, this function provides more flexibility to the researcher for calculating the grade summary.

Table 2. Some of the Functions in the `crsra` Package

Function	Description
`crsra_membershares`	Returns a summary of the total number and the shares of learners in each course broken down by factors such as roles, country, language, gender, employment status, education level, and student status
`crsra_gradesummary`	Returns the total grade summary broken down by the factors mentioned above
`crsra_progress`	Summarizes, for each course item, the total number and the share of learners who ceased activity at that specific course item; also ranks course items by their attrition
`crsra_assessmentskips`	Categorizes skips (learners may "skip" reviewing a peer-assessed submission if there is a problem with it) by their type, such as "inappropriate content," "plagiarism," etc.; also returns a word cloud appearing in peer comments as to why they skipped the submission
`crsra_timetofinish`	Calculates the time to finish a course for each user
`crsra_anonymize`	Anonymizes ID variables (such as Partner hashed user ids) throughout the data set; based on the function `digest` from the package `digest`
`crsra_delete_user`	Deletes a specific user from all tables in the data in case Coursera data privacy laws require you to delete a specific user or set of users from your data
`crsra_tabledesc`	Returns a description for a table
`crsra_whichtable`	Returns a list of tables in which a variable appears

```
## Prints the summary of grades for the course Regression Models:
grade_sum[["Regression Models"]]

#Note that maximum grade possible is 1.
# A tibble: 2 x 2
#  reported_or_inferred_gender  AvgGrade
#                        <chr>     <dbl>
#1                        male 0.7250660
#2                      female 0.6691554
```

2.5 Share of Students in Different Subgroups

The function `crsra_membershares()` summarizes the number of learners across various subgroups. The argument `groupby` can be used to calculate shares of subgroups based on different categories, such as employment status, student status, gender, and education.

```
## Calculates the share of students across various subgroups:
member_shares <- crsra_membershares(my_import, groupby = "empstatus")

## Prints the shares of learners for different employment groups for the course Regression
    Models:
member_shares[["Regression Models"]]

# A tibble: 9 x 3
  y                                 Total  Share
  <chr>                             <int>  <dbl>
1 EMPLOYED_FULL_TIME                 2209 0.680
2 UNEMPLOYED_LOOKING_FOR_WORK         489 0.150
3 EMPLOYED_PART_TIME                  181 0.0600
4 UNEMPLOYED_NOT_LOOKING_FOR_WORK     123 0.0400
5 SELF_EMPLOYED_FULL_TIME             119 0.0400
6 SELF_EMPLOYED_PART_TIME              61 0.0200
7 UNABLE_TO_WORK                       26 0.0100
8 HOMEMAKER                            20 0.0100
```

```
9 RETIRED                           11 0.
```

2.6 Summary of Student Activity by Course Items

The next function, `crsra_progress()`, summarizes, for each course item, the total number and the share of learners who ceased activity at that specific course item. The function ranks course items by their attrition. The column `item_rank` shows the rank of a course item based on where it stands in the course. For instance, the `item_rank` for course item 1 in lesson 1 in module 1 is 1.

```
## Summarizes the total number and the share of learners who ceased activity at each course
   item:
progress <- crsra_progress(my_import)

## Prints the shares for the course Regression Models:
glimpse(progress[["Regression Models"]])

Observations: 86
Variables: 6
\$ item_rank  <int> 85, 1, 8, 9, 12, 2, 24, 20, 78, 3, 84, 86, 69, 19, 71, 14, 79, 10, 7, 15
   , 6, 73, 16, 26, 13, 28, 45, 27, 11,...
$ Total  <int> 6070, 4006, 2554, 1791, 1250, 1186, 1105, 1029, 897, 810, 754, 652, 588, 572,
    565, 556, 486, 472, 453, 444,...
$ Share  <dbl> 0.16, 0.11, 0.07, 0.05, 0.03, 0.03, 0.03, 0.03, 0.02, 0.02, 0.02, 0.02, 0.02,
    0.02, 0.02, 0.01, 0.01, 0.01,...
$ course_item_name  <chr> "Regression Models Course Project", "Welcome to Regression Models"
   , "Introduction to Regression", "Introduc...
$ course_lesson_name <chr> "Course Project", "Introduction", "Introduction to regression and
    least squares", "Introduction to regressi...
$ course_module_name <chr> "Week 4: Logistic Regression and Poisson Regression", "Week 1:
   Least Squares and Linear Regression", "Week ...
```

Defining completion rates as the percentage of enrolled students who complete the course is shown to be a rough measure of a MOOC's success. Jordan (2015) shows that completion rates vary significantly according to course length, start date, and assessment type. Deng, Benckendorff, & Gannaway (2019) point out that measures of learning outcomes lack sophistication and are often based on single variables. The function `crsra_progress` can help researchers define a more granular measure of completion in MOOCs.

2.7 Analysis of Length of Time to Finish Courses

Using the `crsra_timetofinish` function in the `crsra` package, we can first investigate the time difference between the first and last activities in a course for each student. Time to finish is only calculated for those who completed the course. Figure 2 depicts the density plot for time to complete for three of the courses in the specialization. Note that the density plot varies across courses. While for Developing Data Products and Getting and Cleaning Data, a majority of students finish the courses in around 30 days, for Data Science Capstone, a majority of students finish the course in 50 days.

```
## Calculates the time to finish a course for all individuals/courses in the data:
ttf <- crsra_timetofinish(my_import)
```

In the table called `users`, Coursera provides a field for student status of the learner including full-time and part-time students and those who are not degree students. We can look at how time to finish is different for groups with different student status. Pursel, Zhang, Jablokow, Choi, & Velegol (2016) and Kizilcec, Pérez-Sanagustín, & Maldonado (2017) show that demographics can explain a student's activity pattern in MOOCs. We will use the `dplyr` package throughout this section since it facilitates working with data frame–like objects (Wickham, François, Henry, & Müller, 2018). Figure 3 reports this for the course Getting and Cleaning Data and shows that part-time students take longer to finish the course. The peaks of the three distributions are quite close. However, the main difference between groups is that the time-to-finish distribution has a fatter tail for part-time students.

```
## Loads package dplyr:
library(dplyr)
## Time to finish a course for the course Getting and Cleaning Data
ttf_gcd <- ttf[["Getting and Cleaning Data"]]
## Merging the data above with the table users for analysis of time to finish by user groups
```

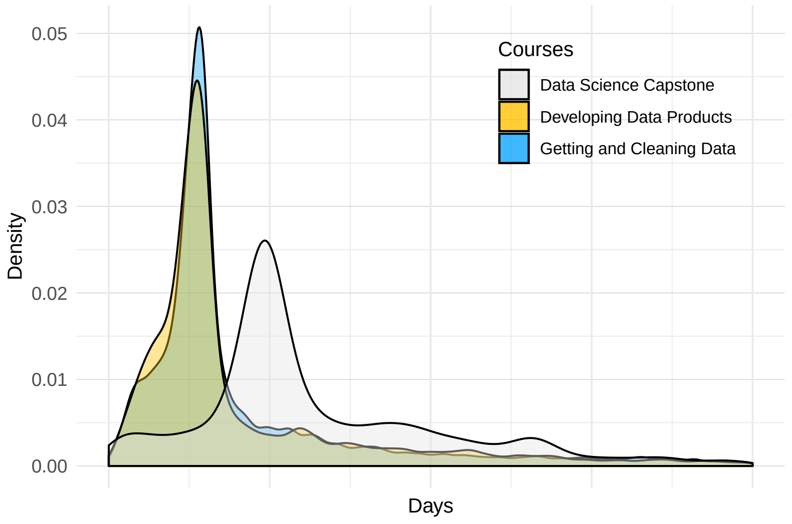

Figure 2. Density plots for time to finish defined as the time difference between the first and last activities across three courses: Data Science Capstone (3,491 students), Developing Data Products (5,684 students), and Getting and Cleaning Data (15,281 students)

```
## jhu_user_id is the learner identifier and is used for merging the two tables.
ttf_gcd_join <- dplyr::left_join(ttf_gcd,
    my_import[["Getting and Cleaning Data"]][["users"]],
    by = "jhu_user_id", `copy`=TRUE)
ttf_gcd_join_narm <- subset(ttf_gcd_join, !is.na(student_status))
```

In the following section, we will use some of the other functions of the package in a more detailed analysis. Since the package may not address all the needs of researchers who intend to analyze the Coursera research exports, we have asked researchers to contribute to the project by making pull requests on the package Github repository[6]. Those interested in a specific functionality that the package does not currently provide can create a Github issue and detail their request.

3. Example Analysis: Student Behaviour on Coursera

3.1 Learning in Online Courses

There are only a handful of studies on students' progress and outcomes in MOOCs. Perna et al. (2014) perform a descriptive analysis of student progress through a set of 16 courses on Coursera. They find that most users accessed course content in the sequential order defined by the instructor of the course. Ho et al. (2014) study 17 courses taught on edX and find that most of the attrition in online courses happens in the first week of course activity (about 50 percent attrition) and that the average percentage of learners who cease activity in the second week declines sharply to 16 percent.

In a more recent paper, Lu, Bradlow, & Hutchinson (2017) focus on bingeing behaviours (consuming content from the same course in succession) among MOOC learners using a clickstream dataset from Coursera to study how the timing of content release affects binge consumption. In the following section, we will show how the `crsra` package can be used to investigate students' progress through MOOCs by using data from the JHU Data Science Specialization on Coursera.

3.2 Student Progress in Online Courses

One of the factors that distinguish MOOCs from traditional classrooms is the flexibility in advancing through the course. While in traditional education, the class length, pace, and completion dates are determined by the instructor of the course, in MOOCs it is the student who, for the most part, has the freedom to choose these factors. We can then look at how many course items students pass in the first week of course activity. One obvious yet interesting finding is large variations across students. For instance, if we look at the course Getting and Cleaning Data, we can use the following code to find the number of course items completed in the first week of course activity. The course has roughly 40 items, including lectures and assignments. The variable `nweek1` captures the number of passed course items in the first week of course activity, calculated as one week after a student's first activity in the class. The density plot in Figure 4 represents the variations across students. For a majority of learners, the number of passed course items in the first week is two. However, the number of those who finish more than 10

[6]The repository is available at `https://github.com/jhudsl/crsra`.

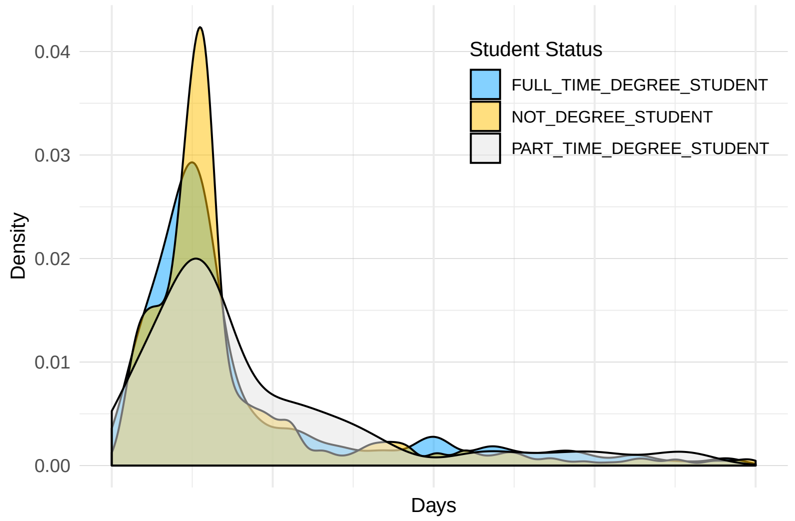

Figure 3. Density plots for time to finish for learners with different student statuses in the course Getting and Cleaning Data (1,221 students)

items in the first course is significant. Also interesting is the double-peak shape of the density plot. The histogram has two peaks at 2 course items (4,789 students) and 12 course items (3,319).

```
passed_items <- my_import[["Getting and Cleaning Data"]][["course_progress"]]
## 604800 is the number of seconds in a week.
## The following code counts the number of course items in the first week
## for each student:
passed_items <- dplyr::filter(dplyr::group_by(passed_items, jhu_user_id),
    course_progress_ts <= min(course_progress_ts) + 604800)
dplyr::summarise(passed_items, nweek1 = n())
```

A third interesting variable to look at when studying students' progress in MOOCs is the time gaps between sessions. In this exercise, we looked at the time lapsed between each pair of consecutive course items for each learner throughout the course Getting and Cleaning Data. We used the following code for the analysis. Note that we have ranked course items based on the time stamp when a student passes them and not based on their natural order defined by the course instructor. For this part, we will use the pipe function (`%>%`) in the `dplyr` package because it is more convenient.

```
## Loads the course_progress table for the course Getting and Cleaning Data:
my_import_progress <- my_import[["Getting and Cleaning Data"]][["course_progress"]]
## The following code calculates the time lapsed between each
## pair of consecutive course items for each learner:
gaps <- my_import_progress %>%
    ## 2 is an indicator that the course item is completed:
    filter(course_progress_state_type_id == 2) %>%
    group_by(jhu_user_id, course_item_id) %>%
    ## This is for keeping only the latest event for each course item:
    filter(course_progress_ts == max(course_progress_ts)) %>%
    ungroup() %>%
    arrange(jhu_user_id, course_progress_ts) %>%
    group_by(jhu_user_id) %>%
    ## This is for converting the time gap to hours:
    mutate(time.dif = as.numeric(course_progress_ts -
                                         lag(course_progress_ts))/3600) %>%
    ## filtering if the time difference is infinity or missing:
    filter(!is.na(time.dif)) %>%
    filter(time.dif != Inf | time.dif != -Inf)
```

Figure 5 shows student progress in the course Getting and Cleaning Data for three sample students. The vertical axis is the gap between two consecutive sessions in hours. These three students are chosen intentionally to show three different learning paths. Panel A shows progress for a student with short gaps between sessions for the first half of the course and longer gaps

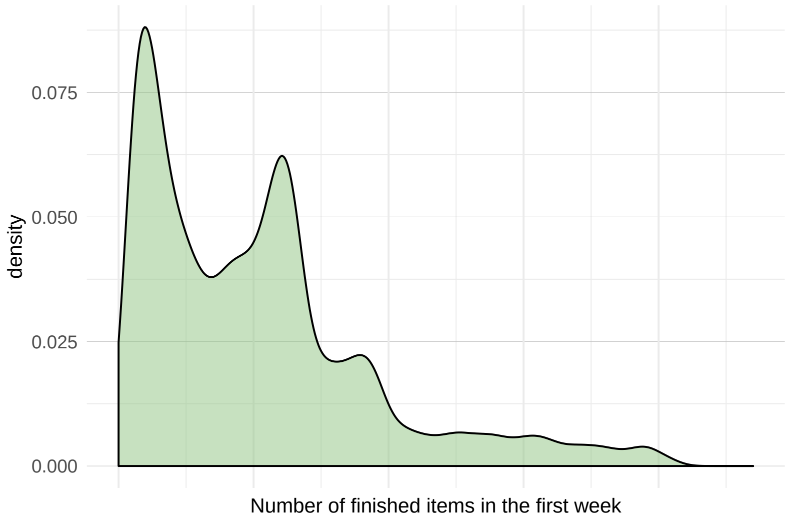

Figure 4. Density plot for the number of passed items in the first week of course activity for the course Getting and Cleaning Data (45,921 students)

toward the end. We call this the "slowing-down" pattern. This pattern is typical of many students in this course. Panel B shows progress for a student with short gaps between sessions in the beginning and the end of the course and longer gaps in the middle. There are fewer students in this group than in the first group. Finally, Panel C shows progress for a student with no apparent pattern in their progress throughout the course. Only a small group of students follow this pattern in our data. Analysis of patterns like these is an open question that this package will allow someone to address in the near future.

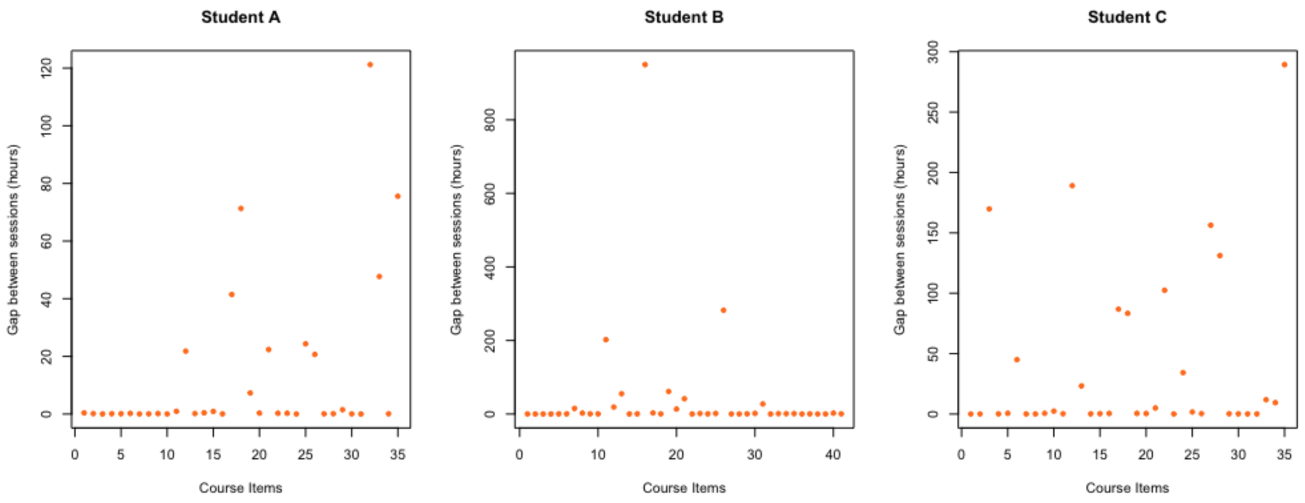

Figure 5. Student progress through the course Getting and Cleaning Data for three sample students. The vertical axis shows the time gap between completing an item and the next item in hours.

Let us look at the average gap between sessions for the first and the second half of the course. We can then calculate how much the average session gap changes from the first half to the second. Across our sample of students registered for the course, the average change in session gap from the first half to the second half is positive and equal to 132 percent. In other words, the gap between sessions more than doubles from the first half of the course to the second half. Figure 6 shows the density plot of this statistic across our sample. The long right tail in the figure supports the notion that most students follow the slowing-down pattern. Given the features of the `crsra` package, this can be further studied in future research.

We can do a similar analysis for some subgroups of our sample. Some of the most interesting categories are gender, educational attainment, and whether the learner paid to take the course. The variable `was_payment` in the table `users_courses__certificate_payments` captures whether the learner has ever paid for a course certificate. This

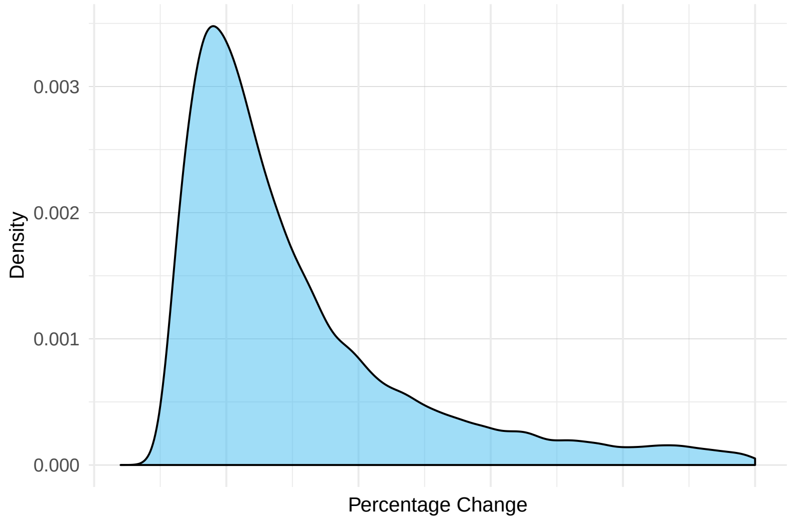

Figure 6. Percentage change in time gaps in course progress between the first and the second half of the course Getting and Cleaning Data (13,200 students)

Table 3. Gaps between Sessions for Different Subgroups of Learners in the Course Getting and Cleaning Data

Categories	Level	Average Activity Time Gap (hours)
Gender	Female	39
	Male	36
Educational Attainment	Less than high school	152
	High school diploma	39
	College (no degree)	39
	Associate degree	23
	Bachelor's degree	32
	Master's degree	38
	Professional degree	22
	Doctoral degree	34
Paid for the course?	Yes	36
	No	39

purchase could be a "single payment" for the course or a "bulk payment" for a specialization that contains the course. The following code is used for the analysis of payers and non-payers, and the results for all categories are shown in Table 3.

```
## The following code calculates the average gap between course items
## for different students based on whether they paid for courses or not:
gaps_payment <- gaps %>%
    group_by(jhu_user_id) %>%
    ## calculates the average of time gaps for each student:
    summarise(avgtime = mean(time.dif)) %>%
    ## joining the data above with the table course_grades for selecting those
    ## who have passed the course:
    inner_join(my_import[["Getting and Cleaning Data"]][["course_grades"]],
                      by = "jhu_user_id", `copy`=TRUE) %>%
    filter(course_passing_state_id %in% c(1, 2)) %>%
    ## joining with the table user_courses__certificate_payments for comparing students:
    left_join(
        my_import[["Getting and Cleaning Data"]][["users_courses__certificate_payments"]],
        by = "jhu_user_id", `copy`=TRUE) %>%
    filter(!is.na(was_payment)) %>%
    group_by(was_payment) %>%
    summarise(avggap=mean(avgtime))
```

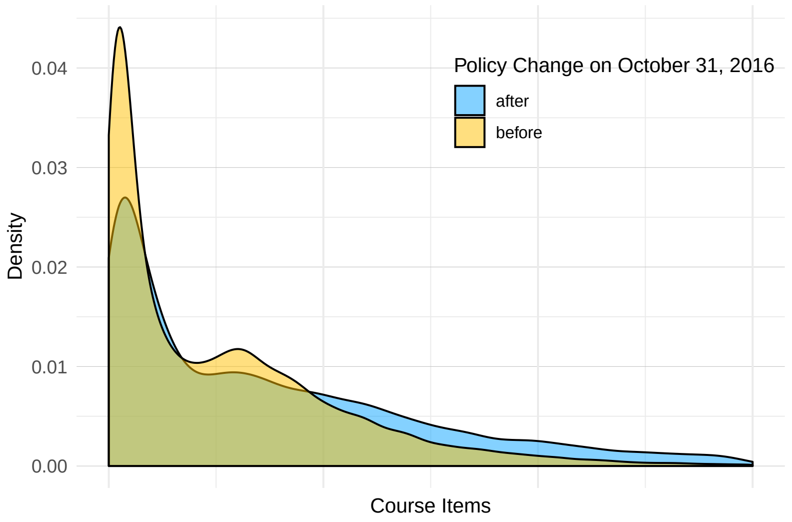

Figure 7. Density plots for the number of passed items in the first week of course activity for Getting and Cleaning Data for those who enrolled in the course before October 31, 2016, and those who enrolled after

3.3 The Effect of a Change in Coursera's Payment Plan on Student Progress

The last exercise in our analysis is to study how Coursera's change in policy from a pay-per-course business model to a subscription model changed students' progression throughout the course. On October 31, 2016, Coursera introduced a new payment system through which they allowed students to purchase access to all content in a specialization on a month-by-month or annual basis[7]. As a result, students would only pay for the amount of time they needed to learn the material. This system replaced the existing model, where students would pay up front for each course regardless of how long it took them to finish the course. A question to ask is whether the switch to this system where payments are tied to the length of time it takes students to complete the class speeds up learning. This is a potential area of research that can be easily performed using the package.

The following code calculates the average number of course items passed in the first week of activity for the two groups: those who enrolled in the course before October 31, 2016, and those who joined after. We hypothesize that those who pay monthly are more likely to finish more items in the first week than those who pay a fixed price.

```
## The following code compares the number of passed items for students with
## different payment methods based on the time they enroll for a course:
passed_items_policy <- passed_items %>%
    ## Merging the data with the table course_memberships that includes
    ## information on when students enrolled for the courses:
    left_join(my_import[["Getting and Cleaning Data"]][["course_memberships"]],
                        by = "jhu_user_id", `copy`=TRUE) %>%
    filter(!is.na(course_membership_ts)) %>%
    ## Creating an indicator for whether a student enrolled in the course
    ## before or after 2016-10-31:
    mutate(subscription = ifelse(course_membership_ts < "2016-11-01 00:00:00",
                                            "before", "after")) %>%
    group_by(subscription) %>%
    summarise(subnw = mean(nweek1))
```

The results suggest that those who enrolled before the policy change on average passed three fewer courses than the group who enrolled after (9 versus 12). Figure 7 shows the density plot of the number of passed items in the first week of activity for the two groups. This comparison, however, has a caveat: there is some selection bias since those who enrolled before and those who registered after October 31, 2016, may be fundamentally different.

4. Discussion

Importing, processing, and analyzing large amounts of data has discouraged researchers from studying MOOCs (Lopez et al., 2017). These researchers have the domain knowledge to ask interesting and important questions on the topic of MOOCs,

[7]Coursera. Coursera blog: Introducing subscriptions for specializations. `https://blog.coursera.org/introducing-subscriptions-for-specializations/`

but the time and skills required to analyze big data to answer such questions act as a barrier to entry. The `crsra` package reduces these barriers by providing tools for cleaning and analyzing Coursera's research data exports. Our motivation for the development of the package was initially to analyze student progress using the JHU Data Science Specialization data. The size and messiness of the data were our main challenges. In this regard, we felt the need for a tool that imports the data into R, cleans the data, and provides some analysis of the courses. We decided later on to make the tool available to all researchers who use Coursera's research data.

Because MOOCs are new, analyzing students' behaviours on MOOCs is essential. One of the main differences between MOOCs and traditional college classrooms is how students progress through MOOCs. While the order and length of course content and the pace at which the course is taught are chosen by the instructor in classrooms, in MOOCs, it is the student who determines when and how to learn the material. Thus, we used the `crsra` package to analyze student progress and pace in one of a sample of courses offered by JHU. One of the least studied aspects of MOOCs is the learning patterns among MOOC learners. With the help of this package, researchers can study what characteristics define why some students learn at a faster pace than others. We provided a benchmark for studying student progress. We hope the package and the analysis provided in this paper will pave the way for future studies on student behaviours on MOOCs.

Acknowledgements

We would like to thank Ira Gooding, the Coursera Research team, and the Johns Hopkins University Data Science Lab for support in completing this project.

Declaration of Conflicting Interest

Dr. Leek receives financial compensation through the Johns Hopkins University Tech Transfer Program from revenue generated by the Johns Hopkins University Data Science Specialization on Coursera. Dr. Muschelli receives financial compensation through the Johns Hopkins University Tech Transfer Program from revenue generated by his course on Neurohacking on Coursera.

Funding

The publication of this article received financial support from the National Institute of General Medical Sciences (NIGMS) of the National Institutes of Health under award number 1R01GM115440-01A1.

References

Bozkurt, A., Akgün-Özbek, E., & Zawacki-Richter, O. (2017). Trends and patterns in massive open online courses: Review and content analysis of research on MOOCs (2008–2015). *The International Review of Research in Open and Distributed Learning*, *18*(5), 118–147. https://dx.doi.org/10.19173/irrodl.v18i5.3080

Deng, R., Benckendorff, P., & Gannaway, D. (2019). Progress and new directions for teaching and learning in MOOCs. *Computers & Education*, *129*, 48–60. https://dx.doi.org/10.1016/j.compedu.2018.10.019

Gašević, D., Kovanović, V., Joksimović, S., & Siemens, G. (2014). Where is research on massive open online courses headed? A data analysis of the MOOC research initiative. *The International Review of Research in Open and Distributed Learning*, *15*(5), 134–176. https://dx.doi.org/10.19173/irrodl.v15i5.1954

Ho, A., Reich, J., Nesterko, S., Seaton, D., Mullaney, T., Waldo, J., & Chuang, I. (2014). HarvardX and MITx: The First Year of Open Online Courses, Fall 2012–Summer 2013. *SSRN Working Papers*. https://dx.doi.org/10.2139/ssrn.2381263

Jordan, K. (2015). Massive open online course completion rates revisited: Assessment, length and attrition. *The International Review of Research in Open and Distributed Learning*, *16*(3), 341–358. https://dx.doi.org/10.19173/irrodl.v16i3.2112

Kizilcec, R. F., Pérez-Sanagustín, M., & Maldonado, J. J. (2017). Self-regulated learning strategies predict learner behavior and goal attainment in Massive Open Online Courses. *Computers & Education*, *104*, 18–33. https://dx.doi.org/10.1016/j.compedu.2016.10.001

Lopez, G., Seaton, D. T., Ang, A., Tingley, D., & Chuang, I. (2017). Google BigQuery for education: Framework for parsing and analyzing edX MOOC data. In *Proceedings of the 4th ACM Conference on Learning @ Scale (L@S 2017),* 20–21 April 2017, Cambridge, MA, USA (pp. 181–184). New York: ACM. https://dx.doi.org/10.1145/3051457.3053980

Lu, T., Bradlow, E., & Hutchinson, W. (2017). Binge Consumption of Online Content. *Working Paper*. Retrieved from https://faculty.wharton.upenn.edu/wp-content/uploads/2017/07/JoyLu$_{J}obMarketPaper_{2}017.pdf$

Maiz Olazabalaga, I., Castaño Garrido, C., & Garay Ruiz, U. (2016). Research on MOOCs: Trends and methodologies. *Porta Linguarum*(I), 87–98.

Pardos, Z. A., & Kao, K. (2015). moocRP: An open-source analytics platform. In *Proceedings of the 2nd ACM Conference on Learning @ Scale (L@S 2015),* 14–18 March 2015, Vancouver, BC, Canada (pp. 103–110). New York: ACM. https://dx.doi.org/10.1145/2724660.2724683

Perna, L. W., Ruby, A., Boruch, R. F., Wang, N., Scull, J., Ahmad, S., & Evans, C. (2014). Moving through MOOCs: Understanding the progression of users in massive open online courses. *Educational Researcher*, *43*(9), 421–432. https://dx.doi.org/10.3102/0013189X14562423

Pursel, B. K., Zhang, L., Jablokow, K. W., Choi, G., & Velegol, D. (2016). Understanding MOOC students: Motivations and behaviours indicative of MOOC completion. *Journal of Computer Assisted Learning*, *32*(3), 202–217. https://dx.doi.org/10.1111/jcal.12131

Reich, J. (2015). Rebooting MOOC research. *Science*, *347*(6217), 34–35. https://dx.doi.org/10.1126/science.1261627

Wickham, H., François, R., Henry, L., & Müller, K. (2018). dplyr: A grammar of data manipulation, R package version 0.7.6 [Computer software manual]. Retrieved from https://CRAN.R-project.org/package=dplyr