Eindhoven, July 2016

REPORT TWO

A Method for Pre-Processing Learning Management System Log Data for Learning Analytics

W.J.M. (Wouter) Nij Bijvank
ir. M.A. (Rianne) Conijn
prof.dr. C.C.P. (Chris) Snijders
dr. U. (Uwe) Matzat
dr.ir. P.A.M. (Ad) Kleingeld

This work was supported in part by a grant from 4TU.Centre for Engineering Education.

Management summary

Data from Learning Management Systems potentially provide a wealth of information about students' online learning behaviour. Unfortunately, this information is stored in large raw log data tables that are hard to transform to actionable tables which can be used for further data analyses. Therefore, in this report we aim to provide a manual for transforming this raw LMS data in data suitable for analysis for teachers and researchers who have little background of filtering, mutating, aggregating, and transforming raw LMS data.

This report is the second one of the project "EXCTRA - EXploiting the Click-TRAil. Assessing the benefits of Learning Analytics". The main objective of the project is to figure out how Learning Analytics can be better used to predict student performance. In this report we show how the raw log data from Learning Management Systems can be translated into meaningful variables which can be used in prediction models. This transformation is also known as pre-processing data.

In this report, we provide a hands-on manual for pre-processing data, based on the pre-processing steps defined by Romero and Ventura (2007). For this method we use R, an open-source software environment. R scripts are provided for every pre-processing step. For the pre-processing data are used from Moodle LMS, with courses from Eindhoven University of Technology. These data allow us to contextualize the decisions made in pre-processing the data. As a proof of concept, the pre-processed data are used for actual analyses. This manual should allow teachers and researchers to be able to transform their raw LMS data with little own input, and only slightly more input when they want to access data from another LMS than Moodle.

Contents

1 Introduction 4
2 Process of learning analytics 5
2.1 Pre-processing data 5
3 Method 7
3.1 Data 7
3.2 Data manipulation environment 7
3.3 Requirements 8
3.4 Output 8
4 Pre-processing LMS data 10
4.1 Import raw data 10
4.2 Data exploration 12
4.3 Data cleaning 16
4.4 Transaction identification 18
4.5 Data transformation and enrichment 19
4.6 Data integration 22
4.7 Pre-processing output table 23
4.8 Summary 24
5 Predicting student performance 25
5.1 Exploring correlations 25
5.1.1 Correlation spreadsheet 25
5.1.2 Correlation bar plot 26
5.1.3 Discussion 27
5.2 Predicting student performance 31
6 Discussion 33
7 Bibliography 35

1 Introduction

Retrieving information about learners' study progress is required for a teacher to focus attention on particular students, evaluate the course setup and account for study performance towards controlling authorities (Campbell, DeBlois, & Oblinger, 2007). The standard approach for teachers is to assess learning progress of their learners through predefined learning outcomes, i.e. concepts to understand and skills to master. Various assessment methods are used on a continuous basis to measure the progress in these learning outcomes (L. W. Anderson, 2005). During a course, interactions between teacher and learners and between learners play an important role in learners reaching study outcomes (Van den Berg & Hofman, 2005). These types of interactions are typically hard to quantify because of their subjective and unstructured nature.

In the last few decades, IT has emerged more and more in education. This has resulted in the wide adoption of Learning Management Systems (LMSs). An LMS can facilitate various aspects of a course, such as the supply of study material or practice material, automated online testing, and discussion groups, to name just a few (Cole & Foster, 2007). An LMS typically keeps a log of every event that has occurred between the system and its users, which could provide new insights into how learners behave in a learning environment. Interpreting and contextualizing this information to improve learning and teaching, increasing student success, and detecting at-risk students, i.e. students who have a high chance of failure, is also known as learning analytics (Agudo-Peregrina, Iglesias-Pradas, Conde-González, & Hernández-García, 2014).

However, the data structure of the logged LMS events does not allow for a straightforward insight into learners' study activity and study progress. Data are located across several data tables and events are stored in a time-sequential structure (Psaromiligkos, Orfanidou, Kytagias, & Zafiri, 2011). Moreover, the event data logged by the LMS is not intended to measure specific behavioural concepts. It is therefore not clear what kind of information about learners' study behaviour can be deduced from the logged LMS data (Agudo-Peregrina et al., 2014).

Several studies have successfully shown that behavioural data from LMSs can be processed in such a way that it can provide useful insights for educational management and development (Romero & Ventura, 2010). Examples include visualisation of the social network in a discussion forum (e.g. Macfadyen & Dawson, 2010), classification to predict exam pass/fail probabilities (e.g. Minaei-Bidgoli & Punch, 2003), and dashboards to provide feedback to students about their learning (e.g. Arnold & Pistilli, 2012). However, a systematic approach to deriving such insights from educational data is not available yet. Moreover, evidence about what types of captured behavioural data provide useful information is still inconclusive (Macfadyen & Dawson, 2010).

Therefore, this report presents a method of systematically extracting, manipulating, and analysing LMS event data to support educational decision-making with the use of existing statistical methods and data mining techniques. A general pre-processing model to derive data suitable for analysis is discussed, including data processing scripts. Moreover, we include how research questions are used in guiding the pre-processing of the data. The model in this report is based on Moodle LMS and

supported with example transformation decisions and data analysis from a real dataset from Moodle, including 21 courses with 3,293 individual students. This is done in such a way that educational researcher and teachers with little background in dealing with raw data are able to transform the data with little own input, and only slightly more input when they want to access data from another LMS than Moodle (as most LMSs use a similar setup for their underlying databases). In this way, they do not require extensive expertise about techniques such as filtering, mutating, aggregating, and transforming LMS data to generate a dataset that is suitable for analysis. Note, we aim to show the principle of pre-processing learner data and hence we restrict the pre-processing to the data and variables used for some typical research questions. As a proof of concept, we conclude with a data analysis example on the given dataset that shows how the pre-processed data can be used for further analyses. The analysis is designed to investigate to what extent a given set of pre-processed behavioural metrics relates to final course grade for a given course.

2 Process of learning analytics

The process of learning analytics can be described in the steps that are illustrated in Figure 1. Based on a hypothesis raw data are gathered, which is pre-processed into a (modified) dataset that can be analysed. The analysis of this modified dataset yields results that can be interpreted for testing the hypothesis (Romero & Ventura, 2013). For instance, if we want to test if students' forum usage predicts their study performance (hypothesis), we need to gather data about how much students have used the forum facilities and their grades for the vocal course (raw data). Then, we combine both datasets into one dataset that is suitable for analysis (modified data). Next, the modified dataset can be analysed using statistical methods or data mining techniques. The results should allow interpretation of whether forum usage relates to study performance. In the current report we mainly focus on the pre-processing step and provide a short example of a possible data analysis as a proof of concept.

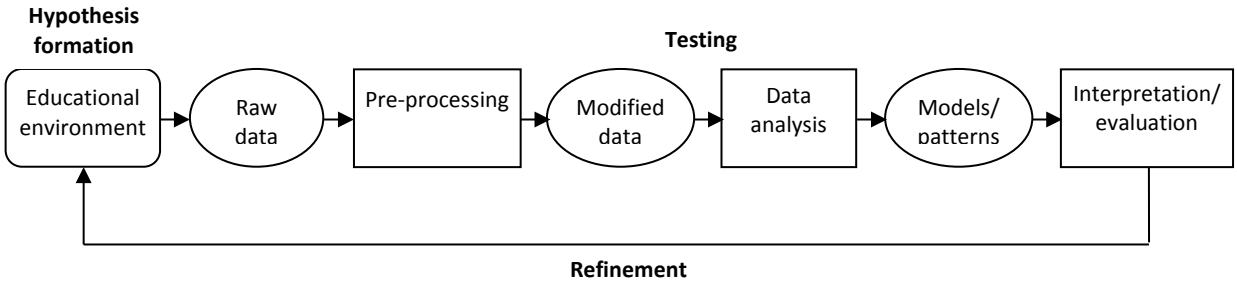


Figure 1: Educational knowledge discovery (LA) as an iterative process, adapted from Romero and Ventura (2013, p. 19)

2.1 Pre-processing data

For the pre-processing of the raw data into a new dataset, several pre-processing steps have to be addressed. In this report we follow the pre-processing steps as discussed in Romero and Ventura (2007), see also Figure 2:

- Data exploration. In this stage the raw data are investigated deeper to understand the characteristics of the LMS log entries, such as the number of courses, students, time periods, event frequencies, etc.
- Data cleaning. Data are filtered to only contain log entries of interest, based on properties such as course, time, event type, etc.
- Transaction identification. In this stage new variables may be created to identify activities in a different format, such as determining the time between activities. Also, in this step the log data can be broken down into smaller units, for example the actions per week or per online session.
- Data transformation and enrichment. An LMS log typically lists individual events based on its time of occurrence. We typically need a dataset format that lists individual users and specifies certain event occurrence frequencies and other characteristics for each user. Therefore in this stage the format of the data is changed based on a different dimension. For example, aggregated measures such as counts of occurrences are created, or numerical attributes into nominal attributes are transformed.
- Data integration. Data from different sources, such as the integration of student grades from a grade list, may need to be integrated to form a complete dataset for analysis.

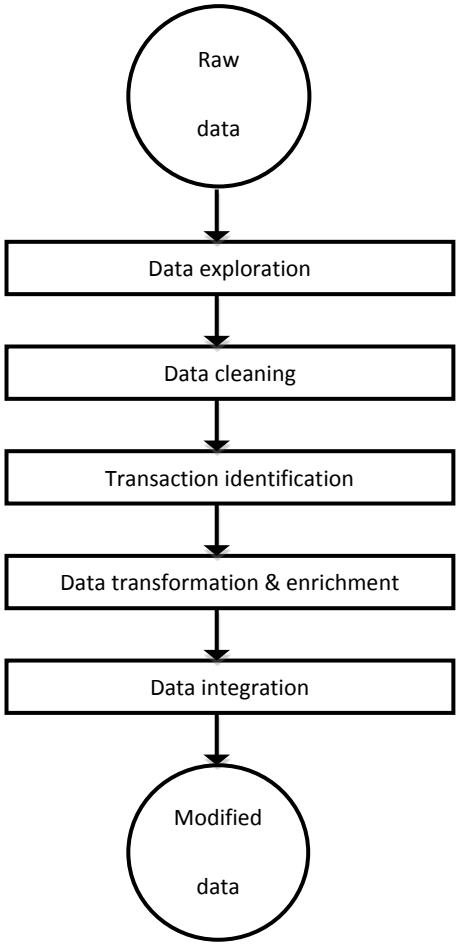


Figure 2: Pre-processing steps for our procedure (Romero & Ventura, 2007)

In summary, data pre-processing entails the gathering of raw data from several sources, filtering out the useful information, and transforming it into a single dataset with a structure that allows for analysis.

3 Method

3.1 Data

For our pre-processing method data are used from Moodle[1], an open source Learning Management System (Piña, 2012). Moodle can be downloaded free of charge, and is one of the most widely used LMSs, supporting almost ten million courses with a total of 86.7 million students in 229 different countries ("Moodle.org: Moodle Statistics," n.d.).

In the current report data are used from Moodle LMS used at Eindhoven University of Technology in the academic year 2013-2014. This includes data from 21 courses and 3,293 individual users who have shown activity in the LMS. During this period the LMS has logged 3,049,262 entries, all of which correspond to actions by users. From these log entries a selection of courses and types of actions are used for further pre-processing. Student grades, currently typically not stored in the LMS, have been collected and merged to the data set.

3.2 Data manipulation environment

To pre-process our data, we use R, a free software environment for statistical computing and graphics[2] (Ihaka & Gentleman, 1996). R is more extensive compared to other statistical packages such as Stata, SPSS, and Excel and can more easily and quick handle large datasets. The datasets often consist of multiple data tables, which is easier to handle in R. Moreover, the raw data is often too large to be fully loaded into other statistical packages.

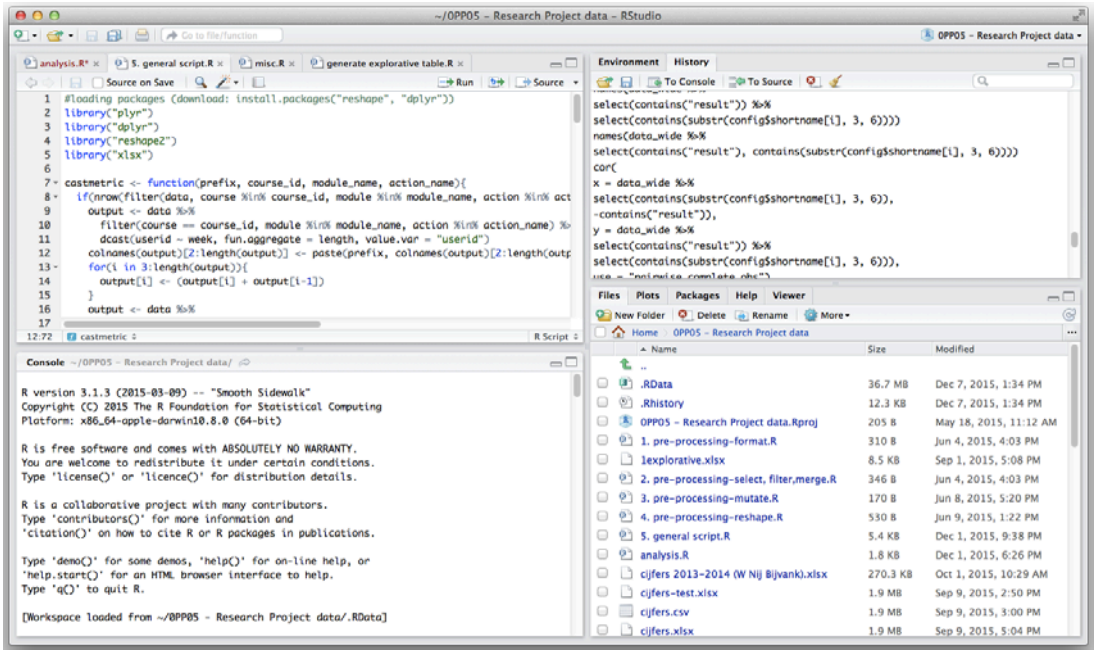


Figure 3: User interface of data manipulation and data analysis tool RStudio

Next to R we use the integrated development environment (IDE) for R: RStudio. RStudio is a user interface for R which provides additional features that improve R's usability and productivity (Racine,

[1] http://moodle.org
[2] https://www.r-project.org/

2012). A screenshot of RStudio is shown in Figure 3. To use RStudio, R needs to be downloaded as well.

R uses many packages provided by users for specific functions. In our data-pre-processing script we use three R packages: the 'xlsx' package[3] (Dragulescu, 2014) to import from and export to Microsoft Excel files and two popular data manipulation packages, 'dplyr'[4] (Wickham & Francois, 2015) and 'reshape2'[5] (Wickham, 2014). These packages can be installed directly in R with the code: `install.packages("reshape2", "dplyr", "xlsx")` or downloaded from the websites. The websites of these packages provide detailed information on how to use these packages.

3.3 Requirements

In order to create a pre-processing method that can support the analysis of a variety of research questions of learning analytics studies, we have applied the following requirements:

- Given that a large portion of learning analytics studies concerns the goal of creating a model for predicting student performance, the output dataset must contain individual study results in terms of preliminary and final grades.
- A number of learning analytics studies attempt to create intervention strategies, i.e. warn students about their predicted study results, based on the online behaviour they have shown that far, to positively influence their study behaviour. Analyses in such studies require behavioural data from given advancements of a course, for example the number of page views after the second and after the third academic week.
- Learning analytics studies have shown a great variance in effects that have been found amongst behavioural metrics depending on the focal course. The resulting dataset must therefore contain specific behavioural data for each individual course, which should enable modelling and comparison of individual courses.
- Because of the great variance in effects amongst behavioural metrics, the output dataset must contain specific variables for every individual type of behavioural metric, such as a specific variable for number of course views and a specific variable for forum discussion views.
- To prevent the pre-processing stage from being considered a complex and technical exercise, a certain degree of automation must be introduced. Data manipulation itself will therefore take place according to pre-defined scripts.

3.4 Output

Figure 4 illustrates the conversion of the data structure in pre-processing the data that we have based on the abovementioned requirements. It also shows a suggested analysis result of a linear regression model. The raw data we start with is the Moodle log table, in which entries have been added sequentially for every event that a user triggered with the system, which represent learners' behavioural data. For every entry attributes are stored that provide more information about the

[3] https://cran.r-project.org/web/packages/xlsx/index.html
[4] https://cran.r-project.org/web/packages/dplyr/index.html
[5] https://cran.r-project.org/web/packages/reshape2/index.html

logged event, such as the user who triggered the event, course in which the event is triggered, time at which the event is triggered, and other variables associated with the event.

The result of the pre-processing stage of the procedure we propose is a new table that is only based on a selection of these events, and where data about the events is now structured in a user-based sequence: every entry is a unique LMS user. For every user there are a number of processed variables, which indicate a metric of certain behaviour, with specific variables for each course, each type of behaviour and each time period within the progression of a course. Moreover, a new variable is created for each course grade that applies to courses within the output dataset.

The output of the pre-processing can in turn be used for statistical analysis, for example in R. As the output consists of only one table with a selection of events, the size of the dataset has decreased to a large extent and consequently the statistical analyses can also be done with Stata, SPSS, or any other general statistical package

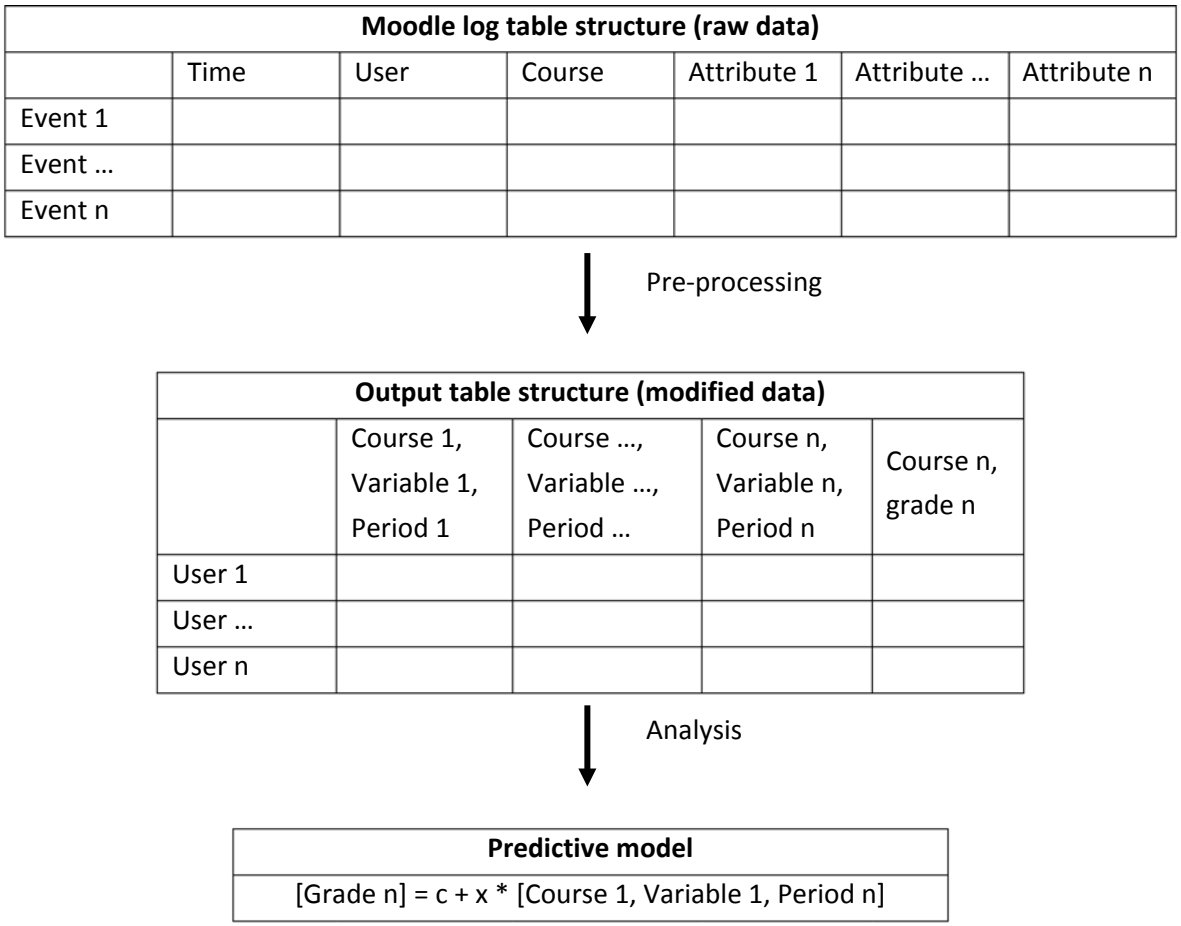


Figure 4: Data structure of the raw data, the modified data, and the analysis result

4 Pre-processing LMS data

The following chapter describes the pre-processing steps as discussed in Romero and Ventura (2007) for the LMS data (see Figure 2). For each step, we define the data conversion and show what the input data looks like at the beginning of this step, and what the output data looks like at the end of this step (which is in turn the input data for the next step).

4.1 Import raw data

Data conversion in this stage:

Input: raw data located at their original sources

Output: raw data imported into RStudio

In this report, we use data from Moodle LMS. Moodle LMS, like any typical web system, stores its data in a Relational Database Management System (RBDMS) (Sumathi & Esakkirajan, 2007). In order to import the data straight from a Moodle database into the RStudio, an R add-on package called 'RODBC' can be used. Instructions on how to import the database tables into RStudio are provided on the webpage[6] (Ripley & Lapsley, 2015).

The most important raw data table is the log table, where all events of all users in all the courses are sequentially listed. The log data table is typically rather large. For example, in our case we have 145,202 events (rows) per course (with 150 students per course on average). In Moodle LMS, data coordinators and teachers have access to this log table, called mdl_log. Data coordinators also have access to tables with additional information about the course, such as course name, content of discussion posts, answer options in a quiz, etc. At this moment, the Moodle database has around 200 tables.

In our pre-processing method we use two Moodle tables. Hence, to replicate the pre-processing as outlined below, one needs access to the following Moodle tables:

- mdl_log: the main log file in which Moodle stores all the actions that have been performed by the system's users as units of events.
- mdl_course: the Moodle table that describes the courses that are facilitated. The table is used to translate the course identifier that is used in the LMS to the institutional course short name and full name.

A number of variables (columns) from these two tables are of interest. The meaning of the variables and their purpose for this procedure are explained in Table 1 and Table 2.

As our aim is to show the principle of the data pre-processing, we only focus on these two tables. There are multiple other tables in Moodle which can for example provide information about the content in the discussion forum, or the answer option a student chose in a quiz. However, for most research questions the mdl_log table provides enough data. If you want to add other tables, this can

[6] https://cran.r-project.org/web/packages/RODBC/index.html

be done in the data exploration step, in the same manner as the mdl_course table is merged with the mdl_log table (lines 15-18, Figure 6).

Table 1: Description of the mdl_log table variables of interest in terms of their meaning and purpose for the current pre-processing procedure

Variable	Meaning	Purpose
time	Timestamp of the occurrence of the event	Any time-related mutation of data is based on this value
userid	The identifier, that indicates each individual user of the Moodle LMS, that initiated the event	The dataset is pre-processed to a format in which each 'item' represents a Moodle user
ip	Computer host IP address from which the event has been initiated	Indicates the approximate geographic location of the users' actions
course	The course, identified with the internal identifier, in which a user initiated the event	Enables course-specific data manipulation
module	Indicates the module in which an action has been initiated. Moodle has a modular functional structure, in which each module encompasses a certain LMS feature, such as "forum", "quiz" or "course"	Enables selecting certain types of events as the activities of interest
action	Describes the type of action that took place within the given module, such as "view", "add" or "enrol"	Similar to "module", it enables selecting certain types of events as the activities of interest

Table 2: Description of the mdl_course table variables of interest in terms of their meaning and purpose for the current pre-processing procedure

Variable	Meaning	Purpose
id	The LMS' internal course identifier, coupled with the mdl_log "course" variable	Reference to course for the other variables in this table
fullname	The full course name that is used within the institution	Disambiguating the LMS' internal course ID
shortname	The course code that is used within institution	Similar to "fullname", it disambiguates the LMS' internal course ID

Teachers often only have access to the log of all activities that take place in the courses they teach. In Moodle, these logs can be accessed and exported via the administration menu into separate mdl_log tables per course. As teachers have access to the course page themselves as well, the mdl_log table is often enough for the data analyses, as they can just look-up the additional data needed on the course page (such as full course name). When you only have access to the mdl_log table (for each course), you can use a simplified version of the pre-processing script, and run it per course. As you

only have one course offering per mdl_log table, you can easily identify the course and you do not need the information in the mdl_course table.

In this pre-processing procedure we also include the study results for each course and each student of interest. In this particular Moodle setup, grading registration was organised outside of the LMS. Therefore a separate grading table is imported to complete the raw dataset. One important prerequisite for importing the grading data separately is the reference from student grade to the 'id' that is utilized in the mdl_log table for identifying each student.

```
1  #loading (and installing) packages needed for whole script
2  if(!require(dplyr)){install.packages('dplyr')}
3    library("dplyr")
4  if(!require(reshape2)){install.packages('reshape2')}
5    library("reshape2")
6
7  #load Moodle tables
8  mdl_log    <- read.table("mdl_log.csv", header=TRUE, sep=",")
9  mdl_course <- read.table("mdl_course.csv", header=TRUE, sep=",")
10
11 #load grade file
12 results    <- read.table("grades.csv", header=TRUE, sep=",")
```

Figure 5: Pre-processing script section for initialization

Figure 5 shows the R script which handles the initialization needed for the pre-processing. First, it installs (if needed) and loads two frequently used data manipulation packages (lines 2-5), which will be used further on in the script. Thereafter, the necessary Moodle and grade data tables are loaded into R. Here, these files were in csv format. Several other data types can be imported in R as well. Mostly a simple search "Import [data type] R" will result in the relevant R package and code to import a file from the specified data type. For example, a grade list in an *.xlsx data format (Microsoft Excel 2007 and higher) can be enabled in R by the use of the 'xlsx' package (Dragulescu, 2014).

The mdl_log and mdl_course tables from Moodle and the grading table together form the raw data that are required for pre-processing the data into a dataset that can meet the requirements of our analysis.

4.2 Data exploration

Data conversion in this stage:

Input: raw data imported into RStudio

Output: descriptive results for our raw data

Before the raw data are processed we first need to filter events from the mdl_log table that are of interest for further processing. To make a decision on which data to extract we first need to explore the contents of the mdl_log table.

We want to get insights into the following aspects of the data:

- Course: a selection must be made for data of Moodle courses that is subjected to further processing.
- Time: given the period of time a course took place, actions of users outside of this period are discarded.
- Type of user action: a user action represents a type of learner behaviour, such as viewing a page or posting a forum message. We are interested in specific types of learning behaviour and how specific types of behaviour relate to the study results. Other types of user actions such as course modules created, user reports viewed, and enrolments deleted are discarded.

The R script in Figure 6 generates a table with some descriptive parameters from an inputted mdl_log table and its corresponding mdl_course table (see Table 1 and Table 2).

```
1  #generate summary statistics for each course
2  explorative <- mdl_log %>%
3    mutate(timestamp =%>%
4    group_by(courseid) %>%
5    summarise(
6      n_user = length(unique(userid)),
7      time_min = min(timestamp),
8      time_q_low = quantile(timestamp, c(1/4)),
9      time_med = median(timestamp),
10     time_q_up = quantile(timestamp, c(3/4)),
11     time_max = max(timestamp)
12     )
13
14 #add course names
15 explorative <- explorative %>%
16   merge(mdl_course, by.x = "courseid", by.y = "id") %>%
17   select(courseid, shortname, fullname, n_user, time_min,
18        time_q_low, time_med, time_q_up, time_max)
19
20 #export table to csv file
21 write.csv(lapply(explorative, as.character),
22   file = "explorative_out.csv",
23   row.names = FALSE)
```

Figure 6: R script that generates table with descriptive parameters about given mdl_log dataset

First, the *time* variable from the mdl_log tables needs to be converted to an R-compatible format (line 3). Next, a series of boxplot descriptive parameters is generated for each course (lines 5-12). After this the “fullname” and “shortname” variables from the mdl_course table are merged with the generated dataset to clarify which course is referred to with each row (line 15-18). Table 3 shows the resulting table that has been outputted by the above R script, based on our raw Moodle dataset. In our case, not all rows represent real courses, also users are enrolled in other groups, such as “oncourse’13” which provides the general login and logout in Moodle, “manuals”, and “test”. Based on the researcher’s question the reader might choose to remove these extra groups from the analyses. Table 4 describes the meaning of each of the variables related to the courses.

Table 3: Descriptive parameters generated by R script, based on our Moodle LMS data

course	shortname	fullname	n_user	time_min	time_q_low	time_med	time_q_up	time_max
1	oncourse'13	Oncourse	3293	2013-08-05 17:01:57	2013-09-17 13:47:52	2013-10-04 08:26:44	2013-10-25 09:21:34	2015-04-10 12:33:02
2	2WAB0	Calculus 2WAB0	1379	2013-08-06 10:08:46	2013-09-19 18:58:51	2013-10-03 15:36:17	2013-10-17 16:05:55	2015-03-01 11:46:45
4	2WBB0	Calculus 2WBB0	1621	2013-08-08 12:00:43	2013-09-22 13:02:37	2013-10-06 13:44:47	2013-10-17 19:18:00	2015-03-30 17:28:18
5	test	Test course	5	2013-08-14 11:42:51	2013-11-19 14:47:21	2014-03-24 16:02:08	2014-04-08 08:40:18	2014-05-06 14:26:27
6	3A1X0	Experimentele Fysica 1	162	2013-08-14 14:49:15	2013-09-10 18:01:31	2013-09-19 21:19:34	2013-09-29 21:20:20	2015-04-10 12:32:28
7	2DL06	Linear Algebra (2DL06)-2013-Q1	129	2013-08-14 14:57:15	2013-09-17 20:34:39	2013-10-01 07:02:34	2013-10-15 10:00:38	2014-02-06 09:32:26
8	2WF40	Set Theory and Algebra (2WF40)	208	2013-08-15 07:11:00	2013-09-18 16:38:42	2013-10-03 20:58:46	2013-10-13 20:52:49	2015-02-10 11:59:37
9	2DL00	Basiswiskunde (2DL00)	150	2013-08-15 10:20:31	2014-04-06 10:00:47	2014-05-13 20:08:54	2014-06-19 19:05:43	2015-03-11 11:58:23
10	2DB03	Calculus voor het schakel-programma van Bouwkunde	203	2013-08-19 13:03:01	2013-09-30 15:06:45	2013-10-20 18:13:45	2013-11-02 21:47:10	2015-03-16 22:02:46
11	2DD40	Wiskunde 1 (2DD40)	298	2013-08-16 16:26:04	2013-11-30 17:38:59	2013-12-29 14:08:39	2014-01-04 14:34:41	2015-01-26 15:05:09
12	Manuals	Manuals	2192	2013-08-19 15:17:58	2013-09-10 19:57:08	2013-09-19 16:06:55	2013-10-17 13:34:28	2015-02-24 18:56:21
13	Contact	Contact	1018	2013-08-19 15:07:26	2013-09-11 20:50:02	2013-09-26 19:37:05	2013-10-24 14:37:29	2015-03-03 16:34:52
14	2WCB0	Calculus 2WCB0	548	2013-08-22 10:46:52	2013-09-16 19:07:29	2013-10-01 09:01:05	2013-10-16 17:31:21	2015-01-02 16:26:26
15	2WF20	Linear Algebra (2WF20)	86	2013-08-22 13:28:15	2013-10-01 11:11:02	2013-10-13 18:24:50	2013-10-23 20:13:21	2014-10-04 17:05:48
16	2DE07	Discrete Mathematics (2DE07)	44	2013-08-23 13:15:15	2013-09-24 14:29:12	2013-10-26 12:20:43	2013-10-31 14:36:34	2014-10-30 14:14:08
17	3A2X0	Experimentele Fysica 2	148	2013-09-13 13:02:19	2013-11-18 09:51:47	2013-12-01 10:37:59	2013-12-14 21:32:26	2014-11-13 08:44:46
18	e-test	Entrance Test	964	2013-09-13 13:58:17	2013-10-08 14:58:59	2013-10-24 07:17:25	2014-01-16 18:57:10	2015-02-02 11:47:29
19	2DN60	Lineaire Algebra en Vectorcalculus (2DN60)	65	2013-10-07 15:03:32	2013-12-30 15:45:21	2014-01-12 17:11:38	2014-01-25 15:35:41	2015-04-10 12:34:15
20	3BOX0	Optica (3BOX0)	77	2013-10-09 09:43:46	2013-12-08 13:34:51	2013-12-09 23:43:38	2013-12-22 17:01:36	2014-08-20 14:49:19
21	3B3X0	Experimentele Fysica 3	76	2014-01-09 20:12:31	2014-04-24 19:04:03	2014-05-09 08:59:23	2014-05-22 14:24:39	2015-01-13 09:06:31
22	2WF05	Algebra and Geometry (2WF05)	10	2014-01-17 07:41:14	2014-02-10 09:04:57	2014-02-27 15:51:45	2014-04-15 10:44:18	2015-03-23 14:36:54
23	2DL06	Linear Algebra (2DL06)	72	2014-02-03 10:57:47	2014-02-19 20:58:19	2014-03-12 13:47:46	2014-03-26 12:05:31	2015-02-12 13:38:22
24	3EEX0	Elektrodynamica (3EEX0)	20	2014-03-13 12:06:04	2014-03-22 13:34:08	2014-03-24 11:06:03	2014-03-30 15:41:40	2015-02-24 07:29:10
25	Logic	Logic	9	2014-04-10 08:50:31	2014-04-11 16:04:45	2014-04-22 11:51:30	2014-05-13 09:50:44	2014-12-17 13:02:30
26	stat	Statistics	41	2014-05-16 11:27:20	2014-05-19 00:29:53	2014-05-20 08:50:17	2014-05-20 09:32:15	2015-04-10 12:31:42
27	3NBB0	Toegepaste natuurwetenschappen	6	2014-05-19 11:53:25	2014-05-21 07:44:05	2014-05-22 13:57:18	2014-06-03 06:53:00	2014-09-03 13:58:41

Table 4: Description of the variables of the table with descriptive parameters

Variable	Description
course	Course identifier that is internally used in the Moodle LMS, corresponding to the *course* variable in the mdl_log table.
shortname	The institutional course code originating from the mdl_course table.
fullname	The institutional course full name originating from the mdl_course table.
n_user	The number of users that have shown activity for the specific course.
time_min	The earliest moment in time that an event has been logged for the specific course.
time_q_low	The lower quartile value of the distribution of events over time for the specific course.
time_med	The median value of the distribution of events over time for the specific course.
time_q_up	The upper quartile value of the distribution of events over time for the specific course.
time_max	The latest moment in time that an event has been logged for the specific course.

More insights into the data, aside from consulting the generated table with descriptive parameters, are needed. Depending on the research question it can, for instance, be needed to know which LMS functionality is applied for each course, for example whether a course used forum functionality.

The script shown in Figure 7 returns a table that shows the frequency per course of clicks in all modules in the mdl_log file. Line 3 can be replaced with another variable from the mdl_log file to get an overview of for example the actions per course (for the meaning of module and action variables see Table 1). An excerpt of the resulting output is shown in Table 5. In our case, a total of 25 Moodle modules were used.

```
1 #create list of frequency of modules, for each course
2 mdl_log %>%
3   group_by(courseid, components) %>%
4   summarise (n = n()) %>%
5   arrange(desc(n)) %>%
6   View()
```

Figure 7: R script that outputs a table that list the frequency of clicks in a module, for each course.

Table 5: Excerpt the output table of the R script from Figure 7, based on Moodle LMS data

Courseid	Component (= module)	n
…		
17	mod_quiz	34104
17	core	11205
17	mod_page	6221
17	mod_wiki	3097
17	mod_resource	2115
17	mod_forum	1844
…		

By means of exploring the data as introduced, a selection of the data for further processing (data cleaning) and analyses can be made for a variety of research questions. For each course that is logged in the Moodle mdl_log file we now know:

- The institutional name of the course with the shortname and fullname variables in the explorative table.
- When user activity took place: based on the time_[…] variables in the explorative table.
- How many users showed activity: with the "n_user" variables in the explorative table.
- The frequency and types of events that took place, based on the use of the module and action variables from the mdl_log file in the R script, which returns types and frequencies of occurrence (Figure 7).

With these insights into our raw data we can decide on what data to keep for further processing and this completes our data exploration stage.

The decision on the selection of these types of events is made on the basis of our literature review (Conijn, Snijders, Matzat, & Kleingeld, 2016). We have selected the types of events that have shown significant correlations with final course grade in the literature. The selected types of events are not exhaustive, but given the demonstrative purposes of this procedure, we limited the number of types for further pre-processing to the ones in Table 6.

Table 6: "module"-"action" - "target" combinations of events that are filtered from the raw mdl_log data

module	action	target	User action
core	viewed	course	A course page is visited
mod_url	viewed	course_module	An external page is visited, originating from the LMS
mod_forum	created	post	A post is added to an existing discussion
	created	discussion	A new discussion is added
	viewed	discussion	A discussion is viewed
mod_quiz	started	attempt	An new quiz attempt is made

We decided to keep data about these events for seven courses (see Table 7). These seven courses have been selected because they have the highest number of students enrolled and they have at least some of the filtered types of LMS events (Table 6).

4.3 Data cleaning

Data conversion in this stage:

Input: raw data imported into RStudio with descriptive results about our raw data

Output: filtered raw data

We can now start the actual pre-processing procedure, starting with the data cleaning (Romero & Ventura, 2007, pp. 139–140). Figure 8 shows the section of the pre-processing script that handles the data filtering in terms of selection of courses, time periods, and module-action types of events. The data filtering process is controlled through an imported csv file, which is a filtered version of the table with descriptive parameters (Table 3). This filtered file only includes the courses we selected for analyses and the "time_min" and "time_max" values for every course. In this way, we only selected data from 1 week before the course started until 1 week after the course was finished. Our version of the edited table of descriptive parameters is shown in Table 7.

```
#import configuration file
config <- read.table("G:/explorative_in.csv", header=TRUE, sep=",")
config$shortname <- paste("c", config$shortname, sep = "_")
config$time_min <- as.numeric(as.POSIXct(config$time_min, origin='1970-
01-01', tz = "UTC"))
config$time_max <- as.numeric(as.POSIXct(config$time_max, origin='1970-
01-01', tz = "UTC"))

#for each selected course, filter events within specified time period
for(i in 1:nrow(config)){
  data = rbind(data.frame(data),
               filter(mdl_log,
                      courseid == config$course[i],
                      timecreated >= (config$time_min[i]),
                      timecreated <= (config$time_max[i])
               )
  )
}
#filter for specified course-action metrics
data <- data[
((data$component == "core"      &   data$action == "viewed"  &
data$target == "course") |
 (data$component == "mod_url"   &   data$action == "viewed"  &
data$target == "course_module") |
 (data$component == "mod_forum" & ((data$action == "created" &
data$target == "post") |
 (data$action == "created" & data$target == "discussion") |
 (data$action == "viewed"  & data$target == "discussion"))) |
 (data$component == "mod_quiz"  &   data$action == "started" &
data$target == "attempt") )
,]
```

Figure 8: Pre-processing script section that handles data filtering

Table 7: Configuration table that is used in the pre-processing script for data selection

course	shortname	fullname	n_user	time_min	time_q_low	time_med	time_q_up	time_max
2	2WAB0	Calculus 2WAB0	1379	2013-08-20 00:00:00	2013-09-19 18:58:51	2013-10-03 15:36:17	2013-10-17 16:05:55	2013-11-10 00:00:00
4	2WBB0	Calculus 2WBB0	1621	2013-08-20 00:00:00	2013-09-22 13:02:37	2013-10-06 13:44:47	2013-10-17 19:18:00	2013-11-10 00:00:00
6	3A1X0	Experimentele Fysica 1	162	2013-08-20 00:00:00	2013-09-10 18:01:31	2013-09-19 21:19:34	2013-09-29 21:20:20	2013-11-10 00:00:00
14	2WCB0	Calculus 2WCB0	548	2013-08-20 00:00:00	2013-09-16 19:07:29	2013-10-01 09:01:05	2013-10-16 17:31:21	2013-11-10 00:00:00
17	3A2X0	Experimentele Fysica 2	148	2013-10-28 00:00:00	2013-11-18 09:51:47	2013-12-01 10:37:59	2013-12-14 21:32:26	2014-02-01 00:00:00
20	3BOX0	Optica (3BOX0)	77	2013-10-28 00:00:00	2013-12-08 13:34:51	2013-12-09 23:43:38	2013-12-22 17:01:36	2014-02-01 00:00:00
21	3B3X0	Experimentele Fysica 3	76	2014-04-08 00:00:00	2014-04-24 19:04:03	2014-05-09 08:59:23	2014-05-22 14:24:39	2014-07-02 00:00:00

In other words, we first generated a table that contained an overview of all courses and their descriptive parameters (Table 3); we now edit that table to only contain courses and their event time

range that we want to keep after filtering (Table 7) and feed this edited version into the R script to instruct the filtering process.

After the import of the configuration file (Figure 8, lines 2-5) data from the mdl_log raw dataset are selected (lines 8-16) based on the courses and time periods that are defined in the configuration table (Table 7). After this the data are filtered to only contain specific types of events (lines 18-26). Events are selected to contain combinations of module and action values that resemble types of user actions. Table 6 gives an overview of what kinds of events are filtered and what kinds of user actions these events resemble. The filtering of these types of events is 'hard-coded' in the pre-processing script but can be edited based on the preferred types of events. The filtering process is performed using the 'dplyr' R package (Wickham & Francois, 2015).

The result of this filtering stage is a dataset with the same structure as the raw mdl_log dataset, but only containing events that occur within the specified courses, time range, and types of user actions.

4.4 Transaction identification

Data conversion in this stage:

Input: filtered raw data

Output: filtered raw data with additional generated variables

Before our data can be transformed into the desirable format we first create new variables based on our current structure because this is less complex than to generate them during transformation. Based on the timestamp for each event we want to know the time between every user event. This 'inactivity' can provide information about study regularity and has been considered to be of predictive value for study performance (Yu & Jo, 2014) and is therefore included into the pre-processing output data.

In our pre-processing script we incorporate two ways of using information about the occurrence of events. First, we generate a new variable "inactive" in our dataset that, for each individual event, indicates the time that passed since the last time an event occurred for that specific user in that specific course (Figure 9, lines 2-4). Second, based on the time variable a new variable "week" is generated that indicates the week number in which the each event occurred (lines 7-8). This variable is used later on to group events in week periods for data transformation.

```
1 #calculate time between actions
2 data <-arrange(data, courseid, userid, timecreated)
3 data$inactive[2:nrow(data)] <- as.numeric(diff(data$timecreated))
4 data$inactive[data$inactive < 0] <- NA
5
6 #generate week number for column casting
7 data$timecreated <- as.POSIXct(data$timecreated, origin='1970-01-01',
  tz = "UTC")
8 data <- mutate(data, week = strftime(data$timecreated, format="%W"))
```

Figure 9: Pre-processing script section that computes time between actions and the week number

4.5 Data transformation and enrichment

Data conversion in this stage:

Input: filtered raw data with additional generated variables

Output: transformed dataset with additional aggregated variables

With the data filtered and additional variables computed we can start 'casting' the data into a new format, according to Figure 4. For the log data from the LMS, the mdl_log table, to be analysable we need to transform the data table structure into a different cross section. The mdl_log table currently has a 'long' format, in which new LMS events, or 'observations', are added as rows in the table, accompanied by attributes that describe the event. We want to create a table in a 'wide' format, which has rows that represent individual users, and columns that represent metrics about the users' behaviour, specific for each course, each type of processed user action, and each time period. The principle of this format transformation and the usage of the R package are described by Anderson (2013). For this transformation we use the R package 'reshape2' (Wickham, 2014).

```
#generate table with columns for periodic activity metric summaries
castmetric <- function(prefix, course_id, component_name, action_name,
target_name){
  if(nrow(filter(data,
                 courseid  == course_id,
                 component == component_name,
                 action    == action_name,
                 target    == target_name)) > 0){
    output <- data %>%
      filter(courseid  == course_id,
             component == component_name,
             action    == action_name,
             target    == target_name) %>%
      dcast(userid ~ week, fun.aggregate = length, value.var =
      "userid")
    colnames(output)[2:length(output)] <- paste(prefix,
                                  colnames(output)[2:length(output)],
                                  sep = "_")
    total = rowSums(output)
    activity = sum(total > 0)
    output[(length(output)+1)] <- total
    names(output)[length(output)]<- paste(prefix, "users", activity,
    sep = "_")

    return(output)
  } else{
    output <- data.frame(NA)
    colnames(output) <- paste(prefix, "no_data", sep = "_")
    return(output)
  }
}
```

Figure 10: The function "castmetric" in the R pre-processing script that casts a specific type of event into the 'wide' format

For casting the data into the wide format, the function "castmetric" is used. This function calculates the weekly activity metrics for each action chosen in the data cleaning step. First, the script checks for the defined course whether the action was present in the course (3-7). If so, the clicks (related to

this action) will be casted using the “dcast” function from R package ‘reshape2’. Here a new table is created with a userid for each row and columns for the counts of actions per every week (line 13). Lines 14-16 change column names into names that indicate the course short name, type of event, and week number. Lines 17-20 add a column for the total number of clicks for the specified action per user, across all filtered weeks. This column name includes the number of users that have shown activity for that specific type of action. If the specified action is not present in the course only one column is created with the values coded as missing (‘NA’) (lines 24-26).

Table 8: Summary parameters that are calculated for each course and each user

Variable	Description	Method
rows	Total number of user events.	Observations counter “n()” from the “dplyr” package.
campus	The ratio of user activity performed on the university campus or university VPN compared to the total user activity.	Divide the count of occurrences in the “ip” variable starting with “131.155” (the university’s network) by the observations counter “n()”.
inactive_q_low	The lower quartile value of the distribution of the generated “inactive” variable.	Using the “quantile” R function
inactive_q_low	The upper quartile value of the distribution of the generated “inactive” variable.	
inactive_med	The median of the distribution of the generated “inactive” variable.	Using the “median” R function
inactive_mean	The mean of the distribution of the generated “inactive” variable.	Using the “mean” R function
inactive_max	The maximum value of the distribution of the generated “inactive” variable.	Using the “max” R function
inactive_var	The variance of the distribution of the generated “inactive” variable.	Using the “var” R function

In addition to casting existing event information into a wide format we also calculate a number of summary statistics. For each user and each course the summary parameters are calculated, according to Table 8. The calculation of these statistics, including the use of transformation to the wide format using the castmetric function, is shown in done in the R script shown in Figure 11. The script loops for every course that has been defined in the configuration table (Table 7). Coding lines 5-16 illustrate the computation of the summary parameters based on the “inactivity” variable that was created in the previous section, as shown in Table 8. Lines 17-25 rename these variables into course-specific names for usability, based on the “shortname” variable from the configuration table (Table 7). These variables are temporarily stored in the “descriptive” table. Lines 26-32 initiate the casting of the types of user events into the individual-columns wide-format. Each of these lines is casting a specific type of event; the events we have chosen in the data filtering step (Table 6). The casting takes place in the castmetric function (Figure 10).

```r
#generate summary statistics for each user of each course
for(i in 1:nrow(config)){
  descriptive <- data %>%
    filter(courseid == config$course[i]) %>%
    group_by(userid) %>%
    summarize(
      rows = n(),
      campus = sum(grepl('131.155', ip)) / n(),
      q_low = quantile(inactive, c(1/4), na.rm = TRUE),
      q_up = quantile(inactive, c(3/4), na.rm = TRUE),
      med = median(inactive, na.rm = TRUE),
      mean = mean(inactive, na.rm = TRUE),
      max = max(inactive, na.rm = TRUE),
      var = var(inactive, na.rm = TRUE))
  names(descriptive) <- c("userid",
    paste(config$shortname[i], "rows", sep = "_"),
    paste(config$shortname[i], "campus",
    round(mean(descriptive$campus, na.rm=TRUE), digits=4), sep = "_"),
    paste(config$shortname[i], "inactive_q_low", sep = "_"),
    paste(config$shortname[i], "inactive_q_up", sep = "_"),
    paste(config$shortname[i], "inactive_med", sep = "_"),
    paste(config$shortname[i], "inactive_mean", sep = "_"),
    paste(config$shortname[i], "inactive_max", sep = "_"),
    paste(config$shortname[i], "inactive_var", sep = "_"))

  #generate weekly actions using castmetric
  actions <- castmetric(paste(config$shortname[i], "course_view", sep
  = "_"), config$course[i], "core", "viewed", "course")
  actions <- merge(actions, castmetric(paste(config$shortname[i],
  "course_view",     sep = "_"), config$course[i], "core",
  "viewed" , "course"), all = TRUE)
  actions <- merge(actions, castmetric(paste(config$shortname[i],
  "url_view",        sep = "_"), config$course[i], "mod_url",
  "viewed" , "course_module"), all = TRUE)
  actions <- merge(actions, castmetric(paste(config$shortname[i],
  "add_post",        sep = "_"), config$course[i], "mod_forum",
  "created", "post"), all = TRUE)
  actions <- merge(actions, castmetric(paste(config$shortname[i],
  "add_discussion",  sep = "_"), config$course[i], "mod_forum",
  "created", "discussion"), all = TRUE)
  actions <- merge(actions, castmetric(paste(config$shortname[i],
  "discussion_view", sep = "_"), config$course[i], "mod_forum",
  "viewed" , "discussion"), all = TRUE)
  actions <- merge(actions, castmetric(paste(config$shortname[i],
  "quiz_attempt",    sep = "_"), config$course[i], "mod_quiz",
  "started", "quiz"), all = TRUE)
  actions[is.na(actions)] <- 0
  # merge actions with descriptives
  if(i == 1){
    data_wide <- merge(descriptive, actions)
  } else{
    data_wide <- merge(data_wide, merge(descriptive, actions), all =
    TRUE)
  }
}
```

Figure 11: The main R pre-processing script section for data transformation and data enrichment

With the original dataset casted into the new format and the additional descriptive variables added we have now finished the data manipulation, at least as far as the Moodle tables are concerned.

4.6 Data integration

Data conversion in this stage:

Input: transformed dataset with additional aggregated variables

Output: transformed dataset with additional aggregated variables and grading variables

The output from the pre-processing procedure should support analyses that use the study result as a dependent variable. Therefore the grading table that is imported as a raw data source must be integrated with the table that contains the independent variables: the processed user actions as described in the previous sections. To merge the grading table with the data table, a unique student identifier is required which is present in both tables. In our case this is the userid. The format of the raw grading data is assumed to be 'long', in which every row contains a single grade. Hence, the grading table needs to be transformed to the wide format (as the data table is in wide format as well). Figure 12 illustrates the transformation of the grading table into the format that is used for data integration.

```
1  #prepare grade list (rename column names)
2  names(results)[names(results) =="cijferwaarde"] <- "grade"
3  names(results)[names(results) =="vakcode"] <- "course"
4  names(results)[names(results) =="datum_resultaat"] <- "grade_submit"
5  names(results)[names(results) =="studiejaar"] <- "course_year"
6  names(results)[names(results) =="idmoodle"] <- "userid"
7  #filter results per user per grade with the earliest grade submit
8  results <- filter(results,
9                    course %in% (substr(config$shortname, 3, 7)),
10                   course_year == "2013") %>%
11   group_by(userid, course) %>%
12   filter(grade_submit == min(grade_submit)) %>%
13 mutate(grade = as.numeric(grade))
14
15 #generate results table in wide format with each row a user and each
   column a grade for a course
16 results_wide <- (dcast(results, userid ~ course, value.var = "grade",
   mean))
17 colnames(results_wide)[2:length(results_wide)] <- paste("c",
18                       colnames(results_wide[2:length(results_wide)]),
19                       "grade", sep = "_")
20 results_wide[is.na(results_wide)] <- NA
21
22 #merge results_wide table with data_wide table
23 data_merged <- merge(data_wide, results_wide, all=TRUE)
```

Figure 12: The R script pre-processing section that handles the transformation of the raw grading data source into the format that is used for integrating the study results

First, the necessary columns of the grade table are renamed (lines 2-6). The grades are filtered to only contain grades from the courses of the configuration table (Table 7) and only from academic year 2013 (the year of our interest) (lines 8-13). After this the processed grades table is transformed into the 'wide' format, in which each row is a user and each column is a grade (lines 16-20). The actual integration takes place in line 23, where the data table and the grades table are merged.

4.7 Pre-processing output table

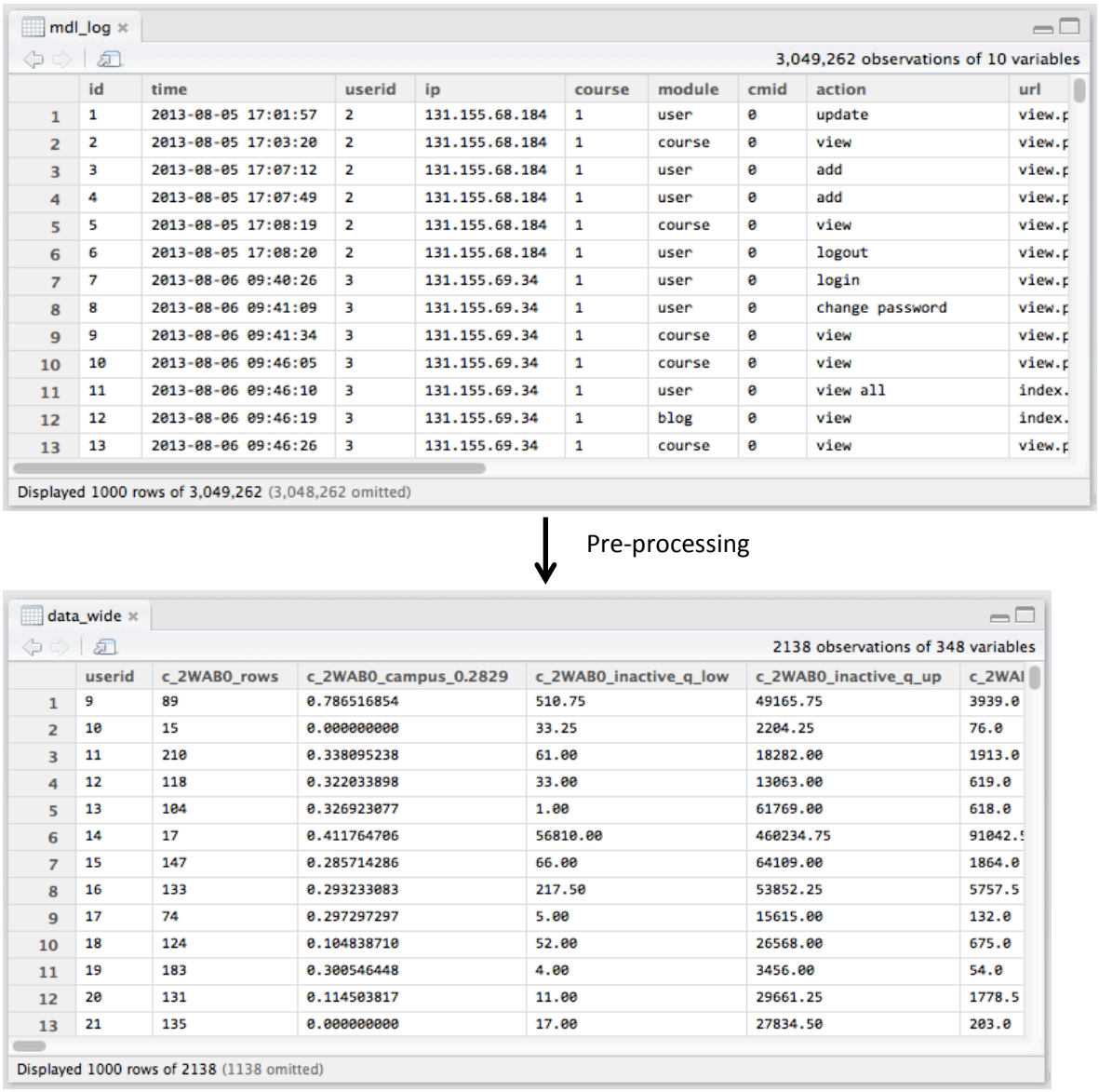
The R pre-processing script creates a table with output data called “data_wide”. Table 9 provides an overview of the structure of the data_wide table, in which the column variables for one course are described. Only the columns for one course are shown. With our data and the configuration (Table 7, 7 courses) the complete data_wide table contains 328 columns according to the structure of Table 9 and 2377 rows that represent individual users.

Table 9: Excerpt of column names of the R pre-processing script output table “data_wide”

Variable	Description
userid	The Moodle user ID
course1_rows	Total number of user events.
course1_campus	The ratio of user activity performed on the university campus or university VPN compared to the total user activity.
course1_inactive_q_low	The lower quartile value of the distribution of the generated “inactive” variable.
course1_inactive_q_low	The upper quartile value of the distribution of the generated “inactive” variable.
course1_inactive_med	The median of the distribution of the generated “inactive” variable.
course1_inactive_mean	The mean of the distribution of the generated “inactive” variable.
course1_inactive_max	The maximum value of the distribution of the generated “inactive” variable.
course1_inactive_var	The variance of the distribution of the generated “inactive” variable.
course1_url_view_users_x	Total number of times a webpage was viewed (x indicates number of users that have shown this type of event).
course1_url_view_week1	Number of times a webpage was viewed in week 1.
course1_url_view_weekn	Number of times a webpage was viewed until week n.
course1_forum_addpost_users_x	Total number of times a forum post was added (x indicates number of users that have shown this type of event).
course1_forum_addpost_week1	Number of times a forum post was added in week 1.
course1_forum_addpost_weekn	Number of times a forum post was added until week n.
course1_forum_adddiscussion_users_x	Total number of times a forum discussion was added (x indicates number of users that have shown this type of event).
course1_forum_adddiscussion_week1	Number of times a forum discussion was added in week 1.
course1_forum_adddiscussion_weekn	Number of times a forum discussion was added until week n.
course1_forum_viewdiscussion_users_x	Total number of times a forum discussion was viewed (x indicates number of users that have shown this type of event).
course1_forum_viewdiscussion_week1	Number of times a forum discussion was viewed in week 1.
course1_forum_viewdiscussion_weekn	Number of times a forum discussion was viewed until week n.
course1_quiz_attempt_users_x	Total number of times a quiz was attempted (x indicates number of users that have shown this type of event).
course1_quiz_attempt_week1	Number of times a quiz was attempted in week 1.
course1_quiz_attempt_weekn	Number of times a quiz was attempted until week n.
course1_grade	Grade that was obtained in this course.

4.8 Summary

In this chapter we have outlined a procedure for pre-processing Moodle log data. The outlined pre-processing procedure is a means of handling the data in a learning analytics study from retrieving raw data from their sources to delivering a dataset that is suitable for analysis. First we imported our raw data (the log data and the grades data) into our data manipulation environment, RStudio. We then introduced means of exploring the log data, on the basis of which it can be decided which data are selected for further processing. Next, we showed how data can be filtered based on our selection. After this we created new variables that are needed for the transformation and enrichment of the data. Finally we integrated the grading data with the new dataset, which completed the pre-processing. The resulting dataset structure is presented and explained in Table 9. An impression of the actual input and output data is shown in Figure 13. In the next chapter we will provide an example analysis on this output data.



	id	time	userid	ip	course	module	cmid	action	url
1	1	2013-08-05 17:01:57	2	131.155.68.184	1	user	0	update	view.p
2	2	2013-08-05 17:03:20	2	131.155.68.184	1	course	0	view	view.p
3	3	2013-08-05 17:07:12	2	131.155.68.184	1	user	0	add	view.p
4	4	2013-08-05 17:07:49	2	131.155.68.184	1	user	0	add	view.p
5	5	2013-08-05 17:08:19	2	131.155.68.184	1	course	0	view	view.p
6	6	2013-08-05 17:08:20	2	131.155.68.184	1	user	0	logout	view.p
7	7	2013-08-06 09:40:26	3	131.155.69.34	1	user	0	login	view.p
8	8	2013-08-06 09:41:09	3	131.155.69.34	1	user	0	change password	view.p
9	9	2013-08-06 09:41:34	3	131.155.69.34	1	course	0	view	view.p
10	10	2013-08-06 09:46:05	3	131.155.69.34	1	course	0	view	view.p
11	11	2013-08-06 09:46:10	3	131.155.69.34	1	user	0	view all	index.
12	12	2013-08-06 09:46:19	3	131.155.69.34	1	blog	0	view	index.
13	13	2013-08-06 09:46:26	3	131.155.69.34	1	course	0	view	view.p

	userid	c_2WAB0_rows	c_2WAB0_campus_0.2829	c_2WAB0_inactive_q_low	c_2WAB0_inactive_q_up	c_2WAI
1	9	89	0.786516854	510.75	49165.75	3939.0
2	10	15	0.000000000	33.25	2204.25	76.0
3	11	210	0.338095238	61.00	18282.00	1913.0
4	12	118	0.322033898	33.00	13063.00	619.0
5	13	104	0.326923077	1.00	61769.00	618.0
6	14	17	0.411764706	56810.00	460234.75	91042.5
7	15	147	0.285714286	66.00	64109.00	1864.0
8	16	133	0.293233083	217.50	53852.25	5757.5
9	17	74	0.297297297	5.00	15615.00	132.0
10	18	124	0.104838710	52.00	26568.00	675.0
11	19	183	0.300546448	4.00	3456.00	54.0
12	20	131	0.114503817	11.00	29661.25	1778.5
13	21	135	0.000000000	17.00	27834.50	203.0

Figure 13: Impression of our data before and after our pre-processing procedure

5 Predicting student performance

As a proof of concept, we now analyse the pre-processed dataset using standard statistical procedures. We first explore variables from the dataset that have the strongest correlation with their respective study results. We then consider how the correlations of a selection of these variables vary over the progression of the course, i.e. how the correlation changed from week to week as the course progressed. Finally, we use standard linear regression to demonstrate a model that can be applied to a selection of the dataset to predict study success.

5.1 Exploring correlations

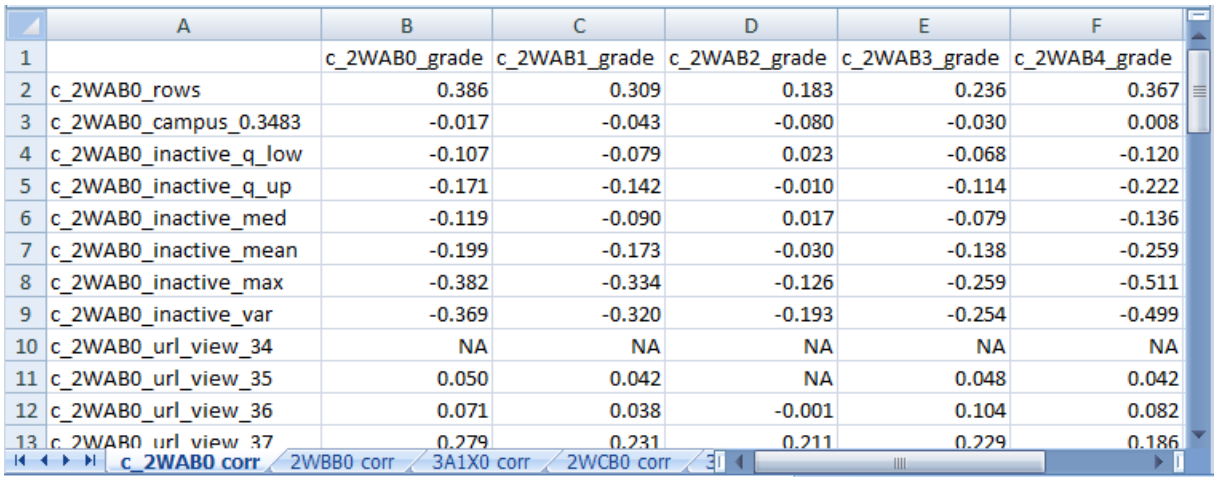
Typically the first step in analysing quantitative data is to explore the correlational relationships within the set of variables that are available. In this situation we are dealing with a pre-processed dataset that contains behavioural metrics from users' course activity and the corresponding study results. We explore the correlation relationships between these metrics and the study results with the use of some R scripts. There are two versions of the script: a version that creates an Excel spreadsheet with the correlation matrix and a version that creates bar plots of the correlations.

5.1.1 Correlation spreadsheet

Figure 14 shows the R script that creates csv files that contain correlational matrices for each course, providing correlations between the actions and the grade(s) in the course. The script runs for every course listed in the configuration table (Table 7), creating a new correlation matrix for every course in a separate file. A correlation matrix of a course contains on one axis all pre-processed metrics except the results of the course and on the other axis the results of the course. The variables that are on x-axis and y-axis of the matrix can however be modified by changing the selection of variables from the pre-processing output table ("data_wide") in lines 6 and 7, respectively. Figure 15 shows an impression of the resulting spreadsheet.

```
1  #create a csv file with correlation matrices of variables
2  for(i in 1:nrow(config)){
3    c <- select(data_wide, starts_with(config$shortname[i]))
4    write.csv(
5              x = cor(
6                      x = select(c, -contains("grade")),
7                      y = select(c,  contains("grade")),
8                      use = "pairwise.complete.obs"),
9    file = paste(config$shortname[i],"corr.csv"))
10 }
```

Figure 14: The R script that outputs a spreadsheet with correlational matrices.



	A	B	C	D	E	F
1		c_2WAB0_grade	c_2WAB1_grade	c_2WAB2_grade	c_2WAB3_grade	c_2WAB4_grade
2	c_2WAB0_rows	0.386	0.309	0.183	0.236	0.367
3	c_2WAB0_campus_0.3483	-0.017	-0.043	-0.080	-0.030	0.008
4	c_2WAB0_inactive_q_low	-0.107	-0.079	0.023	-0.068	-0.120
5	c_2WAB0_inactive_q_up	-0.171	-0.142	-0.010	-0.114	-0.222
6	c_2WAB0_inactive_med	-0.119	-0.090	0.017	-0.079	-0.136
7	c_2WAB0_inactive_mean	-0.199	-0.173	-0.030	-0.138	-0.259
8	c_2WAB0_inactive_max	-0.382	-0.334	-0.126	-0.259	-0.511
9	c_2WAB0_inactive_var	-0.369	-0.320	-0.193	-0.254	-0.499
10	c_2WAB0_url_view_34	NA	NA	NA	NA	NA
11	c_2WAB0_url_view_35	0.050	0.042	NA	0.048	0.042
12	c_2WAB0_url_view_36	0.071	0.038	-0.001	0.104	0.082
13	c_2WAB0_url_view_37	0.279	0.231	0.211	0.229	0.186

Figure 15: Impression of the outputted spreadsheet containing correlation matrices

5.1.2 Correlation bar plot

Figure 16 shows the R-script that creates a PDF file with a bar plot from the correlations, each course in a separate plot on a separate page.

```
pdf("correlation bar plot.pdf", width = 15, height = 10)
for(i in 1:nrow(config)){
  cormat <- cor(
    y = data_wide %>%
        select(contains(substr(config$shortname[i], 3, 6))),
    x = data_wide %>%
      select(contains("grade")) %>%
      select(contains(substr(config$shortname[i], 3, 6))),
    use = "pairwise.complete.obs")
  par(mar = c(20, 3, 3, 1), las = 2, cex = .5)
  plot(x = 0, y = 0, type = "n", xlab = NA, ylab = NA,
       yaxs = "i", yaxt = "n", xaxt = "n", xaxs = "i",
       xlim = c(2.5-4.5, -1.5+4.5 + ncol(cormat) * (nrow(cormat)+ 1)),
       ylim = c(-1, 1))
  rect(xleft = par()$usr[1], ybottom = par()$usr[3], xright =
  par()$usr[2],
       ytop = par()$usr[4],
       col = gray(.8))
  grid(nx = (1 + ncol(cormat)), ny=NA, lty=1, col = "white", lwd = 20)
  grid(nx = NA, ny=20, lty = 1, col = gray(.8))
  barplot(cormat, axes = TRUE, add=T, las=2, space = c(0,1), beside =
  TRUE, width = 1,
          col=rainbow(nrow(cormat)),
          legend.text = TRUE,
          args.legend = list(x = "topleft",
                             y.intersp = 1,
                             title = "Results legend",
                             title.adj = c(.5, .5),
                             bg = "white"),
          title(main = paste("Correlations of metrics vs. results of
          ", config$fullname[i]))))}
dev.off()
```

Figure 16: R script that outputs a grouped bar plot based on the correlational matrix

The script runs for every course in the configuration table (Table 7), similar to the spreadsheet script from Figure 14. Firstly a correlational matrix is generated (lines 1-9). After generating the correlational matrix a grouped bar plot is generated based on the matrix. The plot is drawn with a grey background (lines 10-17), on which a grid is drawn with grey horizontal gridlines (line 19) and vertical guidelines for each group of correlation bars (line 18). After this the actual bar groups are plotted. The "barplot" function groups the correlation values by the x-value of the matrix'. Each bar of a group is plotted with a different colour to support orientation and a legend is plotted to indicate the independent variable (course result) that each colour represents. Figure 17, Figure 18, and Figure 19 show the grouped bar plots that are plotted for three courses from our pre-processed dataset.

The grouped bar plot R script and its corresponding output allow for a graphical overview of correlation relationships amongst variables within the pre-processed dataset. Depending on what correlations need to be investigated the script can be adjusted to present correlation relationships for any set of variables available. Based on interpretation of this plot one can decide to turn back and run a different kind of pre-processing procedure and in doing so create different variables from available raw data, or decide to move on and select certain variables for further analysis.

5.1.3 Discussion

From the bar plots in Figure 17, Figure 18, and Figure 19 we can make a number of observations. The first variable group represents the correlation of the sum of all events initiated by users with the corresponding grades (see Table 8). Across all the courses presented by the figures and also all other courses within our pre-processed dataset, this variable shows a positive correlation with the course grades, although the magnitude may vary. Variable groups 3-9 from each plot represent the descriptive parameters about users' inactivity between events (Table 8). These variable groups vary greatly between the courses represented by the graphs, with variables for "Calculus A" (Figure 17) in opposite direction of the others two courses (Figure 18 and Figure 19). This instability of correlation amongst courses also applies to the "campus" variable, which classifies whether activity was initiated on university campus or outside the campus (Table 8). From variable groups that represent specific types of events over time, such as the "course_url_view" variable (see Table 6), there is a general tendency of higher correlations with the course grades from week to week as the course progresses, which is most evident in variable groups 11-22 of the "Calculus A" course (Figure 17). Lastly, in all courses within our pre-processed dataset that contain activity for event type "quiz_attempt" (see Table 6) these variable groups consistently show high positive correlations with the corresponding course.

Given the variance in the size of the correlations between courses we conclude that creating a general model that applies to all courses is not a trivial exercise.

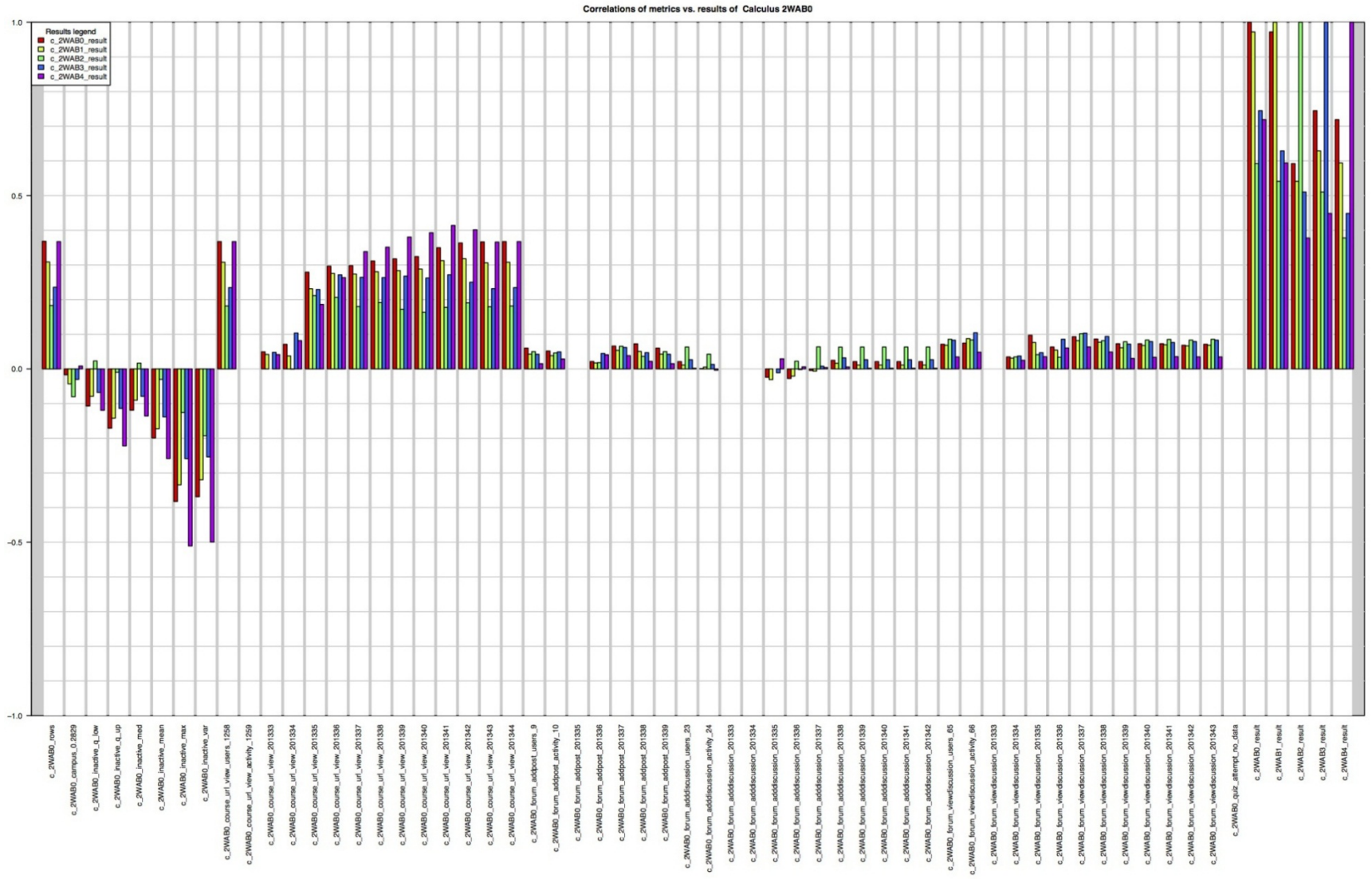


Figure 17: Bar plot from correlations of metrics of “Calculus A” course

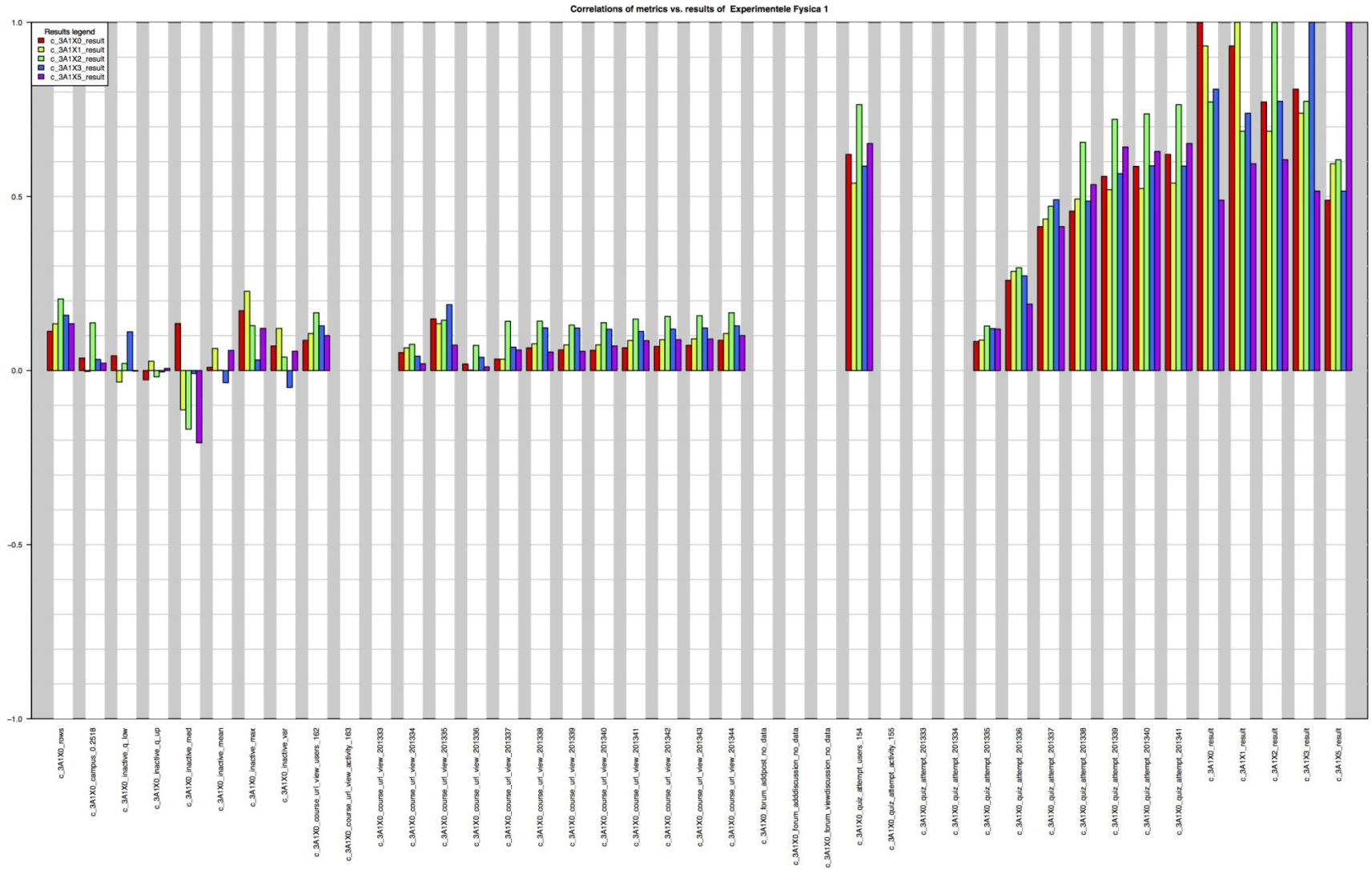


Figure 18: Bar plot from correlations of metrics of "Experimental Physics I" course

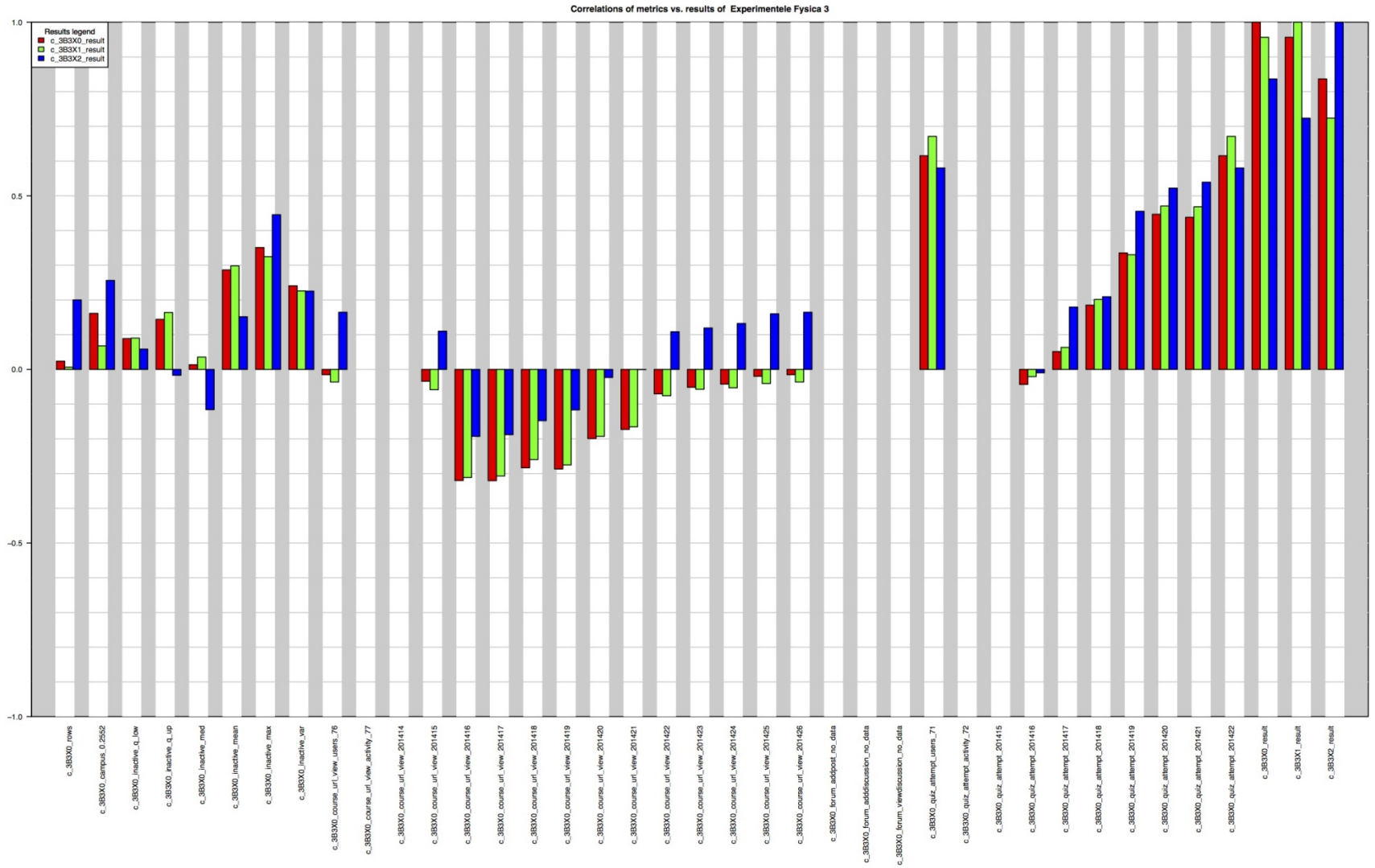


Figure 19: Bar plot from correlations of metrics of "Experimental Physics III" course

5.2 Predicting student performance

Amongst the courses within our pre-processed dataset the most common variables that are available and show correlation with the course grades are the “course_url_view” and “forum_view_view” discussion variables. In the “Calculus A” course (Figure 17) this is most evident. As an illustration, we therefore performed a multiple regression analysis with these two variables from the “Calculus A” course as independent variables and the final course grade as the dependent variable.

We created a linear regression model using R (Kabacoff, 2014). This analysis can, however, be performed with Stata, SPSS, or any other general statistical package. It is, for instance, a straightforward task to export the pre-processed dataset to a spreadsheet file using the “xlsx” package (Dragulescu, 2014) for making it available for other tools.

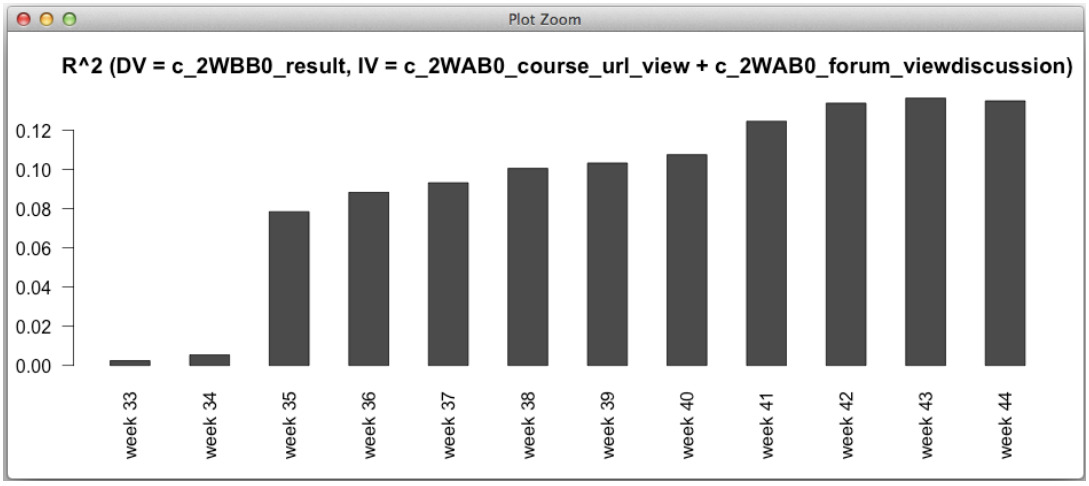
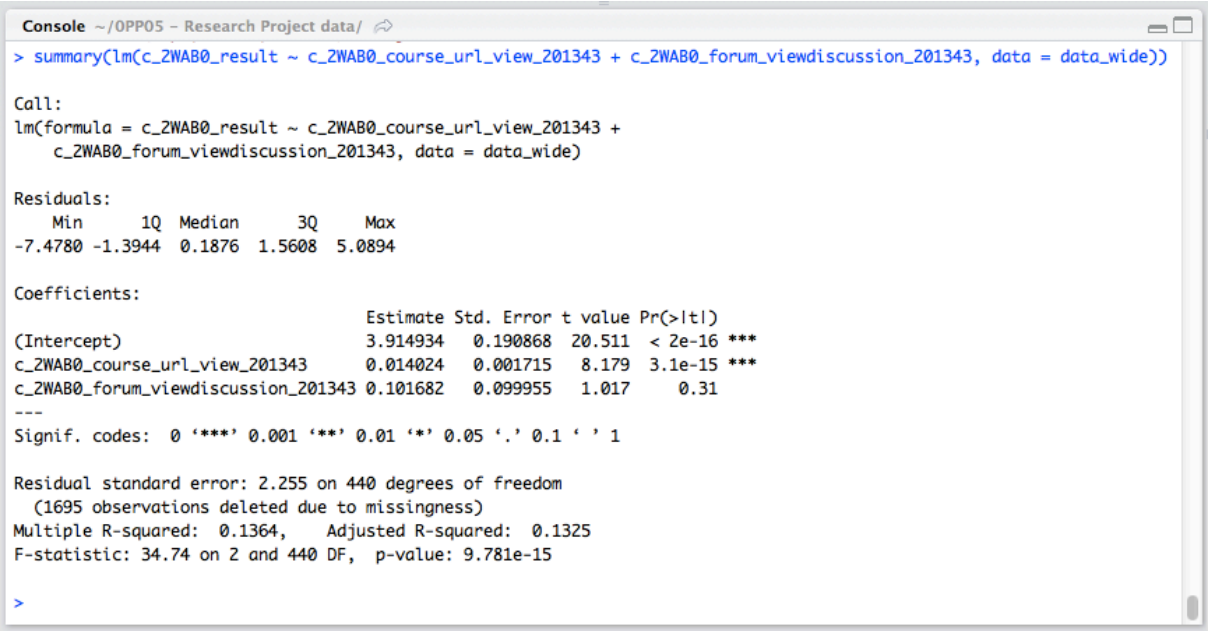


Figure 20: Bar plot of R^2 coefficients of iteratively generated regression models for the “Calculus A” course.

In Figure 20 an overview is given of 12 runs of the multiple linear regression analysis. Every run represents an analysis of the independent variables “course_url_view” and “forum_viewdiscussion” for each week that is present within the pre-processed dataset. From this overview we can observe how the R^2 coefficient increases over the progression of the course in time, so the prediction increases over time. This is as expected, as more data becomes available over time. Compared to the other weeks, the first two weeks of available data (week 33 and 34) show a very low model fit. This could be explained by the fact that those where the two weeks before the course started and hence most users did not use the LMS yet. The best model fit is in week 43, the week of the final exam, with an R^2 of 0.13.

More model parameters, such as the residual statistics, *p*-values, *F*-statistic and adjusted R^2 that are outputted by R as shown in Figure 21 for the multiple linear regression model of week 43.



```
Console ~/OPP05 - Research Project data/
> summary(lm(c_2WAB0_result ~ c_2WAB0_course_url_view_201343 + c_2WAB0_forum_viewdiscussion_201343, data = data_wide))

Call:
lm(formula = c_2WAB0_result ~ c_2WAB0_course_url_view_201343 +
    c_2WAB0_forum_viewdiscussion_201343, data = data_wide)

Residuals:
    Min      1Q  Median      3Q     Max
-7.4780 -1.3944  0.1876  1.5608  5.0894

Coefficients:
                                Estimate Std. Error t value Pr(>|t|)
(Intercept)                     3.914934   0.190868  20.511  < 2e-16 ***
c_2WAB0_course_url_view_201343  0.014024   0.001715   8.179  3.1e-15 ***
c_2WAB0_forum_viewdiscussion_201343 0.101682   0.099955   1.017     0.31
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.255 on 440 degrees of freedom
  (1695 observations deleted due to missingness)
Multiple R-squared:  0.1364,    Adjusted R-squared:  0.1325
F-statistic: 34.74 on 2 and 440 DF,  p-value: 9.781e-15

>
```

Figure 21: Multiple linear regression model from week 43 for the "Calculus A" course

6 Discussion

In this report we have outlined a procedure for pre-processing LMS log data. Using this procedure enables the retrieval of a dataset containing specified variables which is suitable for analysis, based on raw data from LMS logs and a grading table. The method uses the R data manipulation language and performs the pre-processing according to an automated script. To demonstrate the usefulness of the pre-processing methods we have performed a multiple linear regression analysis on a selection of variables from a pre-processed dataset from LMS log data.

Our work is useful for learning analytics researchers because the method we introduce can systematically create behavioural metrics based on raw LMS log data. The decision on what metrics to pre-process can be guided by the data exploration method we have outlined. After pre-processing the structure of the output dataset allows for iterative model creation and testing: new datasets with updated pre-processing parameters can be pre-processed from the raw data in a straightforward fashion. The structure of the output dataset also allows for a variety of analyses within learning analytics. When using this method, researchers can analyse learners' behavioural data available from LMS logs without requiring extensive data manipulations skills that are typically needed for preparing the raw data before analysis.

A certain degree of R programming and general data manipulation skills are, however, required for using our pre-processing procedure, especially when a higher degree of customization of the pre-processing R script is required. Nevertheless, with the outline of the method in this report the script represents a significant degree of automation of the data manipulation process. Currently the pre-processing script supports the LMS log data structure that is used by a Moodle LMS. With some of adjustment our script should support other types of LMS logs as well.

Having access to raw LMS data and grading data and being able to pre-process and analyse certain variables is not always sufficient to understand how certain effects come about: more context about the way the LMS was used could be needed. For example, in our case the variables of the type "quiz_attempt" have shown a consistently high, positive correlation with final course grades. To understand the cause of such correlations we would need to investigate the procedure of performing such a test on the LMS and how the grading would be influenced by doing quizzes. We therefore argue that, to understand how certain behavioural data has been generated when analysing LMS data, it is often needed to investigate under which circumstances an LMS has been used in a course, for instance by contacting a course's responsible teacher.

The guideline offered with this pre-processing method is useful for researchers and for teachers who have some limited data analysis skills, but are not very familiar with scripting routines. Researchers can follow this guideline and can then conduct their own analyses. In addition, this pre-processing procedure makes it possible for teachers as well to use their own preferred statistical methods and visualizations in their course. Together with their knowledge about the way the LMS is implemented in the course, this can provide useful insights in how the LMS is used in the course and how LMS usage is related to student performance, which can be used to improve the course. Moreover, as the

pre-processing can be done while the course is running, teachers have real time data about their course and can even learn about some effects of their teaching while the course is still running.

7 Bibliography

Agudo-Peregrina, Á. F., Iglesias-Pradas, S., Conde-González, M. Á., & Hernández-García, Á. (2014). Can we predict success from log data in VLEs? Classification of interactions for learning analytics and their relation with performance in VLE-supported F2F and online learning. *Computers in Human Behavior*, *31*, 542–550.

Anderson, L. W. (2005). Objectives, evaluation, and the improvement of education. *Studies in Educational Evaluation*, *31*(2), 102–113.

Anderson, S. C. (2013, October 19). An Introduction to reshape2. Retrieved February 20, 2016, from http://seananderson.ca/2013/10/19/reshape.html

Arnold, K. E., & Pistilli, M. D. (2012). Course Signals at Purdue: Using Learning Analytics to Increase Student Success. In *Proceedings of the 2nd International Conference on Learning Analytics and Knowledge* (pp. 267–270). New York, NY, USA: ACM. http://doi.org/10.1145/2330601.2330666

Campbell, J. P., DeBlois, P. B., & Oblinger, D. G. (2007). Academic Analytics: A new tool for a new era. *Educause Review*, *42*(4), 40–57.

Cole, J., & Foster, H. (2007). *Using Moodle: Teaching with the popular open source course management system*. O'Reilly Media, Inc.

Conijn, M. A., Snijders, C. C. P., Matzat, U., & Kleingeld, P. A. M. (2016). *Opportunities and challenges in the emerging field of Learning Analytics: A literature review*. Eindhoven University of Technology.

Dragulescu, A. A. (2014). xlsx: Read, write, format Excel 2007 and Excel 97/2000/XP/2003 files (Version 0.5.7). Retrieved from https://cran.r-project.org/web/packages/xlsx/index.html

Ihaka, R., & Gentleman, R. (1996). R: a language for data analysis and graphics. *Journal of Computational and Graphical Statistics*, *5*(3), 299–314.

Kabacoff, R. I. (2014). Quick-R: Multiple Regression. Retrieved February 20, 2016, from http://www.statmethods.net/stats/regression.html

Macfadyen, L. P., & Dawson, S. (2010). Mining LMS data to develop an "early warning system" for educators: A proof of concept. *Computers & Education*, *54*(2), 588–599.

Minaei-Bidgoli, B., & Punch, W. F. (2003). Using genetic algorithms for data mining optimization in an educational web-based system. In *Genetic and Evolutionary Computation—GECCO 2003* (pp. 2252–2263). Springer. Retrieved from http://link.springer.com/chapter/10.1007/3-540-45110-2_119

Moodle.org: Moodle Statistics. (n.d.). Retrieved May 20, 2016, from https://moodle.net/stats/

Piña, A. A. (2012). An overview of learning management systems. *Virtual Learning Environments: Concepts, Methodologies, Tools and Applications. USA: IGI Global*, 33–51.

Psaromiligkos, Y., Orfanidou, M., Kytagias, C., & Zafiri, E. (2011). Mining log data for the analysis of learners' behaviour in web-based learning management systems. *Operational Research*, *11*(2), 187–200.

Racine, J. S. (2012). RStudio: A Platform-Independent IDE for R and Sweave. *Journal of Applied Econometrics*, *27*(1), 167–172.

Ripley, B., & Lapsley, M. (2015). RODBC: ODBC Database Access (Version 1.3-12). Retrieved from https://cran.r-project.org/web/packages/RODBC/index.html

Romero, C., & Ventura, S. (2007). Educational data mining: A survey from 1995 to 2005. *Expert Systems with Applications*, *33*(1), 135–146.

Romero, C., & Ventura, S. (2010). Educational data mining: a review of the state of the art. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, *40*(6), 601–618.

Romero, C., & Ventura, S. (2013). Data mining in education. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, *3*(1), 12–27. http://doi.org/10.1002/widm.1075

Sumathi, S., & Esakkirajan, S. (2007). *Fundamentals of relational database management systems* (Vol. 47). Springer.

Van den Berg, M., & Hofman, W. (2005). Student success in university education: A multi-measurement study of the impact of student and faculty factors on study progress. *Higher Education*, *50*(3), 413–446.

Wickham, H. (2014). reshape2: Flexibly Reshape Data: A Reboot of the Reshape Package (Version 1.4.1). Retrieved from https://cran.r-project.org/web/packages/reshape2/index.html

Wickham, H., & Francois, R. (2015). dplyr: A Grammar of Data Manipulation (Version 0.4.3). Retrieved from https://cran.r-project.org/web/packages/dplyr/index.html

Yu, T., & Jo, I.-H. (2014). Educational technology approach toward learning analytics: Relationship between student online behavior and learning performance in higher education (pp. 269–270). Presented at the Proceedings of the Fourth International Conference on Learning Analytics and Knowledge, ACM.