May 13, 2019

Working

edX Learner and Course Analytics and Visualization Pipeline

Version 2

Michael Ginda[1], Katy Borner[1], Michael Richey[2], Mark Cousino[3]

[1]Indiana University at Bloomington, [2]The Boeing Corportation, [3]The Boeing Corporation

dx.doi.org/10.17504/protocols.io.zckf2uw

Michael Ginda
Indiana University at Bloomington

ABSTRACT

The edX Student and Course Analytics and Visualization Pipeline is analytics and visualization pipeline using edX course database and user logs, written in R to 1) to extract and process student users and performance data, course structures and event logs; 2) create learner trajectory networks of use and pathways through course content and activity modules; 3) analyze the students use of course content modules; and 4) aggregate student performance and interaction measurements for a given course.

EXTERNAL LINK

https://github.com/cns-iu/edx-learnertrajectorynetpipeline

GUIDELINES

The processing scripts are provided under Apache License 2.0. Contributors provide permission for commercial use, modification, distribution, patent use, and private use. Licensed works, modifications, and larger works may be distributed under different terms and without source code. The script is provided with a limited liability and warranty, use these data processing scripts at your own discretion, and make preservation copies of any source data prior to use.

Additional modifications are likely needed to make use of this pipeline when processing other course data sets that use the edX Data Package format specification.

Organizational implementation of the edX learning management systems may use customized event log tracking systems, and courses may use different types of edX block modules, and logs may include types of events that were not encountered in this project (e.g., error events, or edX discussion forums).

BEFORE STARTING

The scripts also rely on standard edX data export directory structure, which may vary based on local implementations and data provided by an organizations' data czar. An exploratory analysis of the course structure and event logs is advisable at the outset of a project, as well as a comparison of the file names provided and the names expected by the scripts.

The pipeline's scripts create a set of directory structures for processing and analysis outputs, which may be modified, updated in the scripts. These changes should be made after review across all scripts before changes are made, to ensure that processing, analysis and visualization run smoothyl.

Acquire edX Data

1

Review edX Research Guide, which provides documentation for how a user may acquire edX course data from an edX Data

Czar, how to properly and responsibly maintain and use these data in research, and describe in detail the data exports provided for a given edX courses.

The edX Research guide is available for review at https://edx.readthedocs.io/projects/devdata/en/latest/index.html.

When working with data that is outside of your home organization, it is essential to set up a data use agreement (DUA) between the organization holding the data and your institution, in addtion to an IRB. This process must be completed before you can access learner data, which contains information that can be used to re-identify learners.

Data will be provided in a ZIP format that will need to be extracted in Step 3.

edX Data Package

Set up project

2 Install R statistical software

Install R programming language and R Studio Desktop to run the scripts used in this pipeline. Once these pieces of software are installed, you will need to run R Studio. In R studio, you will need to install from the package manager:

- tcltk2 - a GUI interface used to set paths to access data used in the pipeline.
- jsonlite - package used for parsing JSON data
- ndjson - package used for parsing streaming JSON data used in edX event logs.
- Hmisc - data analytics functions
- plyr - data aggregation
- reshape2 - data reshaping functions
- magrittr - piping functions
- stringr - string processing and manipulation
- ggplot2 - statistical visualization package
- GGally - supplements ggplot2
- colorspace - color palette generation

R programming language 3.3.3 of later
source by The R Foundation

R Studio Desktop 1.1.463
by The R Studio, Inc.

3 Install Gephi Network Analysis and Visualization statistical software.

Gephi 0.8.2
by The Gephi Consortium

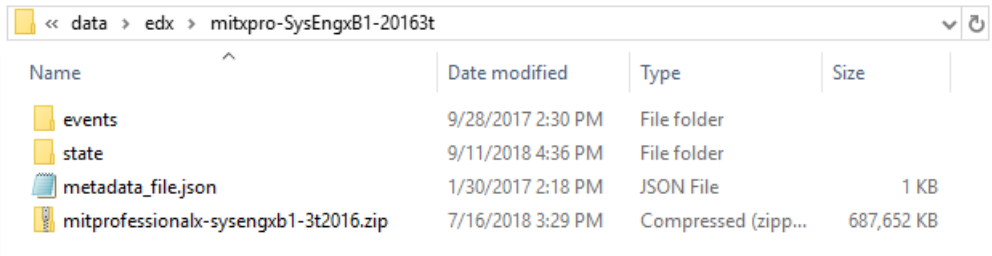
Spatial Ranking
source by Mathieu Bastian

Gephi provides a means of visualizing edX Course hiearchies (see step 15), and student's learner trajectory networks in figure 2 (see step 16). Make sure that you install the Spatial Ranking layout algorithms for replication of Figure 1.A and 1.C.

4 Set up the project directory workspaces that separates course level data provided in the edX data package, and a space for processing outputs, analysis and visualization results. An example directory structure used in this protocol

5 Extract edX Data Package into user identified directory space. The pipeline expects a directory structure that separates original edX course data and processing results (Z:\\project\data\edx\{Course_identifier}\).

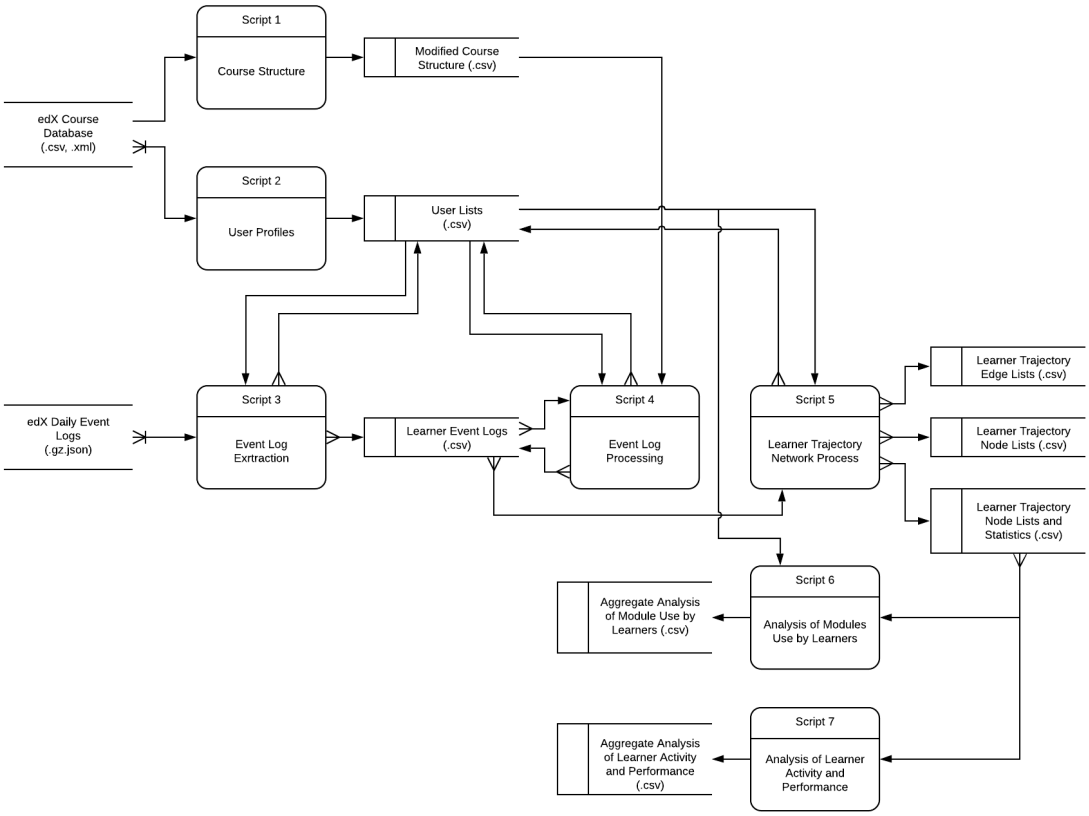
Each course in our edX directory uses a common structure to separate course and user data (state) and daily events logs (events).



Example edX course data directoy extracted from an edX Data Package.

Processing edX Course Data

6 Review the processing scripts used in this analysis to understand how they function alone and as part of the overall pipeline. The UML process flow diagram provides an overview of how original data and processing outputs are used between scripts in the pipeline.



UML Data Flow Diagram of edX Learner and Course Analytics Pipeline

7 Load and Run Script edX-1-courseStructureMeta.R in R Studio.

This script extracts a the course structure from the edX Data Package filesis used in processing log files and creating the node lists in learner trajectory networks.

Make sure to set accurate pathways for reading in original data and saving processing outputs. Note that the script will create directories for processing results automatically, unless these are removed or commented out by a user.

8 Load and Run Script edX-2-studentUserList.R

This script processes user profile data sets from the edX Data Package to identify active students in the course, and exclude instructors, teaching assistants and beta testers from the user log data sets. The script generates a list of edX user IDs.

9 Load and Run Script edX-3-eventLogExtractor.R

This script processes the daily edX course's event tracking logs (which use streaming JSON format) for active students in the course.

Logs are collected for each day of the course, combining all students actions in one file. The script loops through the users identified in the list of active students generated by the edX-2-studentUserList.R to extract a raw event log for each student in the course.

The logs are saved as individual CSV files. The processing speed of this script will be based on the number of students and their volume of recorded activity.

10 Load and Run Script edX-4-eventLogFormatter.R.

This script processes the individual active students event logs, extracted by the *edX-3-eventLogExtractor.R script.* The script uses the course structure data set generated by *edX-1-courseStructureMeta.R* script as part of the log processing.

The script allows a researcher to identify the types of events that are maintained in the final event logs for a student for analysis. All events to the are aligned to the lowest level of the course structural hierarchy; provides temporal ordering and event period calculations and outlier estimates. The script loops through the identified list of active students, and sorts students into further groups based on the size of their processed log files (for example, the script separates students with fewer than 10 events to remove them from the analysis). The script creates of two new lists of studen users based on analysis of the processed event logs: active and inactive students who were not excluded by *edX-2-studentUserList.R script.*

The preset parameters in the script were used with the goal of identifing meaningful user initiated interactions, while preserving the maximal amount of log responses. The script provides detailed comments explaining processing choices.

11 Load and Run Script edX-5-learnerTrajectoryNet.R.

This script creates a learner trajectory network for each student in the course based on the individual's processed event

logs and user list generated by *edX-4-eventLogFormatter.R*. The script first creates an edge list to document transitions between modules in a course, and then creating a node list, which calculates the student's interaction statistics for each low level module in the course.

The script exports a node and an edge lists for each student as CSV files, as well as a JSON formatted learner trajectory network that combines the nodes and edge lists data sets are combined into a single file.

12 Load and Run Script edX-6-moduleUseAnalysis.R.

This script uses the node lists generated by *edX-5-learnerTrajectoryNet.R* and lists of student IDs generated by *edX-4-eventLogFormatter.R*. The script aggregates the node lists from individual students' learner trajectory networks to provide an analysis centering the course structure overall student interactions and activity. Analysis is completed for the lowest modules in the course hierarchy. The results are saved as a CSV data that can be joined to the course structure data set produced by *edX-1-courseStructureMeta.R* script

13 Load and Run Script edX-7-studentFeatureExtraction.R.

This script uses the course metadata generated by the *edX-1-learnerTrajectoryNet.R* script, and output user list and processed student logs and list of student IDs generated by the *edx-4-eventLogFormatter.R* script. The script loops through the list of student processed event logs to create a set of frequency statistics of student activity in an EdX course (e.g. number of sessions, events, unique modules, event_types), calculations of temporal use of content (overall, and relevant module and event types).

Visualization of Results

14 Visualizations were created based on the outputs of the edX Learner and Course Analytics pipeline scripts detailed in the previous section. These visualizations are replicated on the project GitHub site using Rmarkdown scripts and sample data files generated from an edX course that are provided as part of the project repository.

Sample data is described in the edX Learner and Course Analytics and Visualization Pipeline - Sample Data Index. Each of the sample data is described briefly, and along with a view of the data structure produced by R. The file also links to a detailed data dictionary describing each variable in a sample data set.

15 Figure 1.A represents a course structure for an edX course. The course structural hierarchal tree was were visualized using Gephi 0.8.2, using the sample Data A was, which was generated using Script 1, found in step 7.

Data A was loaded as a node list, and a copy was modified to keep the mod_hex_id and parentID as the source and target nodes in a new edge list CSV. Both files were loaded into Gephi using the tools Data Laboratory. The network was laid out using the Spatial Ranking (https://gephi.org/plugins/#/plugin/spatial-ranking) layout plugin for Gephi. Nodes are laid out along the X-axis based on the nodes'.

16 The script used to generate Figure 1.B is provided as a GitHub RMarkdown page available for review here, https://cns-iu.github.io/edx-learnertrajectorynetpipeline/edx-8-figure1-B.html

Sample Data A and Data C were used to create these visualizations.

17 Figure 1.C and Figure 2 visualize student's learning trajectory networks for the for the analyzed edX course. The networks were visualized

using Gephi 0.8.2, using the network exports of script 7 found in step 13.

The network uses JSON format that includes a list of node and list of edges. Data is also exported as CSV formatted node and edge list that are used for visualizing individual student's network data. See Sample Data Set B for an example of data used to create these visualizations.

The nodes represent course structure, specifically all content and activity modules; and description of the student's interaction with the content over the entire course. The edge list represents the student's transitions between course content nodes

The node and edge list CSV files were loaded into Gephi using the tools Data Laboratory. The networks shown visualized using two different layouts:

- The network shown in Figure 1.C was laid out using the Spatial Ranking (https://gephi.org/plugins/#/plugin/spatial-ranking) layout plugin for Gephi. Nodes are layed out on the X-axis based on the nodes'.
- The networks shown in Figure 2 use a force directed layout using the Atlas2 algorithm bundled with Gephi.

18 The script used to generate Figure 3 and 4 is provided as a GitHub RMarkdown page available for review here, https://cns-iu.github.io/edx-learnertrajectorynetpipeline/edx-8-figure3and4.html.

Sample Data E was used to create these visualizations.

19 The script used to generate Figure 5 is provided as a GitHub RMarkdown page available for review here, https://cns-iu.github.io/edx-learnertrajectorynetpipeline/edx-8-figure5.html

Sample Data D was used to create these visualizations.

20 The script used to generate Figure 6 is provided as a GitHub RMarkdown page available for review here, https://cns-iu.github.io/edx-learnertrajectorynetpipeline/edx-8-figure6.html

Sample Data D was used to create these visualizations.