Using CourseKata Data

Last updated: 2020-05-06

Getting Started

Note: This file is included in all data downloads for completed courses on CourseKata.

The easiest way to get up and running with class data from our course is to use our custom R package 'CourseKataData' which can be found at https://github.com/UCLATALL/CourseKataData. The package will unpack and pre-process the data for you to make it a little more manageable. If you aren't comfortable with using R, we still recommend that you use this package to process the data before continuing with your preferred statistical methods.

The absolute simplest way to get running is only a few lines of code in R:

```
# install a package to install the processing package
install.packages('devtools')

# install the processing package
devtools::install_github('UCLATALL/CourseKataData')

# load the package
library(CourseKataData)

# process the data (change the path to point to your downloaded zip file)
process_data('path/to/your/data')
```

By default, the data will be loaded into R objects. R has a number of tools for exporting objects to standard formats. A common format that can be read by most programs (including Excel, Stata, SAS, SPSS) is the comma-separated value (CSV) format. You can export to CSV format with the following code:

```
write.csv(responses, 'path/to/save/the/csv/to', na = '', row.names = FALSE)
# responses is only one of the objects that is created by process_data()
# make sure you export all the data you need
```

Automatically Merge Multiple Downloads

If you have downloaded multiple data download zip files from CourseKata, don't worry, this package takes care of merging the files for you. To load multiple data downloads into R, specify a vector of zip files or directories to load:

```
zip_paths <- c("path/to/first/zip", "path/to/second/zip", "path/to/a/directory")
process_data(zip_paths)
```

More Detail

The rest of this document is the full README that is included on the 'CourseKataData' repository. In addition to explaining some of the other functions in the package, there is a section that explains each of the variables in each of the downloaded data files (**Data Structure**).

CourseKataData

The goal of CourseKataData is to help researchers working with CourseKata courses on data science, statistics, and modeling. The data downloaded from CourseKata can be useful in the more-or-less "raw" form that it comes in, but you will usually want to process the data before working with it. This package includes useful functions for cleaning and tidying the raw course data from completed courses.

Installation

Download the package directly from this repository using devtools:

```
library(devtools)
install_github("UCLATALL/CourseKataData")
```

Usage

This section details the usage of the functions in the CourseKataData package. If you would like to read more about the actual structure of the data downloaded from CourseKata, scroll down to the **Data Structure** section.

One and Done

The easiest way to get started working with your data is to just download it and run `process_data` with the path to the zip file (no need to unzip the file first).

```
library(CourseKataData)

process_data("path/to/downloaded/data.zip")
```

If you would like to unzip the file first (which can sometimes be faster), you can also pass the name of the directory to `process_data()`. If your data was in `path/to/downloaded/data.zip` and you extracted it to `path/to/downloaded/data`, then the call should be to the latter path:

```
process_data("path/to/downloaded/data")
```

When you run this function it will load all of the data and create six data frames (actually tibbles, but they work pretty much the same). These are the names of the data frames that are created:

- `classes`
- `responses`
- `items`

- tags
- media_views
- page_views

If there is already an R object with one of these names, you will be given the opportunity to abort the processing or overwrite the existing object.

Automatically Merge Multiple Downloads

If you have downloaded multiple data download zip files from CourseKata, don't worry, this package takes care of merging the files for you. To load multiple data downloads into R, specify a vector of zip files or directories to load:

```
zip_paths <- c("path/to/first/zip", "path/to/second/zip", "path/to/a/directory")
process_data(zip_paths)
```

Time Zones By default the data is parsed using the "UTC" time zone. If you would like to convert the date time data in each table to a different time zone you can specify it. In this example, your computer system's time zone will be used:

```
process_data("path/to/downloaded/data", time_zone = Sys.timezone())
```

Splitting Responses By default, all of the responses in the course are included in the `responses` table. These responses can be semantically split into three parts: the surveys at the beginning and end of the course, the practice quizzes at the end of each chapter, and the rest of the items in the text. If you would like to split the responses like this, there is a handy `split_responses()` function that will return a list of three tibbles:

```
parts <- split_responses(responses)
parts$in_text
parts$quizzes
parts$surveys
```

Piece by Piece

If you would instead like to process only part of the data, you can use the helpful sub-process functions that exist for each of the file types in the data download:

- process_classes()
- process_responses()
- process_items()
- process_tags()
- process_media_views()
- process_page_views()

As with `process_data()`, each of these functions will accept the path to a CourseKata data download zip file or the directory of the extracted zip file, or you can simply supply the name of the file that you want to process with the function.

```
# zip file
process_classes("path/to/downloaded/data.zip")

# directory
process_classes("path/to/downloaded/data)
```

The classes data file contains information about each class in the data download (see more about the structure of the data download and what is in each file in the **Data Structure** section below). However, the other files are specific to each class included in the download. If you want to extract a specific class, you can specify the `class_id`, which should correspond to the name of the class's folder in the `classes` folder of the download.

```
# a single class
process_page_views(
  "path/to/downloaded/data.zip",
  class_id = "dad3c954-3cb5-11ea-b81d-1d385b778009"
)

# two classes
process_page_views(
  "path/to/downloaded/data.zip",
  class_id = c(
    "dad3c954-3cb5-11ea-b81d-1d385b778009",
    "1a99bb15-4d19-4e3f-bb60-6332141573ed"
  )
)
```

Into the Deep with Responses

The responses take a significant amount of processing. If you would like to do this in parts (perhaps to inject your own processing at some point along the way), each of the finer processing functions are available. The following is equivalent to `process_responses()`:

```
read.csv("path/to/downloaded/data/classes/[class_id].csv") %>%
  convert_types_in_responses() %>%
  ensure_data_in_responses() %>%
  map_response_options()
```

The part functions are purely functional, which means that they can be run in any order. If you alter the response data and and a function doesn't know how to handle it any more, it will tell you what is missing or needed.

Data Structure

Data downloaded from CourseKata comes in a zip file with following the structure (this example assumes data for two classes were downloaded):

```
[datetime of download]
|   classes.csv
|   README.pdf
```

```
|
\---classes
    +---[class id of the first class]
    |       items.csv
    |       media_views.csv
    |       page_views.csv
    |       responses.csv
    |       tags.csv
    |
    \--- [class id of the next class]
            items.csv
            media_views.csv
            page_views.csv
            responses.csv
            tags.csv
```

Each of the following six sections will describe what you should expect in each of the `.csv` files.

`classes.csv`

The classes data is organized into a table with a single row for each class included in the data download. Note that each of the `class_id` values will correspond to the name of a subfolder in the `classes` folder. Here is a description of each variable in the classes table:

Column	Description
class_id	a unique identifier for this particular class
course_name	the GitHub repository for the course
release	the GitHub branch for the course
teacher_id	a unique identifier for the teacher on CourseKata
lms	the Learning Managment System the class was deployed on
setup_yaml	a JSON string including the full structure of the course (converted to a `list` object when the data is processed)

`responses.csv`

The response data is organized into a table with a variable number of columns (depending on `lrn_option_<n>`, see column description below) and a number of rows equivalent to the number of responses made to questions in the course. This table will likely be very large (200 students will yield around 300,000 responses). Here is a description of each variable in the responses table:

Column	Description
class_id	a unique identifier for this particular class
course_name	the GitHub repository for the course
release	the GitHub branch for the course
student_id	a unique identifier for each student on CourseKata
item_id	a unique identifier for this particular question
item_type	whether this is a *learnosity* or *datacamp* item
chapter	the chapter that the item appears in
page	the page that the item appears on
prompt	the question prompt for this response

Column	Description
response	the value of the response: either the value (for shorttext, plaintext, ratings, datacamp, etc.) or an array of numbers indicating the position of the multiple choice answers chosen and which correspond to the columns `lrn_option_<number>`.
points_possible	the number of points possible if the completely correct answer is given
points_earned	the number of points earned
dt_submitted	a datetime object indicating when the response was submitted (timezone: GMT/UTC)
attempt	the number of times the question has been attempted, including the current attempt
user_agent	the browser user agent string for the user (see for details: https://developer.chrome.com/multidevice/user-agent; also, this R package for parsing: `uaparserjs`)
lrn_session_id	the unique ID for this user session on Learnosity
lrn_response_id	the unique ID for this particular response on Learnosity
lrn_items_api_version	the version of the Learnosity Items API for this item
lrn_response_api_version	the version of the Learnosity Response API for this response
lrn_activity_id	the unique ID for the activity on Learnosity
lrn_question_reference	the unique ID for the question on Learnosity
lrn_question_position	for multi-question items, the position of the question in the item
lrn_type	the Learnosity type of the question (e.g. mcq, shorttext, plaintext, rating, etc.)
lrn_dt_started	a datetime object indicating when the responses was started (timezone: GMT/UTC)
lrn_dt_saved	a datetime object indicating when the responses was saved (timezone: GMT/UTC)
lrn_status	the status of the question response on Learnosity (e.g. "Completed")
lrn_option_	for multiple choice questions (`lrn_type`: "mcq"), the value of the option at position n
lrn_response_json	the fully-detailed JSON response object (converted to a `list` object when the data is processed)

For more about the distinction between Learnosity Activities, Items, and Questions, see the documentation at https://authorguide.learnosity.com/hc/en-us.

`page_views.csv`

The page view data comes in a table of student-page-datetime cases. That is, there is a row for every time a student accessed a page in the course. The table has five variables:

Column	Description
class_id	a unique identifier for this particular class
student_id	a unique identifier for each student on CourseKata
chapter	the chapter the page view occurred in
page	the page that was viewed
dt_accessed	a datetime object indicating when the page was accessed (timezone: GMT/UTC)

`media_views.csv`

The media view data refers to student interactions with the videos in the course. It comes in a table of student-video cases. That is, there is a row for each video for each student. The table has five variables:

Column	Description
class_id	a unique identifier for this particular class
student_id	a unique identifier for each student on CourseKata
chapter	the chapter the video is in
page	the page that the video is on
type	the type of media object (currently only "video")
media_id	the unique identifier for the video
dt_started	a datetime object indicating when the video was first started (timezone: GMT/UTC)
dt_last_event	a datetime object indicating when the video was last interacted with (timezone: GMT/UTC)
proportion_video	the proportion of the video that student has watched, summed across all interactions with the video
proportion_time	the proportion of the full video runtime that the student has spent watching a video, regardless of which part of the video they watched (e.g. if the video is 10 seconds long, and the student watches the first 5 seconds three times, this value should be 1.5)
log_json	the fully-detailed JSON object about student interactions with the video (converted to a `list` object when the data is processed)

`items.csv`

Items are organized in a table where each row represents all of the data for a particular question in the class. There are 18 columns in the table, though they will never *all* be relevant for a particular item. Columns prefixed with `dcl_` are only filled for DataCamp-Light items (the R-sandboxes) and columns prefixed with `lrn_` are only filled for Learnosity items. Here are descriptions of the columns in the table:

Column	Description
class_id	a unique identifier for this particular class
item_id	a unique identifier for this particular question
item_type	whether this is a **learnosity** or **datacamp** item
chapter	the chapter that the item appears in
page	the page that the item appears on
dcl_pre_exercise_code	the code run invisibly to set up the module
dcl_sample_code	the code in the module when it first loads
dcl_solution	the code that appears in the solution tab
dcl_sct	the solution checking code (see the `testwhat` package for details)
dcl_hint	the text that appears in the hint box
lrn_activity_reference	the unique ID for the activity on Learnosity
lrn_question_reference	the unique ID for the question on Learnosity
lrn_question_position	for multi-question items, the position of the question in the item
lrn_template_name	the template used to create the item
lrn_template_reference	a unique ID for the item template on Learnosity
lrn_item_status	the status of the item on Learnosity (e.g. "published")
lrn_question_data	the fully-detailed JSON object that sets up the item (converted to a `list` object when the data is processed)

For more about the distinction between Learnosity Activities, Items, and Questions, see the documentation at https://authorguide.learnosity.com/hc/en-us.

`tags.csv`

Tags are not currently utilized heavily within CourseKata but may have a larger role in the future (e.g. tagging specific learning outcomes). For completeness, the table is described here, but it will likely not be of much use.

The tags table is organized at the item-tag level where each tag for each item has its own row. There are three columns in the table:

Column	Description
item_id	a unique identifier for this particular question
tag	the tag given to this item
tag_type	the hierarchical parent tag for this tag (e.g. "Chapter" holds all of the chapter name tags)

Contributing

If you see an issue, problem, or improvement that you think we should know about, or you think would fit with this package, please let us know on our issues page. Alternatively, if you are up for a little coding of your own, submit a pull request:

1. Fork it!
2. Create your feature branch: `git checkout -b my-new-feature`
3. Commit your changes: `git commit -am 'Add some feature'`
4. Push to the branch: `git push origin my-new-feature`
5. Submit a pull request :D