Data Description

This is a time-series code competition, you will receive test set data and make predictions with Kaggle's time-series API. Please be sure to review the Time-series API Details section closely.

Files

train.csv

- `row_id`: (int64) ID code for the row.
- `timestamp`: (int64) the time in milliseconds between this user interaction and the first event completion from that user.
- `user_id`: (int32) ID code for the user.
- `content_id`: (int16) ID code for the user interaction
- `content_type_id`: (int8) 0 if the event was a question being posed to the user, 1 if the event was the user watching a lecture.
- `task_container_id`: (int16) Id code for the batch of questions or lectures. For example, a user might see three questions in a row before seeing the explanations for any of them. Those three would all share a `task_container_id`.
- `user_answer`: (int8) the user's answer to the question, if any. Read -1 as null, for lectures.
- `answered_correctly`: (int8) if the user responded correctly. Read -1 as null, for lectures.
- `prior_question_elapsed_time`: (float32) The average time in milliseconds it took a user to answer each question in the previous question bundle, ignoring any lectures in between. Is null for a user's first question bundle or lecture. Note that the time is the average time a user took to solve each question in the previous bundle.
- `prior_question_had_explanation`: (bool) Whether or not the user saw an explanation and the correct response(s) after answering

the previous question bundle, ignoring any lectures in between. The value is shared across a single question bundle, and is null for a user's first question bundle or lecture. Typically the first several questions a user sees were part of an onboarding diagnostic test where they did not get any feedback.

questions.csv: metadata for the questions posed to users.

- `question_id`: foreign key for the train/test content_id column, when the content type is question (0).
- `bundle_id`: code for which questions are served together.
- `correct_answer`: the answer to the question. Can be compared with the train `user_answer` column to check if the user was right.
- `part`: the relevant section of the TOEIC test.
- `tags`: one or more detailed tag codes for the question. The meaning of the tags will not be provided, but these codes are sufficient for clustering the questions together.

lectures.csv: metadata for the lectures watched by users as they progress in their education.

- `lecture_id`: foreign key for the train/test content_id column, when the content type is lecture (1).
- `part`: top level category code for the lecture.
- `tag`: one tag codes for the lecture. The meaning of the tags will not be provided, but these codes are sufficient for clustering the lectures together.
- `type_of`: brief description of the core purpose of the lecture

example_test_rows.csv Three sample groups of the test set data as it will be delivered by the time-series API. The format is largely the same as train.csv. There are two different columns that mirror what information the AI tutor actually has available at any given time, but with the user interactions grouped together for the sake of API performance rather than strictly showing information for a single user at a time. *Some users will appear in the hidden*

test set that have NOT been presented in the train set, emulating the challenge of quickly adapting to modeling new arrivals to a website.

- `prior_group_responses` (string) provides all of the `user_answer` entries for previous group in a string representation of a list in the first row of the group. All other rows in each group are null. If you are using Python, you will likely want to call `eval` on the non-null rows. Some rows may be null, or empty lists.

- `prior_group_answers_correct` (string) provides all the `answered_correctly` field for previous group, with the same format and caveats as `prior_group_responses`. Some rows may be null, or empty lists.

Time-series API Details

- Refer to the starter notebook for an example of how to complete a submission. The time-series API has changed somewhat from previous competitions!

- You should not try to submit anything for the rows that contain lectures.

- The API provides user interactions groups in the order in which they occurred. Each group will contain interactions from many different users, but no more than one `task_container_id` of questions from any single user. Each group has between 1 and 1000 users.

- Expect to see roughly 2.5 million questions in the hidden test set.

- The API will load up to 1 GB of the test set data in memory after initialization. The initialization step (`env.iter_test()`) will require meaningfully more memory than that; we recommend you do not load your model until after making that call. The API will also consume roughly 15 minutes of runtime for loading and serving the data, but will also obfuscate the true runtime for all submissions.

- The API loads the data using the types specified above (int32 for `user_id`, int8 for `content_type_id`, etc).