Capstone Data Wrangling
ACADEMIC MASTERY STUDY

All work described in this report can be viewed in the initial_data_exploration notebook.

Download **grockit_all_data.zip** data files from the Grockit competition on Kaggle, found here: https://www.kaggle.com/c/WhatDoYouKnow/data. Employ Python technologies to import and inspect the raw **training.csv** data, isolate relevant variables, calculate new variables, organize the dataframe, and resolve missing, invalid, corrupted or duplicate values. Begin creating data visualizations to inspect variables and analyze outliers.

Import Packages, Dataset & Inspect Raw Data

After the raw training dataset was imported and saved as a dataframe, initial inspection established that there are 4,851,475 rows, or observations. Each observation holds information about a question attempted by a user. There are 17 columns, or variables. Of particular interest are **user_id** which identifies individual users and **question_id** which identifies unique questions. Since the study focuses on the performance of both, a count of each was retrieved before eliminating any records in the wrangling process. There were 179,106 users and 6,045 unique questions before data cleaning.

Organize Columns

Variables related to time were converted to datetime objects. The columns **outcome** and **answer_id** were dropped since these are not included in the test dataset. The **question_set_id** column was also dropped since, according to the dataset documentation provided, most question sets only have one question. The question_id is a better identifier.

A timedate column was created: round_ended_at. The **round_ended_at** column was filled with answered_at values if present. If no value was found in any answered_at field, then the value from deactivated_at was used to populate round_ended_at.

Before moving on with the dataframe, the space-separated tag strings in the tag_string column for each observation were converted to a comma delimited list of tag identifiers. This column was renamed **tag_ids**, also for semantic accuracy. Columns were then reordered to prioritize user_id, time data and question_id.

Evaluate & Resolve Missing Data

NULL values in the dataframe indicate that the value is missing, invalid or corrupted. Looking at the count of NULL values for every column in the dataframe revealed that timedate variables contained missing data. NULL values were expected for **answered_at** since this denotes that the question timed-out. Missing values for **date_of_test** were also no surprise since it was known that users may or may not enter an expected test date. NULL values for **round_started_at** and **deactivated_at** were less understood and, therefore, examined more closely.

Since time data may be an important factor in improving the efficiency of testing applications, all 69 observations with missing time data for all timedate variables were eliminated. Another 21 rows were dropped for missing **answered_at** and **deactivated_at** values since round duration cannot be calculated in those rows.

3,824,208 of the 4,851,475 observations in the raw dataset were missing values for **date_of_test**. That is nearly 80%, but this does not indicate how many users entered no test date data. A look at NULL values by user revealed that 87% of users did not provide an expected test date. This variable can not be adequately analyzed so the column was dropped.

At this point, most of the obvious NULL values in the dataframe were in the **answered_at** column and 10 remained in the **deactivated_at** column. Since observations where both are NULL were eliminated, there was no harm in keeping all of the remaining NULL values. Either variable provides the necessary information for round_ended_at, so ignoring one missing value is no problem.

Not at all obvious from traditional inspection of the dataframe was the fact that NULL values existed in the **game_type** column. This was learned from looking at the labels in the category_labels.csv file. Information found in the forum where the datasets are hosted claim a value of 6 represents a data collection failure. Upon inspecting observations for these values, they were dropped mainly because most, if not all, of the observations involve rounds with end times earlier than start times.

Evaluate & Resolve Duplicate Data

Since one user cannot physically start or end a round at the exact same time more than once, records were inspected for such a scenario. This revealed many duplicates, but it also revealed many observations where round_started_at times were the same as round_ended_at times. Given the reasonable expectation that it is impossible to start and end a round at the same time, 4264 observations meeting such conditions were dropped.

After weird time issues were resolved, duplicate observations for the combination of **user_id**, **round_started_at** and **round_ended_at** were dropped.

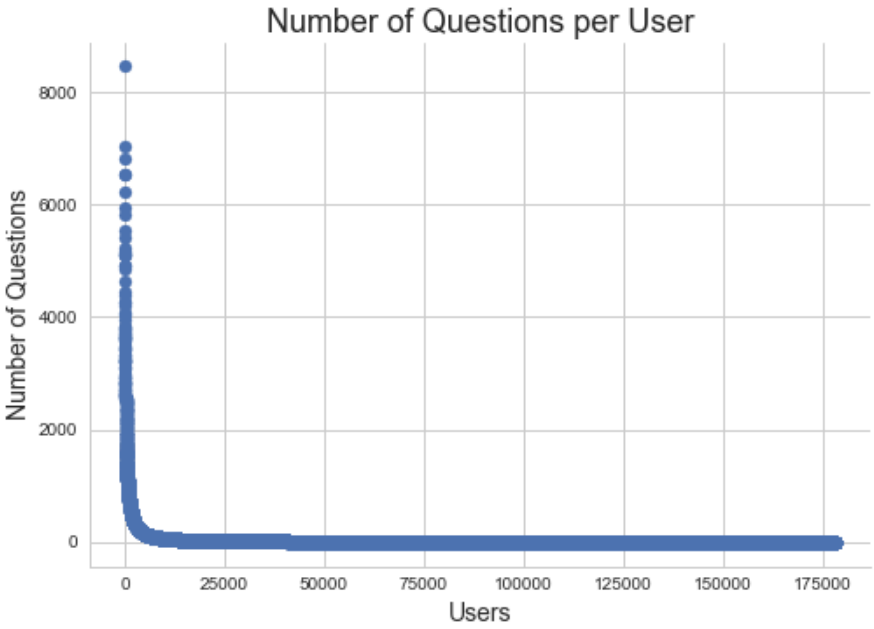
Compute and Explore Time Data

The **round_duration** column was created and filled by subtracting round_started_at values from round_ended_at values.

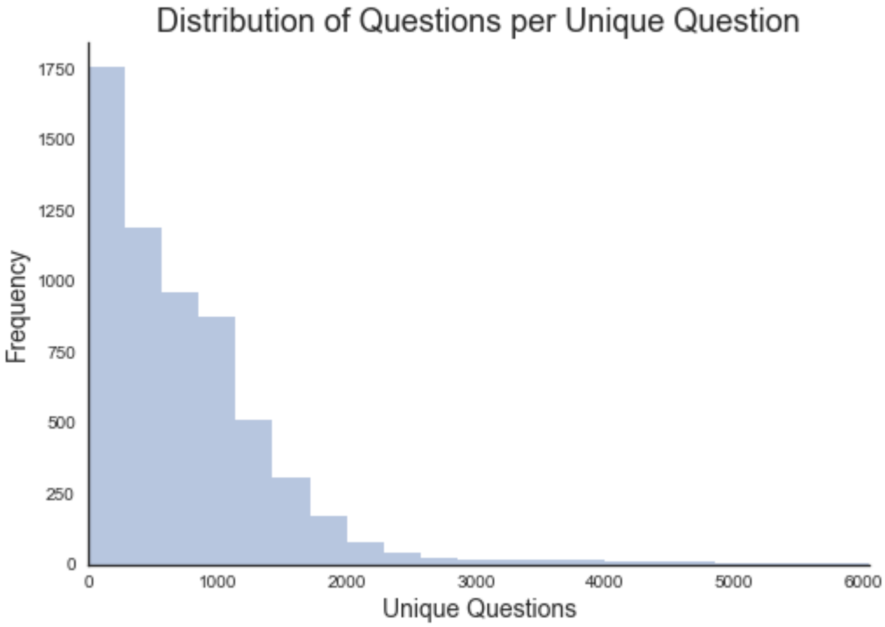
A view of time data statistics revealed that some rounds were days long and many were over within only a few seconds. 167,564 observations outside a threshold of 5 seconds to 10 minutes were eliminated. This left 4,679,166 rows in the working dataframe, for a total of 172,309 dropped records, or 3.6% of the raw dataset. This also left 178,342 users in the dataframe, so less than a half percent of the users were eliminated during data cleaning.

Initial Exploration

The most interesting of the initial views of variables were those of questions per user and questions per unique question:



NOTEBOOK CELL #43
It looks like a small percent of users attempted a large number of questions and a large percent attempted a small number of questions. This may be important to keep in mind moving forward.



NOTEBOOK CELL #46
The long tail here was more unexpected. Perhaps the questions with low frequency are grouped somehow that explains why they are underrepresented? The distribution of groups and tracks may point to such. Nonetheless, this is another feature to keep in mind.

Important Features Before and After Data Wrangling:

	Total Observations	Total Variables	Number of Users	Number of Uniques Questions
Before Cleaning	4,851,475	17	179,106	6,045
After Cleaning	4,679,166	15	178,342	6,045
ELIMINATED	172,309	(2 added) 4	764	0

Conclusions

The major outliers appear to be related to number of questions per user. There are extremes at both ends, but the most dramatic seems to be the number of users attempting very few questions. There are also many unique questions underrepresented in the dataset. A look at the test groups and tracks may shed light on this.

Follow further analysis in the exploratory_data_analysis notebook.