Machine Learning Engineer Nanodegree

Capstone Project

Determination of students' interaction patterns with an intelligent tutoring system and study of their correlation with successful learning

Dmytro Iakubovskyi, November 6, 2017

I. Definition

Project Overview

Usually, the process of learning is described by the learning curve. Numerous studies of different types of learning suggest the scale-invariant learning curve defined by the "Power Law of Learning":

$$E(x) = b\, x^{-d}\ ,$$

where ***x*** is the number of attempts to solve a given problem, ***E(x)*** is the average error rate, ***b*** is the difficulty parameter and ***d*** is the learning parameter. Despite its simplicity, the Power Law of Learning describes the observed variety of learning curves remarkably well, see [1-6].

The goal of this project is to study learning curves provided by an intelligent tutoring system (ITS). ITS are designed to give the learners an instantaneous automatic feedback concerning the correctness of their solutions, see, e.g., [7] for an ITS review. Many previous works (see, e.g., [8-10] for reviews) have used machine learning to analyse the ITS output.

Modern ITS can provide large amounts of relevant data. In this project, I am going to investigate the ASSISTments dataset [11] publicly available at PSLC DataShop [12]. It contains detailed records from several hundreds of schoolchildren regularly solving math problems using an intelligent tutoring system (in addition to regular math classes) during 2004-2005, 2005-2006 and 2006-2007 school years. This dataset is large enough for a detailed statistical analysis. In total, it contains about 3.3M instances collected from several thousands of students, and 100+ features (including all information necessary to determine our target variable — the learning parameter). The critical element of the ASSISTments system is that it proposes several "scaffolding" questions to coach students that failed on original questions.

The ASSISTments dataset has been already used in many related studies. For example, [13, 14] reported that despite very limited interaction with the system (one 20-minute or 40-minute lesson every two weeks [15], depending on the school), the students improved their knowledge during such interaction. Also, [15-17] found that the models based on skill learning tracking predicted the final test score better than a simple model based only on time variable and the difficulty parameter ***b***. Also, [18] studied in details the "gaming" phenomenon where "a learner attempts to succeed in an educational environment by exploiting properties of the system's help and feedback rather than by attempting to learn the material".

Problem Statement

Because students mostly study a subject during regular classes, I expect that they come with different backgrounds. It should result in a variety of students' interaction patterns with an ASSISTments system. The goal of the project is ***to determine this variety*** and ***to study its correlation with successful learning*** (parametrised by a learning parameter ***d***). This problem is naturally split into three steps:

- to cluster students based on their interaction patterns with the tutor (***Step 1***);

- to determine the value of learning parameter for each student (***Step 2***);
- to rank the obtained clusters based on learning parameter distributions (***Step 3***).

The raw data presented by ASSISTments are in the form of system log files for all students interacting during the session (with 3.3M instances in total). So, I consider the determination of learning parameter ***as a part of the pre-processing stage*** that will produce a "condensed" dataset with only several thousands of instances, one for each student. This dataset will be suitable for cluster studies performed during the next step (note that information about learning parameter will ***not*** be used to determine the clusters). Finally, the obtained groups will be ranked based on learning parameter of students that belong to these clusters.

Metrics

During clustering (***Step 1***), I maximise the Silhouette Coefficient which is the average value of individual scores ***s*** calculated for each sample:

$$s = \frac{b - a}{max(a, b)}$$

where ***a*** is the mean distance between a sample and all other points in the same class, ***b*** is the average distance between a sample and all different points in the *next nearest cluster*. This score is bounded from -1 and +1 and are higher for compact and well-separated groups, similar to expectations. Because it strongly depends on distance scalings, I also use an appropriate scaling (i.e., MinMaxScaler after log-transformation).

During learning parameter determination (***Step 2***), the values of learning parameter ***d*** and difficulty parameter ***b*** for each student will be determined by non-linear regression of individual learning curves. Note that ASSISTments dataset provides an outcome ("CORRECT" or "INCORRECT") for every problem attempted by students, and I constructed new feature ***x*** (the number of attempts to solve a problem). Usual approaches (such as LinearRegression with R2 score minimisation) are not directly applicable: the power law of learning is not linear, and log-log transformation (which can make the fitting function linear) fails if the error rate is zero. Instead, I minimise the ***C-statistic*** (assuring Poissonian distribution of counts instead of Gaussian and thus being widely used in high-energy astronomy [19, 20] which deals with the very small number of counts per bin):

$$C = 2 \sum_{i=1}^{N} (e_i - n_i \ln e_i),$$

where ***N*** is the number of bins (in our case — total number of attempts made by the student), n_i is the error probability (0 for "CORRECT" and 1 for "INCORRECT" attempt), and e_i is the learning curve prediction given the values of learning parameter ***d*** and difficulty parameter ***b***.

II. Analysis *(approx. 2-4 pages)*

Data Exploration

Throughout this project, I used raw data from ASSISTments dataset available at PSLC DataShop. The dataset contains three txt files with detailed records of students' interactions with ASSISTments system during 2004-2005, 2005-2006 and 2006-2007 school years that include 0.58M, 0.69M and 1.45M instances, respectively. Each row represents the unique interaction of a student, i.e., attempting to solve a math problem or looking for a related hint given by the system.

After dropping features that contain only NaNs or single fields such as ‘Level (Curriculum)' always equal to ‘Assistment', the ASSISTments dataset contains the following "meaningful" features:

- **‘Anon Student Id’** - unique anonymous student identifier;

- **‘Session Id’** - identifier of a session started by a student;

- **‘Time’** - time of a record (during Preprocessing, it will be transformed to **‘Day’** by removing hh: mm: ss information);

- **‘Duration (sec)’** - instance duration in seconds;

- **‘Student Response Type’** - type of student responses (‘ATTEMPT' when attempting to solve a problem or ‘HINT_REQUEST' when requesting a hint);

- **‘Tutor Response Type’** - ‘RESULT’ if student made an attempt and ‘HINT_MSG’ otherwise;

- **‘Problem Name’** - unique problem identifier;

- **‘Problem View’** - number of sessions (see **‘Session Id’**) under which the problem is viewed;

- **‘Step Name’** - number and description of the step. For each problem, step number is set to 0 at the beginning of the session and is increased by 1 after making 3 attempts or requesting a "scaffolding" hint. Contrary to **‘Selection’** (see below), does contain null instances.

- **‘Is Last Attempt’** - is 1 for a "scaffolding" hint, a third attempt during the step or when coming to the next problem. Otherwise, it is set to 0.

- **‘Outcome’** - ‘HINT’ for a hint, ‘CORRECT’ or ‘INCORRECT’ for an attempt;

- **‘Selection’** - almost identical with **‘Step Name’** (e.g., it contains the same step number) but does not contain null instances.

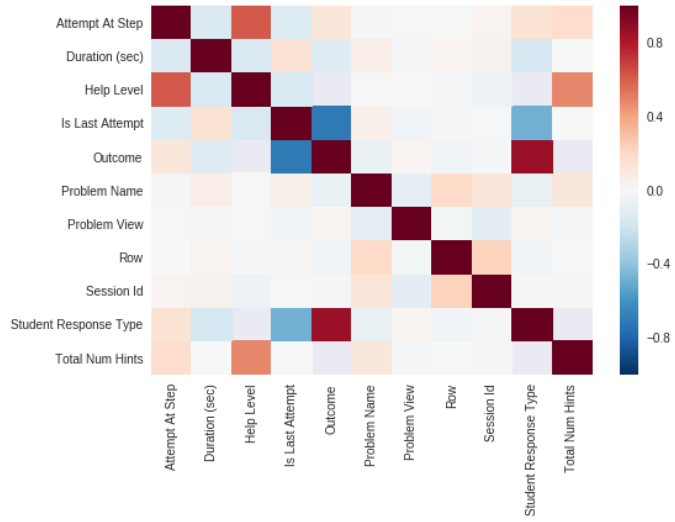
- **‘Feedback Classification’** - NaN for attempts, ‘HINT_REQUEST’ or ‘SCAFFOLD’ for hints;

- **‘Help Level’** - number of subsequent hints (NaN for attempts);

- **multiple ‘KC*’ features** — detailed classification of a given question;

- **‘School’** and **‘Class’** - school and class identifiers.

Below is the correlation map between numerical features:



As we see, there are some substantial (anti)correlations between the following features:

• correlation (***0.87***) between **'Outcome'** and **'Student Response Type'** (hints contribute to the largest values of both features);

•correlation (***0.61***) between **'Attempt At Step'** and **'Help Level'** (making more steps for attempts means making more hints);

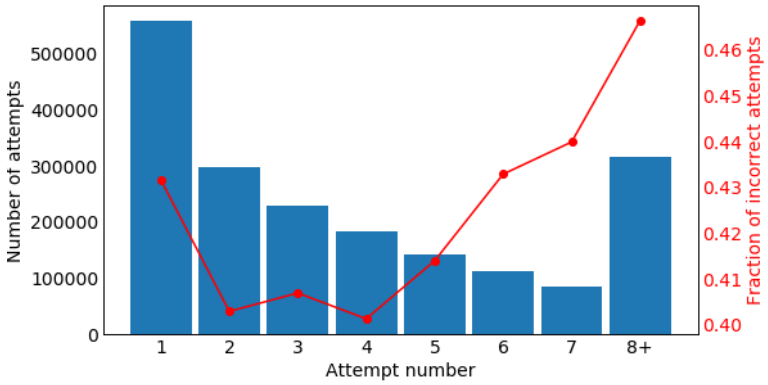
•correlation (***0.49***) between **'Help Level'** and **'Total Num Hints'** (number of subsequent hints correlate with the total number of hints);

•anti-correlation (***-0.72***) between **'Is Last Attempt'** and **'Outcome'**;

•anti-correlation (***-0.48***) between **'Is Last Attempt'** and **'Student Response Type'** (a product of correlation between **'Outcome'** and **'Student Response Type'** and anti-correlation between **'Is Last Attempt'** and **'Outcome'**).

Exploratory Visualization

To visualise the process of learning in ASSISTments system, I created a new feature ***x*** (the number of attempts to solve a problem) directly related to the "Power Law of Learning", grouped all 1,914,379 attempts made by 8,980 students by values of this new feature, and plotted them together with error rate (percentage of incorrect attempts):



As we can see from this visualisation, the ***stacked*** learning curve behaves ***differently*** from naive "Power Law of Learning" expectations. Namely, while for a small number of attempts the error rate (shown by the red curve) decreases according to expectations, it ***gradually increases*** starting from the fifth attempt. In Results section below, I discuss the origin of this difference.

Algorithms and Techniques

Step 1 is a typical clustering (unsupervised classification) problem. For this action, I will use the variety of clustering algorithms implemented in the scikit-learn library for machine learning and described in details here. Due to limited complexity of the problem (< 10 000 samples and an unknown number of desired clusters), the MeanShift algorithm is expected to be the best. However, I am going to compare the MeanShift algorithm performance with three widely used clustering algorithms — KMeans, GaussianMixture and AgglomerativeClustering — and to choose the algorithm with the most substantial Silhouette Coefficient selected as a relevant evaluation metrics.

KMeans clustering algorithm aims at minimising ***inertia*** — the sum of squared distances to the nearest cluster centre:

$$\sum_{i=0}^{n} \min_{\mu_j \in C}(||x_j - \mu_i||^2)$$

After initialisation of cluster centres, KMeans algorithm adjust students closest to a particular centre (the process often visualised with the concept of Voronoi diagrams), as members of a corresponding cluster. After that, the cluster centre position is shifted to the mean position of the cluster members, and the procedure repeats.

MeanShift clustering algorithm aims at discovering clusters in a smooth density of samples. Unlike KMeans algorithm, there is a possibility to start not with a pre-defined number of clusters, but with a pre-defined method of smoothing the field (parametrised by the ***estimate_bandwidth*** function). This is helpful if the number of clusters cannot be estimated, e.g. by using visualisation.

GaussianMixture clustering algorithm is a ***probabilistic*** model which assumes that all instances are generated from a finite mixture of Gaussian distributions with unknown parameters. The goal of the algorithm is first to reconstruct these parameters, and then to predict the probability of contributing a given instance to each of the clusters.

Finally, AgglomerativeClustering clustering algorithm uses the ***bottom-up approach***: it starts from the number of clusters equal to the number of instances and tries to merge them by minimally increasing the "linkage distance" (I used the default Ward linkage which minimises the sum of squared Euclidian distances within the cluster).

As we see, the chosen clustering algorithms have ***very different*** structures, so choosing between them may help to overcome the "No Free Lunch" theorem.

Step 2 requires determination of the learning parameters for each student. Based on the Power Law of Learning, I expect that the error rate ***E(x)*** for an ***individual*** student will be the *power law* function of the number ***x*** of student's attempt. As discussed before in ***Metric*** section, the appropriate algorithm is the non-linear regression based on minimisation of the C-statistic [19, 20].

Finally, step 3 does not need machine learning algorithms only using the outcomes of steps 1 and 2. Throughout this step, the statistical significance is estimated from the two-sided t-test with unequal sample sizes and variances (also known as Welch's t-test).

Benchmark

I selected benchmark model based on the finding of [18] that some students showed explicit "gaming" behaviour when using ASSISTments. An example of "gaming" action is the student Mary who, according to [13], "used the system 4 hours and 17 minutes, finished 114 items with 20% correct ... went through 356 scaffolding questions with 20% correct and asked for 705 hints, which is enormous compared to her classmates". As a result, students could be grouped into at least *two* clusters - "gaming" vs "non-gaming" students.

According to my hypothesis, during the "gaming" periods students click on buttons very fast. This should result in higher percentage of attempts and hints with very small instance durations. To quantify this hypothesis, during ***Step 1*** I constructed two special features that trace "gaming" behaviour:
- **'frac_3s_atts'** — fraction of attempts answered within a concise time (3 seconds or less);

- **'frac_1s_hints'** — fraction of hints looked by a concise time (1 second or less).
After that, I define "gaming" cluster by applying requirement:

'frac_3s_atts' > *atts_threshold* OR **'frac_1s_hints' > *hints_threshold***,

where ***atts_threshold*** and ***hints_threshold*** are the numbers between 0.001 and 0.999. Consequently, all other students are chosen to "non-gaming" cluster. Parameters ***atts_threshold*** and ***hints_threshold*** are derived from maximising our clustering metric score, Silhouette Coefficient.

This simple benchmark model shows a high score, ***0.4473***, with corresponding ***atts_threshold*** = 0.8172 and ***hints_threshold*** = 0.7342. But the obtained "gaming" cluster contains ***only 14 students*** (out of 8,980). So in fact, the benchmark clustering model ***does not reveal real clustering structure*** providing a group of several clustered outliers instead. In **Methodology** section, I describe more exact clustering method based on KMeans algorithm (finding it to be the best compared with MeanShift, GaussianMixture and AgglomerativeClustering algorithms).

III. Methodology *(approx. 3-5 pages)*

Data Preprocessing

Because ASSISTments datasets are huge (1.6 GBytes for a sum of 2004-2005, 2005-2006 and 2006-2007 years datasets) and contain much information irrelevant for further study, I start with pre-processing step that:

- selects 9 features relevant for further analysis (**'Anon Student Id', 'Session Id', 'Duration (sec)', 'Student Response Type', 'Problem Name', 'Problem View', 'Attempt At Step', 'Outcome', 'Day'**). All these features (except **'Anon Student Id'** and **'Day'**) are converted to numbers, and the **'Day'** feature is obtained from **'Time'** by removing hh: mm: ss information.

- combines together 2004-2005, 2005-2006 and 2006-2007 years datasets (2,717,403 instances in total);

- constructs a new relevant feature — the number ***x*** of times that a student attempted to solve a given problem — and creates visualisation shown in **Exploratory Visualization** subsection;

- writes constructed dataset to a gzipped HDF5 format file ***data.hdf.gz*** (17 Mbytes) that contains 2,717,403 instances (all present in initial ASSISTments datasets) and nine features described above.

I do not find any abnormalities in these data and thus use ***data.hdf.gz*** for further analysis.

Implementation

After preprocessing stage, I proceed with the primary project implementation. As discussed before, it splits on steps 1, 2, and 3.

Step 1 starts with preparation of preprocessed data. For each of 8,980 students in ASSISTments, I engineered the following features:

- **'num_sess'** — number of unique sessions opened;

- **'num_days'** — number of days used the system;

- **'num_probs'** — number of unique problems attempted;

- **'num_atts'** — total number of attempts made;

- **'num_hints'** — total number of hints requested;

- **'frac_corr_atts'** — fraction of correctly answered attempts;

- **'frac_3s_atts'** — fraction of attempts answered within a concise time (3 seconds or less);

- **'frac_1s_hints'** — fraction of hints looked by a concise time (1 second or less);

- **'time_atts'** — total time spent on attempts;

- **'time_hints'** — total time spent on hints;

- **'max_probl_views'** — number of maximal problem views averaged for all assessed problems;

- **‘max_atts’** — number of maximal attempts averaged for all assessed problems.

Further study of data distribution with **pd.describe()** and **pd.plotting.scatter_matrix()** shows substantial deviations from the Gaussian distribution. Because clustering algorithms performance strongly depends on distance scalings, I rescale most[1] of the features with MinMaxScaler after log-transformation. After that, I ran clustering algorithms for different subsets of features and reported the algorithm with the largest Silhouette Coefficient.

Step 2 determines learning parameters. For each student, I adjusted each of their attempts the error number ***E*** (0 for a ‘CORRECT’ and 1 for an ‘INCORRECT’ attempt) and the number ***x*** of student’s attempt (implemented during the pre-processing stage). After that, I fit the obtained sequence ***E(x)*** with the *power law* function representing the “Power Law of Learning”. Both power law model parameters — learning parameter ***d*** and difficulty parameter ***b*** for each student — are determined from non-linear regression based on minimisation of C-statistic [19, 20].

Finally, ***Step 3*** combines the results of the first two steps summarising the typical learning policy within each cluster and the resulting learning parameter for that procedure.

Refinement

In this section, I describe the improvement made during ***Step 1***. The goal of this step is to split students into ASSISTments clusters given their interaction pattern with the system. Because my benchmark model does not provide the significant result (although its best obtained Silhouette Coefficient is quite high, ***0.4473***, the corresponding clusters contain 14 and 8,966 students, so it can only be used to mark a small group of outliers), I proceed with more detailed clustering model.

I started with KMeans clustering algorithm because of its speed and reasonable performance. Because of the well-known “Curse of dimensionality”, I do not run KMeans algorithm for the *whole* sequence of features introduced in **the Implementation** section. Instead, I started with the pair of features having *the most* significant Silhouette Coefficient. After running ~6 minutes, KMeans algorithm detected two clusters of comparable size (with 7,686 and 1,294 students, respectively) with the Silhouette Coefficient ***0.6946***, much higher than for our benchmark model (***0.4473***).

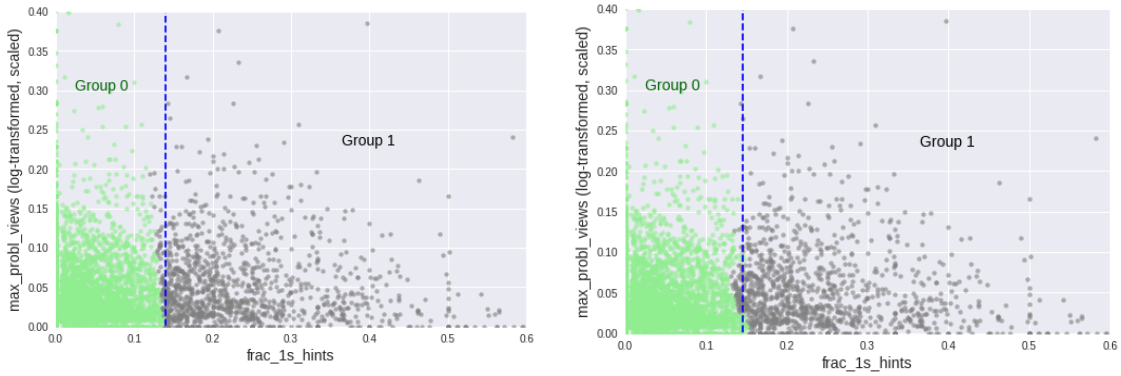
Then, I run the MeanShift, GaussianMixture and AgglomerativeClustering algorithms with the same pair of columns, **‘frac_1s_hints’** and **‘max_probl_views’**. During these runs, the main MeanShift parameter bandwith was based on the result of the estimate_bandwidth function with ***quantile*** parameter ranging from 0.01 to 0.99 with the step 0.01. Because MeanShift algorithm tends to produce tiny clusters for large Silhouette Coefficient values, I also restricted this algorithm to create the largest group containing *no more than 85%* of the total ASSISTments students, the best result from the corresponding KMeans algorithm. As a result of this restriction, the largest MeanShift score is only ***0.5470*** (compared to ***0.9008*** without the limitation — however, the effect without the restriction is meaningless because it provides 2 clusters, one of them with just two students). Because the running time for KMeans, GaussianMixture and AgglomerativeClustering algorithms increases substantially very large number of clusters, I run all of them run with rather moderate number of clusters, from 2 to 5. The largest score for GaussianMixture is only ***0.3924*** (for a model with 5 clusters). Together with the MeanShift score, *it is far too small for further tuning,* so I will not use them further during this project. On the other hand, the largest score of the AgglomerativeClustering algorithm is ***0.6976***, slightly larger than for KMeans (***0.6946***), and

1 Note that **‘frac_corr_atts’**, **‘frac_3s_atts’** and **‘frac_1s_hints’** are already scaled between 0 and 1 and therefore do not need rescaling; to isolate their outliers I have not log-transformed them as well.

the largest cluster (of 2) contains ***86.5%*** of all students, also quite similar to KMeans result (***85.6%***). However, because the AgglomerativeClustering algorithm does not produce a significant improvement over KMeans but it ~8 times longer, ***I will only use KMeans for further analysis***.

Then, I analysed the dataset with remaining features added one by one, to improve the Silhouette Coefficient. However, because adding none of the remaining columns improved the score, I remained only **'frac_1s_hints'** and **'max_probl_views'** features responsible for the best score of ***~0.69***.

The figure below shows the outcome of the KMeans (left) and AgglomerativeClustering (right).



It contains the distribution of students over the two axes used for clustering, **'frac_1s_hints'** and **'max_probl_views'**. The obtained two clusters can be well separated by a blue vertical line corresponding to **'frac_1s_hints'** of **0.14**. Because **'frac_1s_hints'** was introduced to trace the "gaming" behaviour of students, it can be concluded that such the obtained clusters correspond to "non-gaming" (**Group 0**) versus "gaming" (**Group 1**) behaviour. Because, as noted above, **Group 0** contains ***~86%*** of all students for both algorithms, I can estimate the percentage of student resembling "gaming" behaviour as ***14%***, in accordance with the estimate of [18] that "across the systems studied, a reasonably substantial minority of students (10-40%) appear to engage in some variety of gaming behavior, at least some of the time".

Next, I further clusterized **Group 0**, trying to find the pair of columns for which the score is the largest. By using KMeans, it is best split by columns **'frac_3s_atts'** and **'max_probl_views'** with maximal score ***0.714***. Similar to initial splitting, it sorts out 391 students with large **'frac_3s_atts'** (approximately corresponding to **'frac_3s_atts'** > 0.14) (**Group 2**), which I interpret as a particular kind of "gaming". Together with **Group 1**, the **total number of students showing "gaming" behaviour** is 1685, or **18.8% of all ASSISTments students**., in full accordance with the estimate of [18].

Further iterative application of the KMeans clustering algorithm to the ***most substantial*** remaining group reveals the following clusters:

- **Group 3**, 1109 students with small **'time_hints'** ("non-gaming" behaviour, small usage of hints);

- **Group 4**, 2422 students with small **'num_sess'** and **'num_probs'** ("non-gaming" behaviour, large usage of hints, small experience);

- **Group 5**, 1947 students with medium **'num_sess'** and **'num_probs'** ("non-gaming" behaviour, large usage of hints, medium experience);

- **Group 6,** 1817 students with large **‘num_sess’** and **‘num_probs**’ (“non-gaming” behaviour, large usage of hints, extensive experience).

Throughout the project, the code execution was straightforward, with no significant difficulties appeared. Although I am surprised by much smaller performances of the MeanShift and GaussianMixture algorithms than I expected from the well-known scikit-learn algorithm cheat-sheet, the observed behaviour is still consistent with the “No Free Lunch” theorem. Because, in contrast, I am particularly impressed with excellent performance with the simple KMeans algorithm, I have not investigated the reasons why the MeanShift and GaussianMixture algorithms fail. Also, I found the “iterative” clustering procedure (i.e., running the clustering algorithm on the largest cluster found in the previous step) as a handy tool to determine the very detailed structure of the dataset.

IV. Results *(approx. 2-3 pages)*

Model Evaluation and Validation

From ***Step 1*** I obtained that all 8980 ASSISTments students can be grouped by the following 6 clusters:

- **Group 1**, 1294 students with large ***‘frac_1s_hints’*** (“gaming” behaviour);

- **Group 2**, 391 students with small ***‘frac_1s_hints’*** and large ***‘frac_3s_atts’*** (“gaming” behaviour);

- **Group 3**, 1109 students with small **‘time_hints’** (“non-gaming” behaviour, small usage of hints);

- **Group 4**, 2422 students with small **‘num_sess’** and **‘num_probs’** (“non-gaming” behaviour, large usage of hints, small experience);

- **Group 5**, 1947 students with medium **‘num_sess’** and **‘num_probs’** (“non-gaming” behaviour, large usage of hints, medium experience);

- **Group 6,** 1817 students with large **‘num_sess’** and **‘num_probs**’ (“non-gaming” behaviour, large usage of hints, large experience).

For a sensitivity check, I randomly chosen a subset containing 90% of ASSISTments students and repeated the same analysis 3 times (with random_state parameters 0, 1 and 2). The results are remarkable stable: the procedure results in the same 6 clusters although the fraction of cluster population slightly changes, see Table below:

	Final run	**Test, test_size = 0.1, random_state = 0**	**Test, test_size = 0.1, random_state = 1**	**Test, test_size = 0.1, random_state = 2**
Group 1	1294 (14%)	1148 (14%)	1170 (14%)	1163 (14%)
Group 2	391 (5%)	451 (6%)	350 (4%)	372 (5%)
Group 3	1109 (12%)	1001 (12%)	954 (12%)	1047 (13%)
Group 4	2422 (27%)	2151 (27%)	2230 (28%)	2182 (27%)
Group 5	1947 (22%)	1734 (21%)	1739 (22%)	1754 (22%)
Group 6	1817 (20%)	1597 (20%)	1639 (20%)	1564 (19%)

During ***Step 2***, I calculate the values of learning parameter ***d*** and difficulty parameter ***b*** for each student. During ***Step 3***, these values were compared to different clusters obtained as a result of ***Step 1***.

Main results obtained during ***Step 3***:

- “Non-gaming" students having large "experience" with ASSISTments system (group 6) have the significantly smaller fraction of incorrect attempts (**‘frac_incorrect_atts’**) compared with "non-gaming" students with small (group 4) or medium (group 5) experience. They also make fewer attempts to solve the problem, use more hints and make more problem views. In other words, they use the ASSISTments system more efficiently.

- the difference of average learning parameter ***d*** from zero value decreases with the number of attempts made by the student;

- deviation of the averaged learning curve from power law at a large number of attempts (> 5) is systematic and present for all groups (see visualisation in Conclusion section below).

Justification

Because my benchmark clustering model is too simplistic and thus ***does not reveal real clustering structure*** (providing a group of several clustered outliers instead), it ***does not have much sense*** to compare the obtained results with the benchmark clustering model. However, the finding that students can be split into 2 groups with "gaming" (our groups 1 and 2) and "non-gaming" (our groups 3-6) behaviour can be considered as such a benchmark.

Comparison summary between these two groups:

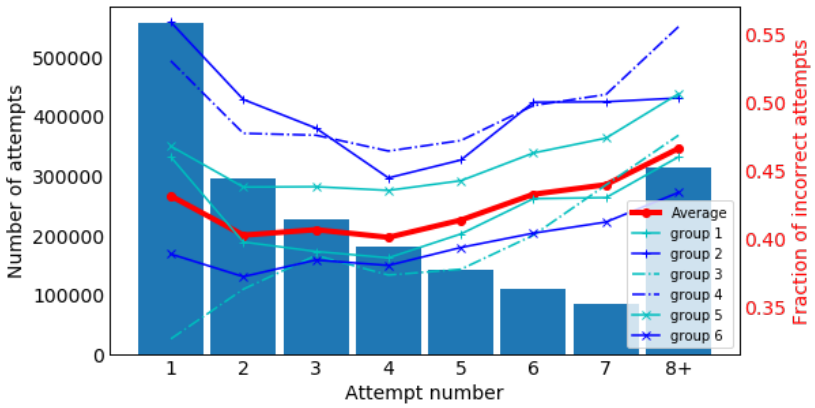
- much more significant number of attempts and hints for "gaming" group with the far more substantial fraction of very short attempts and hints (expected);

- no difference between **‘frac_incorrect_atts’** (somewhat unexpected; means that there are similar fractions of students with mixed math background in both groups).

V. Conclusion *(approx. 1-2 pages)*

Free-Form Visualization

An important feature of ASSISTments dataset is its deviation from the "Power Law of Learning”. After looking the incorrect attempts fraction from all 6 groups, I found during Step 1 it is clear that the observed increase of **‘frac_incorrect_atts’** for the larger number of attempts (> 5) is a common feature for all groups. Moreover, group 3 possess an “anti-learning” behaviour:



Reflection

During this project, I split 8980 ASSISTment students into 6 groups, much beyond previous expectations that suggested only 2 groups, with "gaming" and "non-gaming" behaviour. As a result, "gaming" students become split on 2 groups (group 1 with large ***‘frac_1s_hints’*** and

group 2 with small ***'frac_1s_hints'*** and large ***'frac_3s_atts'***) whereas "non-gaming" students are split on 4 groups (group 3 with small **'time_hints'**, group 4 with small **'num_sess'** and **'num_probs'** , group 5 with medium **'num_sess'** and **'num_probs'** and group 6 with large **'num_sess'** and **'num_probs'**). In addition to splitting students into groups, I determined the learning parameter ***d*** from the "Power Law of Learning", and used it to characterise the obtained clusters.

The first unexpected behaviour is that the learning parameter does not allow to distinguish the obtained clusters. Moreover, learning parameter tends out to zero value with the growth of user experience, such as the number of attempts.

Another issue is the presence of anti-learning, see the free-form visualisation above. I think that the reason of "anti-learning" is "overcomplexity" of some problems. I would expect that solving "simple" problems should not require many attempts. However, if the problem is too complicated, a student can "give up" and try to select "random" answer which should increase the fraction of incorrect attempts. This expectation naturally explains the fact that the effect of "anti-learning" is the strongest in group 3 where almost no hints are used — naturally, it is harder to find out the solution for a complex problem without hints.

Improvement

I expect two significant improvements of the obtained result. The first is to study the effect of "anti-learning" in more details. e.g. by grading problems by their "complexity" (in my final solution, each problem is counted equally). Second, more advanced features can be engineered to trace better "gaming" behaviour. For example, a user can experience some short period of "gaming" (e.g. due to the problem complexity or a software problem), so it may be reasonable to identify such periods and exclude it from the user's statistics to better evaluate their "real" progress.

References:

[1] Snoddy, G. S. (1926). *Learning and stability: a psychophysiological analysis of a case of motor learning with clinical applications*. Journal of Applied Psychology, 10(1):1 – 36.

[2] Crossman, E. R. F. W. (1959). *A theory of the acquisition of speed-skill*. Ergonomics, 2(2):153–166.

[3] Card, S. K., Moran, T. P., and Newell, A. (1983). *The psychology of human-computer interaction*. Lawrence Erlbaum Associates, Hillsdale, NJ.

[4] Fitts, P. M. and Posner, M. I. (1967). *Human performance*. Brooks Cole, Belmont, CA.

[5] Newell, A. and Rosenbloom, P. S. (1981). *Mechanisms of skill acquisition and the law of practice*. In Anderson, J. R., editor, Cognitive skills and their acquisition, pages 1–55. Lawrence Erlbaum Associates, Hillsdale, NJ.

[6] Anderson, J. R. (1982). *Acquisition of cognitive skill*. Psychological Review, 89(4):369–406.

[7] Nwana, H. S. (1990). *Intelligent Tutoring Systems: an overview*. Artificial Intelligence Review, 4, 251-277.

[8] Hämäläinen, W., Vinni, M. (2006) *Comparison of Machine Learning Methods for Intelligent Tutoring Systems*. In: Ikeda M., Ashley K.D., Chan TW. (eds) Intelligent Tutoring Systems. ITS 2006. Lecture Notes in Computer Science, vol 4053. Springer, Berlin, Heidelberg.

[9] Romero, C., & Ventura, S. (2010). *Educational Data Mining: A Review of the State of the Art*. Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on. 40. 601 - 618.

[10] Shahiri, A. M., Wahidah, H., Nur'aini, A.R. *A Review on Predicting Student's Performance Using Data Mining Techniques*. Procedia Computer Science, Volume 72, 2015, Pages 414-422.

[11] Razzaq, L., Feng, M., Nuzzo-Jones, G., Heffernan, N.T., Koedinger, K. R., Junker, B., Ritter, S., Knight, A., Aniszczyk, C., Choksey, S., Livak, T., Mercado, E., Turner, T.E., Upalekar. R,

Walonoski, J.A., Macasek. M.A., & Rasmussen, K.P. (2005). *The Assistment Project: Blending Assessment and Assisting*. In C.K. Looi, G.McCalla, B. Bredeweg, & J. Breuker (Eds.) Proceedings of the 12th Artificial Intelligence in Education. Amsterdam: ISO Press. pp. 555–562.

[12] Koedinger, K.R., Baker, R.S.J.d., Cunningham, K., Skogsholm, A., Leber, B., Stamper, J. (2010) *A Data Repository for the EDM community*: The PSLC DataShop. In Romero, C., Ventura, S., Pechenizkiy, M., Baker, R.S.J.d. (Eds.) Handbook of Educational Data Mining. Boca Raton, FL: CRC Press.

[13] Feng, M., Heffernan, N.T, Koedinger, K.R. (2006). *Addressing the testing challenge with a web-based e-assessment system that tutors as it assesses*. In Proceedings of the 15th International World Wide Web Conference. pp. 307-316. ACM Press: New York, NY. 2006.

[14] Feng, M., Heffernan, N., Beck, J, & Koedinger, K. (2008) *Can we predict which groups of questions students will learn from?* In Baker & Beck (Eds.). Proceedings of the 1st International Conference on Education Data Mining. pp.218-225. Montreal, 2008.

[15] Feng, M., Heffernan, N. T., & Koedinger, K. R. (2006). *Predicting state test scores better with intelligent tutoring systems: developing metrics to measure assistance required*. In Ikeda, Ashley & Chan (Eds.) Proceedings of the Eighth International Conference on Intelligent Tutoring Systems. Springer-Verlag: Berlin. pp 31–40.

[16] Feng, M., Heffernan, N.T, Mani, M., & Heffernan C. (2006). *Using Mixed Effects Modeling to Compare Different Grain-Sized Skill Models*. In Beck, J., Aimeur, E., & Barnes, T. (Eds). Educational Data Mining: Papers from the AAAI Workshop. Menlo Park, CA: AAAI Press. pp. 57-66.

[17] Feng, M., Heffernan, N. (2007) *Assessing Students' Performance Longitudinally: Item Difficulty Parameter vs. Skill Learning Tracking*. Paper presented at the National Council on Educational Measurement 2007 Annual Conference, Chicago.

[18] Baker, R., Walonoski, J., Heffernan, N., Roll, I., Corbett, A., Koedinger, K. (2008) *Why Students Engage in "Gaming the System" Behavior in Interactive Learning Environments*. Journal of Interactive Learning Research, v19 n2 p185-224.

[19] Cash, W. (1979) *Parameter estimation in astronomy through application of the likelihood ratio*. Astrophysical Journal v228, p. 939-947.

[20] Statistics in X-ray spectral fitting package (XSPEC), Appendix B of XSPEC manual, https://heasarc.gsfc.nasa.gov/docs/xanadu/xspec/manual/XSappendixStatistics.html