Analytics from Scratch:

Learning from your learners

Nick Ettlinger

Who is Nick?

- Data analyst at EdX
- Degree in Statistics and Machine learning
- Passionate about learning

What you will learn

- How to approach your data with an analytical mindset
- Useful analytics using your Open edx data
- Code snippets to take home

Three areas we will cover

1 | Course Quality

Who are the successful learners?

What are the engaging courses?

2 | Learner Activity

Which learners have been active recently?

Which courses have the most active learners?

3 | Course Recommendations

Want to recommend new courses for a learner after they finish one?

Wish you had a quantitative way to look at course similarity?

Answering a question using data

- Capture the business problem
- Find and understand the necessary data
- Get the meat out of the data
- Follow-up and next steps

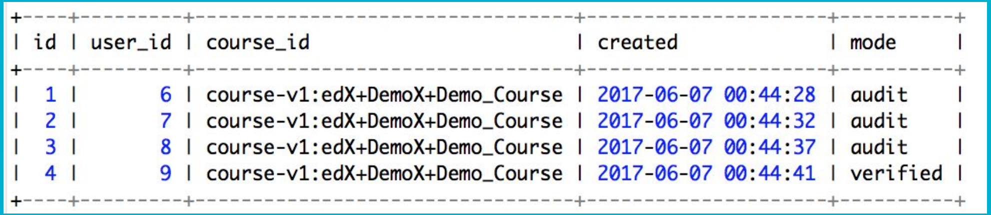
Course Quality

Course quality: business understanding

Course completion rate is a metric that counts the percent of enrolled users who completed the course by passing.

- High completion rate
  - Spread the word
  - Learn why learners like it?
- Low completion
  - Split into pieces
  - Adjust difficulty level
  - Learners stuck somewhere?

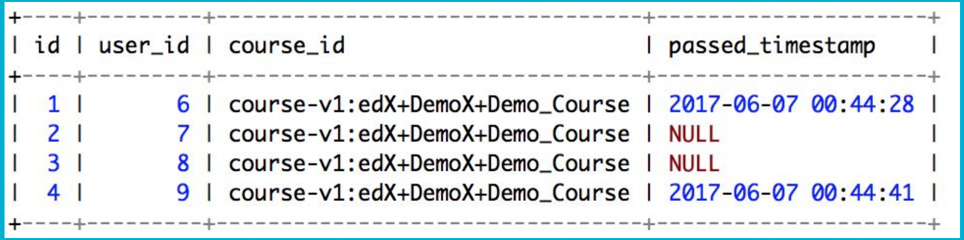
How do we know if a learner has completed a course?



```
+----+---------+-----------------------------------+---------------------+----------+
| id | user_id | course_id                         | created             | mode     |
+----+---------+-----------------------------------+---------------------+----------+
|  1 |       6 | course-v1:edX+DemoX+Demo_Course | 2017-06-07 00:44:28 | audit    |
|  2 |       7 | course-v1:edX+DemoX+Demo_Course | 2017-06-07 00:44:32 | audit    |
|  3 |       8 | course-v1:edX+DemoX+Demo_Course | 2017-06-07 00:44:37 | audit    |
|  4 |       9 | course-v1:edX+DemoX+Demo_Course | 2017-06-07 00:44:41 | verified |
+----+---------+-----------------------------------+---------------------+----------+
```

student_courseenrollment tracks the current enrollment state for all learners in all courses.

A learner has an enrollment if they have started a course



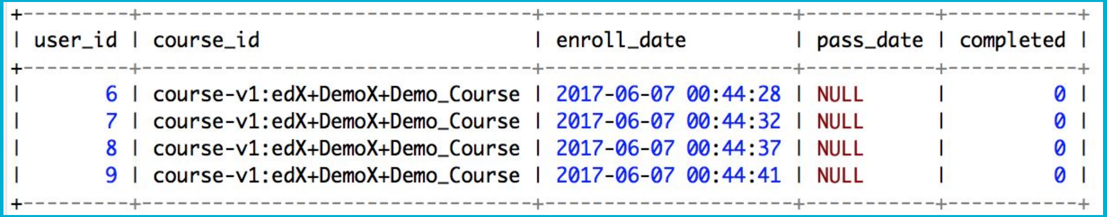
```
+----+---------+-----------------------------------+---------------------+
| id | user_id | course_id                         | passed_timestamp    |
+----+---------+-----------------------------------+---------------------+
|  1 |       6 | course-v1:edX+DemoX+Demo_Course | 2017-06-07 00:44:28 |
|  2 |       7 | course-v1:edX+DemoX+Demo_Course | NULL                |
|  3 |       8 | course-v1:edX+DemoX+Demo_Course | NULL                |
|  4 |       9 | course-v1:edX+DemoX+Demo_Course | 2017-06-07 00:44:41 |
+----+---------+-----------------------------------+---------------------+
```

grades_persistentcoursegrade tracks course grade for all learners in all courses.

Learners with a passed_timestamp that is not null have completed the course or had reached a passing grade at some time in the past.

User completion status

Each row represents a student in a course

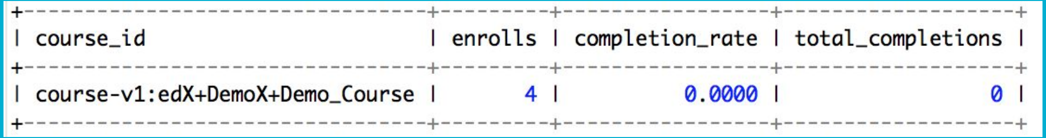


```
+---------+----------------------------------+---------------------+-----------+-----------+
| user_id | course_id                        | enroll_date         | pass_date | completed |
+---------+----------------------------------+---------------------+-----------+-----------+
|       6 | course-v1:edX+DemoX+Demo_Course | 2017-06-07 00:44:28 | NULL      |         0 |
|       7 | course-v1:edX+DemoX+Demo_Course | 2017-06-07 00:44:32 | NULL      |         0 |
|       8 | course-v1:edX+DemoX+Demo_Course | 2017-06-07 00:44:37 | NULL      |         0 |
|       9 | course-v1:edX+DemoX+Demo_Course | 2017-06-07 00:44:41 | NULL      |         0 |
+---------+----------------------------------+---------------------+-----------+-----------+
```

```
SELECT
  enroll.user_id,
  enroll.course_id,
  created AS enroll_date,
  passed_timestamp AS pass_date,
  CASE
    WHEN passed_timestamp IS NOT NULL THEN 1
    ELSE 0
  END AS completed
FROM
  edxapp.student_courseenrollment AS enroll
    LEFT JOIN
  (SELECT
    user_id, course_id, passed_timestamp
  FROM
    edxapp.grades_persistentcoursegrade
  WHERE
    passed_timestamp IS NOT NULL) AS grades
ON grades.user_id = enroll.user_id
    AND grades.course_id = enroll.course_id
```

Course completion rates

Each row represents a course



```
+-----------------------------------+---------+-----------------+-------------------+
| course_id                         | enrolls | completion_rate | total_completions |
+-----------------------------------+---------+-----------------+-------------------+
| course-v1:edX+DemoX+Demo_Course   |       4 |          0.0000 |                 0 |
+-----------------------------------+---------+-----------------+-------------------+
```

```
SELECT
  course_id, COUNT(*) AS enrolls,
  AVG(completed) AS completion_rate,
  SUM(completed) AS total_completions
FROM
  (SELECT
    enroll.user_id,  enroll.course_id,
      created AS enroll_date,
      passed_timestamp AS pass_date,
      CASE
        WHEN passed_timestamp IS NOT NULL THEN 1
        ELSE 0 END AS completed
  FROM
    edxapp.student_courseenrollment AS enroll
  LEFT JOIN (SELECT
    user_id, course_id, passed_timestamp
  FROM
    edxapp.grades_persistentcoursegrade
  WHERE
    passed_timestamp IS NOT NULL) AS grades ON
grades.user_id = enroll.user_id
    AND grades.course_id = enroll.course_id) A
GROUP BY course_id
```

Follow up and extensions

Low ≠ Bad Course

High ≠ Good Course

Typical course: 1% - 5%

Possible extensions

- Adjustments for learners who haven't had time to finish yet.
- Breaking out the completion rate per education level to account for learner ability level.
- Take a look at a high completion rate course and see why learners find it particularly engaging.

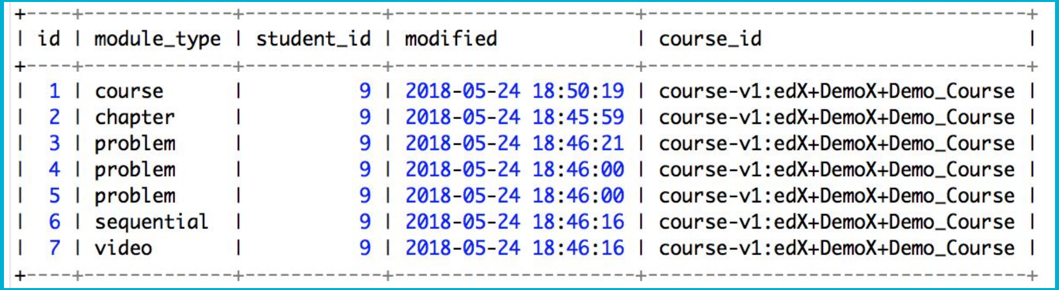
Active Learners

Learner activity leads to learning

Looking at users who have and haven't been on your platform lately can help inform a number of decisions.

- Learners: Who should you email / contact
- Courses: Current activity in my course
- Platform: Monthly active users is good for comparisons

Showing up is the first step



```
+----+-------------+------------+---------------------+----------------------------------+
| id | module_type | student_id | modified            | course_id                        |
+----+-------------+------------+---------------------+----------------------------------+
|  1 | course      |          9 | 2018-05-24 18:50:19 | course-v1:edX+DemoX+Demo_Course |
|  2 | chapter     |          9 | 2018-05-24 18:45:59 | course-v1:edX+DemoX+Demo_Course |
|  3 | problem     |          9 | 2018-05-24 18:46:21 | course-v1:edX+DemoX+Demo_Course |
|  4 | problem     |          9 | 2018-05-24 18:46:00 | course-v1:edX+DemoX+Demo_Course |
|  5 | problem     |          9 | 2018-05-24 18:46:00 | course-v1:edX+DemoX+Demo_Course |
|  6 | sequential  |          9 | 2018-05-24 18:46:16 | course-v1:edX+DemoX+Demo_Course |
|  7 | video       |          9 | 2018-05-24 18:46:16 | course-v1:edX+DemoX+Demo_Course |
+----+-------------+------------+---------------------+----------------------------------+
```

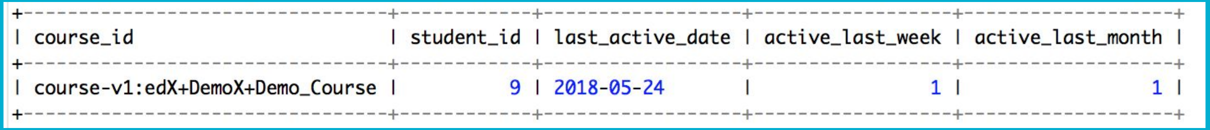
courseware_studentmodule tracks the individual block state for all learners in all courses

A “block” is a small section of course content

Blocks are modified each time a learner loads a new page in the LMS

How active are you?

Each row represents a student in a course



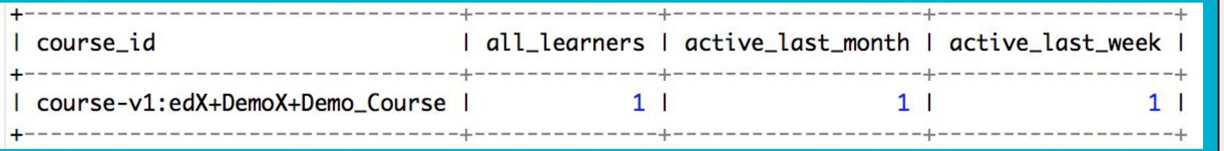
```
+-------------------------------+------------+------------------+------------------+-------------------+
| course_id                     | student_id | last_active_date | active_last_week | active_last_month |
+-------------------------------+------------+------------------+------------------+-------------------+
| course-v1:edX+DemoX+Demo_Course |          9 | 2018-05-24       |                1 |                 1 |
+-------------------------------+------------+------------------+------------------+-------------------+
```

```
SELECT
  course_id,
  student_id,
  DATE(MAX(modified)) AS last_active_date,
  CASE WHEN
      MAX(modified) >
          DATE_ADD(CURRENT_TIMESTAMP(),
        INTERVAL - 7 DAY)
    THEN 1 ELSE 0
  END AS active_last_week,
  CASE WHEN
      MAX(modified) >
          DATE_ADD(CURRENT_TIMESTAMP(),
        INTERVAL - 1 MONTH)
    THEN 1 ELSE 0
  END AS active_last_month
FROM
  edxapp.courseware_studentmodule
GROUP BY course_id , student_id;
```

Active learners per course

Each row represents a course

Results are aggregated from prior query



```
+-----------------------------------+---------------+-------------------+------------------+
| course_id                         | all_learners  | active_last_month | active_last_week |
+-----------------------------------+---------------+-------------------+------------------+
| course-v1:edX+DemoX+Demo_Course   |             1 |                 1 |                1 |
+-----------------------------------+---------------+-------------------+------------------+
```

```
SELECT
  course_id,
  COUNT(*) AS all_learners,
  SUM(active_last_month) AS active_last_month,
  SUM(active_last_week) AS active_last_week
FROM
  (SELECT
    course_id, student_id,
      CASE WHEN MAX(modified) >
          DATE_ADD(CURRENT_TIMESTAMP(),
          INTERVAL - 1 MONTH)
          THEN 1 ELSE 0
      END AS active_last_month,
      CASE WHEN MAX(modified) >
          DATE_ADD(CURRENT_TIMESTAMP(),
          INTERVAL - 7 DAY)
          THEN 1 ELSE 0
      END AS active_last_week
  FROM
    edxapp.courseware_studentmodule
  GROUP BY course_id , student_id) A
GROUP BY course_id
```

Follow up and extensions

So what?

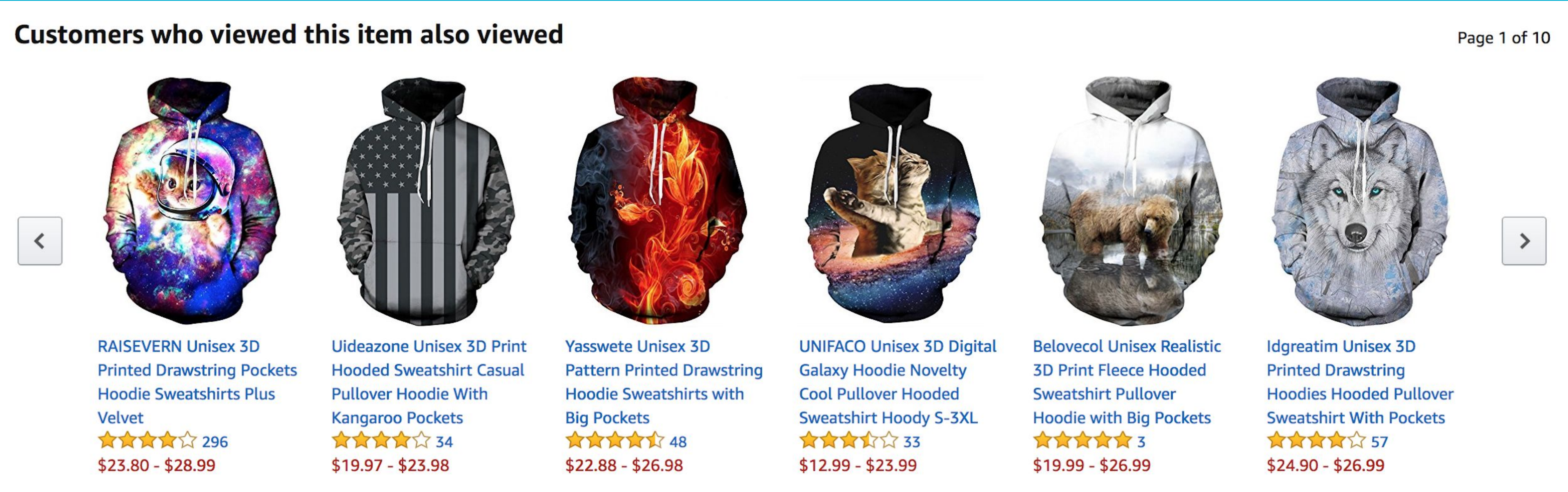
Now we know who was recently learning and who wasn't. We also know how much activity is happening in each of our courses.

Possible extensions

- Counting total active learners on whole site
- Make a list of inactive users and encourage them to come back.
- Take a look at a very active course and see why learners find it particularly engaging

Course Recommendations

“Learners who took this course also took”


Customers who viewed this item also viewed

Page 1 of 10

RAISEVERN Unisex 3D Printed Drawstring Pockets Hoodie Sweatshirts Plus Velvet
296
$23.80 - $28.99

Uideazone Unisex 3D Print Hooded Sweatshirt Casual Pullover Hoodie With Kangaroo Pockets
34
$19.97 - $23.98

Yasswete Unisex 3D Pattern Printed Drawstring Hoodie Sweatshirts with Big Pockets
48
$22.88 - $26.98

UNIFACO Unisex 3D Digital Galaxy Hoodie Novelty Cool Pullover Hooded Sweatshirt Hoody S-3XL
33
$12.99 - $23.99

Belovecol Unisex Realistic 3D Print Fleece Hooded Sweatshirt Pullover Hoodie with Big Pockets
3
$19.99 - $26.99

Idgreatim Unisex 3D Printed Drawstring Hoodies Hooded Pullover Sweatshirt With Pockets
57
$24.90 - $26.99

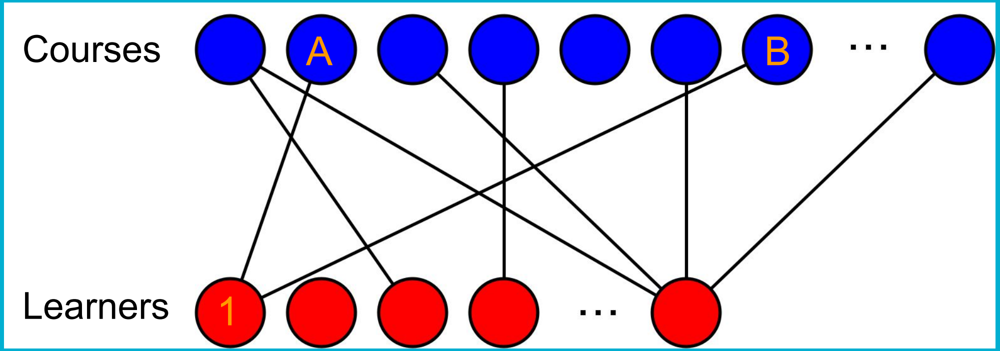
How our data is structured

Learners who enroll in multiple courses generally indicate that both those topics are interesting to them.

In aggregate, if courses A and B share a lot of the same learners, we would expect that a learner in A probably will like course B.

Our chart here visualizes such a relationship

However, it is possible to collapse the diagram, Showing only courses, not learners

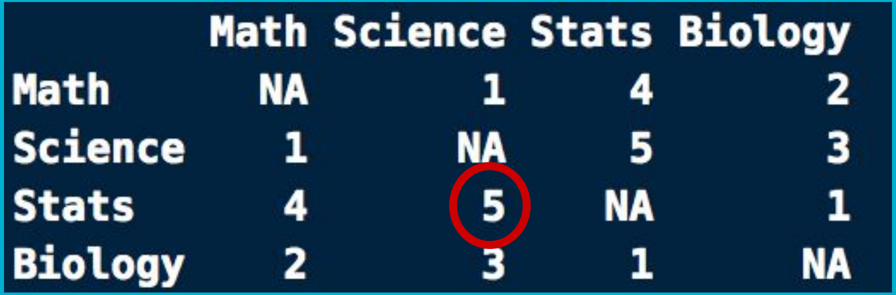


A co-enrollment matrix tells us how many learners take a given pair of courses

We will create a link between two courses if a learner enrolls in both of them.

1. Apply this to all of our data, gives us a co-enrollment matrix.
2. Stats + Science has the highest number of co-enrolling learners.
3. -> So do we suggest Science to the Stats students?

For example, five students enrolled in both the science and stats course offered by FakeX.



```
           Math Science Stats Biology
Math         NA       1     4       2
Science       1      NA     5       3
Stats         4       5    NA       1
Biology       2       3     1      NA
```

You Took:	We Suggest
Stats	Science

How do we make this matrix?

We pull a list of all enrollments, and then using a little linear algebra, we are able to collapse the data to give us co-enrollments.

1. This converts the list of user-course pairs into a matrix of one column per unique user and one row per unique course.
2. We take the cross product of that matrix, and now we have unique courses as rows and columns.

```
#this is the raw data
enrolls = run_query("
        SELECT user_id, course_id
        FROM edxapp.student_courseenrollment")
#check the data
str(enrolls)
#building the matrix
user.fac = factor(enrolls[,1])
course.fac = factor(enrolls[,2])
cm = sparseMatrix(
        as.numeric(user.fac), as.numeric(course.fac),
        dimnames = list(as.character(levels(user.fac)),
        as.character(levels(course.fac))),
        x = 1)
# calculating co-occurrences (matrix times transpose of matrix)
cv = t(cm) %*% cm
# setting self references
diag(cv) = 0
dict = unique(enrolls[,2])
dict = sort(dict)
```

Generating the recommendation

We have two ways of making a course recommendation:

1. (Unweighted - prefer popular) Using the course with the largest count of co-enrollments with your course.
2. (Weighted - adjust for popularity) Dividing the co-enrollment counts by the number of enrollments in the other course. I.e. the percent of enrollments in course B that are shared with course A.

```
typed = c(
          'UTAustinX/UT.7.01x/3T2014',
          'UQx/Write101x/3T2014')
input = cv[typed, ]
#calculate the number of co-enrollments for each of our courses
unweighted = colSums(input)
#calculate the weighted # of co-enrollments
weighted = unweighted/colSums(cv[,names(unweighted)])
#building sorted reference tables for both types
un_dict = dict[order(unweighted, decreasing = T)]
#un_dict$score = sort(unweighted, decreasing = T)
w_dict = dict[order(weighted,decreasing = T)]
#view the top 6 for each category
head(un_dict)
head(w_dict)
```

Follow up and extensions

You Took:	We Suggest
English Grammar and Style	Principles of Written English, Part 1
Foundations of Data Analysis	English for Doing Business in Asia - Speaking

So what?

- Simple framework
    - You already have the data
- Flexible framework
    - One or more input courses
    - One or more recommendations

Possible extensions

- Scoring current learners and sending personalized email suggestions.
- Make a quiz for new visitors and use the quiz responses to generate suggestions for them.
- Add “you might also be interested in..” sections to your site.

Audience participation:

What are you doing with your data?

Thank you!

Github code - https://github.com/Nickett3/OpenedX2018-Analytics/

Openedx Slack - @Nick the Data guy

Questions?