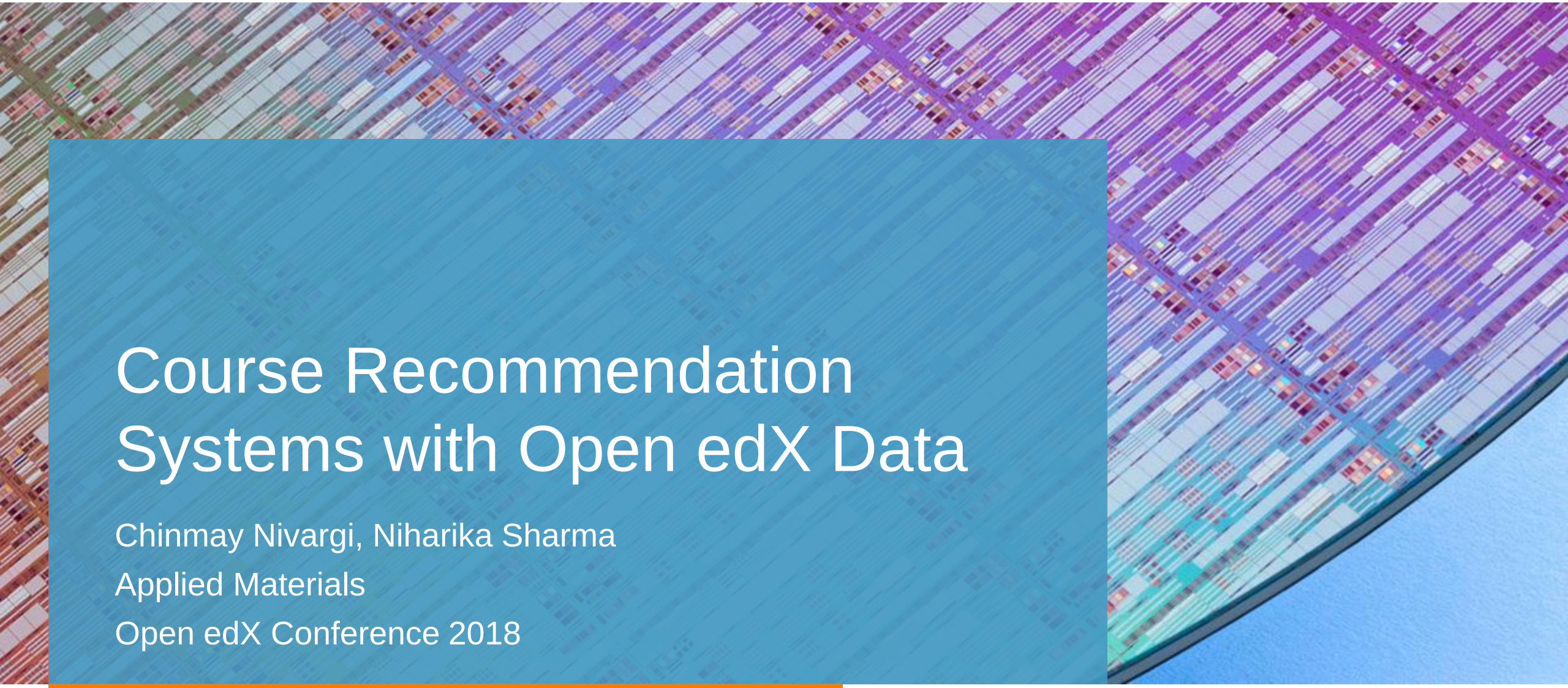


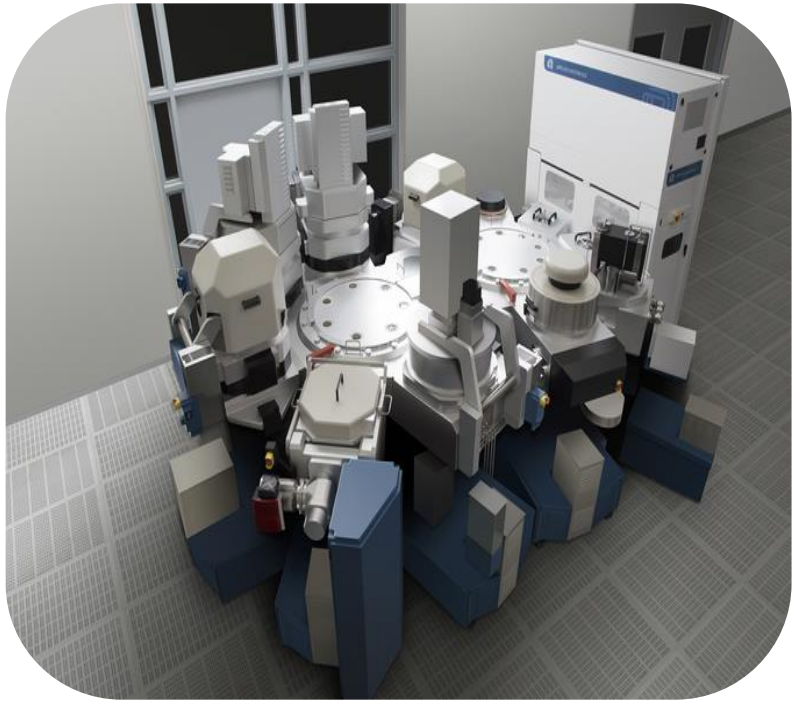
Course Recommendation Systems with Open edX Data

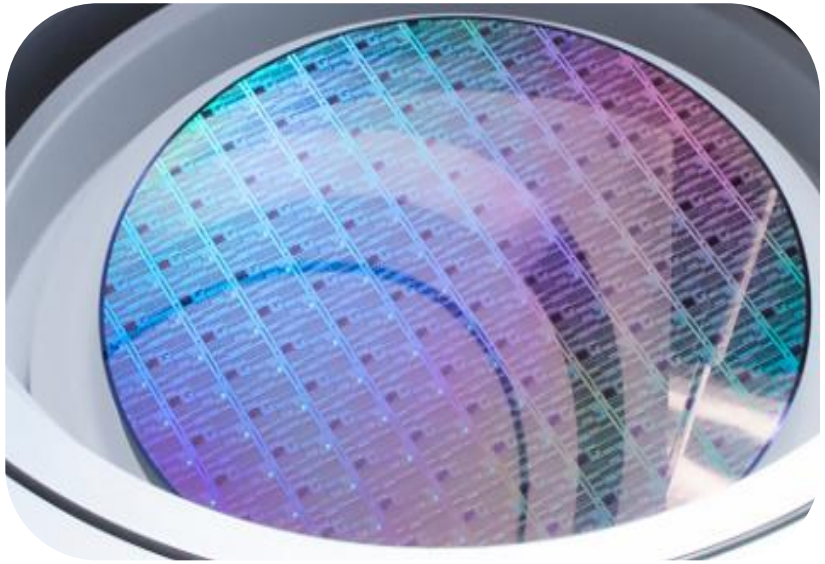
Chinmay Nivargi, Niharika Sharma
Applied Materials
Open edX Conference 2018

Chinmay Nivargi

- Product Manager
- Stanford University → Applied Materials
- Active with MOOCs and online education since 2012
- Third Open edX Conference – Stanford → Madrid → Montreal!
- Chinmay_Nivargi@amat.com, @chinmayn

APPLIED MATERIALS®





Leader in materials engineering solutions used to produce virtually every new chip and advanced display in the world

appliedx

- Self-hosted platform powered by Open edX for unique engineering, technology, business process training and learning needs
- Over 250 courses,100,000+ course instances, completed by ~20,000 users worldwide in 3 years
- In-house development team for customizations, self-service recording infrastructure worldwide, democratized content creation aided by instructional design and media production teams

Why did we need a Course Recommendation System?

- Explosion of content
- Varied, specialized roles with limited universal curricula
- Lack of resources for human curation
- Limitations with Open edX Search
- Improvement of content discovery in a learning culture, in the absence of a high touch marketing model

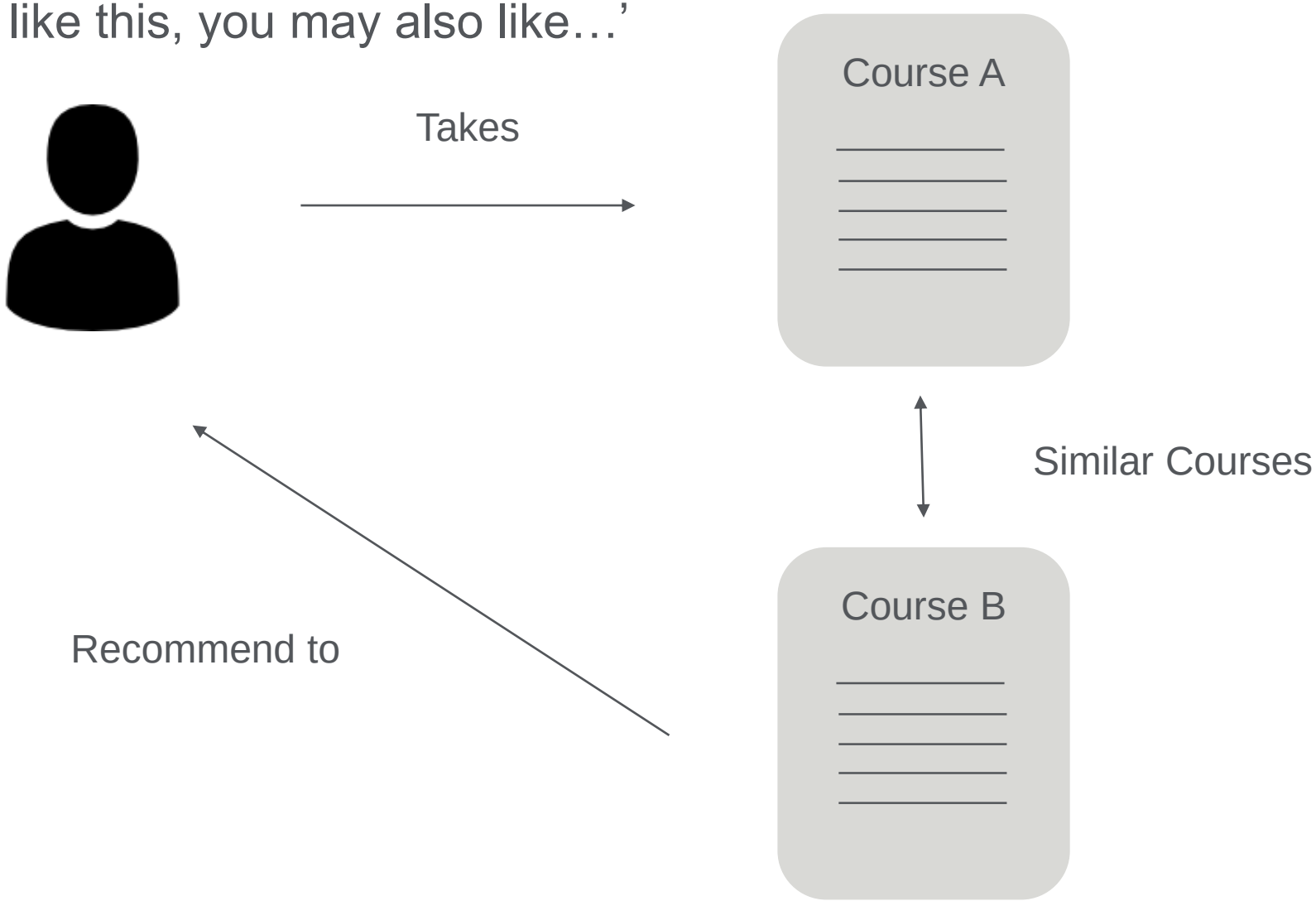
This talk

- Recommendation Systems – What are they?
- Data Sources in Open edX for Course Recommendations
- Methodology for building and deploying Recommendation Systems in Open edX
- Performance, Challenges and Extensions

Recommendation Systems

Types of Recommendation Systems

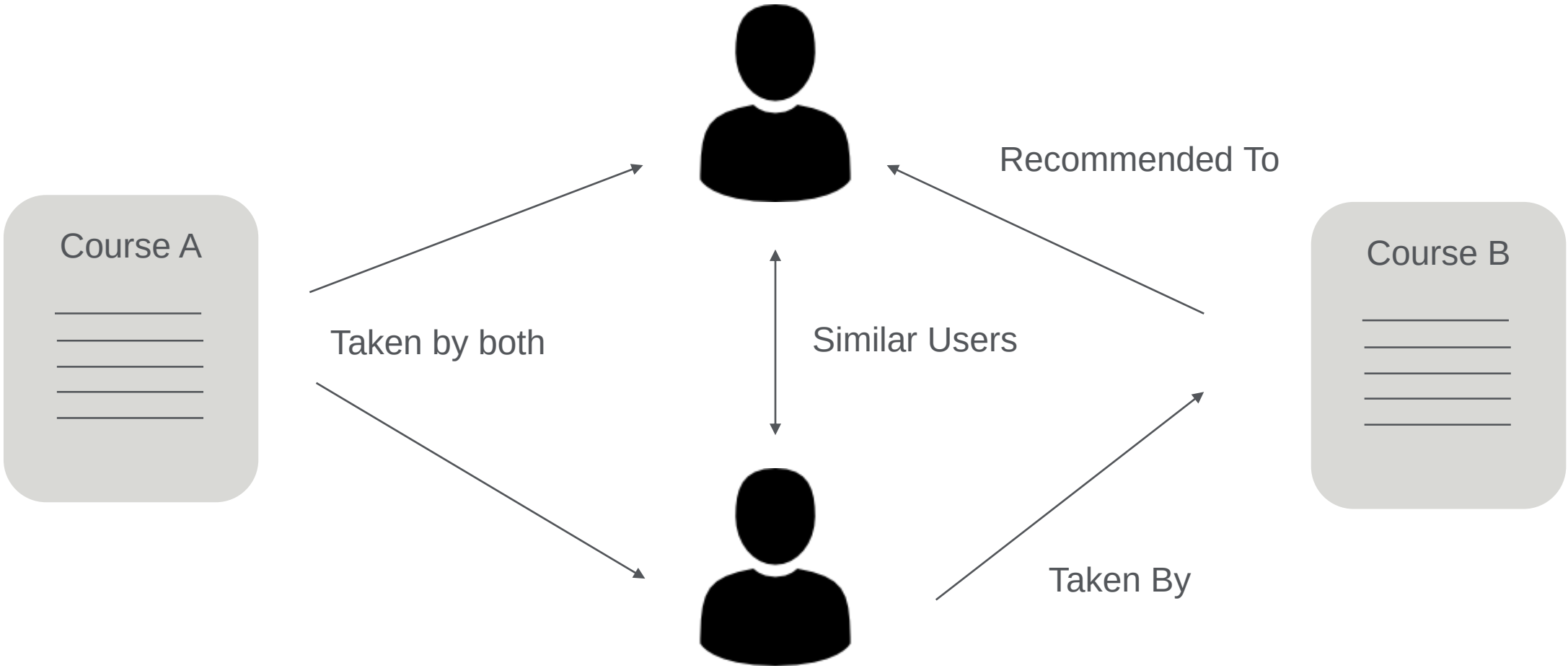
Content based – ‘if you like this, you may also like…’



Source: Icons made by Freepik from www.flaticon.com

Types of Recommendation Systems

User based ‘collaborative filter’ – ‘Users like you also liked …’



Source: Icons made by Freepik from www.flaticon.com

Content-based Recommenders

Require clean, preprocessed data with high signal-to-noise

Solves the ‘cold start’ problem for new users and content

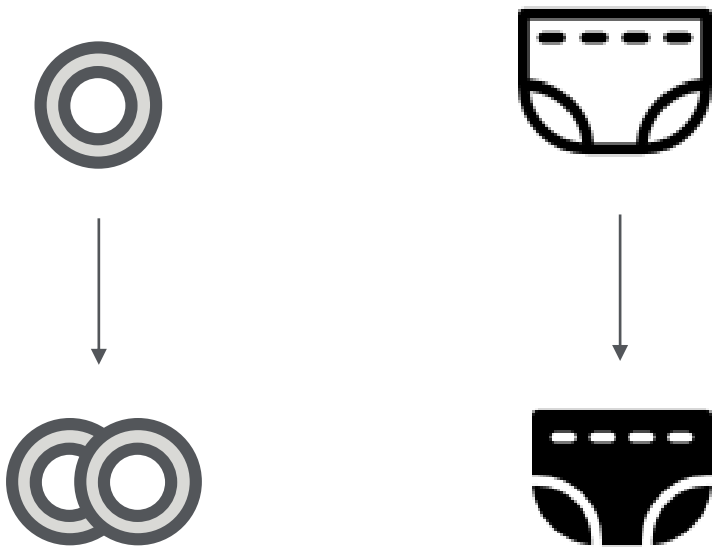
Collaborative Filters

Forgiving of lack of clean data, but requires a lot of data for meaningful results

Suffers from a cold start - cannot recommend to new users, or new content

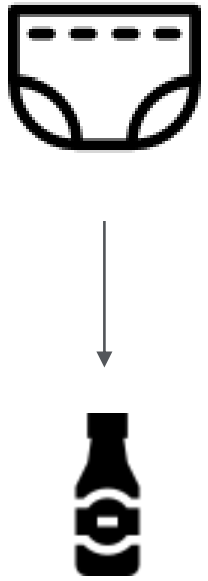
Content-based Recommenders

“If you liked The Lord of the Rings, here’s Lord of the Rings 2… if you bought diapers, here are more diapers you may like!”



Collaborative Filters

“People are more likely to buy beer, if they also bought diapers!”



Source: Icons made by Freepik from www.flaticon.com

Ideally…

Hybridizing the two approaches gives the best overall recommender.

Netflix does this and more to serve up the best recommendations

WINNER!

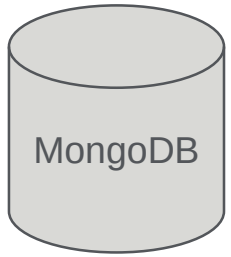
Content-based Recommender!

Why did we not go with a Collaborative Filter?

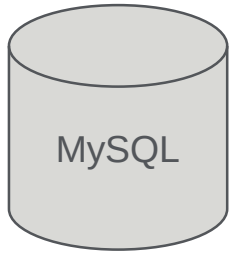
- Plenty of new users and content – cold start problem
- Lack of capturing intent – courses were, often, assigned to users, instead of selected by users
- Lack of ratings across whole catalog – capturing only implicit, unary data (did a user take a class or not)
- Potential extension outside of Open edX ecosystem – universal content comparison easier to implement with content-based recommender
- As content increases and we have more data, we can include collaborative filter as well (spoiler: we are, already)

Data Sources in Open edX

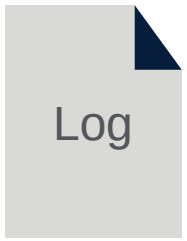
Data Sources available in Open edX



Course Metadata – name, description, topic, etc.	Modulestore
Course Content – chapters, sequentials, units	
Video Transcripts	Files (link)



Demographics (age, location, etc.)	auth_user
Course Engagement and Completion records	courseware_studentmodule
Ratings / Surveys	courseware_studentmodule



Clickstream Events	Files

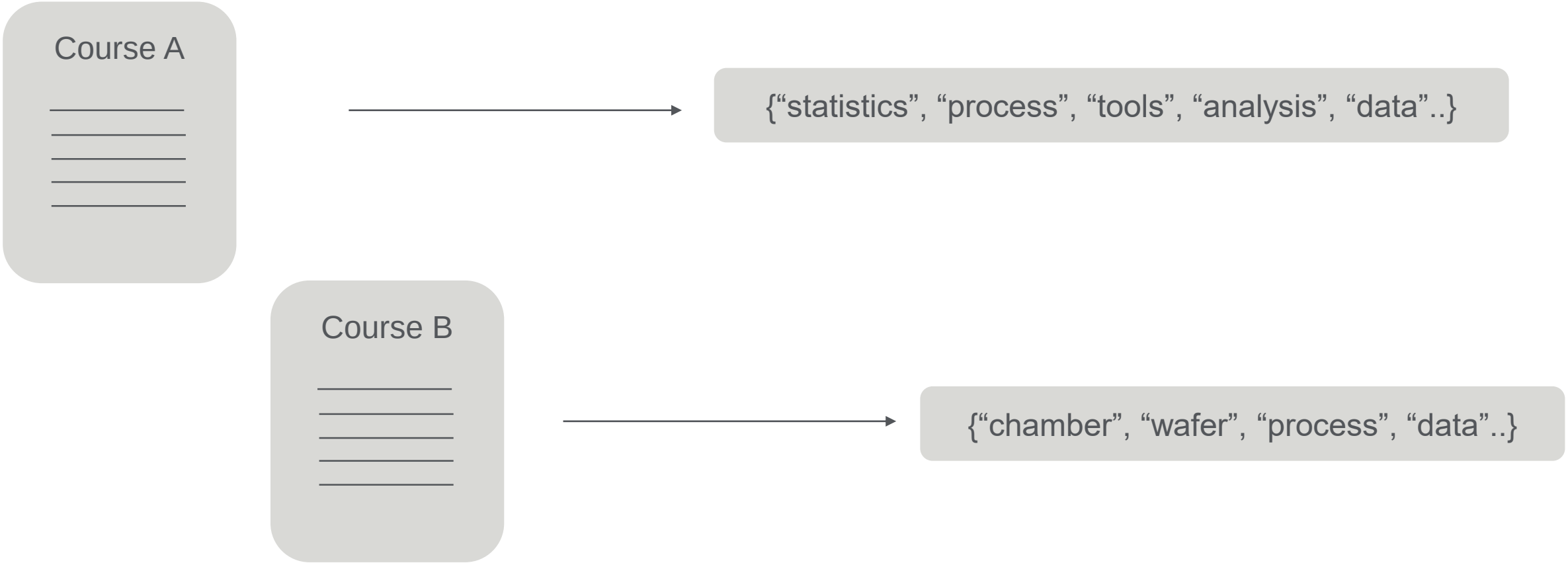
Methodology

Methodology for an Content-Based Recommender

1. Obtain as much data as you can on the course and determine appropriate weightings for the data
2. Determine a ‘similarity score’ between pairs of courses
3. Rank order courses similar to course under consideration
4. Present similar courses to user based on certain filters
5. Based on user feedback, change weighting and iterate

Methodology : How similar is Course A to Course B?

1. Scrape course data from mongoDB, or the open edX LMS directly for courses to determine a ‘bag-of-words’ that represent courses A and B (and all courses in the catalog). Normalization of the words (stemming, removing stopwords, etc.) follows.



Methodology : How similar is Course A to Course B?

2. Represent each course's bag-of-words as a vector with dimension equal to the number of words in the dictionary across the catalog.

TF-IDF (Term Frequency – Inverse Document Frequency) was used as the weighting for every word

$$w_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right)$$

$tf_{i,j}$ = number of occurrences of i in j
df_i = number of documents containing i
N = total number of documents

	Word 1	**Word 2**	**…**	**Word n**
Course A (**A**)	$w_{1,A}$	$w_{2,A}$		$w_{n,A}$
Course B (**B**)	$w_{1,B}$	$w_{2,B}$		$w_{n,B}$

Image source: deeplearning4j.org

Methodology : How similar is Course A to Course B?

3. Compute how similar the vectors are to each other using a similarity score (cosine similarity)

	Word 1	Word 2	…	Word n
Course A (**A**)	$w_{1,A}$	$w_{2,A}$		$w_{n,A}$
Course B (**B**)	$w_{1,B}$	$w_{2,B}$		$w_{n,B}$

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$$

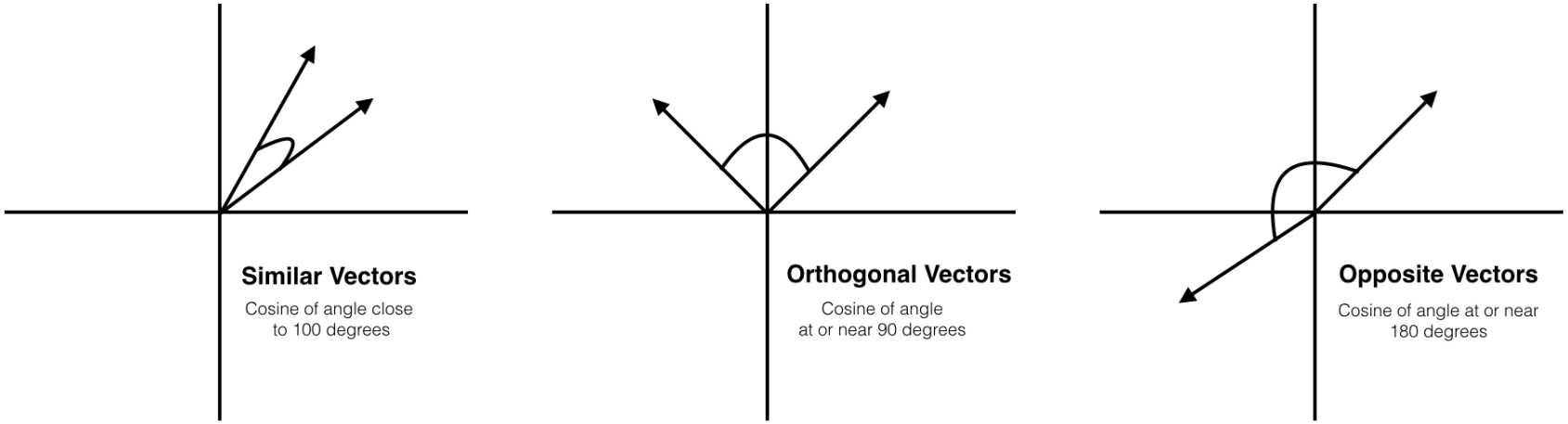
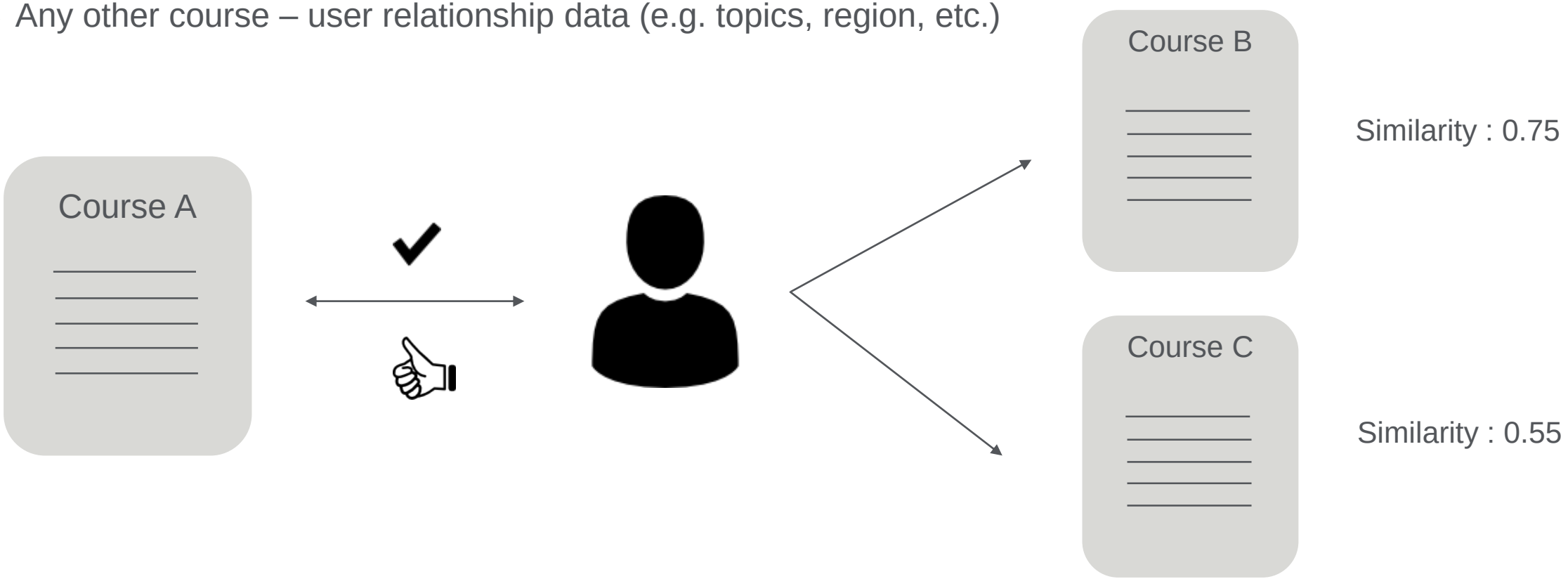


Image source: Mastering Machine Learning with Spark 2.x by Michal Malohlava, Max Pumperla, Alex Tellez

Methodology: Show Recommended Course to User

4. Once similar courses are known and rank ordered for every course pair, determine courses of interest to user based on:

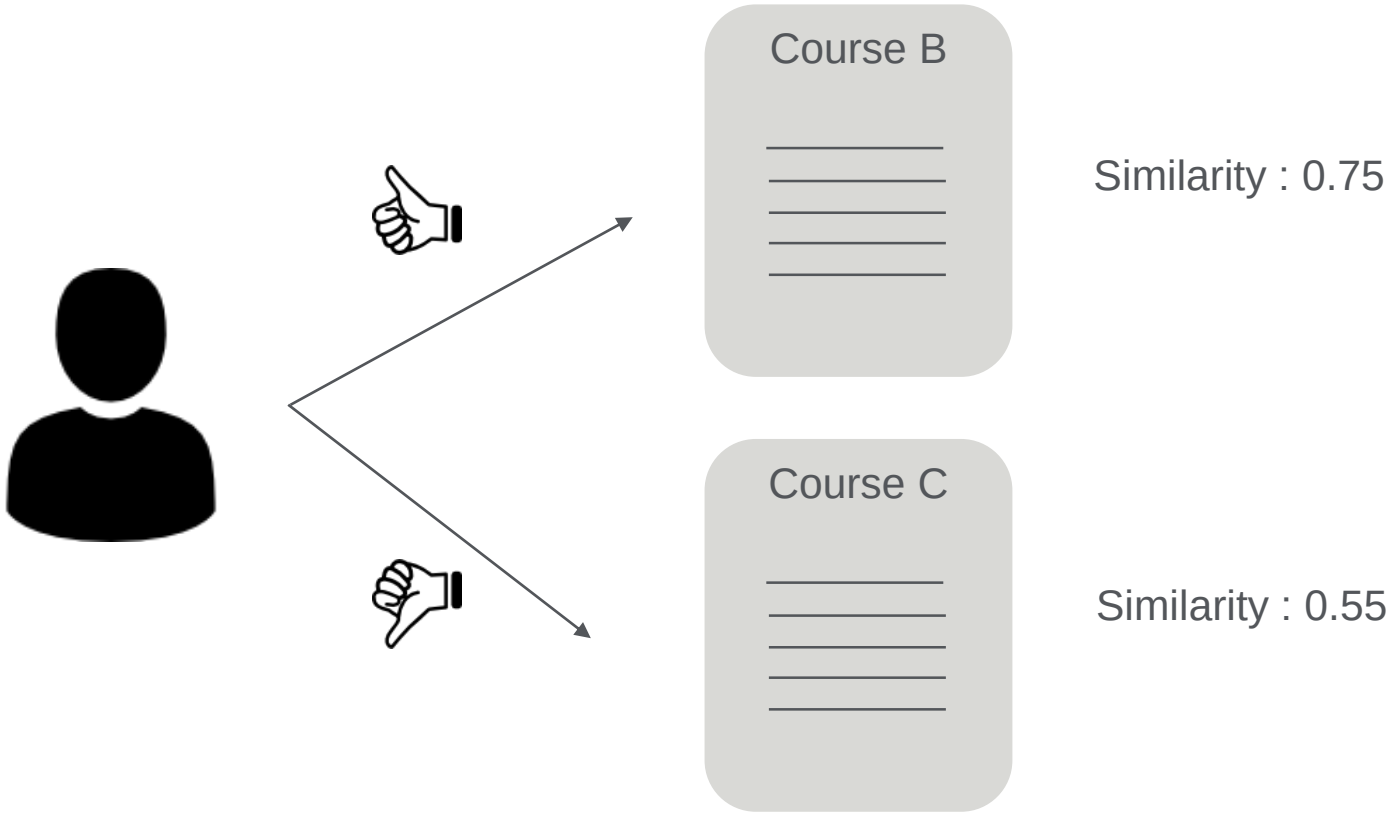
- Courses previously completed by user
- Courses rated by user (if available)
- Any other course – user relationship data (e.g. topics, region, etc.)



Source: Icons made by Freepik from www.flaticon.com

Methodology: Show Recommended Course to User

5. Present to user, measure and iterate!



Source: Icons made by Freepik from www.flaticon.com

Implementation

- At course completion event, users are presented with courses that they may like, served up using an API that pulls from a recommendations database that updates daily

- Users may also be recommended new courses directly on their dashboard based on previously completed courses

```
GET /re/v1.0/recommendations?uid=136712
{
"data": {
"Differentiation, Value, & Sustainability": 0.7482,
"Market Requirements Specification": 0.4375,
"Supply Chain Guidelines": 0.4630
},
"status": "success"
}
```

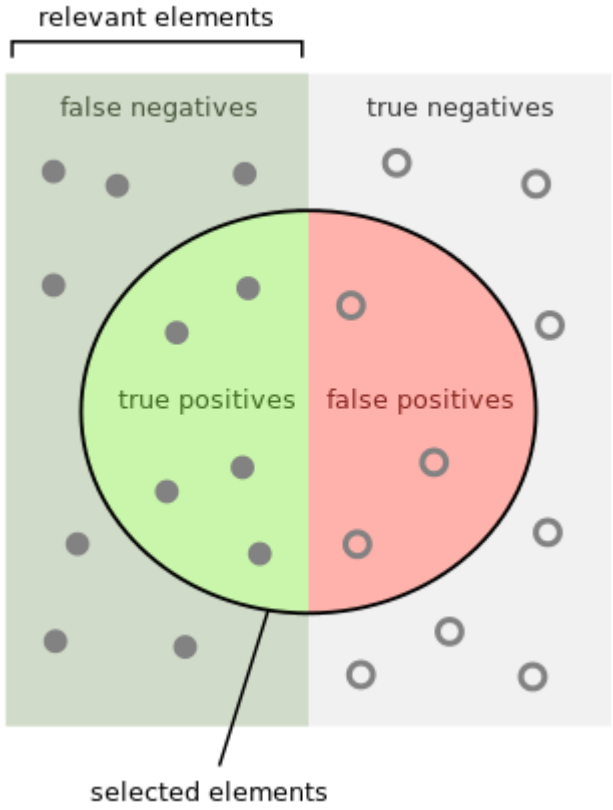
Performance, Challenges and Extensions

How good is a recommender?

‘Academic’ Metrics

- Recall

  From (split) data, how many
  of the courses taken by user are retrieved?



- Catalog Coverage

  What fraction of the course catalog can
  be recommended?

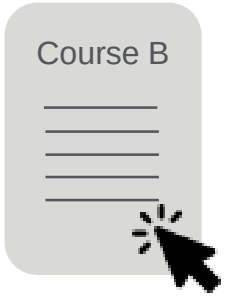
Image source: Wikipedia

How good is a Recommender?

Real World Performance

- Click-through Rate

  Percentage of Users clicking on recommended course



- Conversion Rate

  Percentage of Users enrolling in recommended course

- Rank Metrics

  How many of the top recommendations presented are relevant to user?

Source: Icons made by Freepik from www.flaticon.com

Performance Logging

Collecting data for measuring performance and tuning


```
{
"username": “Click_Bot@amat.com",
"user_id": “13672",
"current_course": "course-v1:PCM+309+309",
"action_type": “Interested",
"action_object": "PCM/PCMB3/304",
"created_date": "2018-05-14T17:27:36.972101Z"
},
{
"username": “Bitcoin_Miner@amat.com",
"user_id": “516213",
"current_course": "course-v1:PCM+309+309",
"action_type": "Not Interested",
"action_object": "course-v1:PCM+PCMC6+315",
"created_date": "2018-05-14T17:27:39.201876Z"
}
```

Learnings

Some (not-so) obvious results

- So much jargon!
- Taking an intro course leads to a recommendation of a summary course in a series (which recaps the intro)
- Some similar courses shouldn’t be similar (older versions, regional courses, etc.)
- So much, yet so little data!

Data quality is the biggest challenge!

- Too few meaningful key words!
- Standardize your course structure and descriptions. At least 100 words!
- Set up governance for your incoming content, even if organically created
- Think about your content and metadata at time of platform version upgrades

Current and Future Work

We want to…

- Hybridize content and user-based recommendations
- Leverage additional data – course ratings, organization and team information, job roles, etc.
- Incorporate user feedback (real world performance) to tune recommender
- Extend of Open edX based recommender to universal content catalog

Acknowledgements

- Data Science – Niharika Sharma
- appliedx dev team – Vijayaraghavan Sundararaman, Rahul Shenoy
- Aneesh Nainani
- Open edX Conference Team!


APPLIED
MATERIALS®
make possible