

Wild Wild Tests

MLOps World, June 2022

Intro

About me



ENGINEERING

- Founder of Tooso, acquired by TSX:CVO, now Director of AI
- Passionate about MLOps, OS contributor

AI RESEARCH & EDUCATION

- 25+ papers in top NLP/ML venues
- Co-organizer of SIGIR eCom
- Teaching MLSys at NYU Tandon

Today

- RecSys 101: what are recommender systems (RS)? Why are they important? Why is testing them hard?
- Quantitative tests: what people usually do in Information Retrieval (IR)?
- Behavioral testing: what is it? How to do it?
- The RecList package: theory and practice

NOTE: This is **NOT** a talk on *how to build recommender systems*. I'm happy to point you to relevant docs if you wish to do so!

RecList by the community for the community

RecList is an open source project originally built by practitioners at Coveo, Stanford, Bocconi and KOSA.AI.

Its development is currently supported by the generosity of our awesome friends at [Comet](#) and [Neptune](#) (and more coming soon)!



comet

Build better ML models faster



Metadata Store for MLOps

RecSys 101



RecSys are everywhere

- RS are ubiquitous in our digital life, think:
 - Movies on Netflix
 - Books (or anything else!) on Amazon
 - Grocery on Instacart
 - etc.
- The RS market will be worth ~15 BN by 2026*
- RS quality is **super important...**

Market Overview

The Recommendation Engine market was valued at USD 2.12 billion in 2020, and it is expected to reach USD 15.13 billion by 2026, registering a CAGR of 37.46% during the period of 2021-2026. With the growing amount of information over the internet along with a significant rise in the number of users, it is becoming essential for companies to search, map, and provide them with the relevant chunk of information according to their preferences and tastes.

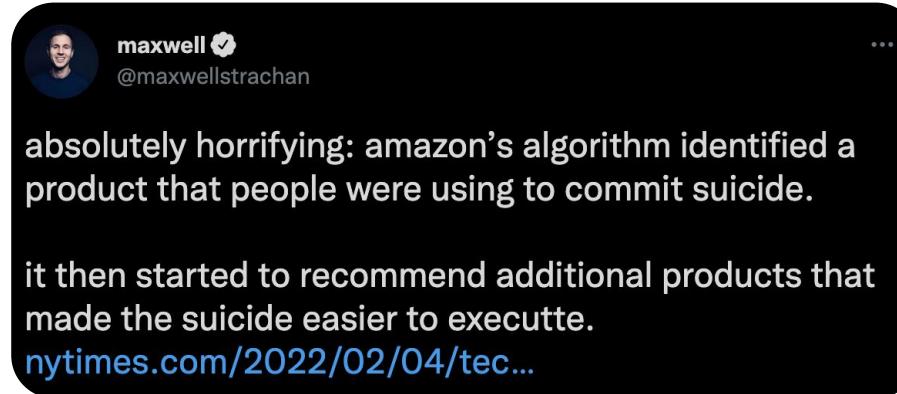
* conditions apply

38% of shoppers
stop shopping when
seeing poor
recommendations.*

* eMarketer

...and dangerous

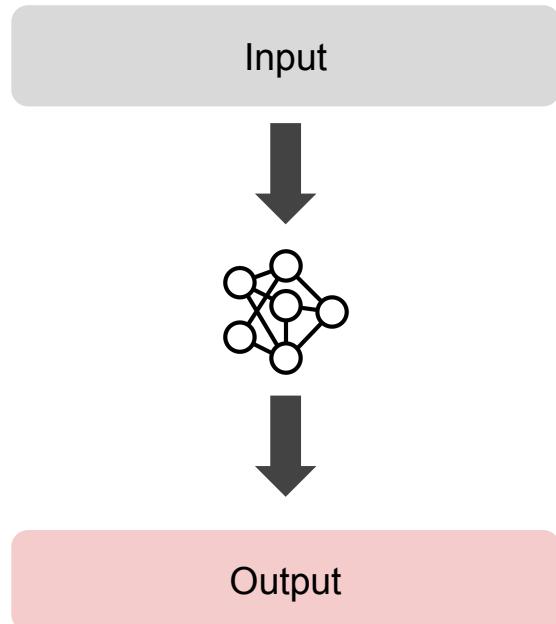
- Even when RS are *mostly right*, one mistake may have disastrous consequences for:
 - the final users...
 - ...and the RS makers



“Failure” is
(almost) *not* an
option

RecSys by use case

- RS can be understood easily by use case and input-output:



RecSys by use case

- RS can be understood easily by use case and input-output:
 - Item as input, item as output: similar vs complementary

Similar books based on genre

Spring in Action (Sponsored)

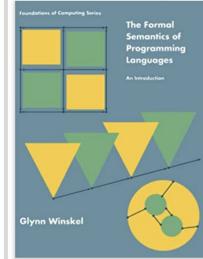
Types and Programming Languages

Essentials of Programming Languages

Book Title	Author	Edition	Type	Rating	Price
Spring in Action, Sixth Edition	Craig Walls	Sixth Edition	Paperback	4.5 stars (10 reviews)	\$53.99
Types and Programming Languages	Benjamin C. Pierce	(The MIT Press)	Hardcover	4.5 stars (66 reviews)	\$64.54
Essentials of Programming Languages	Daniel P. Friedman and Mitchell Wand	Third edition	Hardcover	4.5 stars (14 reviews)	\$73.95

Input

Formal Semantics of Programming Languages
by Glynn Winskel ▾ (Author)
4.5 stars 17 ratings
[Look inside](#) ↴



Hardcover \$45.07 Paperback \$29.98 - \$70.00 Other Se See all 5 versio

[Buy used:](#) \$29.98

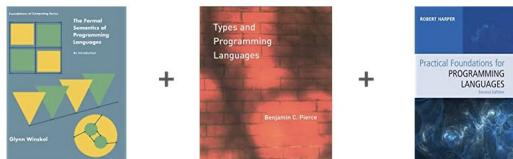
[Buy new:](#) \$70.00

In Stock.
Ships from and sold by Amazon.com.
May be available at a lower price from other sellers, p Prime shipping.

RecSys by use case

- RS can be understood easily by use case and input-output:
 - Item as input, item as output: similar vs complementary

Frequently bought together



Total price: \$183.47

Add all three to Cart

Some of these items ship sooner than the others. [Show details](#)

- This item:** Formal Semantics of Programming Languages by Glynn Winskel Paperback \$70.00
- Types and Programming Languages (The MIT Press) by Benjamin C. Pierce Hardcover \$64.54
- Practical Foundations for Programming Languages by Robert Harper Hardcover \$48.93

Input

A screenshot of an Amazon product page for 'Formal Semantics of Programming Languages' by Glynn Winskel. The page shows the book cover, author information, ratings (4.5 stars from 17 reviews), and purchase options. It includes links for 'Look inside' and 'Buy used' or 'Buy new'.

Formal Semantics of Programming Languages
by Glynn Winskel (Author)
4.5 stars
17 ratings
Look inside

Hardcover
\$45.07

Paperback
\$29.98 - \$70.00

Buy used:
\$29.98

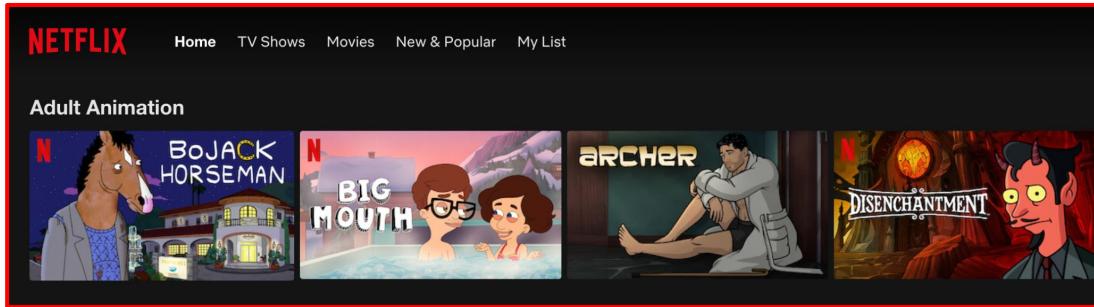
Buy new:
\$70.00

In Stock.
Ships from and sold by Amazon.com.
May be available at a lower price from other sellers, p
Prime shipping.

RecSys by use case

- RS can be understood easily by use case and input-output:
 - Item as input, item as output: similar vs complementary
 - User as input, item as output: “for you”

Input



RecSys by use case

- RS can be understood easily by use case and input-output:
 - Item as input, item as output: similar vs complementary
 - User as input, item as output: “for you”
 - User as input, user as output: people you may know

People in the Technology industry you should follow

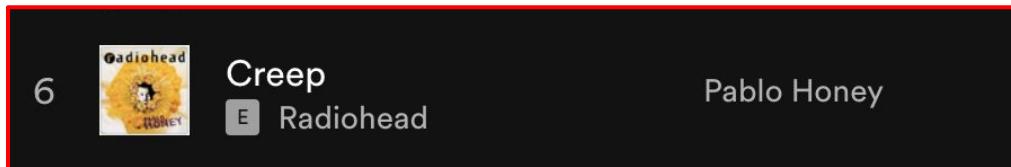
Profile Picture	Name	Title	Follower Count	Action
	Reid Hoffman	Entrepreneur. Product and Business Strategist. Investor....	2,605,741 followers	Follow
	Ryan Holmes	Founder and Board Member at Hootsuite	Talks about #startups, #marketing, and #technology	Follow
	Scott Galloway	Clinical Professor of Marketing, NYU Stern & Founder, Section4	403,092 followers	Follow

Input



RecSys by use case

- RS can be understood easily by use case and input-output:
 - Item as input, item as output: similar vs complementary
 - User as input, item as output: “for you”
 - User as input, user as output: people you may know
 - Session as input, item as output: what are you doing next?



Input

1	Girls & Boys Blur
2	Rock 'n' Roll Star - Remastered Oasis
3	Common People Pulp
4	Lucky Man The Verve
5	Linger The Cranberries

RecSys by use case (with refs!)

- RS can be understood easily by use case and input-output:
 - Item as input, item as output: similar vs complementary
 - User as input, item as output: “for you”
 - User as input, user as output: people you may know
 - Session as input, item as output: what are you doing next?

(There's more!)

- Product comparisons

This Item: Samsung UN55TU8000 55" 8 Series Ultra High Definition Smart 4K Crystal TV with a Samsung HW-T650 Bluetooth Soundbar with Dolby Audio Wireless Subwoofer (2020)	SAMSUNG HW-Q60T 5.1ch Soundbar with 3D Surround Sound and Acoustic Beam (2020)	SAMSUNG HW-T650 3.1ch Soundbar with 3D Surround Sound (2020)	Samsung UN55TU8000 55" 8 Series Ultra High Definition Smart 4K Crystal TV with a Samsung HW-Q60T Wireless 5.1 Channel Soundbar and Bluetooth Subwoofer (2020)
Add to Cart	Add to Cart	Add to Cart	Add to Cart
Customer Rating	★★★★★ (13)	★★★★★ (504)	★★★★★ (2314)
Price	\$995 ^{\$8}	\$497 ^{\$9}	\$397 ^{\$99}
Sold By	Watts TV	Amazon.com	Amazon.com
Connectivity Technology	Wi-Fi, Bluetooth, USB, HDMI, Ethernet	Bluetooth	Bluetooth, HDMI
Item Dimensions	9.90 x 48.40 x 30.80 inches	38.60 x 4.10 x 2.30 inches	38.60 x 3.50 x 2.30 inches
Item Weight	31.30 lbs	—	9.90 x 48.40 x 30.80 inches
Speaker Type	Subwoofer, In-Wall, Center Channel, 3D, Soundbar, Multi-Room	Surround Sound	Surround Sound
			Built-In

[R] 8 Jul 2021

"Are you sure?": Preliminary Insights from Scaling Product Comparisons to Multiple Shops

Patrick John Chia^{*}
Coveo
Montreal, Canada
pchia@coveo.com

Bingqing Yu^{*}
Coveo
Montreal, Canada
cyu@coveo.com

Jacopo Tagliabue[†]
Coveo Labs
New York, NY
jtagliabue@coveo.com

ABSTRACT

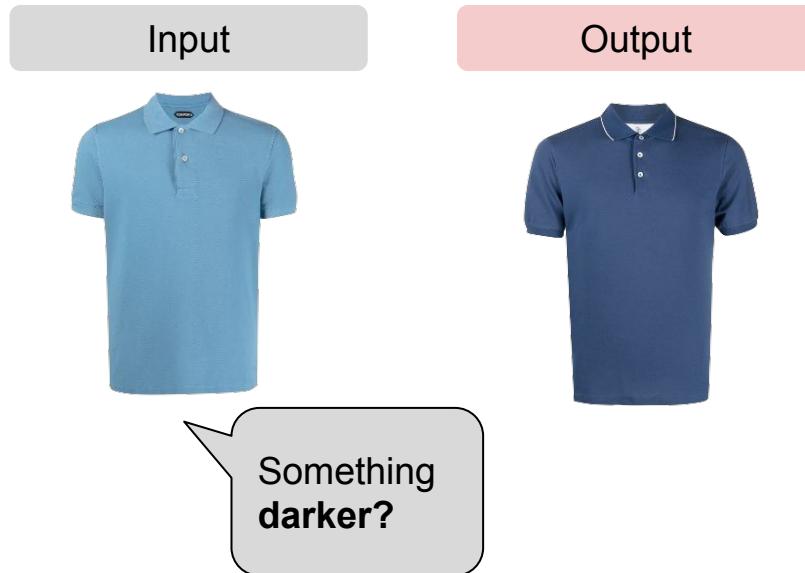
Large eCommerce players introduced comparison tables as a new type of recommendations. However, building comparisons at scale without pre-existing taxonomy data remains an open challenge, especially within the operational constraints of shops in the long tail. We present preliminary results from building a comparison pipeline designed to scale in a multi-shop scenario: we describe our design choices and run extensive benchmarks on multiple shops to stress-test it. Finally, we run a small user study on property selection and conclude by discussing potential improvements and highlighting the questions that remain to be addressed.

CCS CONCEPTS

tables when well-designed not only promote products that are relevant, but also intentionally select products which help customers better understand the range of available features. However – as demonstrated by the sub-optimal alternatives in Fig. 1 – building a CE is far from trivial even for players with full ownership of the data again. In this paper, we share preliminary lessons learned when building CEs in a B2B scenario: that is, designing a scalable pipeline that is deployed across multiple shops. As convincingly argued in [4, 27, 29], multi-tenant deployments require models to generalize to dozens of different retailers: a successful CE is therefore not only hard to build, but valuable to a wide range of practitioners – on one side, practitioners outside of humongous websites, who want to enhance their shop in the face of rising pres-

(There's more!)

- Gradient Recommendations



“Does it come in black?”
CLIP-like models are zero-shot recommenders

Patrick John Chia* Coveo, Montreal pchia@coveo.com	Jacopo Tagliabue Coveo Labs, New York jtagliabue@coveo.com	Federico Bianchi Bocconi University, Milan	Ciro Greco Coveo Labs, New York	Diogo Goncalves Farfetch, Porto
---	---	--	---	---

Abstract

Product discovery is a crucial component for online shopping. However, item-to-item recommendations today do not allow users to explore changes along selected dimensions: given a query item, can a model suggest something similar *but* in a different color? We consider item recommendations of the *comparative* nature (e.g. “something darker”) and show how CLIP-based models can support this use case in recommendation that introduces explicit directionality into the mix, by allowing exploration in selected directions *through natural language*: “something darker” will move the user from the *white dress* to the *grey dress*. In particular, we summarize our contributions as follows. First, we introduce GradREC as a new type of recommendation experience *and* a technical contribution – to the best of our knowledge, GradREC is the first *zero-shot* approach for language-based comparative recom-

Testing RecSys (the old way)

Recommendations in IR

- RS evaluation is traditionally performed using standard IR metrics over the test set (i.e. held-out data points):
 - recommendations are a *ranking* problem!

Test Input



Test Truth



Model Output



Recommendations in IR

- RS evaluation is traditionally performed using standard IR metrics over the test set (i.e. held-out data points):
 - recommendations are a *ranking* problem!

Test Input



Test Truth



correct!

Model Output



Standard metrics

- Since RS are evaluated as rankers, it is not surprising that a quick scan through recent editions of RecSys and SIGIR highlights as main metrics:
 - MRR
 - HIT RATE
 - NDCG

Standard metrics

- Hit Rate@k
 - Ask the model to predict k movies
 - If the target is inside the k predictions, increase hit count
 - Divide hit count by total predictions

Standard metrics

- Hit Rate@k
 - Ask the model to predict k movies
 - If the target is inside the k predictions, increase hit count
 - Divide hit count by total predictions

Test Truth



Model Output



$$\text{HR}@3 = 1 / 2 = 0.5$$

Standard metrics

- MRR
 - Calculate reciprocal rank for all predictions
 - Average the RR
 - Intuition: position matters!

$$\text{MRR} = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{\text{rank}_i}.$$

Test Truth



Model Output



Standard metrics

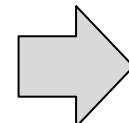
- MRR
 - Calculate reciprocal rank for all predictions
 - Average the RR

$$\text{MRR} = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{\text{rank}_i}.$$

Test Truth



Model Output



$$\text{MRR} = ((1 / 2) + 0) / 2 = 0.25$$

... and their flaws

- IR metrics won't generally consider anything above the fact that the target item is in the top-k recommendations.
 - *Sometimes recommendations are not just bad, but deeply illogical*

New jobs for CTO
Santa Monica, CA

HIGH PAYING

Executive Assistant
Recharge Payments - Los Angeles, CA
\$54K - \$112K (Glassdoor est.)
4.7 ★ Easy Apply
Highly Rated: Career Opportunity, Compensation & Benefits, Work/Life Balance, Culture & Values, Senior Leadership

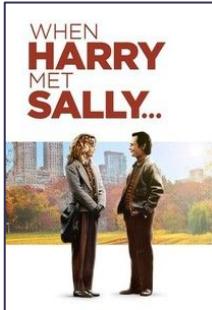
See All Jobs

A screenshot of a job search interface showing a specific listing for an Executive Assistant position at Recharge Payments in Santa Monica, CA. The listing is highlighted as 'HIGH PAYING'. It includes the company logo (a blue square with a white 'R'), the job title, location, salary range, rating (4.7 stars), and an 'Easy Apply' button. Below the listing, it says the job is highly rated for various factors like career opportunity and compensation. At the bottom of the card is a green 'See All Jobs' button.

... and their flaws

- IR metrics won't generally consider anything above the fact that the target item is in the top-k recommendations.
 - *Sometimes pseudo-feedback is not the whole truth: both models below have HR@3=1!*

Test Truth



Model #1



Model #2



Bonus topic: what is a proper split?

- At test time, we ask our model to rank the target items for the query item / user / session etc.
- The literature reports different ways of building that target set, so comparisons between studies / models need to be careful when merging results.

Precision-Oriented Evaluation of Recommender Systems: An Algorithmic Comparison

Alejandro Bellogín, Pablo Castells, Iván Cantador
Universidad Autónoma de Madrid, Escuela Politécnica Superior
Ciudad Universitaria de Cantoblanco, 28049 Madrid, Spain
{alejandro.bellogin, pablo.castells, ivan.cantador}@uam.es

ABSTRACT

There is considerable methodological divergence in the way precision-oriented metrics are being applied in the Recommender Systems field, and as a consequence, the results reported in different

the Mean Average Error (MAE), and the Root Mean Squared Error (RMSE). Although dominant in the literature, some authors have argued this evaluation methodology is detrimental to the field since the recommendations obtained in this way are not the

Behavioral Testing

What is behavioral testing?

1. Define expected output for a given (set of) input(s), *regardless of the composition of the test set.*
2. Test model compliance with the expected output, *regardless of the model inner workings.*

Beyond Accuracy: Behavioral Testing of NLP Models with CHECKLIST

Marco Túlio Ribeiro
Microsoft Research
marcotcr@microsoft.com

Tongshuang Wu
Univ. of Washington
wtshuang@cs.uw.edu

Carlos Guestrin
Univ. of Washington
guestrin@cs.uw.edu

Sameer Singh
Univ. of California, Irvine
sameer@uci.edu

Abstract

Although measuring held-out accuracy has

A number of additional evaluation approaches have been proposed, such as evaluating robustness (Ribeiro et al., 2018; Ribeiro et al., 2018).

Examples from NLP

Sentence

I am a [PROTECTED]
[NOUN]

Template filling

I am a **black women**

???

I am an **asian man**

???

Sentiment

Sentence

The pilot wasn't
fantastic.

Negative

Parse in (question, "no") form

Was the pilot
fantastic? No.

???

Examples from NLP

Sentence

I am a [PROTECTED]

BERT-base fails **100%** of
the time

I am a black women

!!!

I am an asian man

???

Sentiment

Sentence

The pilot wasn't

Microsoft fails **96.8%** of the
time, Amazon **81.6%**,
BERT-base **55.4%**

fantastic? No.

!!!

Lessons from CheckList

1. Behavioral testing improve reliability by testing *generalization* regardless of training distribution: experts and human knowledge is valued *even if* (or, *because*) not necessarily reflected in the test set / data collection etc.
2. Behavioral testing *is hard and painful to do by hand*: to make it feasible, we need good software and “tricks” to scale tests up!

Beyond Accuracy: Behavioral Testing of NLP Models with CHECKLIST

Marco Tulio Ribeiro
Microsoft Research
marcotcr@microsoft.com

Tongshuang Wu
Univ. of Washington
wtshuang@cs.uw.edu

Carlos Guestrin
Univ. of Washington
guestrin@cs.uw.edu

Sameer Singh
Univ. of California, Irvine
sameer@uci.edu

Abstract

Although measuring held-out accuracy has

A number of additional evaluation approaches have been proposed, such as evaluating robustness (Ribeiro et al., 2018; Ribeiro et al., 2018).

RecList

RecList at a Glance

RecList is ~~CheekList~~ for RecSys brings together best practices for reproducibility in research and robustness in deployments.



Open source



Peer reviewed



Community

RecList

RecList is an open source library providing behavioral, "black-box" testing for recommender systems. Inspired by the pioneering work of Ribeiro et al. 2020 in NLP, we introduce a general plug-and-play procedure to scale up behavioral testing, with an easy-to-extend interface for custom use cases.

While quantitative metrics over held-out data points are important, a lot more tests are needed for recommenders to properly function in the wild and not erode our confidence in them: for example, a model may boast an accuracy improvement over the entire dataset, but actually be significantly worse than another on rare items or new users; or again, a model that correctly recommends HDMI cables as

```
from reclist.datasets import CoveoDataset
from reclist.recommenders.prod2vec import CoveoP2VRecModel
from reclist.reclist import CoveoCartReclist

coveo_dataset = CoveoDataset()

model = CoveoP2VRecModel()
model.train(coveo_dataset.x_train)

# Instantiate rec_list object
rec_list = CoveoCartReclist(
    model=model,
    dataset=coveo_dataset
)
```

ar 2022

Beyond NDCG: behavioral testing of recommender systems with RecList

Patrick John Chia*
Coveo
Canada
pchia@coveo.com

Jacopo Tagliabue
Coveo Labs
United States
jtagliabue@coveo.com

Federico Bianchi
Bocconi University
Italy
f.bianchi@unibocconi.it

Chloe He
Stanford University
United States
chloeh@stanford.edu

Brian Ko
KOSA AI
United States
sangwoo@kosa.ai

ABSTRACT

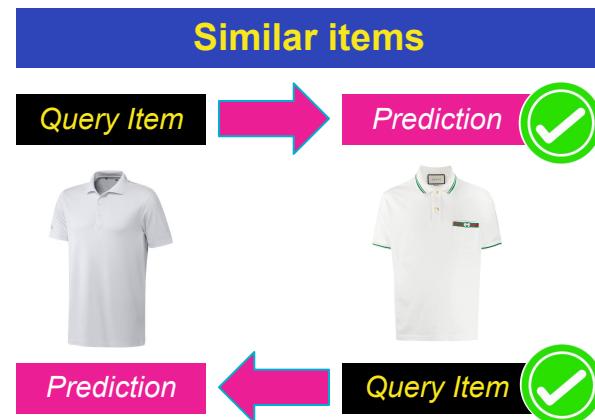
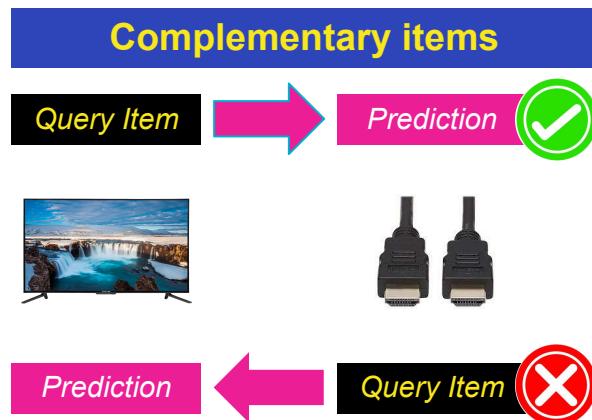
As with most Machine Learning systems, recommender systems are typically evaluated through performance metrics computed over

"bar bursts into flames, killing everyone" – B. Keller
(random tweet).

In recent years, recommender systems (henceforth RSe) have played

Some RecSys behavioral principles

Principle #1: in complementary vs similar recSys, query and target items have different formal relationships.



Some RecSys behavioral principles

Principle #2: some mistakes are *worse than others*.

Test Truth



Model #1



Model #2



This is a reasonable mistake!

Some RecSys behavioral principles

Principle #3: the ~~heart~~ power-law is deceitful above all things (mind the niches!)

HR: $57 / 110 = 0.52$



#: 100, H: 50, HR: 0.5

#: 10, H: 7, HR: 0.7

HR: $57 / 110 = 0.52$



#: 100, H: 56, HR: 0.56

#: 10, H: 1, HR: 0.1

Main abstractions

- A **RecList** is a collection of `recTests` to be run on a `recDataset` for a given `recModel`
 - Running a `RecList` means answering the following question: “how does this model perform on this dataset, according to these metrics?
- `RecList` ships with popular datasets and ready-made tests, but (see below) you can swap everything out with custom stuff
 - Bring your dataset, your model, your tests!

```
1  class MyRecList(RecList):  
2  
3      @rec_test(test_type='stats')  
4      def basic_stats(self):  
5          """  
6              Basic statistics on training, test and prediction data  
7          """  
8          from reclist.metrics.standard_metrics import statistics  
9          return statistics(self._x_train,  
10                          self._y_train,  
11                          self._x_test,  
12                          self._y_test,  
13                          self._y_preds)
```

Quick start

```
from reclist.datasets import CoveoDataset
from reclist.recommenders.prod2vec import CoveoP2VRecModel
from reclist.reclist import CoveoCartRecList

if __name__ == "__main__":

    # get the coveo data challenge dataset as a RecDataset object
    coveo_dataset = CoveoDataset()

    # re-use a skip-gram model from reclist to train a latent product space, to be used
    # (through knn) to build a recommender
    model = CoveoP2VRecModel()
    model.train(coveo_dataset.x_train)

    # instantiate rec_list object, prepared with standard quantitative tests
    # and sensible behavioral tests (check the paper for details!)
    rec_list = CoveoCartRecList(
        model=model,
        dataset=coveo_dataset
    )
    # invoke rec_list to run tests
    rec_list(verbose=True)
```

Pick a Dataset

Pick a Model

Train a model

Run your RecList

Explore with our [Colab tutorial](#)

A ML tool in a MLOps world ([GitHub](#))

1

```
# imports
from coveo import CoveoDataset
from coveo.recommenders.prodvec import CoveoP2VRecModel
from coveo import CoveoCartRecList

if __name__ == "__main__":
    # get the coveo data challenge dataset as a RecDataset object
    coveo_dataset = CoveoDataset()

    # re-use a skip-gram model from reclist to train a latent product space, to be used
    # (through knn) to build a recommender
    model = CoveoP2VRecModel()
    model.train(coveo_dataset.x_train)

    # instantiate rec list object, prepared with standard quantitative tests
    # and sensible behavioral tests (check the paper for details!)
    rec_list = CoveoCartRecList(
        model=model,
        dataset=coveo_dataset
    )
    # invoke rec_list to run tests
    rec_list(verbose=True)
```



METAFLOW



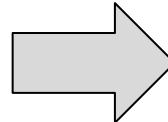
PREFECT



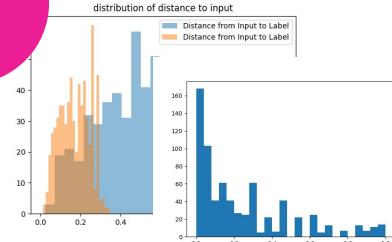
Apache
Airflow

2a

```
"_metadata": {"run_time": 16141478181413, "model_name": "CoveoP2VRecModel", "reclist": "CoveoCartRecList", "tests": ["basic_stats", "price_test", "coverage_at_k", "hit_rate_at_k", "hits_distribution", "dist_to_query"]} ...
```



2b



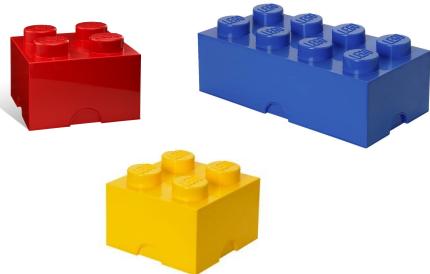
3

OS Connectors



“Talk is cheap,
show me the
code”

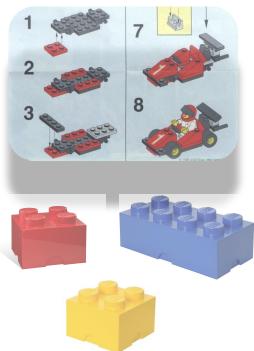
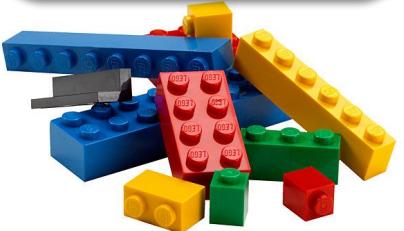
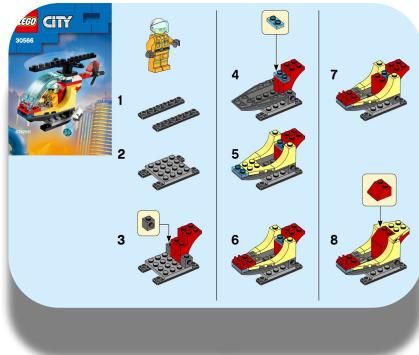
Ready-to-go, with endless possibilities



LEVEL #1

- You bring your model, we provide RecTests and a tried-and-tested way to combine them!

Ready-to-go, with endless possibilities



LEVEL #1

- You bring your model, we provide RecTests and a tried-and-tested way to combine them!

LEVEL #2

- You bring your model and your RecTests, and add them to ours!

Ready-to-go, with endless possibilities



LEVEL #1

- You bring your model, we provide `RecTests` and a tried-and-tested way to combine them!

LEVEL #2

- You bring your model and your `RecTests`, and add them to ours!

LEVEL #3

- Build *anything* you want, and use `RecList` abstractions as the “interlocking system” gluing it all together!

Research and production

Research

Experimentation

Reporting

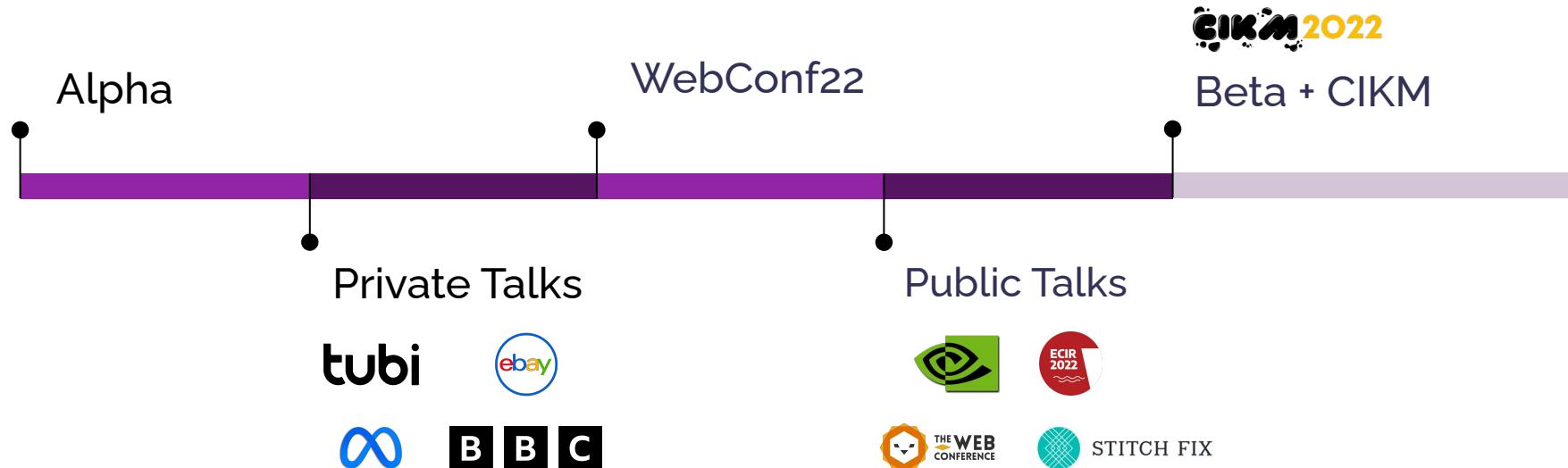
Production Systems

CI/CD

Monitoring/Debugging

What now?

RecList Beta is in the making!



CIKM Data Challenge

RecList will be at CIKM 2022 for a data challenge on rounded evaluation of Recommender Systems! Stay tuned for more information!



CIKM2022

**31st ACM International
Conference on Information and
Knowledge Management**

Check out [RecList](#) on
[Github](#) and give us a star!

Wanna contribute to the
project? Get in touch!

