Rubber Ducky Labs

# Recommender Systems Operations

Alexandra Johnson, Co-Founder and CEO

# About Rubber Ducky Labs

- Co-founders met at MLtools startup SigOpt (acquired by Intel) in 2018
- CS / Software backgrounds
- Years of experience in ML and dev tooling
- Fell in love with recommender systems in fashion tech

## Why are recommender systems important?

A great recommender system is worth $$$$

Consumers expected hyperpersonalized content

Hyperpersonalized content = more interactions, more conversions

ML allows you to hyperpersonalize product / content recommendations at scale

Recommender systems are the systems (ML + data + data pipelines) that deliver hyperpersonalized content to users

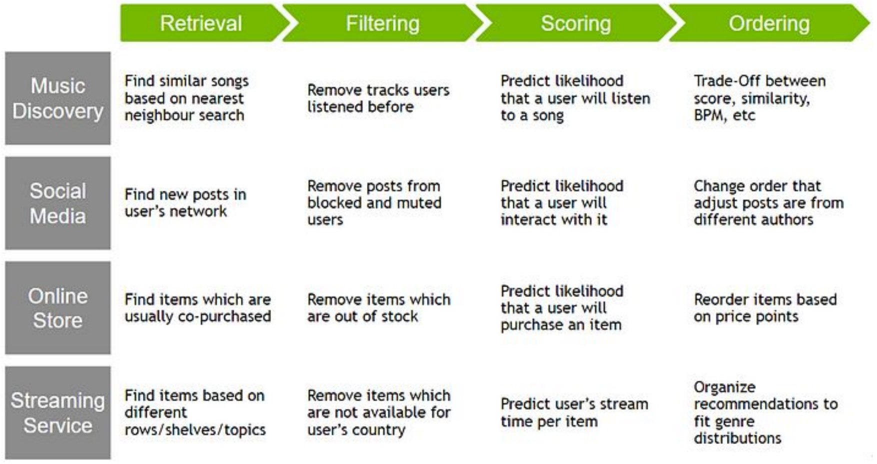
# Why RecSys is more than ML models

The dream: one ML model to rule them all

The reality: many different ML models to account for different objectives and latency concerns

Four key steps: retrieval -> filtering -> scoring -> ordering

Optional: Additional candidate generators + merging, re-rank, and cutoff steps, business logic



| | Retrieval | Filtering | Scoring | Ordering |
|---|---|---|---|---|
| Music Discovery | Find similar songs based on nearest neighbour search | Remove tracks users listened before | Predict likelihood that a user will listen to a song | Trade-Off between score, similarity, BPM, etc |
| Social Media | Find new posts in user's network | Remove posts from blocked and muted users | Predict likelihood that a user will interact with it | Change order that adjust posts are from different authors |
| Online Store | Find items which are usually co-purchased | Remove items which are out of stock | Predict likelihood that a user will purchase an item | Reorder items based on price points |
| Streaming Service | Find items based on different rows/shelves/topics | Remove items which are not available for user's country | Predict user's stream time per item | Organize recommendations to fit genre distributions |

Recommender Systems, Not Just Recommender Models, Oldridge & Higley, NVIDIA

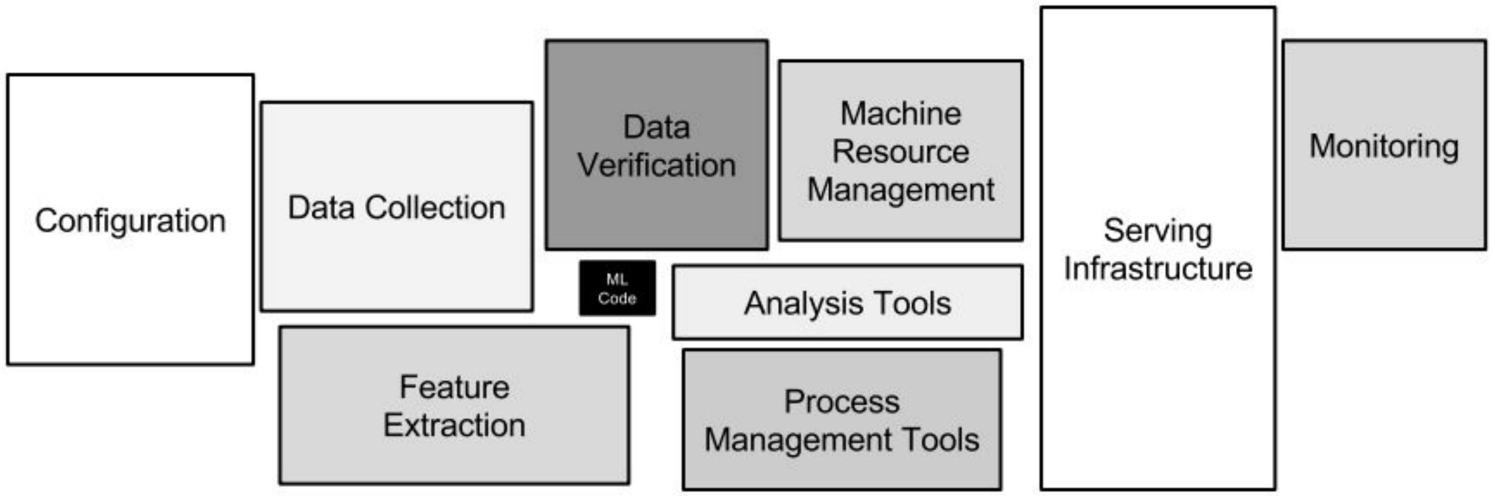


Figure 1: Only a small fraction of real-world ML systems is composed of the ML code, as shown by the small black box in the middle. The required surrounding infrastructure is vast and complex.

Hidden technical debt in Machine learning systems, Sculley et al

## We’ve learned a lot about recommender systems tooling

Product discovery conversations concentrated in 2022

50+ recommender systems teams

E-commerce (biggest category), streaming media, consumer marketplaces, fintech, and video games

Everyone built internal tooling above and beyond standard ML model training + serving

## Problem: Long tail of bugs

ML-systems can surface offensive or off-brand content

Hard to incorporate domain knowledge into ML models

Executive yells at ML engineer: Why is our app recommending ski jackets in June?

## Solution: Non-ML business logic

*“Managing operations for recommender systems cost us one ML engineer’s worth of engineering time, but two ML engineer’s worth of morale”*

— John McDonnell, former Director of ML (Recommender Systems) at Stitch Fix

*“I do this stuff as I have to do it, not because I’m really passionate about doing it.”*

– Operationalizing Machine Learning: An Interview Study, Shankar et al

## Problem: ML team overwhelmed by requests to add rules

Marketers, merchandisers, and salespeople incentivized by specific business outcomes

Constant stream of requests for tweaks and boosts to the ML team

## Solution: Self-serve marketing boost system

Consolidated business logic layer within ML system

Frontend UI for stakeholders

Challenges: too many (1000s) of rules, ML model is too constrained

## Problem: Triaging bugs is slow

Unlike traditional software, you can’t write out all of the test cases

“This item is good, why am I not showing it?”

“Why did this user get recommended that item?”

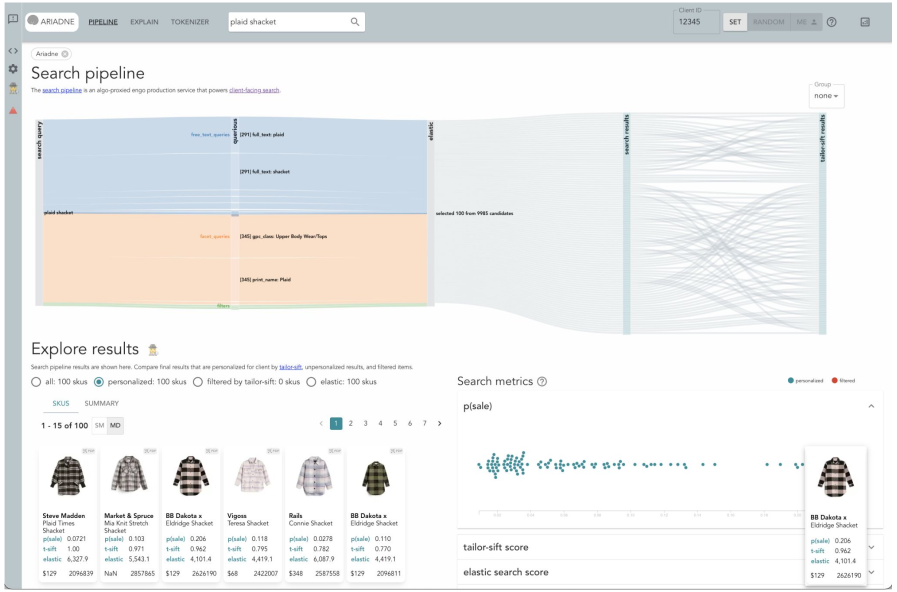
# Solution: Debug tooling for the ML system

One step past ML model observability to ML system observability

Brings together: input / output logging, model scores, product catalog data

Answers: Where in the system is the problem?

Great debugging tooling forces great system architecture: modularity, logging



Search pipeline

The search pipeline is an algo-proxied engo production service that powers client-facing search.

Explore results

Search pipeline results are shown here. Compare final results that are personalized for client by tailor-sift, unpersonalized results, and filtered items.

all: 100 skus [✓ personalized: 100 skus] filtered by tailor-sift: 0 skus elastic: 100 skus

1 - 15 of 100

| Steve Madden Plaid Times Shacket | Market & Spruce Mia Knit Stretch Shacket | BB Dakota x Eldridge Shacket | Vigoss Teresa Shacket | Rails Connie Shacket | BB Dakota x Eldridge Shacket |
|---|---|---|---|---|---|
| p(sale) 0.0721 | p(sale) 0.103 | p(sale) 0.206 | p(sale) 0.118 | p(sale) 0.0278 | p(sale) 0.110 |
| t-sift 1.00 | t-sift 0.971 | t-sift 0.962 | t-sift 0.795 | t-sift 0.782 | t-sift 0.770 |
| elastic 6,327.9 | elastic 5,543.1 | elastic 4,101.4 | elastic 4,419.1 | elastic 6,087.9 | elastic 4,419.1 |
| $129 2096839 | NaN 2857865 | $129 2626190 | $68 2422007 | $348 2587558 | $129 2096811 |

BB Dakota x Eldridge Shacket
p(sale) 0.206
t-sift 0.962
elastic 4,101.4
$129 2626190

Ariadne: building a custom observability UI for personalized search, Bilenko and Hashemi

**Rubber Ducky Labs**

## Problem: Difficult to build intuition

Data scientists, MLEs, analysts, PMs:

Want to understand “what’s going on in my recommender system”

Desire to “move beyond the averages”

Need to build arguments for which areas to invest in next

Make decisions about whether or not to launch experiments or features

## Solution: Slice and dice at scale

Custom-built tools allow slicing and dicing by user or product attributes

Pattern #1: Identify underperforming segment, build new feature

Pattern #2: Slice and dice experiment results on traditionally underperforming segments: new users, power users, geos, other sensitive facets

Huge infrastructure challenges when building at scale

## Problem: Slow iteration cycles

One ML team, many placements, many carousels

Building new models / pipelines is time consuming and expensive

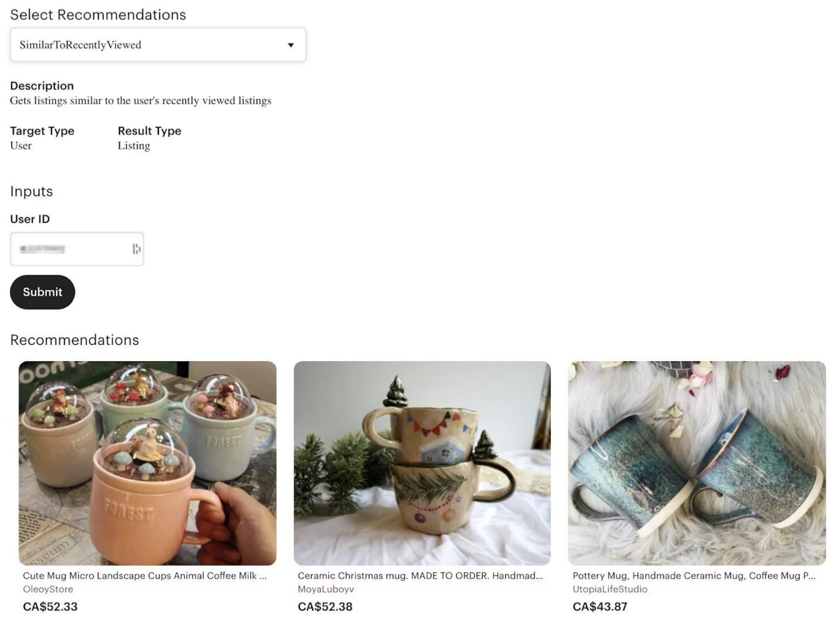
# Solution: Recommender systems playground

Available to engineering / product stakeholders

See available models

Simulate recommendations

Build into placements / carousels



Select Recommendations

SimilarToRecentlyViewed

**Description**
Gets listings similar to the user's recently viewed listings

| **Target Type** | **Result Type** |
|---|---|
| User | Listing |

Inputs

**User ID**

[redacted]

Recommendations

Cute Mug Micro Landscape Cups Animal Coffee Milk ...
OleoyStore
**CA$52.33**

Ceramic Christmas mug. MADE TO ORDER. Handmad...
MoyaLuboyv
**CA$52.38**

Pottery Mug, Handmade Ceramic Mug, Coffee Mug P...
UtopiaLifeStudio
**CA$43.87**

Building a Platform for Serving Recommendations at Etsy, Blondeau

## Rubber Ducky Labs builds these tools

We build tools just like the ones described here

Our goal is to help you shorten recsys debug and analysis cycles

We want you to improve your recommender system, faster

We are currently taking on a limited number of design partners

Questions? Comments? Collaborations?
Alexandra@rubberduckylabs.io