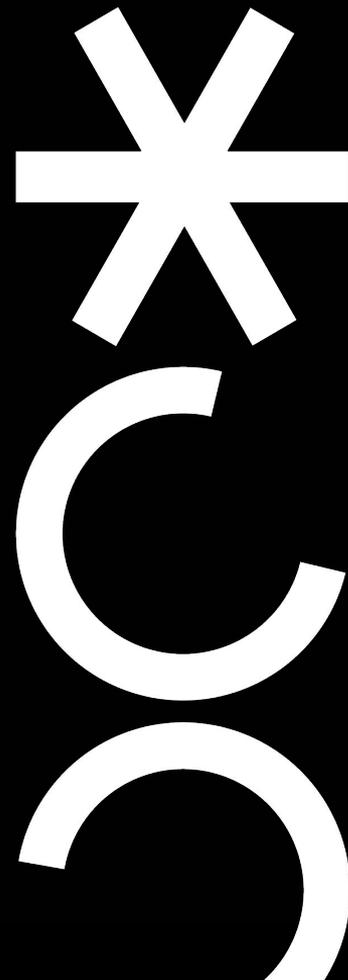




# Пишем рекомендательную систему музыки на Java

Андрей Кузнецов, ML Team Lead, Одноклассники



# Познакомимся



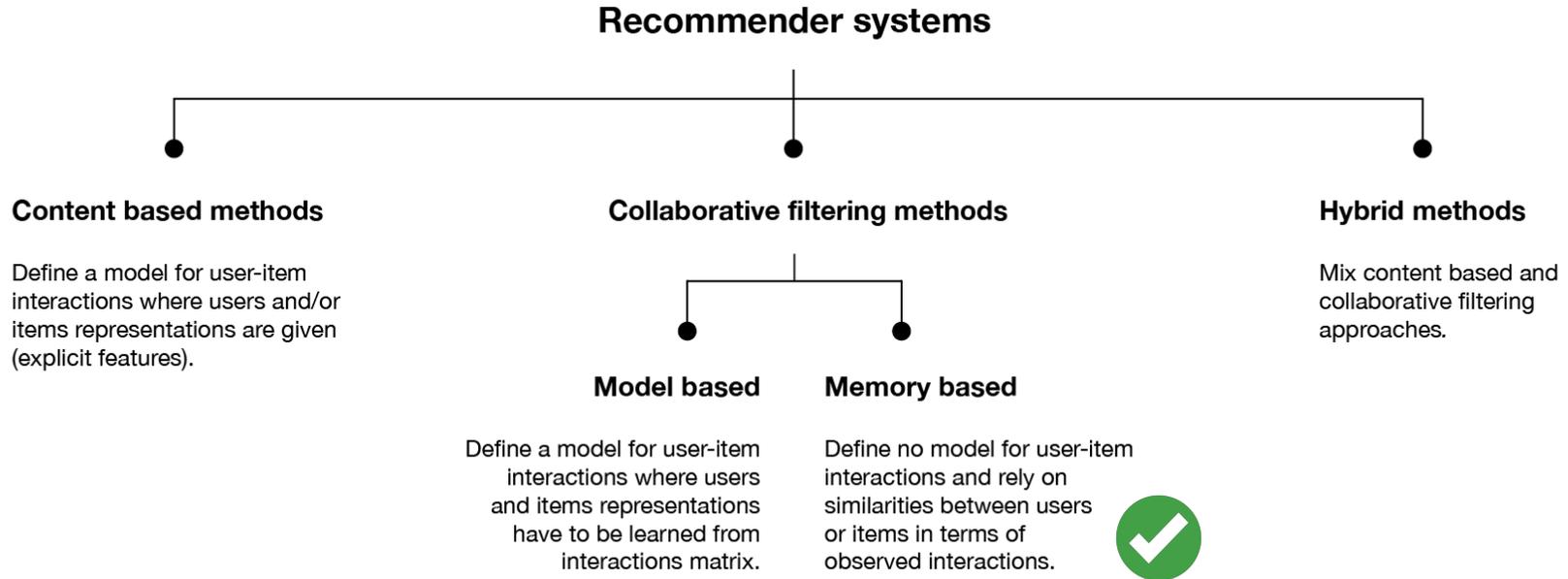
- ML Team Lead в Одноклассниках, VK. Занимаюсь разработкой рекомендательных систем
- Преподаю курс по Большим Данным в ИТМО
- кандидат технических наук





# Рекомендательные системы

# Виды рекомендеров. Принцип построения



# Виды рекомендеров. Назначение

1. user-item
2. item-item
3. item-user
4. user-user (?)



# Музыкальный item2item рекомендер в ОК

The screenshot displays a music player interface for the artist Nightwish. At the top left is the artist's profile picture. To its right, the text reads "Исполнитель Nightwish" and "32 альбома". Below this are four buttons: "▶ Слушать", "+ Добавить", "↻ Похожие песни", and "⋮ Ещё".

Below the buttons is a section titled "Похожие исполнители" (Similar Artists), which contains a grid of ten circular album covers, each with the artist's name underneath:

- Xandria
- Within Temptation
- Walk In Darkness
- Tarja
- Exit Eden
- Apostolica
- Sirenia
- Amaranthe
- Mono Inc.
- RabieS

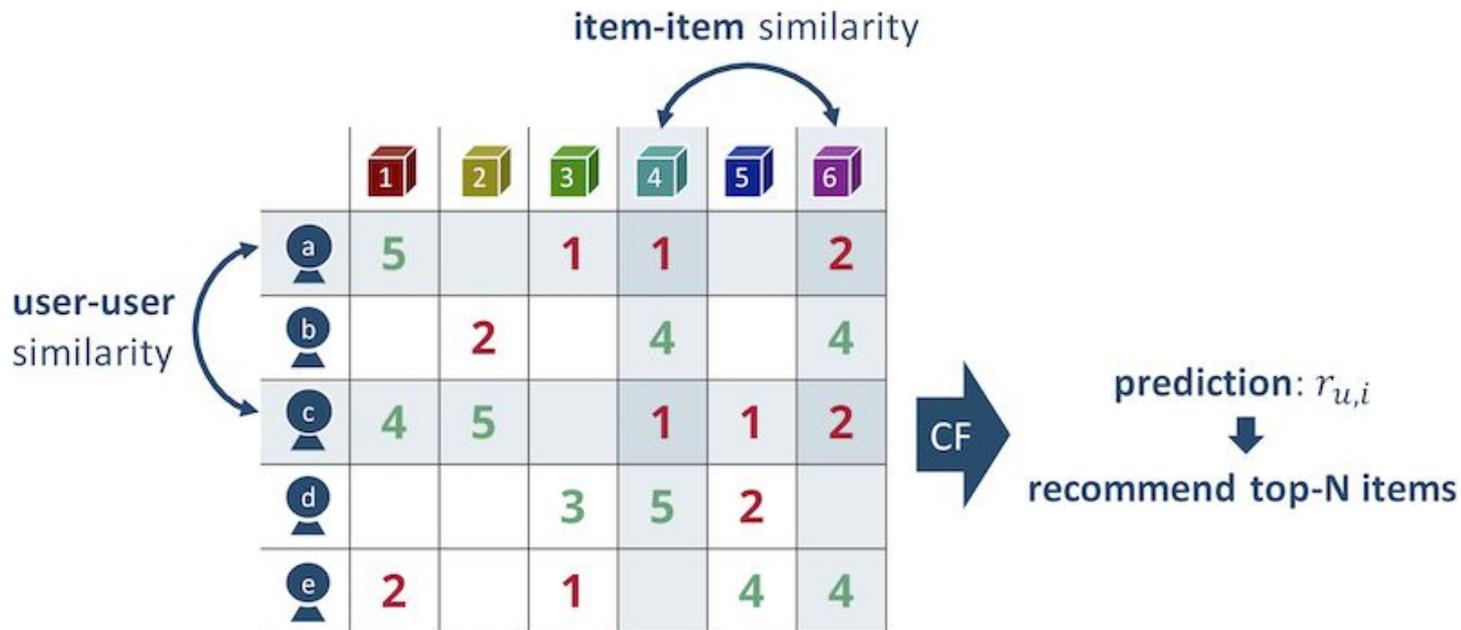


# Описание задачи

- item2item recommender
- Публичный датасет  
<https://www.kaggle.com/datasets/pcbreviglieri/lastfm-music-artist-scrobbles>
- Java
- kNN алгоритм



# Коллаборативная фильтрация



# Матрица похожести исполнителей

iU

	2	4	2		4
			1	1	2
	1	4	1	5	
	1			3	1
		2	5		
	5		4		2
					
					
					
					
					

x

Ui

						
	5		1	1		2
		2		4		4
	4	5		1	1	2
			3	5	2	
	2		1		4	4

=

ii

						
	45	20	7	9	12	26
	20	<b>29</b>	0	13	5	18
	7	0	<b>11</b>	16	10	6
	9	13	16	<b>43</b>	11	20
	12	5	10	11	<b>21</b>	18
	26	18	6	20	18	<b>40</b>





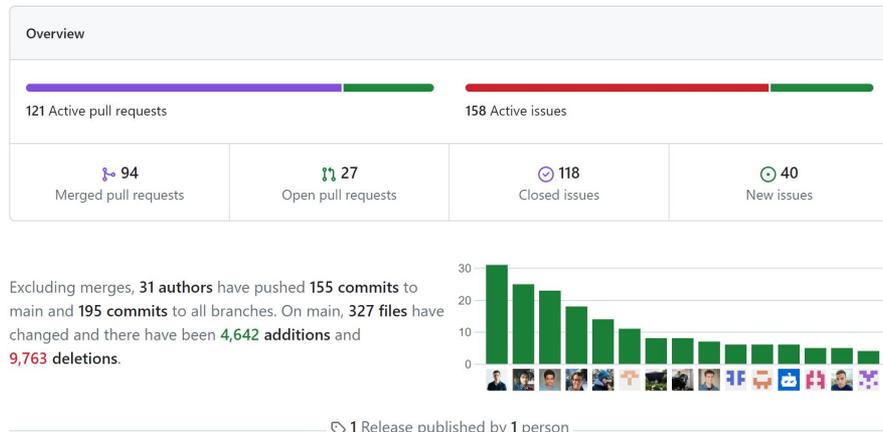
# Выбор библиотеки

# Библиотеки линейной алгебры на Java

## Python - Numpy (pandas)

September 8, 2023 – October 8, 2023

Period: 1 month



## Scala - Breeze (Spark internal)

dlwh 2.1.0 ✓ 28cfe3a on Aug 22, 2022 3,725 commits

github/workflows	Create scala.yml	2 years ago
benchmark	Bump dev.ludovic.netlib to 3.0.0	last year
macros	bump spire to 0.18.0	last year
math	bump spire to 0.18.0	last year
natives	clean up natives	2 years ago
project	bump scala 3.1 version	last year
viz	more making tests happy	2 years ago
.gitignore	Update git ignore for VScode / metals settings	2 years ago
.scalafmt.conf	Scalafmt	6 years ago
DEVELOP.md	notes the new codegen framework	2 years ago
LICENSE	Refactor ADMM based ProximalMinimizer to breeze.optimize.quadratic f...	9 years ago
NOTICE	infiniteIteration API in FirstOrderMinimizer takes initialState;PQN b...	9 years ago
README.md	2.1.0	last year



# Библиотеки линейной алгебры на Java

- Colt
- Commons Math
- ✓ • Efficient Java Matrix Library (EJML)
- Jama
- jblas
- JScience (Older benchmarks only)
- Matrix Toolkit Java (MTJ)
- OjAlgo
- Parallel Colt
- Universal Java Matrix Package (UJMP)
- Elegant Linear Algebra for Java (la4j)

## Works in Process

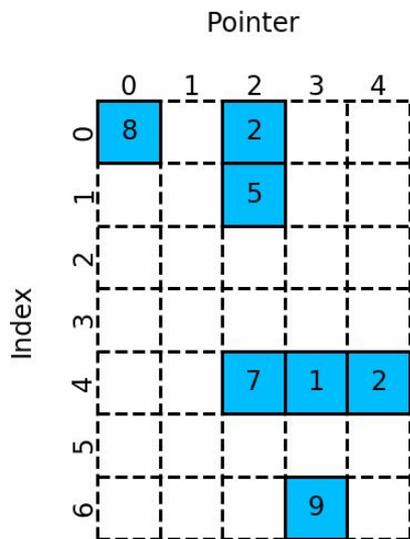
- ND4J working through documentation and missing features issues [TICKET](#)



<https://lessthanoptimal.github.io/Java-Matrix-Benchmark/>



# Эффективное хранение sparse матриц



© Matt Eding

## CSC

Index Pointers



Indices



Data

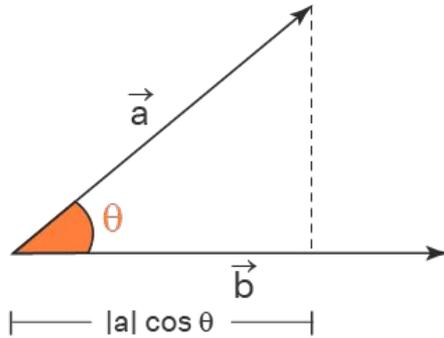


101 Ways to Store a Sparse Matrix

<https://medium.com/@jmaxg3/101-ways-to-store-a-sparse-matrix-c7f2bf15a229>

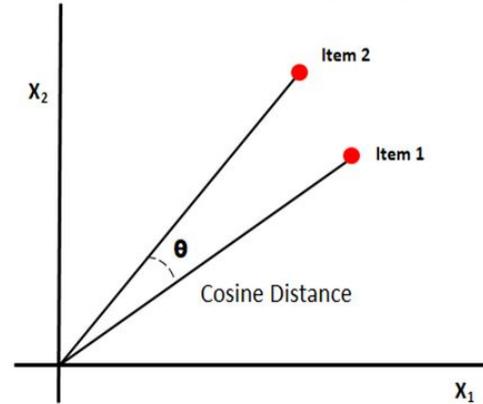


# Расстояния между векторами



$$\vec{a} \cdot \vec{b} = |a| |b| \cos \theta$$

## *Cosine Distance/Similarity*



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





# Рекомендательные системы на больших данных

# Инфраструктура данных



17 Kafka clusters

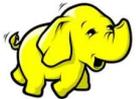
180 servers  
540 TB



Compute  
10k+batch jobs / day

160TB RAM, 60k cores  
1.5 PB NVME

+ **1 Эксабайт  
Медиа  
в объектном  
хранилище**



200 PB  
+ 35 PB / year  
+ 10 TB raw new / day



Inserting into 72 servers  
- 1,5 TB / day  
- **50,000,000,000** rows / day

HDD 150/600 TB  
NVME 60/210 TB  
RAM 5.5

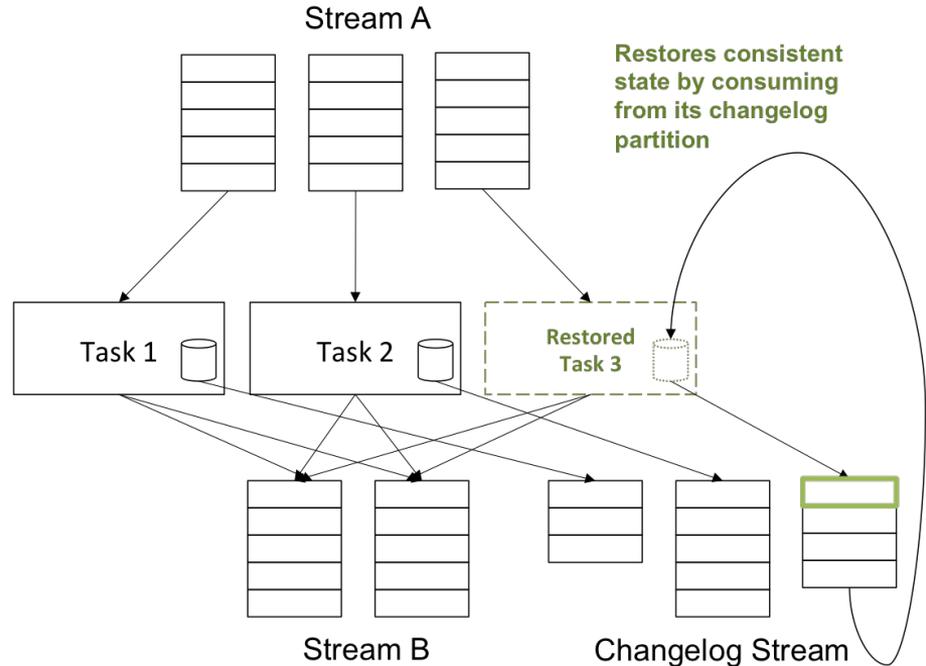


Дешевле, надёжнее, проще. Хранение петабайтов фото и видео в ОК  
[https://youtu.be/uuGbbJhS7o8?si=tG44sFr0KB\\_r6DxF](https://youtu.be/uuGbbJhS7o8?si=tG44sFr0KB_r6DxF)



# Потоковая обработка. Samza

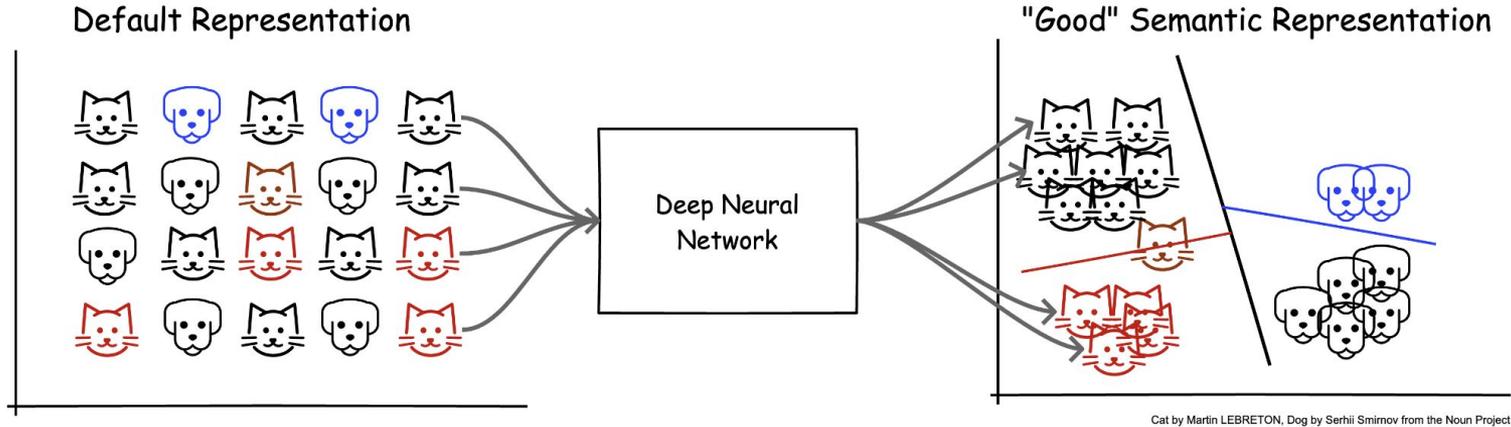
- Samza - стриминговый Java-фреймворк от LinkedIn
- Таски запускаются в контейнерах Yarn
- Состояние может храниться в KV-стораджках (in-memory, HDFS, Kafka)
- Чек-пойнты в отдельный топик Кафки





# Векторный поиск

# Задача векторного поиска



Ищем наиболее близкие объекты в многомерном векторном пространстве



# Векторный поиск. FAISS

В задаче векторного поиска есть два основных требования: **искать быстро** и **хранить эффективно**.

FAISS использует несколько подходов:

1. Asymmetric distance computation (ADC)
2. Inverted file (IVF)
3. Product quantization (PQ)

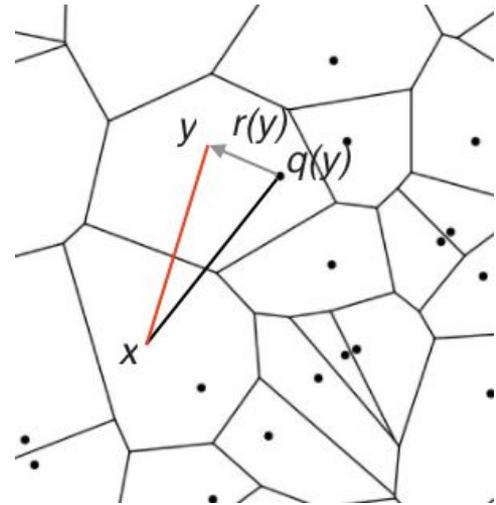


Faiss is a library for efficient similarity search  
<https://github.com/facebookresearch/faiss>



# Векторный поиск: FAISS. ADC

1. Воспользуемся идеей корзинок и разделим наше пространство на кластеры с помощью K-Means
2. Для репрезентации корзины возьмем вектор центрального элемента
3. При запросе находим ближайший центр
4. Достаем элементы кластера через IVF

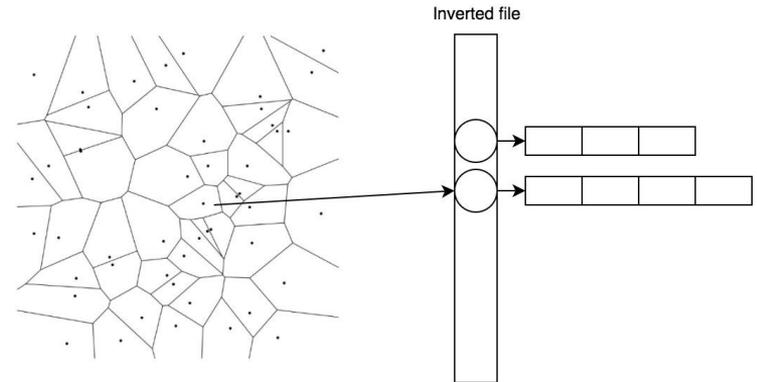


Методы приближенного поиска ближайших соседей  
<https://habr.com/ru/companies/vk/articles/338360/>



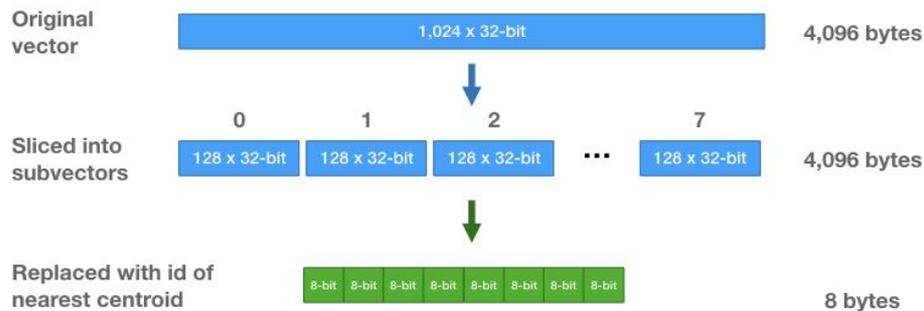
# Векторный поиск. FAISS. IVF

Для центров кластера просто храним список элементов в нем  
Элементы выдаем для честного расчета близости



# Векторный поиск: FAISS. PQ

1. Вычитаем элементы центроида из вектора
2. Делим полученный вектор на корзинки
3. Каждую из частей кластеризуем и заменяем индексом центра



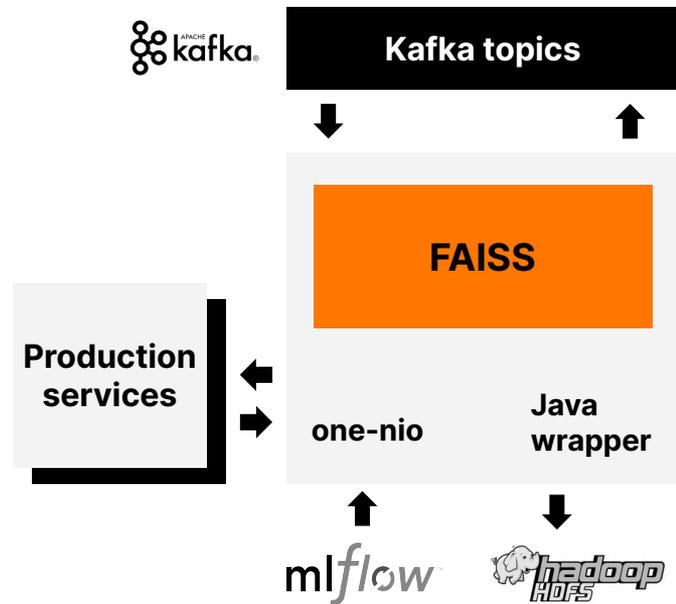
Product Quantizers for k-NN

<https://mccormickml.com/2017/10/13/product-quantizer-tutorial-part-1/>



# OK ANN

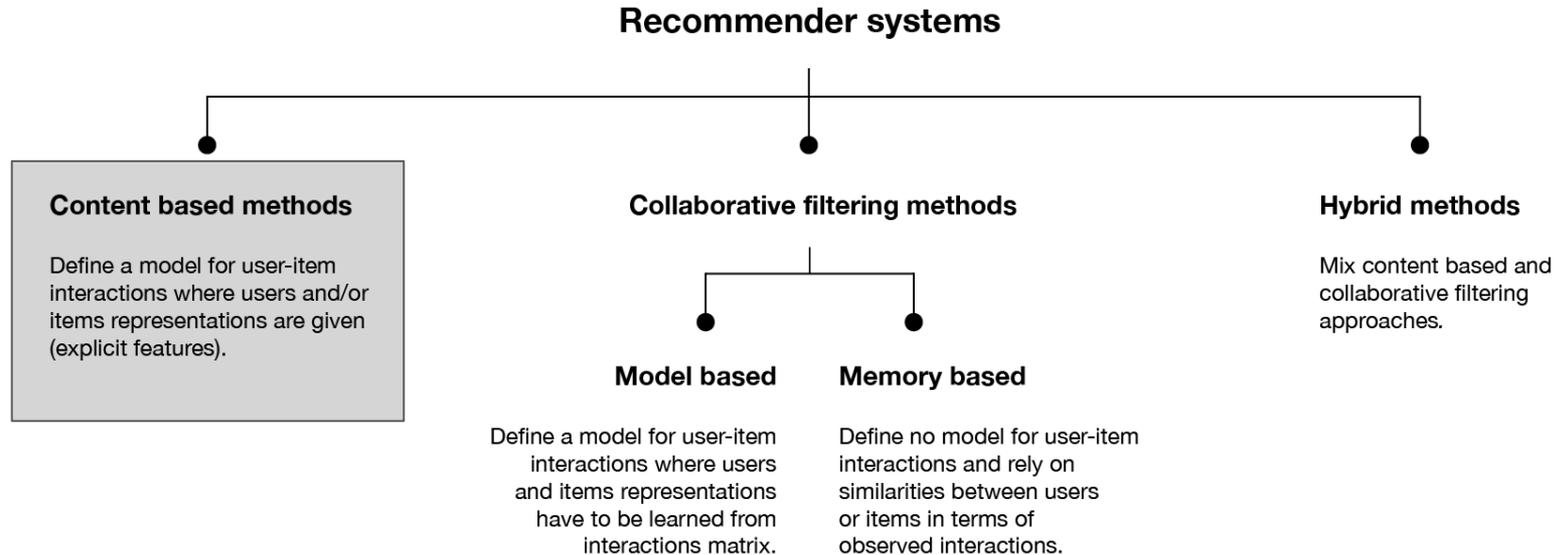
1. Сервис для быстрого поиска ближайших соседей в высокоразмерных векторных пространствах
2. Потокно читаем из Кафки объекты для хранения и пишем в FAISS индекс
3. Умеем в изменяемые индексы
4. Чек-пойнты сохраняем в HDFS
5. Обученные индексы сервим и версионруем с помощью MLflow



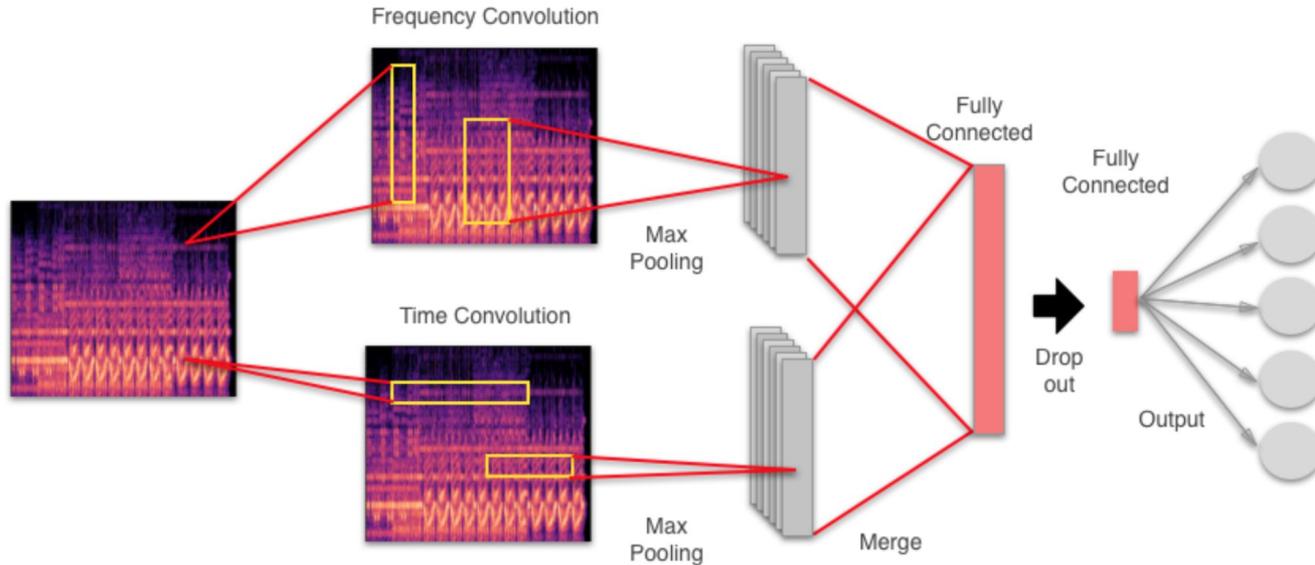
one-nio is a library for building high performance Java servers  
<https://github.com/odnoklassniki/one-nio>  
Vector database built for scalable similarity search  
<https://milvus.io/>



# Виды рекомендеров



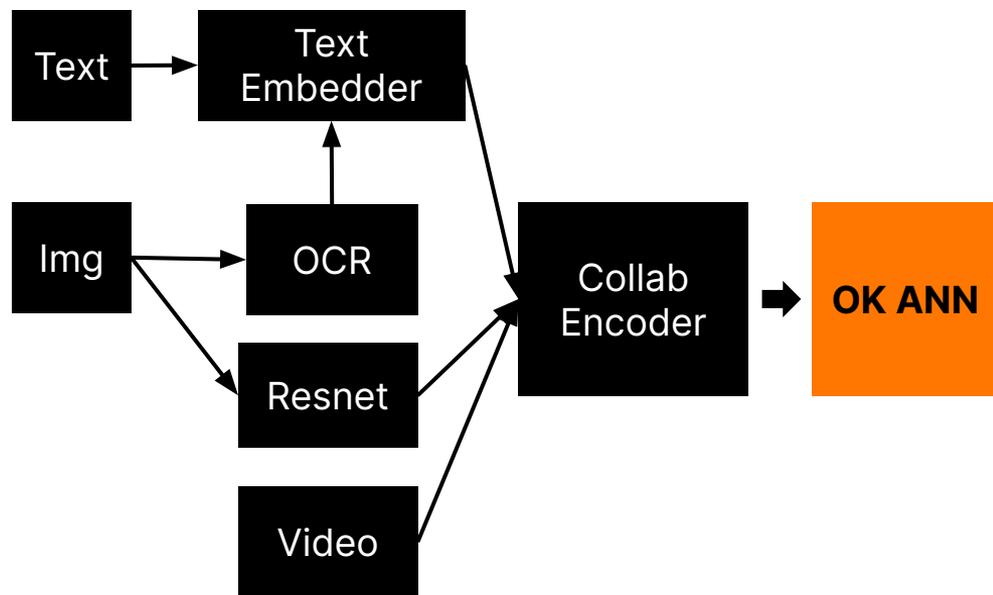
# Контентные нейросети



Neural content-aware collaborative filtering for cold-start music recommendation <https://arxiv.org/pdf/2102.12369.pdf>



# Контентные нейросети



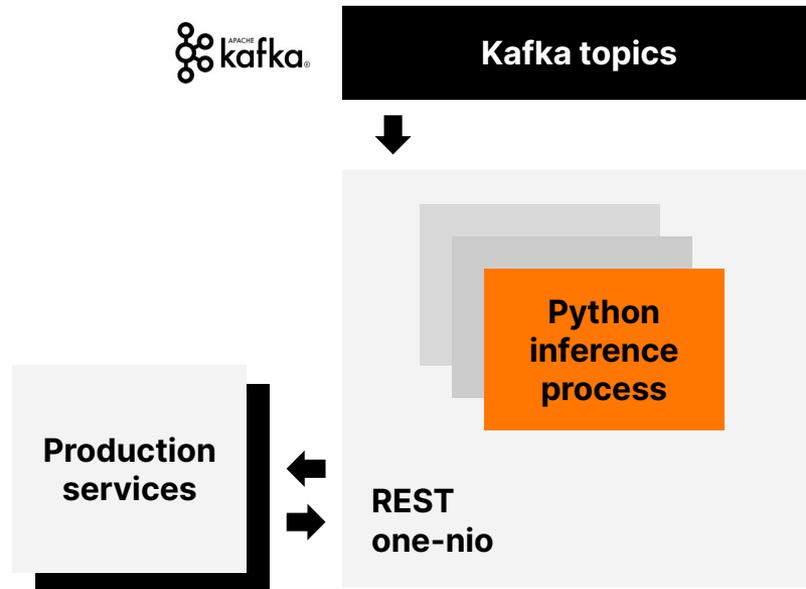
Multimodal embeddings at Google

<https://cloud.google.com/vertex-ai/docs/generative-ai/embeddings/get-multimodal-embeddings>



# Нейроплатформа ОК

1. Деплоим инстансы для инференса в нашем облаке
2. Применяем пайплайны моделей к объектам
3. Пишем результаты в Кафку асинхронно или отвечаем по ремоутингу или REST синхронно



The Triton Inference Server  
<https://github.com/triton-inference-server/server>





**Классические  
рекомендеры все еще  
рулят (или нет)?**

# ChatGPT recommender



Recommend me artist if I like Long Distance Calling, Pelican and Cpyб 22:45 ✓

- If you like Long Distance Calling, you might also enjoy albums by Toundra, Collapse Under the Empire, or God Is An Astronaut.
- If you enjoy Pelican, you might also like albums by Russian Circles, Red Sparowes, or Cult of Luna.
- If you're a fan of Cpyб, you might enjoy albums by Molchat Doma, Kino, or Motorama (all of whom are Russian bands with a similar sound).

22:45



A Survey on Large Language Models for Recommendation  
<https://arxiv.org/abs/2305.19860>

# ChatGPT recommender

Recommend me an similar artist to Sum 41 with explanation why 0:55 ✓

An artist similar to Sum 41 would be Blink-182. Both bands emerged in the late 1990s and became prominent in the early 2000s pop punk scene. Here's why Blink-182 is a recommended choice:

1. Catchy Pop Punk Sound: Both Blink-182 and Sum 41 are known for their infectious and energetic pop punk sound that combines catchy melodies, fast-paced guitar riffs, and relatable lyrics. This similarity in style ensures that fans of Sum 41 would find familiar elements in Blink-182's music.
2. Humor and Wit: Both bands incorporate humor and wit into their lyrics, often addressing teenage angst, relationships, and everyday struggles with a light-hearted and self-aware approach. This shared element adds an entertaining dimension to their music, making their songs relatable and enjoyable for fans.

3. Lyrical Themes: Both Blink-182 and Sum 41 explore themes of youth, rebellion, and self-identity, which resonate with listeners going through similar experiences. Their lyrics often touch upon personal struggles, growing up, and the challenges of navigating the complexities of teenage life.
4. Popularity and Influence: Both bands have had a significant impact on the pop punk genre and gained a substantial fan following over the years. Their music has resonated with a wide audience, making them influential figures in the punk rock scene.

Overall, if you enjoy the energetic pop punk sound, catchy melodies, humor-infused lyrics, and relatable themes of Sum 41, Blink-182 would be a great artist to explore as they share many of these characteristics. 0:55



A Survey on Large Language Models for Recommendation  
<https://arxiv.org/abs/2305.19860>

# Выводы

1. Написать примитивный рекомендер на Java просто
2. Сложные рексистемы требуют инфраструктуры, которая либо написана на Java, либо ее удобно обернуть в Java.
3. Большие языковые модели пока уступают классическим рексистемам.



Код примера

[https://github.com/netcitizenrus/Joker\\_2023\\_ex](https://github.com/netcitizenrus/Joker_2023_ex)



**Спасибо за  
внимание!**

Андрей Кузнецов, ML Team Lead, Одноклассники

