K-means

Groups similar profiles into clusters

→ requires training data
→ can adapt to user patterns
Pros: Adapts Cons: LOTS data needed

HYBRID Approach:

- SBERT initial embeddings
- K-means to cluster similar profiles
- cosine-similarity → final match (inside clusters)

$O(n^2) \xrightarrow[\text{K-means}]{} O(n \times k)$

→ efficient
→ SBert embeds allows good compai.

MongoDB

- user id
- name

everything

→ embed_bio

→ embed_pref

Pinecone

1 record

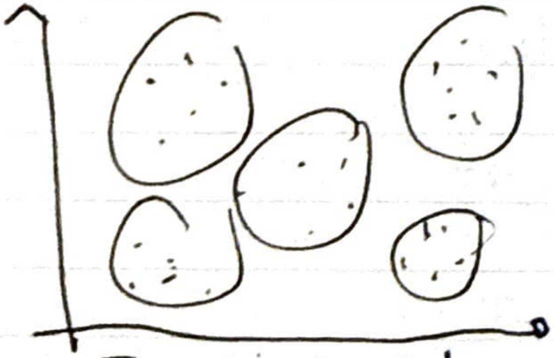
- id
- ~~vector~~ values = vector
- metadata = { vector type
  faculty
  edh
  ...
  .. }

*vector type changes
1 index
2 namespaces

① k-means : periodically

- pull embedding from All

- assign user bio → clusters id



BIO's vector namespace

- store clusters_id in Mongo

② filter clusters id only

find simil. score

Π_1 = user1.bio v.s user2.pref

Π_2 = user2.bio v.s user1.pref

$\Pi = (\Pi_1 \times \Pi_2)/2$

→ Meta_data filtering w/ weights

→ final score = $\alpha \cdot \Pi + \beta \cdot \text{Meta}$

→ Sort by final scores

→ Provide top $\underset{\substack{\uparrow \\ \text{or } N}}{3}$ best matches