Rakuten – Viki Challenge

Le Nguyen The Dat

About me

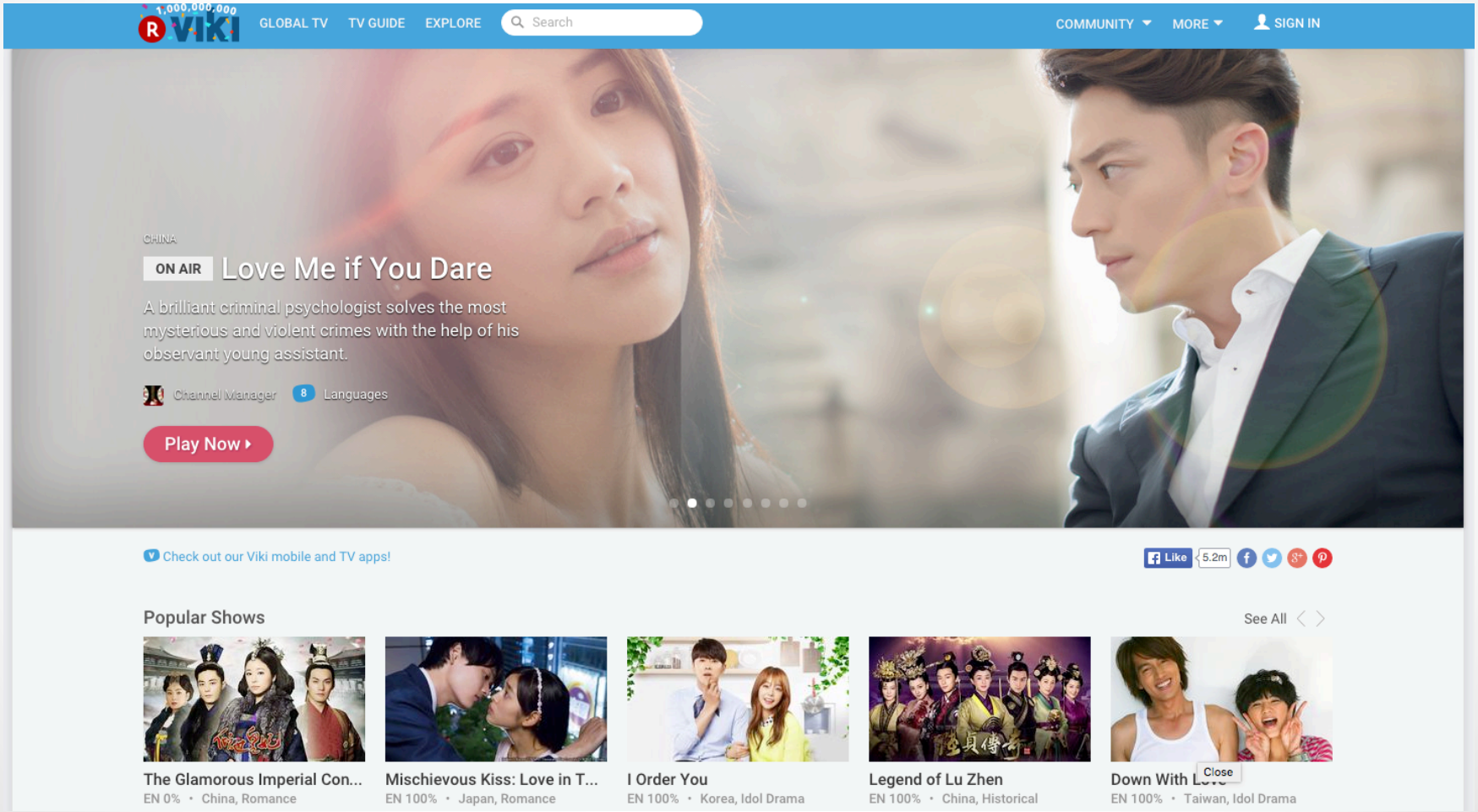
- 2010: MSc. Computer Science – Oxford University
- 2011: Research Engineer – A*STAR DSI
- 2013: Data – ZALORA Group
- 2015: Data – Commercialize TV

https://github.com/lenguyenthedat

https://sg.linkedin.com/in/lenguyenthedat

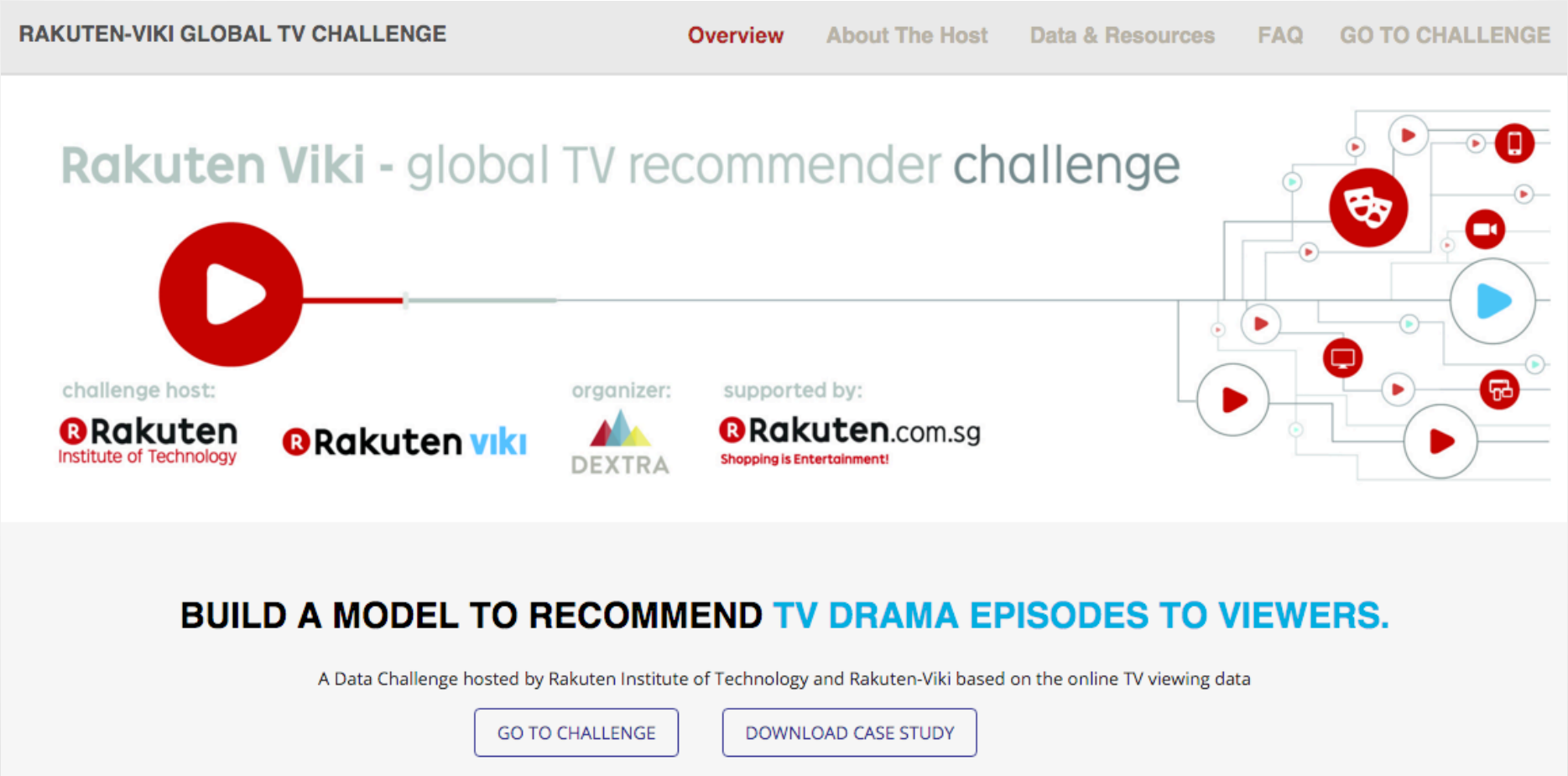
Challenge descriptions

https://www.viki.com/



Challenge descriptions

http://www.dextra.sg/challenges/rakuten-viki-video-challenge/



Challenge descriptions

Data:

- (880,000) User Attributes *(country – gender)*
- (600) Video attributes *(country – language – genre – owner – casts)*
- (4,880,000) User viewing behavior *(video – user – score)*

Task:

- Recommendation engine - prediction for each user *(user – top 3 videos)*
- Insights

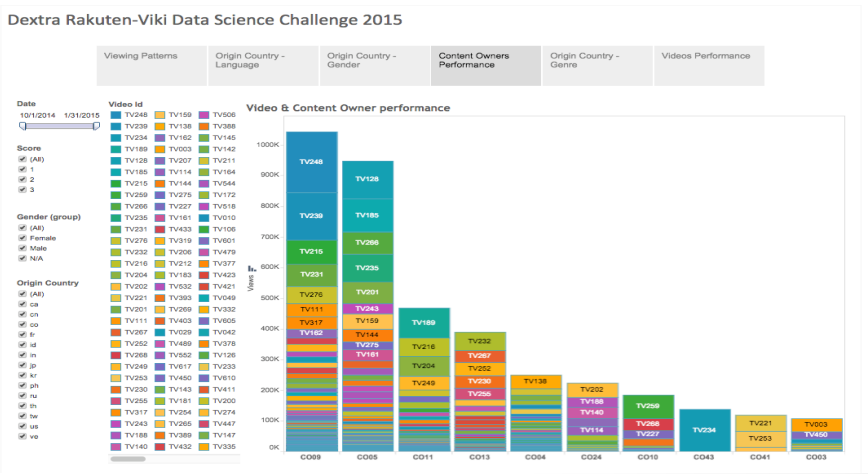
Case study:

- http://www.dextra.sg/wp-content/uploads/2015/09/CaseStudy_Viki.pdf

Useful Links

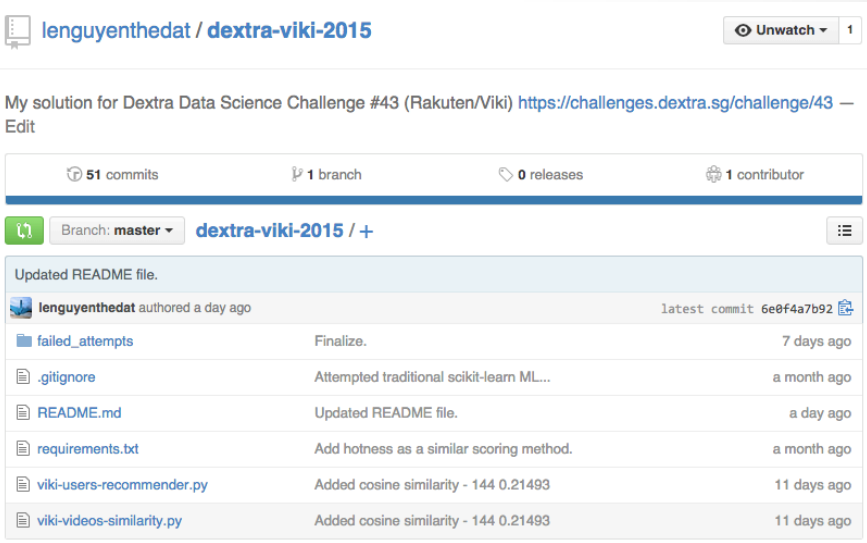
Tableau Public Visualization:

http://tiny.cc/viki-viz



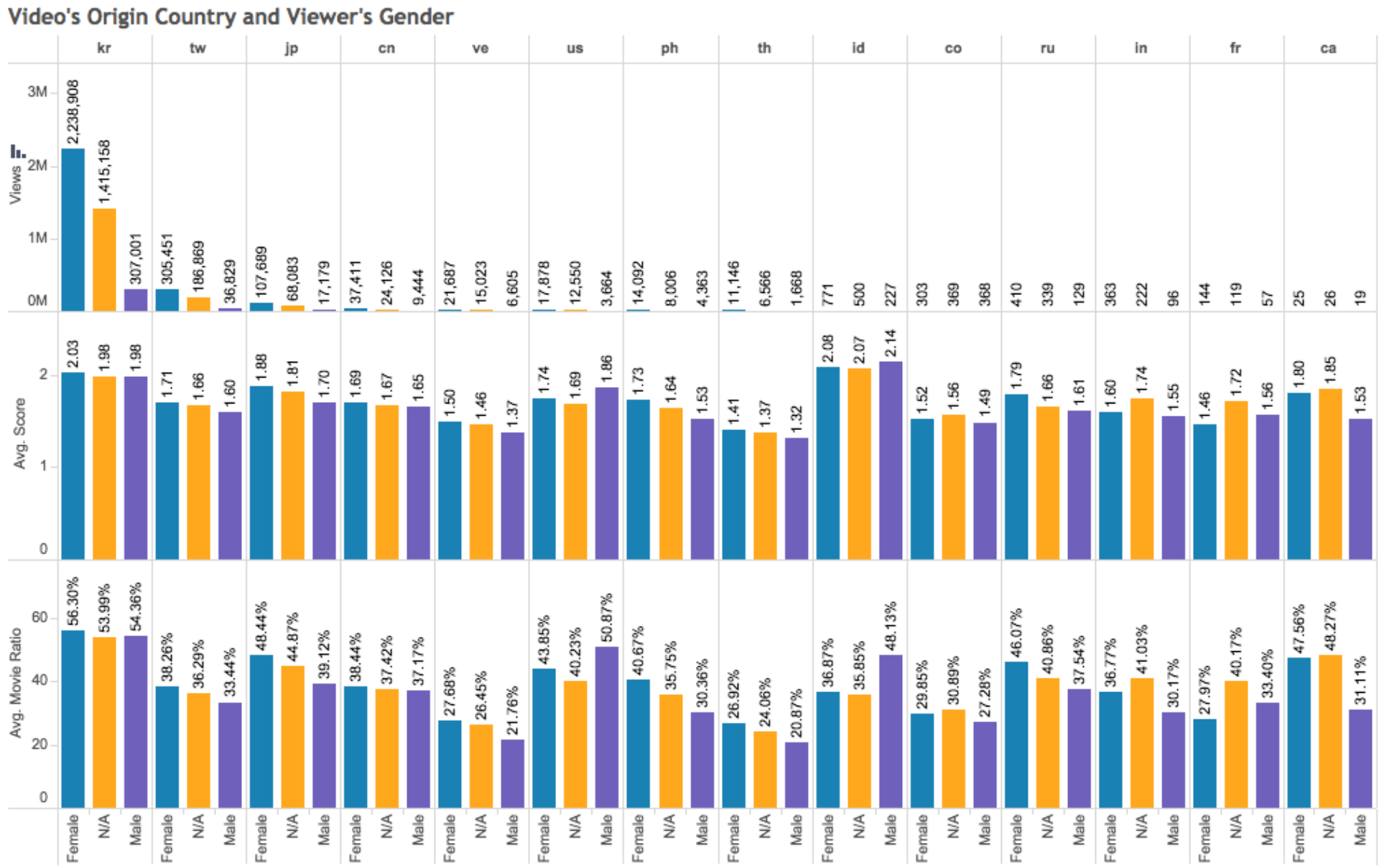
Source Code:

http://tiny.cc/viki-src

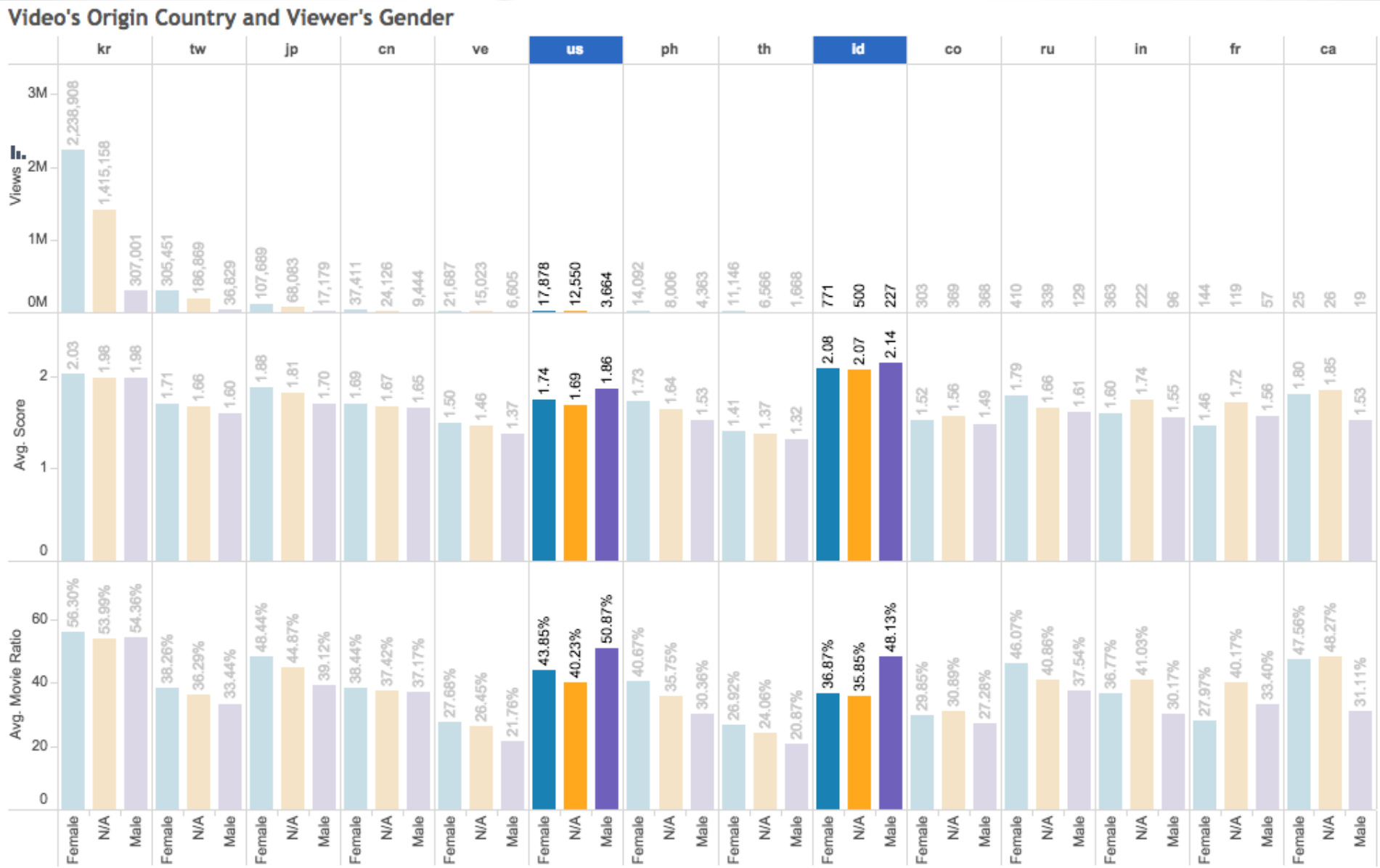


Preliminary Analysis

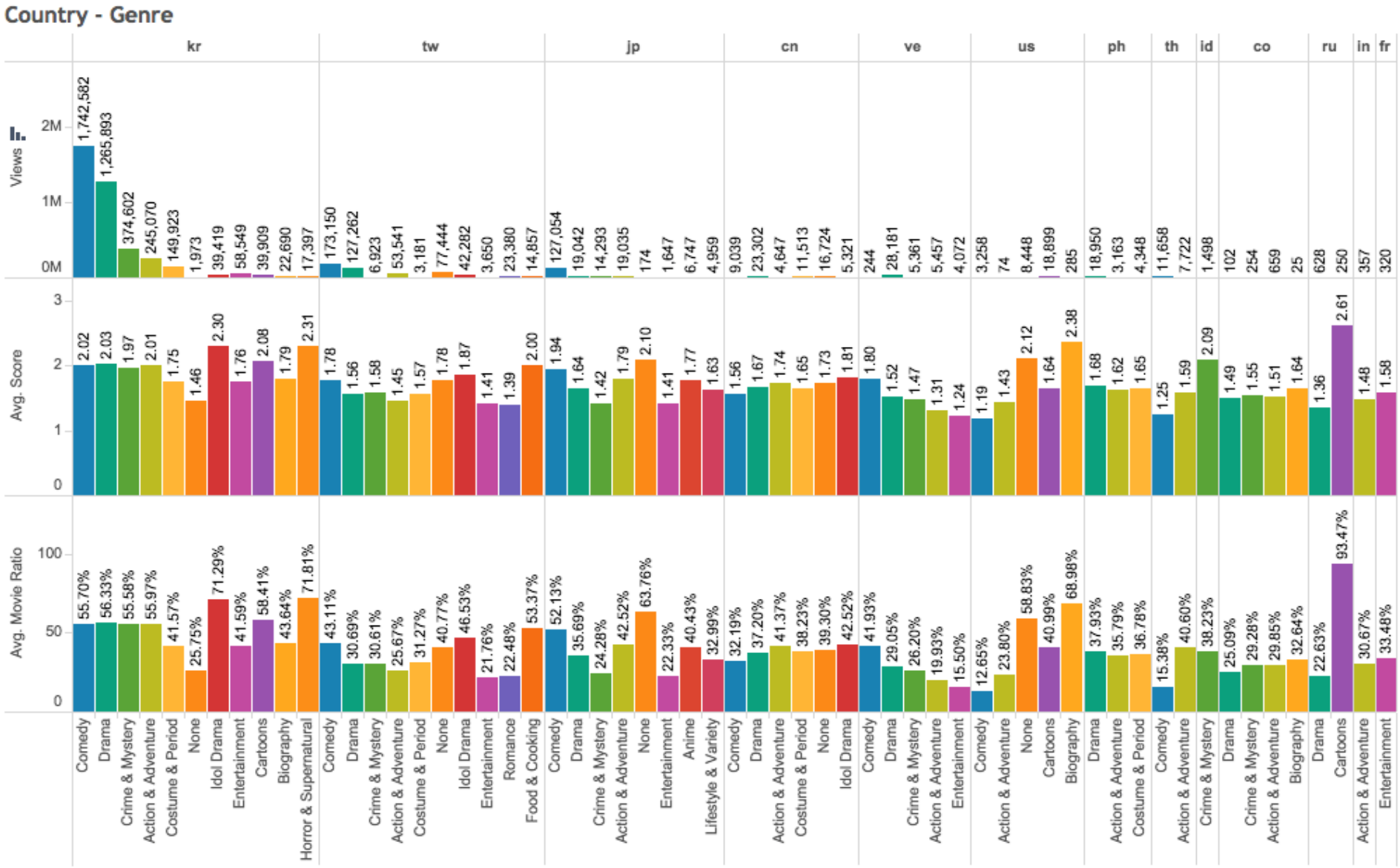
Analysis – Gender



Analysis – Gender

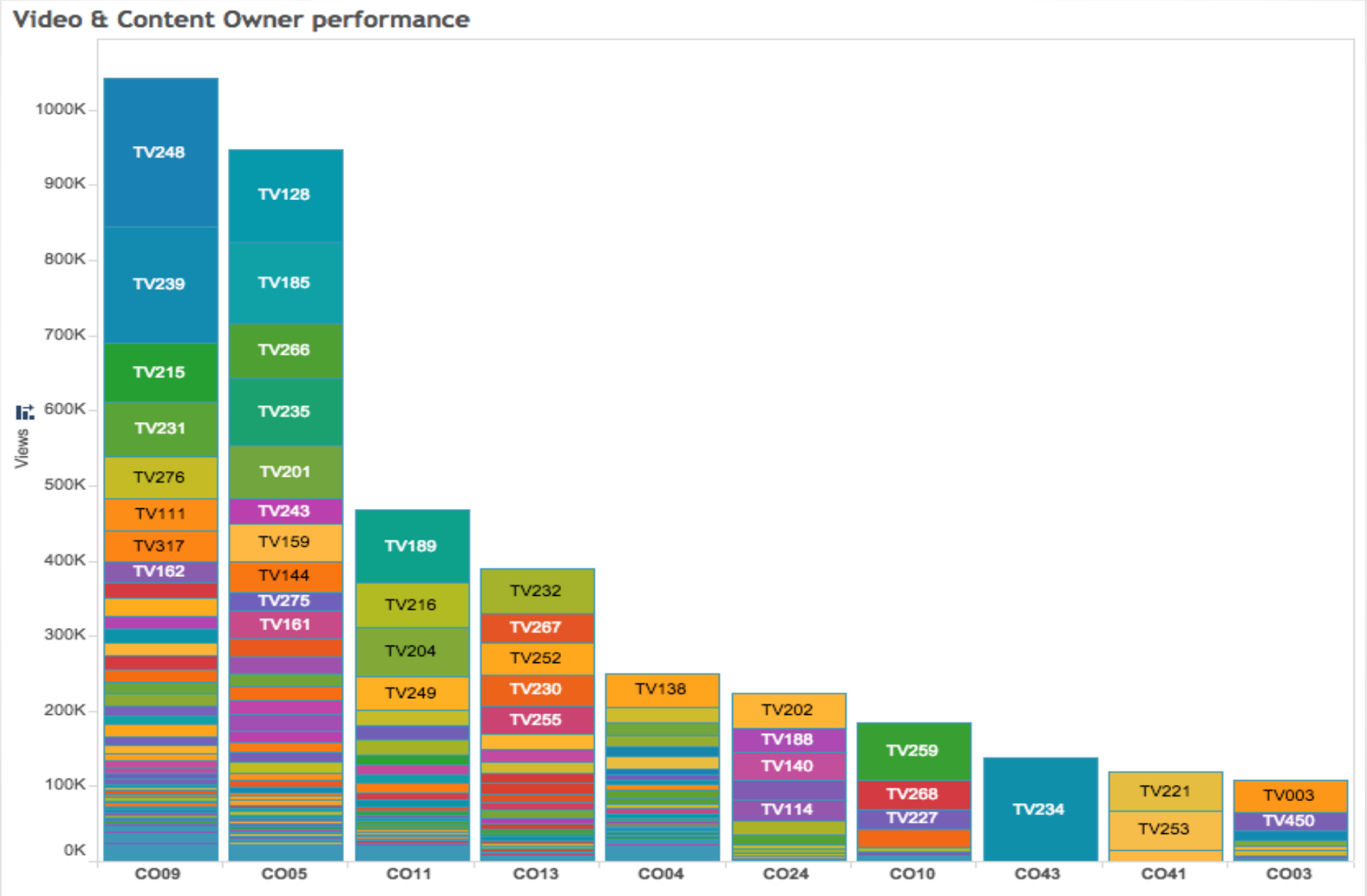


Analysis – Genre

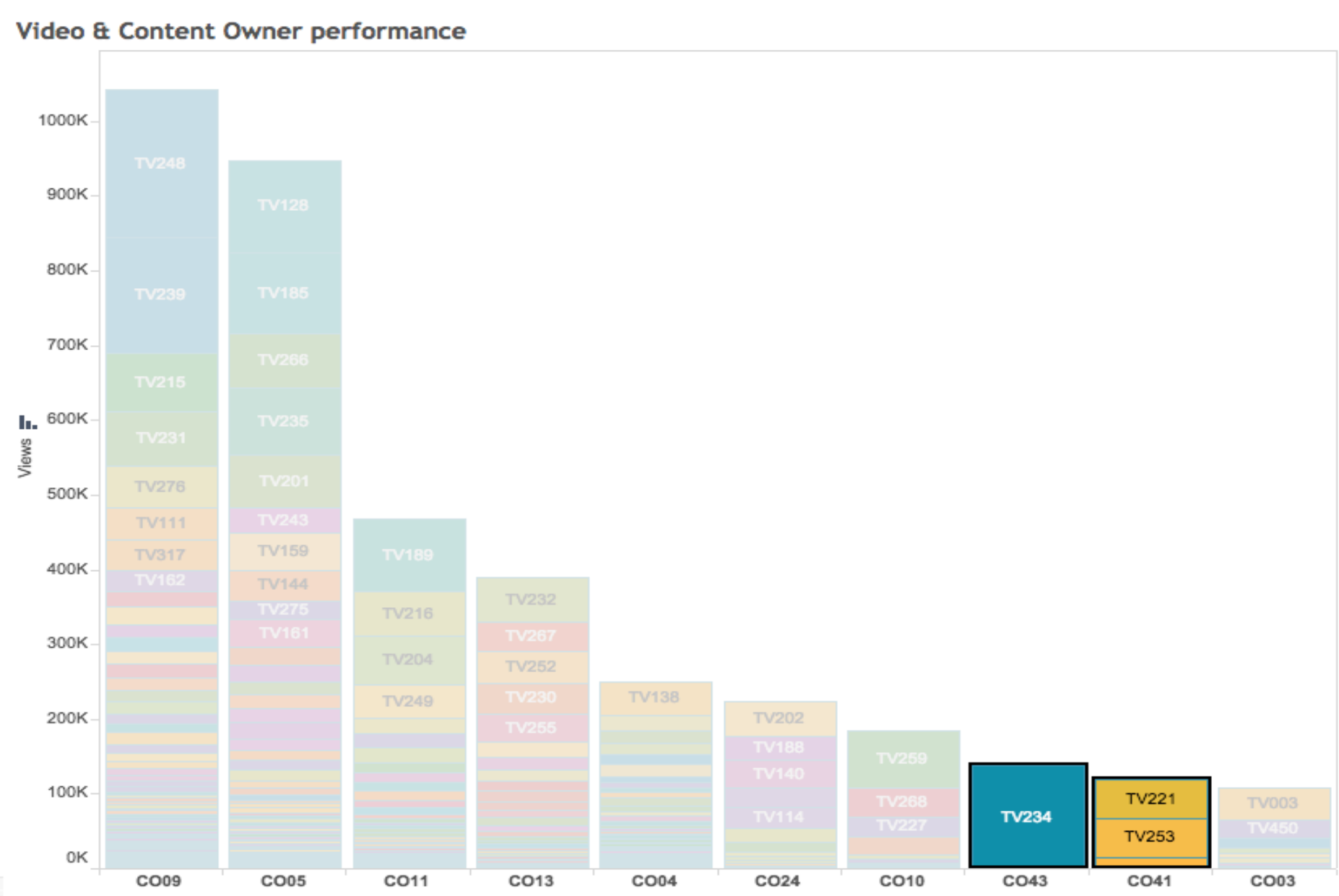


Analysis – Genre

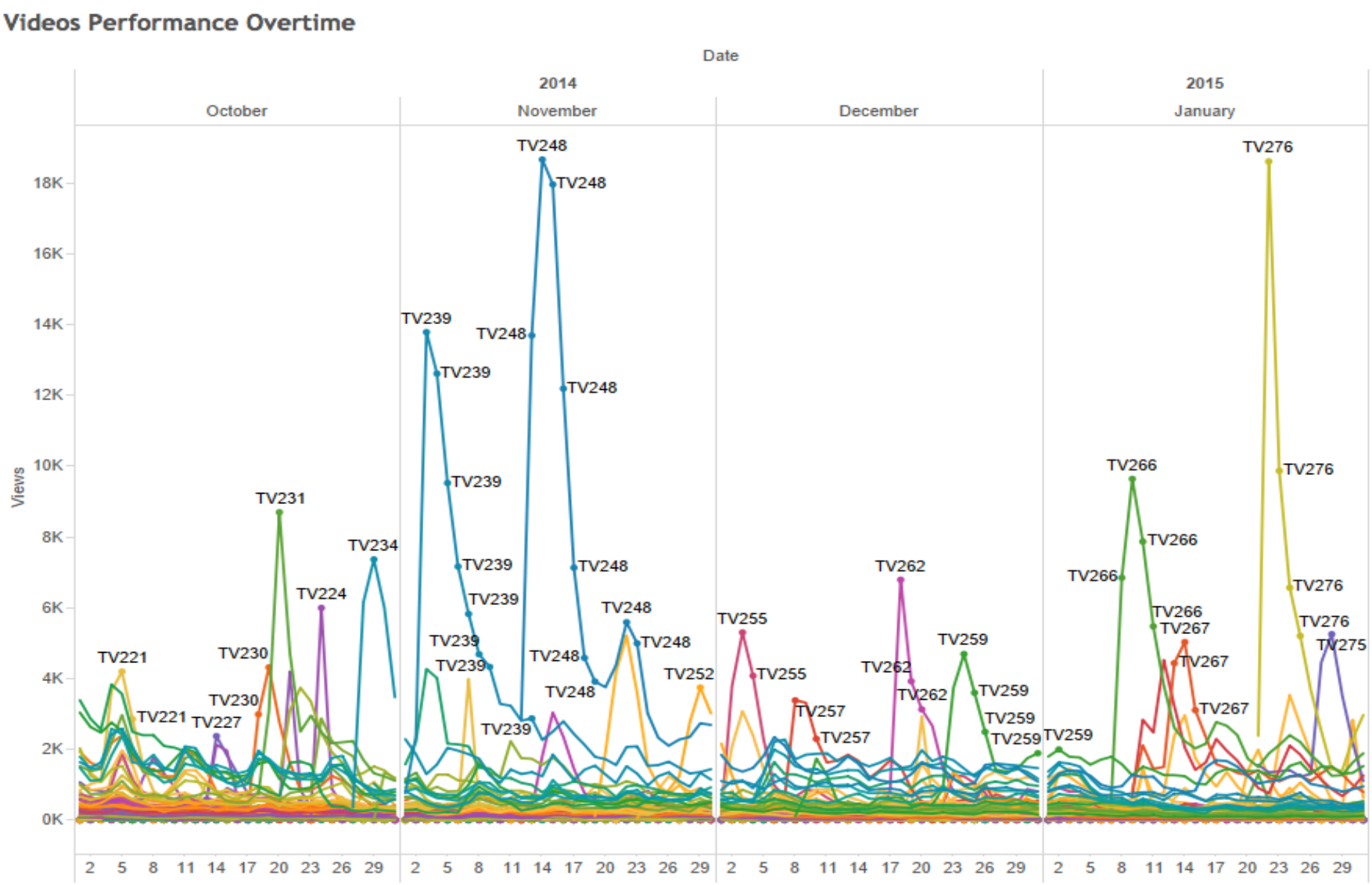
Analysis – Content Owner



Analysis – Content Owner

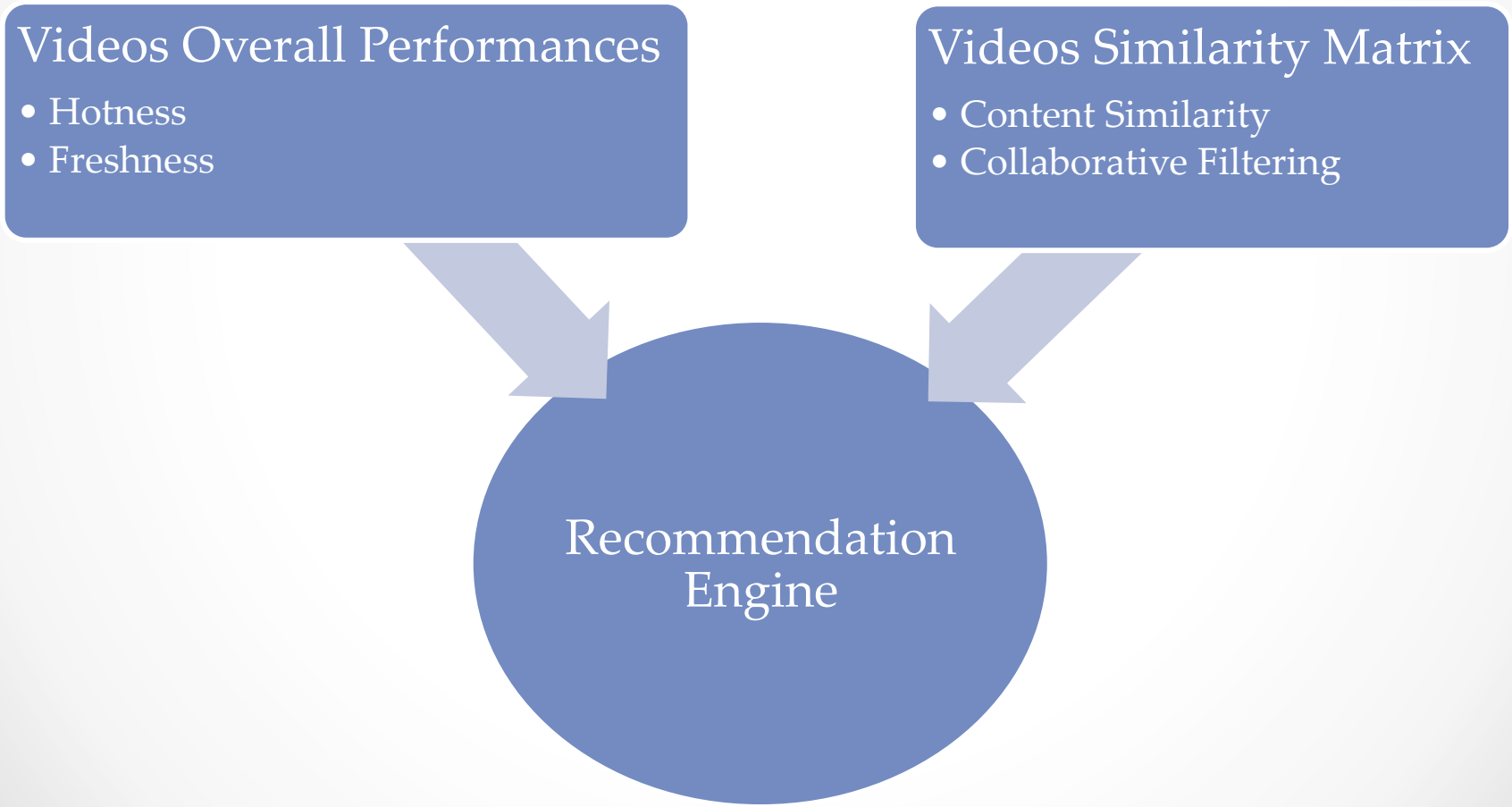


Analysis – Videos Traffic

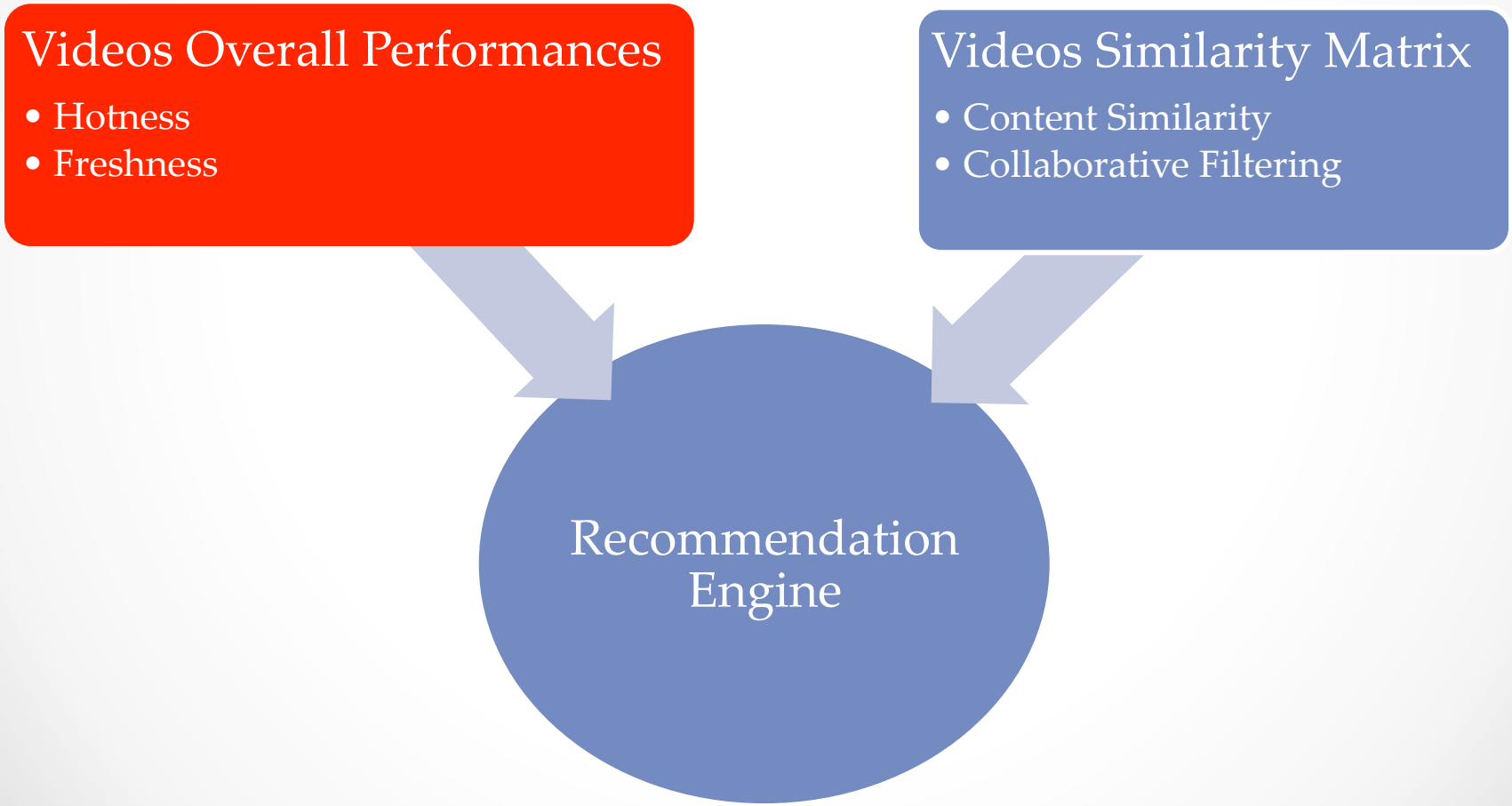


Algorithm Overview

Training phase



Training phase



Training phase

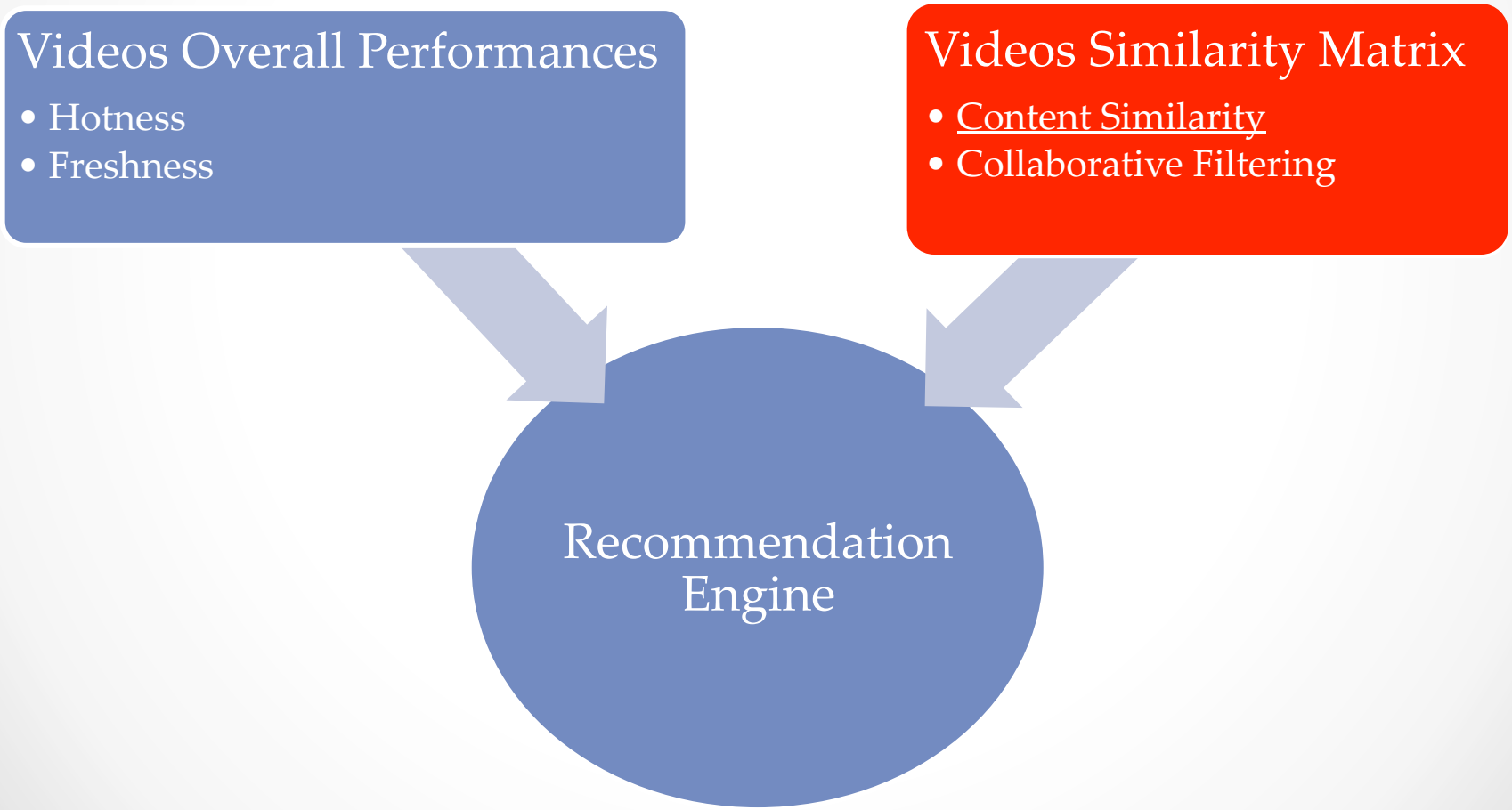
Videos overall performances:

$$Hotness* = \frac{\sum usersWatched}{firstDate - lastDate}$$

*With gender filter applied

$$Freshness = \frac{1}{\left(broadcastDate - currentDate\right)^2}$$

Training phase



Training phase

Videos similarity Matrix – Content Similarity

- Original Country:

$$V_1.country == V_2.country$$

- Original Language:

$$V_1.language == V_2.language$$

- Adult Content:

$$\left(V_1.adult == 1\right) \& \left(V_2.adult == 1\right)$$

- Content Owner:

$$V_1.contentOwner == V_2.contentOwner$$

Training phase

Videos similarity Matrix – Content Similarity

- Episode Count:

$$\frac{\min\left(V_1.episodeCount, V_2.episodeCount\right)}{\max\left(V_1.episodeCount, V_2.episodeCount\right)}$$

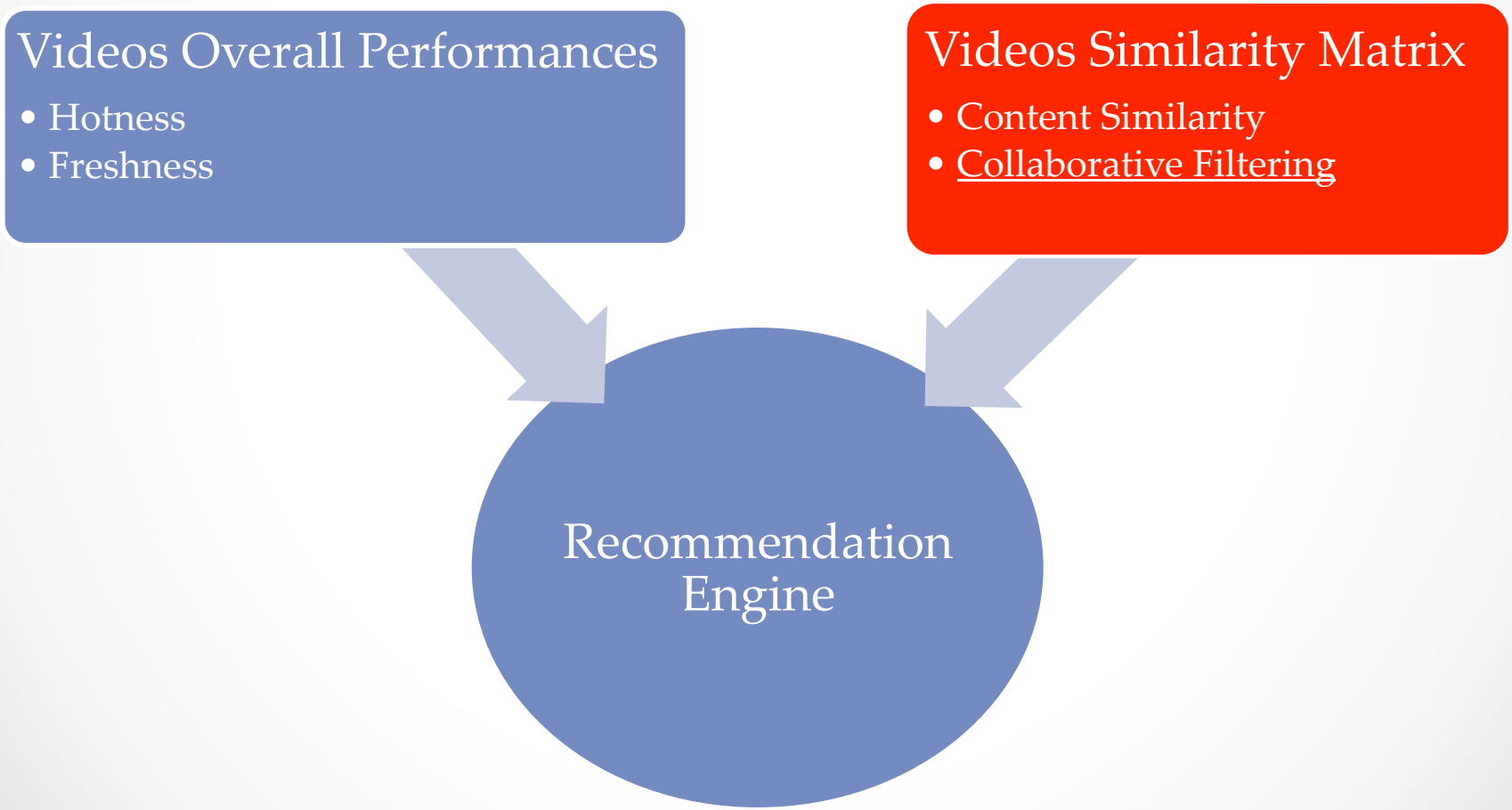
- Genre:

$$J\left(v_1, v_2\right) = \frac{G_1 \cap G_2}{G_1 \cup G_2}$$

- Cast:

$$J\left(v_1, v_2\right) = \frac{C_1 \cap C_2}{C_1 \cup C_2}$$

Training phase

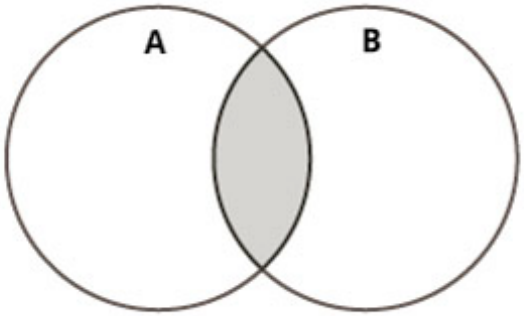


Training phase

Videos similarity Matrix – Collaborative Filtering

- Jaccard Index - https://en.wikipedia.org/wiki/Jaccard_index:

$$J(v_1, v_2) = \left| \frac{U_1 \cap U_2}{U_1 \cup U_2} \right|$$



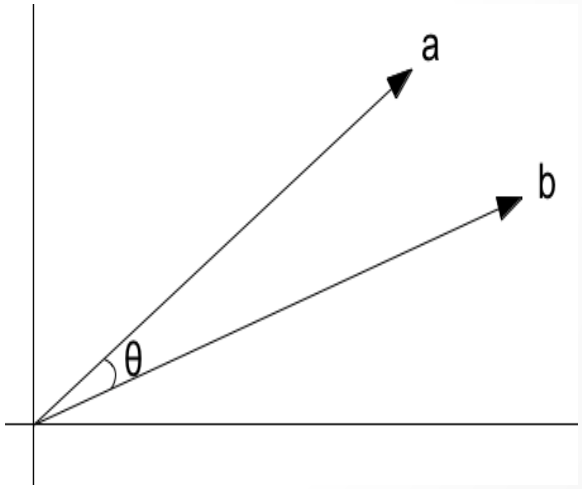
- Set theory
- Ratio of intersection gives similarity score
- Sensitive to sparse input – limit to only top 25% videos

Training phase

Videos similarity Matrix – Collaborative Filtering

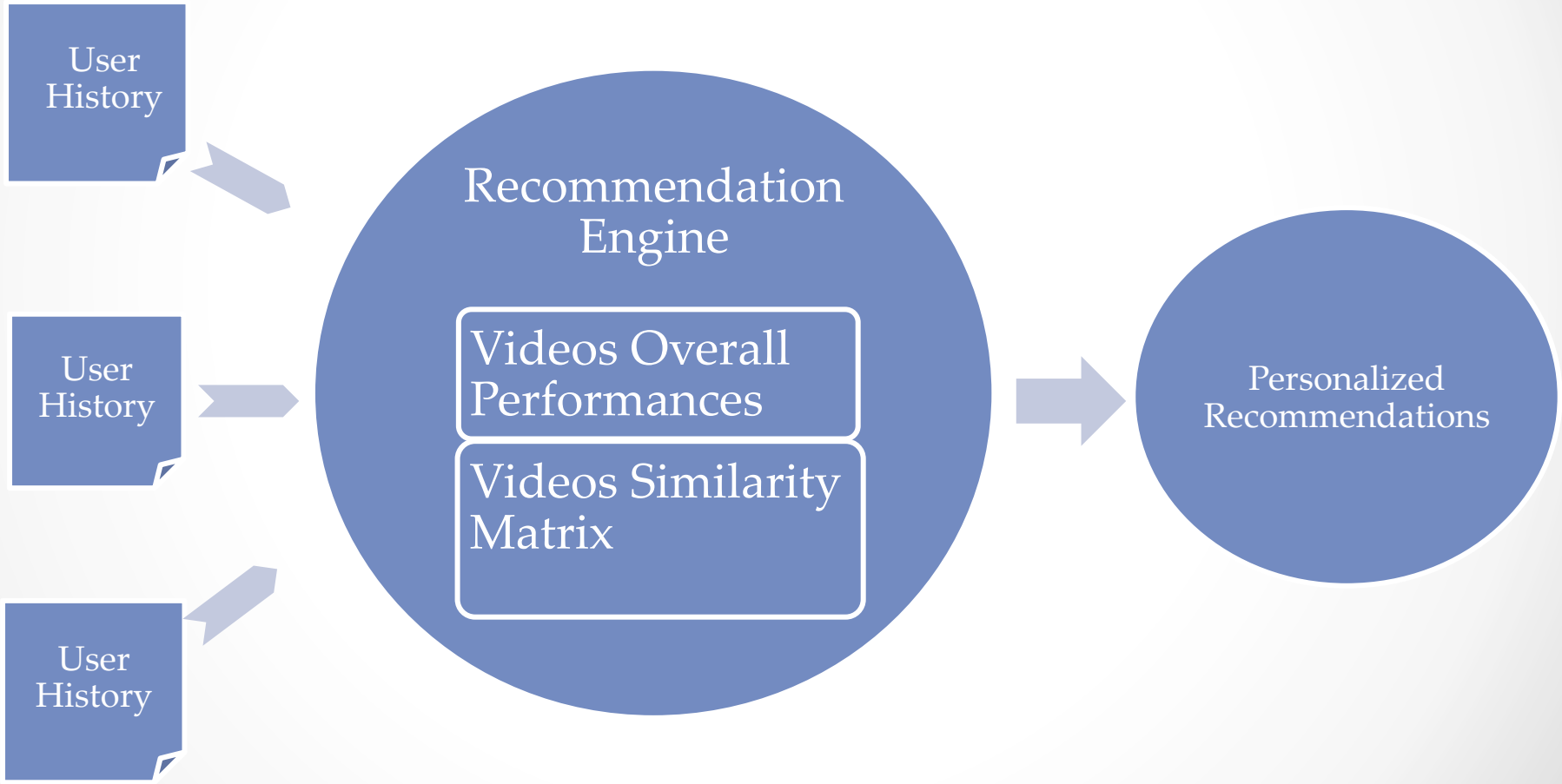
- Cosine Similarity - https://en.wikipedia.org/wiki/Cosine_similarity:

$$\cos(v_1, v_2) = \frac{\vec{U}_1 \bullet \vec{U}_2}{\|\vec{U}_1\| \|\vec{U}_2\|} = \frac{\sum_{i=1}^{n} U_{1,i} \bullet U_{2,i}}{\sqrt{\sum_{i=1}^{n} U_{1,i}^2} \times \sqrt{\sum_{i=1}^{n} U_{2,i}^2}}$$



- Vector space model
- Angle between 2 vectors gives similarity score.
- Good for sparse input – can apply gender filter.

Personalization phase



Performance

Overall time & space complexity:

$$O\left(uv^2\right)$$

- u: number of users (880,000)
- v: number of videos (600)

Advantages:

- Lightweight – fits in 8GB Macbook Air!
- Scalable (fully distributed with SparklingPandas)

Applications

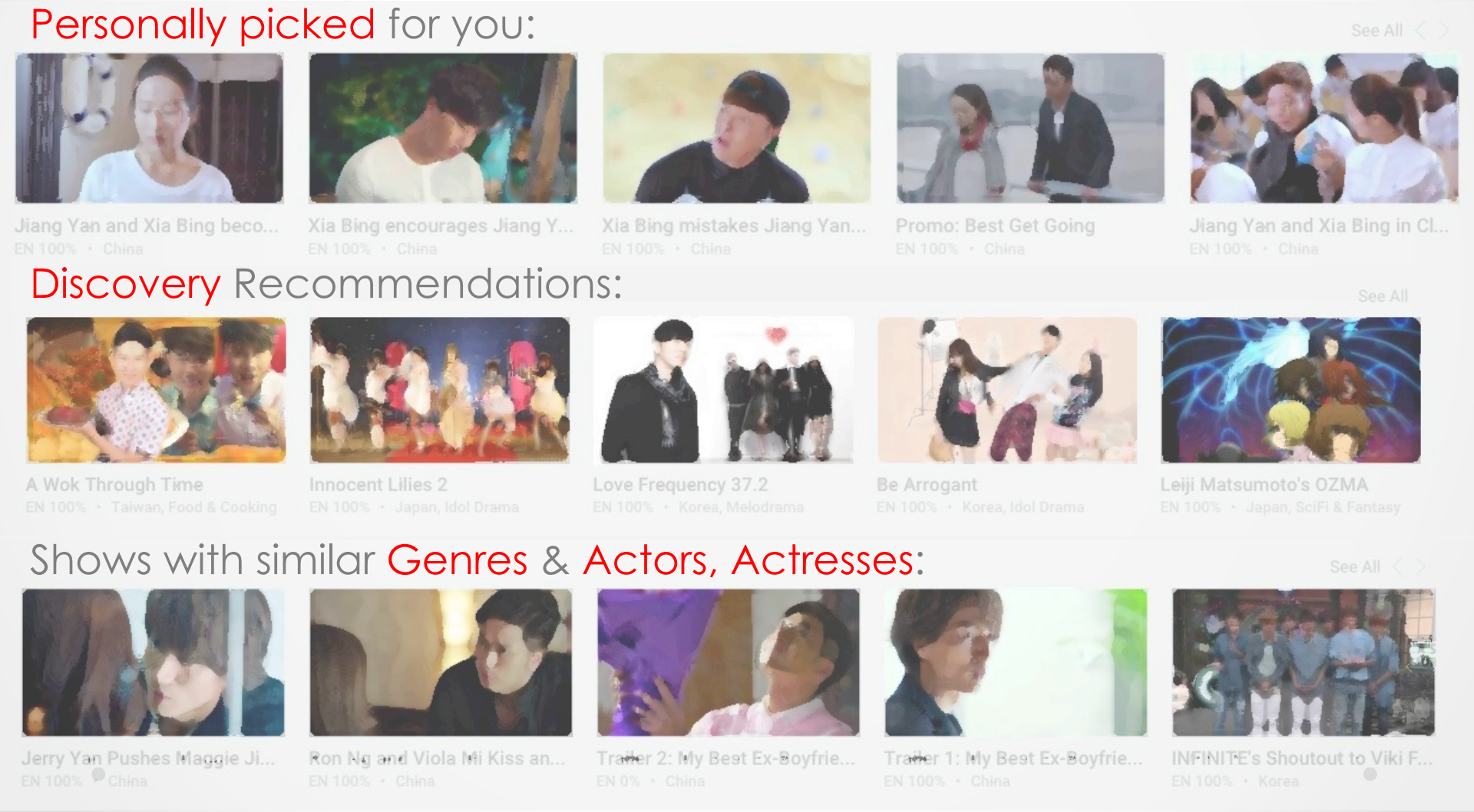
Flexibility:

- Custom weightages for:
  - Features
  - Collaborative filtering similarity scores
  - Video performances (hotness or freshness)
  - Individual User - Video scores

Not just an engine but a framework:

- To create different recommendation engines.

Applications



Suggestions

- **Additional useful data sets:**
  - Explicit user rating is also very important.
  - User's contributions data (subtitles).
  - User's and video's interactions data (live comments).
- **Training & Testing data:**
  - Should exclude top videos.

  (Promoted on front-page or banners.)
- **Evaluation method:**
  - Equal test set splits will give an overall better result.

  (Models that work well with Feb 2015 data might not work very well with March 2015 data)

Technology stack

- **Tableau Public**
  - Free to download
  - Publicly shared workbooks
  - Interactive visualizations and insights

- **Python**
  - Pandas: data analysis library
  - Scikit-Learn: machine learning library
  - iPython Notebook: IDE for data analysis
  - Other libraries:
    - Spotify's annoy: approx. nearest neighbors calculation
    - PySpark's Mllib: spark's machine learning
    - panns: approx. nearest neighbors search
    - python-recsys: recommendation system

Thank you!