

1. Preprocessing the data

- ```
posts.head()
```
- |   | post_id | creator_id | content_type | tags           | text          | tags_clean    |
|---|---------|------------|--------------|----------------|---------------|---------------|
| 0 | P1      | U44        | video        | sports, food   | sports food   | sports food   |
| 1 | P2      | U26        | video        | music, travel  | music travel  | music travel  |
| 2 | P3      | U32        | text         | sports, travel | sports travel | sports travel |
| 3 | P4      | U6         | image        | music, gaming  | music gaming  | music gaming  |
| 4 | P5      | U32        | image        | food, fashion  | food fashion  | food fashion  |

2. Building Post TF-IDF Matrix to represent posts based on their tags. Words like 'sports' , 'food' - get weights.
3. Finding the tags, the user has already engaged with.

```
user_eng_texts

{'U1': 'music sports music sports food music gaming sports literature travel sports gaming fitness literature art literature music fashion art music food gaming tech sports literature literature music sports sports literature travel',
 'U10': 'tech music tech literature sports fitness literature art art music sports music fashion literature travel literature fitness literature food sports literature food fashion travel literature music food gaming gaming music travel fashion sports art',
 'U11': 'literature fitness sports food art food tech gaming literature music literature travel music fashion sports literature travel fashion sports literature music sports literature tech travel sports literature food travel sports music travel literature fitness',
 'U12': 'fitness fitness sports sports travel music gaming art food sports tech food literature music gaming tech travel fashion food fitness sports art art food sports travel gaming tech literature music music',
 'U13': 'fitness tech sports literature art tech food sports literature literature music gaming gaming food fitness travel sports music food literature fitness literature tech food literature fitness music sports art',
 'U14': 'gaming tech gaming food gaming tech sports food literature music fitness art food music fashion literature travel sports sports music tech literature food sports gaming music food sports literature food literature gaming',
 'U15': 'sports food sports music fitness tech sports art literature fitness sports sports fitness literature sports music travel sports gaming fitness food sports food sports'}
```

4. Building User Profiles representing each user's interests and their engagement history.

```
appending each user's engaged-post tags to their profile doc
for i, uid in enumerate(users['user_id']):
 users_profile_docs[i] += ' ' + user_eng_texts.get(uid, '')

users_profile_docs

[1] ['sports art gaming music sports music sports food music gaming sports literature travel sports gaming fitness literature art literature music fashion art music food gaming tech
sports literature literature music sports sports literature travel music sports music sports food music gaming sports literature travel sports gaming fitness literature art
literature music fashion art music food gaming tech sports literature literature music sports sports literature travel',
'travel food fashion music fashion music gaming food music sports literature art travel music sports food sports art music sports fitness food fitness fitness literature gaming
fitness art art food sports literature tech tech travel fashion music fashion music gaming food music sports literature art travel music sports food sports art music sports
fitness food fitness fitness literature gaming fitness art art food sports literature tech tech travel fashion',
'sports travel fashion music sports art gaming fitness gaming travel fashion music travel food fitness food fitness food travel food tech fitness tech gaming fitness gaming
tech gaming sports fashion food sports music sports art gaming fitness gaming travel fashion music travel food fitness food fitness food travel food tech fitness tech gaming
fitness gaming tech gaming sports fashion food sports']
```

Combines **user interests** + **all posts the user has engaged with**.


5. Transforming User Profiles with TF-IDF; representing user profiles in the same vector space as posts.

```
user docs to tf-idf vectors
user_tfidf = tfv.transform(users_profile_docs)
```

6. Computing User-Post Similarity to find score; how relevant each post is to each user. High value means a post is more relevant to the user's interests & past engagements.
7. Excluding Already Engaged Posts to avoid recommending posts the user has already interacted with by assigning them a really small weight i.e.  $-1e9$  to eradicate their chance of making it to the top 3.
8. Now finally generating the top 3 recommendations to show top 3 most relevant posts for the user.


## Metrics

To assess recommendation quality, we are using the following metrics computed on a **train/test split** of engagement data:

1.  Precision@3: 0.03333333333333333

**Precision@K** is a fraction (0 to 1) of recommended items that are actually relevant (engaged). So 0.33 signifies the fraction of top-3 recommended posts that the user actually engaged with in the test set.

On average, only 3% of recommendations were relevant in the test set. This was expected due to sparse user engagement (most users engage with very few posts. with top-3 recommendations, hitting the right post is statistically unlikely) and **exclusion of previously engaged posts** to avoid overfitting (**reduces chance of matching historical engagements in test, lowering Precision@3.**)

2.  NDCG@3: 0.0459216382193173

Measures ranking quality, giving higher weight to relevant posts appearing higher in the recommendation list. Indicates that relevant posts (from test engagements) occasionally appear **higher in the top-3**. **Overall ranking effectiveness is low due to content-based TF-IDF limitations.**

## Possible Extensions

- Including additional post features (e.g., content type, creator popularity) to improve relevance of the recommendations that are shown to the user. A data set with more features might prove to be helpful.
- Using **semantic embeddings** (BERT, sentence transformers) for richer content representation i.e capturing **synonyms, context and meanings** of posts and user profiles, improving similarity computation and thereby improving recommendation quality.
- Adding **hybrid models** to combine content-based (TF-IDF or embeddings) with collaborative filtering (user-item interactions; e.g., users who liked post X also liked post Y).
- Consider **personalization weights** for recent engagement or interest tags giving higher importance to posts the user engaged with recently to reflect **current interests**.
- **Cold-Start Handling:**
  - **New Users:** Using only user profile interests to generate recommendations.
  - **New Posts:** Recommending to users with matching interests based on content similarity until engagement data accumulates.