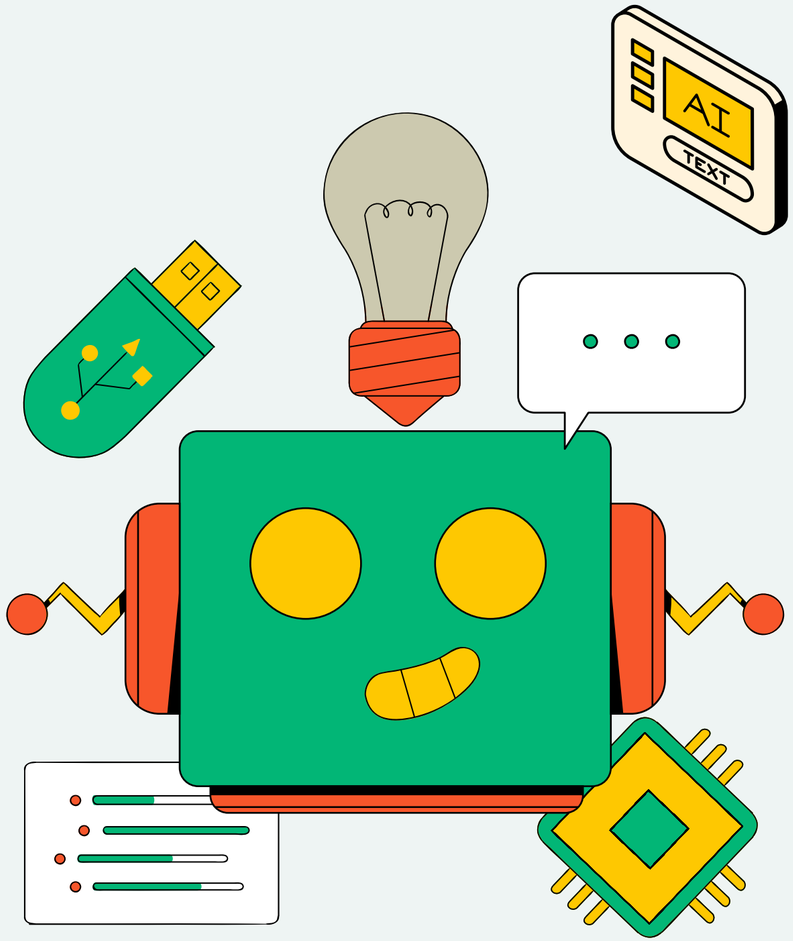
THYNK UNLIMITED

WE LEARN FOR THE FUTURE



NLP FOR MOVIE RECOMMENDATIONS

A CONTENT-BASED APPROACH USING TF-IDF AND COSINE SIMILARITY

1. **ARTHAZ ANTHONY -** 2702320860
2. **EVAN FREDERICKSEN HARTONO -** 2702322853
3. **KENNETH ANGELO SULAIMAN -** 2702373715
4. **OSEL CITTA CHEN -** 2702235134
5. **YOSEPRIL ZHOUNGGI -** 2702369346

PRESENTATION OUTLINE

- Introduction
- TFIDF
- Cosine Similarity
- Penerapan pada program
- Evalutation Metric (ILS)

INTRODUCTION

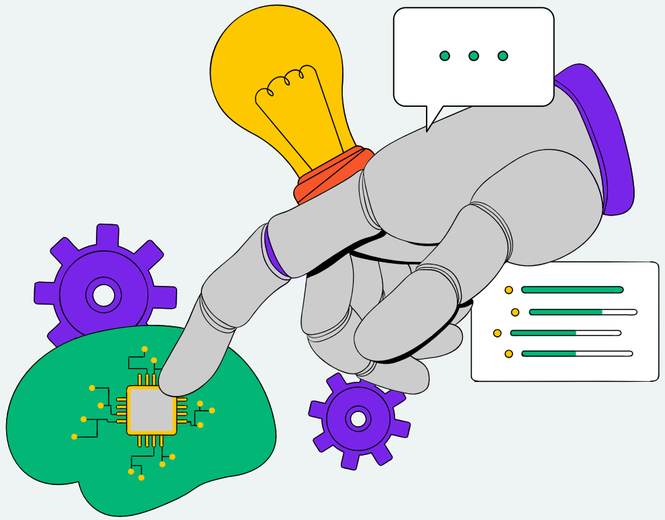
- **Aplikasi: Analisis deskripsi film untuk rekomendasi berbasis konten.**
- **Tujuan: Merekomendasikan film serupa berdasarkan teks deskripsi**

TF-IDF

TF-IDF (Term Frequency-Inverse Document Frequency): Metode statistik untuk menilai pentingnya kata dalam dokumen relatif terhadap korpus.

- Komponen:

Term Frequency (TF): Mengukur frekuensi munculnya suatu kata (t) dalam sebuah dokumen (d). Frekuensi yang lebih tinggi menunjukkan tingkat kepentingan yang lebih besar.
Rumus: TF(t, d) = (Jumlah kemunculan t di d) / (Total kata di d)
Inverse Document Frequency (IDF): Mengurangi bobot kata (t) yang sering muncul di banyak dokumen (d), sambil meningkatkan bobot kata (t) yang jarang muncul. Jika sebuah kata muncul di lebih sedikit dokumen, maka kemungkinan lebih bermakna dan spesifik.
Rumus: IDF(t, D) = log((Jumlah dokumen di D) / (Jumlah dokumen yang mengandung t + 1))
Skor TF-IDF:
Bobot kata berdasarkan TF dan IDF.
Rumus: TF-IDF(t, d, D) = TF(t, d) × IDF(t, D)
Fungsi: Mengubah teks menjadi vektor numerik, di mana setiap dimensi adalah kata unik dengan bobot TF-IDF.



$$\text{TF}(t, d) = \frac{\text{Number of times term t appears in document d}}{\text{Total number of terms in document d}}$$

$$\text{IDF}(t, D) = \log \frac{\text{Total number of documents in corpus D}}{\text{Number of documents containing term t}}$$

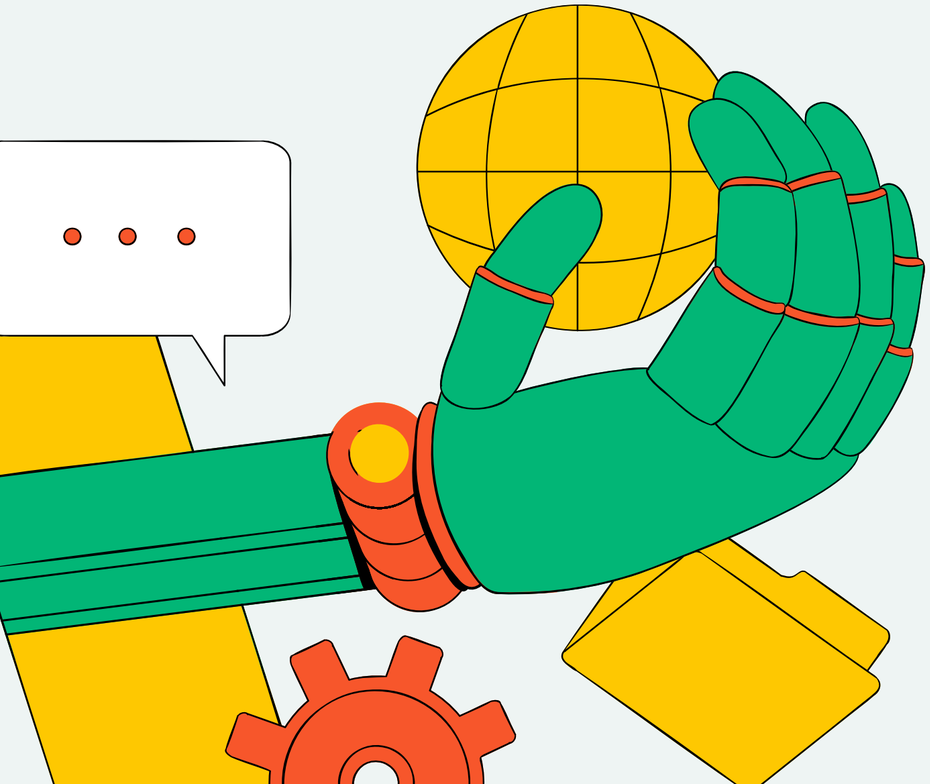
$$\mathbf{tf}(t, d) = \frac{f_d(t)}{\max_{w \in d} f_d(w)}$$

$$\mathbf{idf}(t, D) = \ln\left(\frac{|D|}{|\{d \in D : t \in d\}|}\right)$$

$$\mathbf{tfidf}(t, d, D) = \mathbf{tf}(t, d) \cdot \mathbf{idf}(t, D)$$

COSINE SIMILARITY

(MENGUKUR KEMIRIPAN VEKTOR)



- Rumus: cosine_similarity(A, B) = (A · B) / (|A| × |B|)

A · B: Dot product vektor A dan B
|A|, |B|: Panjang (norm) vektor A dan B

- Interpretasi: Nilai 1 berarti vektor identik, 0 berarti tidak ada kemiripan.
- Aplikasi: Mengukur kemiripan antara vektor TF-IDF deskripsi film.

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

PENERAPAN DALAM REKOMENDASI FILM

DATA PREPARATION

- Tokenisasi
- Lowercase
- Penghapusan stop-word pada deskripsi film.

TF-IDF MATRIX

- Fitur-fitur yang dalam bentuk teks akan diubah menjadi bentuk numerik.
- Hitung vektor TF-IDF untuk setiap deskripsi.

COSINE SIMILARITY

- Hitung kemiripan antara vektor film yang dipilih dan semua film lain.

FLOW PEMROSESAN

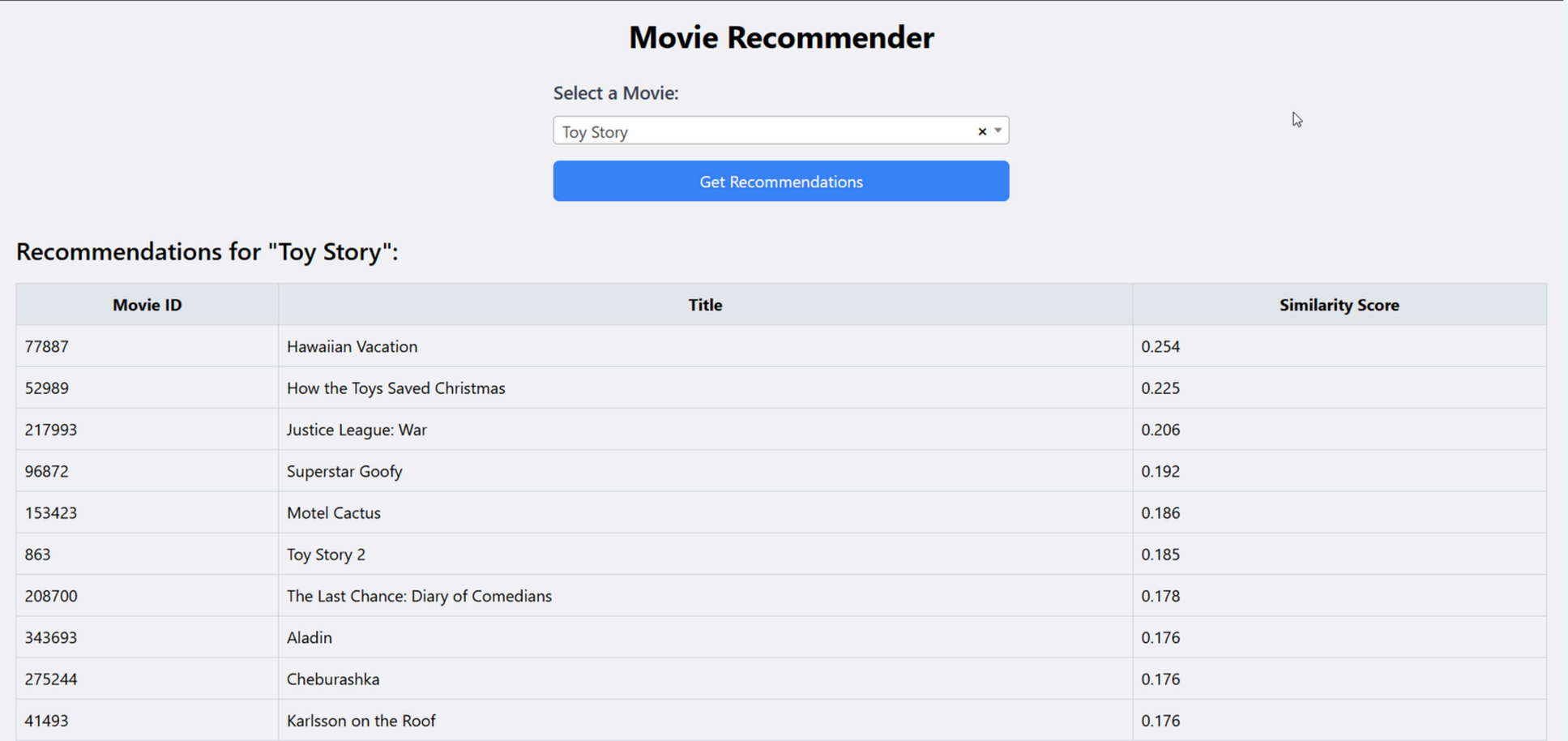
- Deskripsi Film:
  - Toy Story: "Led by Woody, Andy's toys live happily in his room until a new toy, Buzz Lightyear, arrives and challenges Woody's leadership."
  - Jumanji: "When siblings Judy and Peter discover an enchanted board game, they are thrust into a magical adventure with toys and challenges."
- Pra-pemrosesan: Kata unik: "cerita", "persahabatan", "petualangan", "kisah", "fokus".
- TF-IDF:
  - Hitung TF dan IDF untuk setiap kata.
  - Misal, ‘toys’ muncul di kedua deskripsi, berkontribusi pada score similarity
  - "Cerita" dan "kisah" mungkin memiliki IDF lebih tinggi jika jarang muncul.
- Cosine Similarity: Hitung kemiripan antara vektor Film A dan B.
- Hasil: 0.0762

Word	Toy Story	Jumanji
adventure	0.00	0.30
andy	0.25	0.00
arrives	0.25	0.00
board	0.00	0.30
buzz	0.25	0.00
challenges	0.18	0.21
discover	0.00	0.30
enchanted	0.00	0.30
game	0.00	0.30
happily	0.25	0.00
judy	0.00	0.30
leadership	0.25	0.00
led	0.25	0.00
lightyear	0.25	0.00
live	0.25	0.00
magical	0.00	0.30
new	0.25	0.00
peter	0.00	0.30
room	0.25	0.00
siblings	0.00	0.30
thrust	0.00	0.30
toy	0.25	0.00
toys	0.18	0.21
woody	0.50	0.00

	genres	id	overview	tagline	title
0	[{'id': 16, 'name': 'Animation'}, {'id': 35, '...	862	Led by Woody, Andy's toys live happily in his ...	NaN	Toy Story
1	[{'id': 12, 'name': 'Adventure'}, {'id': 14, '...	8844	When siblings Judy and Peter discover an encha...	Roll the dice and unleash the excitement!	Jumanji
2	[{'id': 10749, 'name': 'Romance'}, {'id': 35, ...	15602	A family wedding reignites the ancient feud be...	Still Yelling. Still Fighting. Still Ready for...	Grumpier Old Men
3	[{'id': 35, 'name': 'Comedy'}, {'id': 18, 'nam...	31357	Cheated on, mistreated and stepped on, the wom...	Friends are the people who let you be yourself...	Waiting to Exhale
4	[{'id': 35, 'name': 'Comedy'}]	11862	Just when George Banks has recovered from his ...	Just When His World Is Back To Normal... He's ...	Father of the Bride Part II

```python
def get_content_recommendations(movie_id, tfidf_matrix, movies_df, top_k=10):
    movie_id = int(movie_id) if isinstance(movie_id, (int, str)) else movie_id

    logger.info(f"Processing recommendations for movie_id: {movie_id}")
    idx = movies_df.index[movies_df['movieId'] == movie_id].tolist()
    idx = idx[0]

    cosine_sim = cosine_similarity(tfidf_matrix[idx:idx+1], tfidf_matrix)[0]

    sim_scores = list(enumerate(cosine_sim))
    sim_scores = sorted(sim_scores, key=lambda x: x[1], reverse=True)
    sim_scores = sim_scores[1:top_k+1]

    movie_indices = [i[0] for i in sim_scores]
    similarity_scores = [i[1] for i in sim_scores]
    result = movies_df[['movieId', 'title']].iloc[movie_indices].copy()
    result['similarity_score'] = similarity_scores
    logger.info(f"Found {len(result)} recommendations for movie_id={movie_id}")
    return result
```

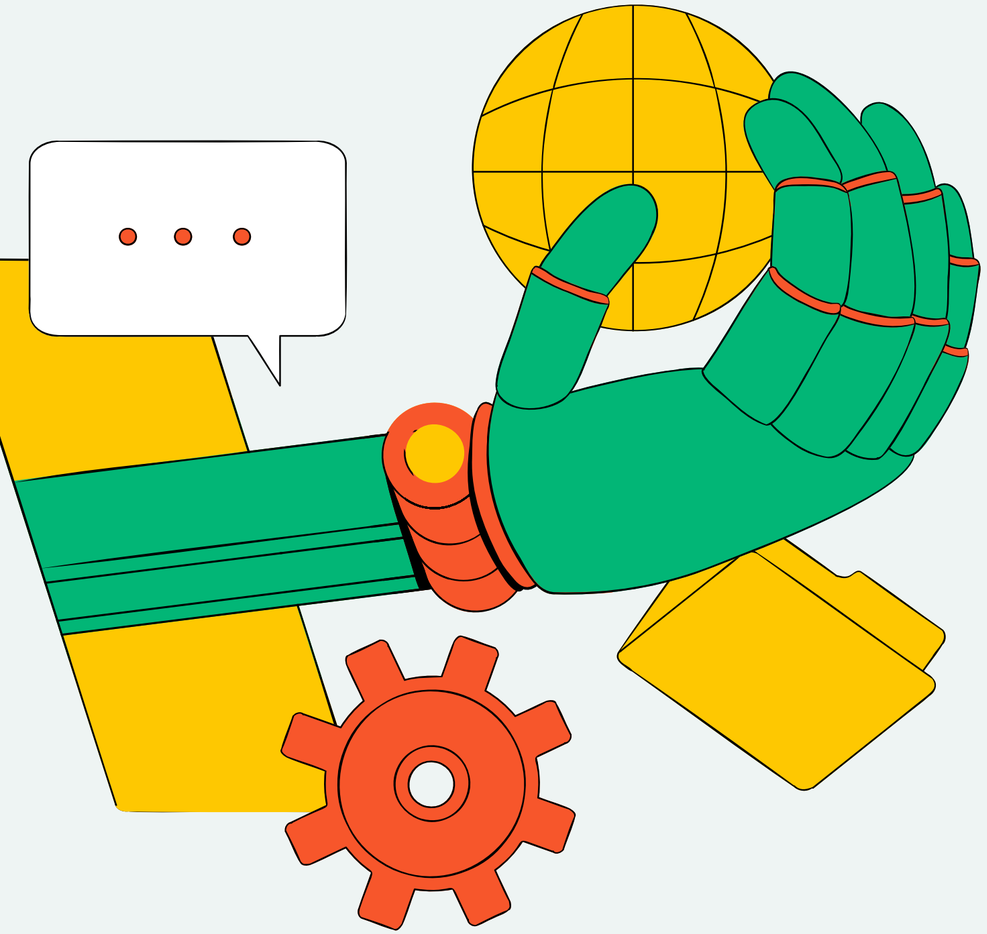
UI OVERVIEW



Movie ID	Title	Similarity Score
77887	Hawaiian Vacation	0.254
52989	How the Toys Saved Christmas	0.225
217993	Justice League: War	0.206
96872	Superstar Goofy	0.192
153423	Motel Cactus	0.186
863	Toy Story 2	0.185
208700	The Last Chance: Diary of Comedians	0.178
343693	Aladin	0.176
275244	Cheburashka	0.176
41493	Karlsson on the Roof	0.176

Gambar di atas memperlihatkan rekomendasi dari film Toy Story, secara nalar manusia seharusnya rekomendasi ToyStory 2 berada pada peringkat pertama rekomendasi namun keyataannya dari program yang dibuat, Toy Story 2 berada pada peringkat ke-6, hal ini menandakan bahwah sistem rekomendasinya belum sempurna untuk melakukan rekomendasi secara menyeluruh

EVALUATION METRICS INTRA-LIST SIMILARITY (ILS)



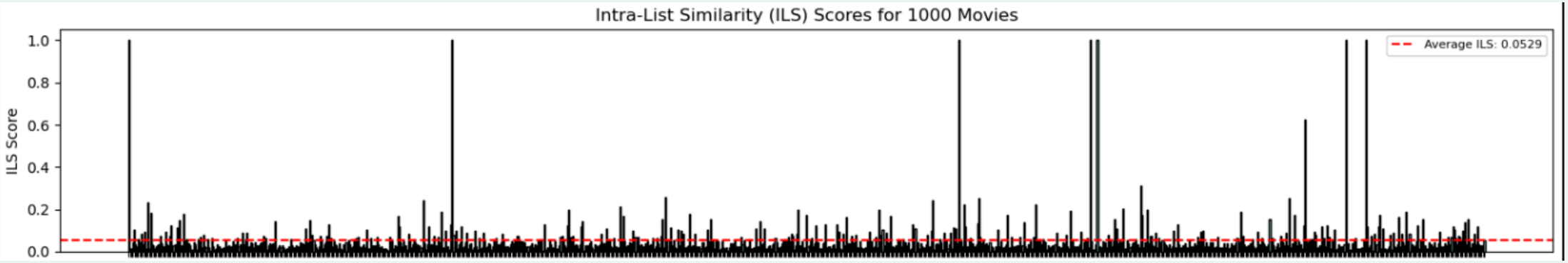
What is ILS?

- Mengukur rata-rata pairwise cosine similarity antar item dalam daftar rekomendasi (top-10 movies).
- Mengevaluasi diversity: Skor ILS rendah (~0) = high diversity; skor tinggi (~1) = low diversity.
- Rentang ideal: 0.3–0.6 untuk keseimbangan relevance dan diversity.

Why Use ILS

- Aligned: Menggunakan TF-IDF dan cosine similarity yang ada, tanpa data baru.
- No Ground Truth: Evaluasi intrinsik, ideal tanpa user feedback.
- Insightful: Deteksi masalah diversity (misalnya, score ILS=1.0 menunjukkan duplikasi).
- Efficient: Komputasi yang ringan Ringan

EVALUATION METRICS INTRA-LIST SIMILARITY (ILS)



Hasil Evaluasi (1000 sampled movies)

- **Rentang ILS:** 0.0094–1.0 sementara itu terdapat beberapa outlier seperti pada movie ID 71063, “The Flying Dutchman” dimana menghasilkan score = 1.
- **Rata-rata ILS:** 0.0529 (menunjukkan high diversity).

Analisis

- **Strength:** High diversity di sebagian besar daftar, menghindari saran berulang.
- **Weakness:**
  - Rata-rata ILS rendah (0.0588) berisiko kurang relevance
  - outlier ILS=1.0 mengindikasikan masalah data seperti duplikasi.
- **Hal yang dapat dilakukan:**
  - Investigasi outlier
  - Hitung Coverage: Proporsi katalog yang direkomendasikan