Movie Finder

Group: 08

Patrick Eckel: Design Document, Frontend, Data Transfer
Marcus Gugacs: Design Document, Frontend, Recommender, CLI, Evaluation, Questionnaire, Report, Presentation
Martin Tobias Klug: Design Document, Subtitle Fetching, Summarization Pipeline, Report
Lukas Leitner: Design Document, Data Preprocessing, Report, Presentation

https://www.github.com/llmpaq/movie-finder

01.2025

Introduction

Motivation

- Lot’s of video content online
- Many different streaming providers
- Central place for content curation
- Value users time
- Personalized recommendations

Introduction

Research Question

- Can we build a central system which provides recommendations of various streaming services to effectively reduce the users effort of finding content?

Data

Movie Dataset

- Original Columns:
  - id, title, genres, original language, overview, popularity, production companies, release date, budget, revenue, runtime, status, tagline, vote average, vote count, credits, keywords, poster path, backdrop path, recommendations
- Reduced to:
  - id, title, genres, original language, overview, popularity, vote average, credits, keywords, poster path, release year
- Added column: rich features

Data

Subtitles

- Subtitles provided by API (Key required)
- Download / processed on demand
- Raw subtitles
- Preprocessed by removing:
  - timestamps, ids, html tags/entities, parentheses, brackets, braces, musical notes, metadata, speakers, empty lines

Methods

Sequence Transformer

- Model: sentence-transformers/all-mpnet-base-v2
- Semantic text embedding
  - Used to similarity between user query and movie features
- MPNet allows for dense vector representation
  - optimal for semantic sentence similarity
- Processing chunks of max 512 Tokens

Methods

Emotion Classifier

- Model: j-hartmann/emotion-english-distilroberta-base
- Based on DistilRoBERTa
- Classify emotions in english text
    - Supports: Anger, disgust, fear, joy, neutral, sadness, surprise
- Mapping user mood preference to support emotions
- Measure alignment

Methods

TF-IDF Vectorization

- Generate vector representation of text (scikit)
- Enable similarity matching
- Required text preprocessing:
  - Lemmatization (WordNetLemmatizer)
  - Stop word removal (StopWords)
  - Special character cleaning
  - Case normalization
  - Minimum token length

Methods

Movie Introduction Summarization

- Model: facebook/bart-large-cnn
  - Summarization pipeline
  - Based on BART
- Purpose:
  - Creates introductory summary from pre-processed movie subtitles
  - Uses first chunk (1024 tokens) of subtitles for better performance and to avoid spoilers

Methods

<u>Keyword Extraction</u>

- Model: KeyBERT
  - Based on BERT embeddings (unsupervised)
  - Semantic similarity for ranking
- Purpose:
  - Extracts key themes from cleaned movie subtitles
  - Returns top 3 keywords / key themes

System

Overview

1. Initial Filtering

    1.1. Language

    1.2. Era (release year timespan)

    1.3. Genre

    1.4. Minimum popularity

    1.5. Minimum vote average

System

Overview

2. Feature Processing

   2.1. Load cached semantic embeddings or compute them

   2.2. Generate TF-IDF Matrix

   2.3. Encode query text (combined user input)

   2.4. Calculate emotion alignment score

System

Overview

3. Semantic Computation

    3.1. Cosine similarity of semantic

    3.2. TF-IDF cosine similarity

    3.3. Emotional <-> Mood alignment score

    3.4. Weighted score computation

Results

Questionnaire: Analysis

- Internal team evaluation
- Standardized questionnaire
- Repeated evaluation:
  - 3 Runs
  - Over multiple days
- Data: averaged over all runs (per evaluator)

Results

Questionnaire: Results

	User Interface (school grade)	Recommendation Quality (school grade)	Response Time (seconds)	Confidence (percent)	Overall Rating (school grade)
Patrick Eckel	1	1	26,33	74,83	1,33
Marcus Gugacs	1	1,07	28,67	72,42	1,67
Martin Klug	1,07	1	27,67	67,67	2
Lukas Leitner	1	1,07	15,67	45,25	2
Average	**1,02**	**1,04**	**24,59**	**65,04**	**1,75**

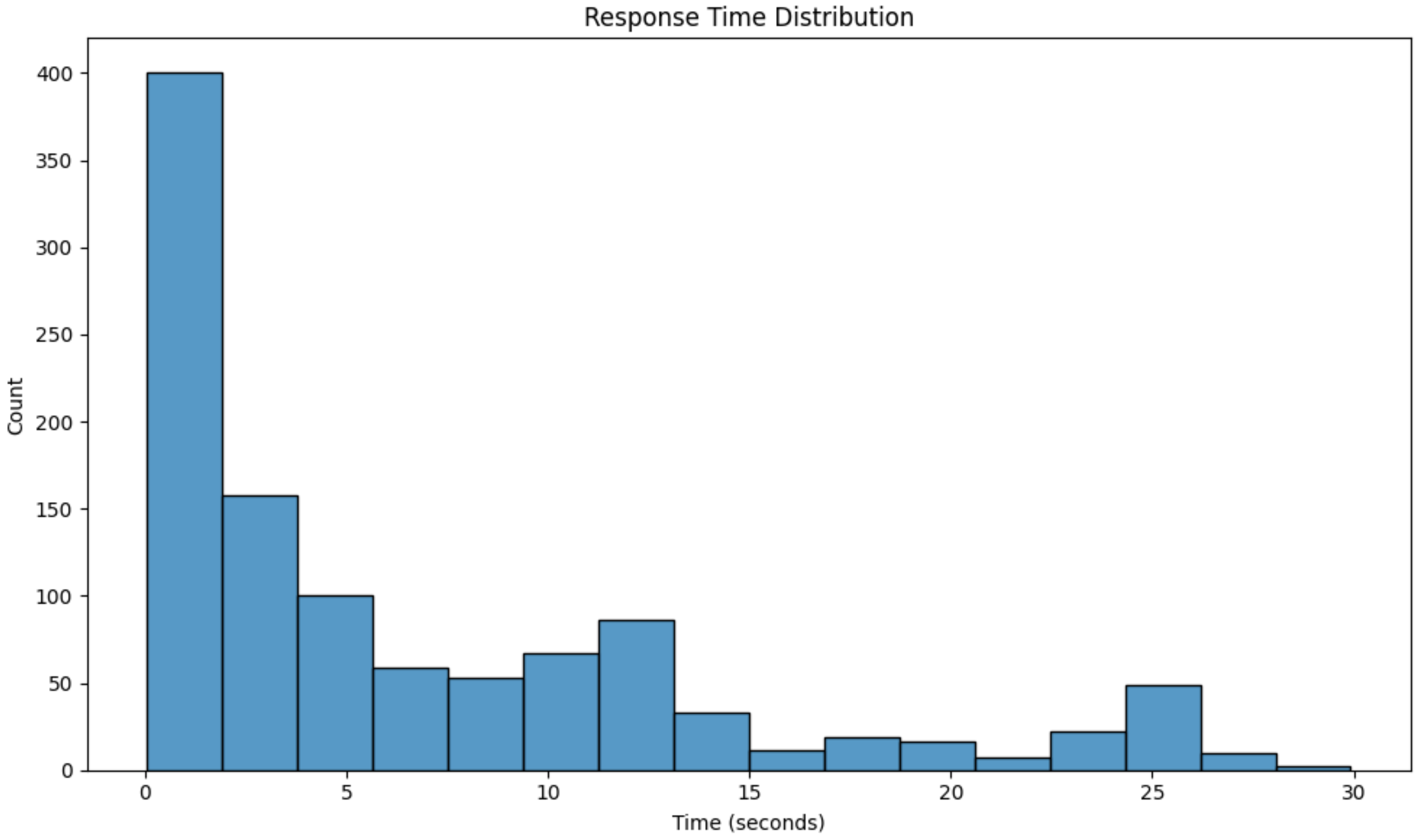
Results

Evaluation: Analysis

- Automated python script: „objective“ evaluation
- Test cases for each possible input
- Random test cases sampling (600 test cases)
- Running each test 2 times (1200 tests)
- Using 40% of dataset (approx. 583k movies)
- Storing measured data
- Calculating statistical measurements

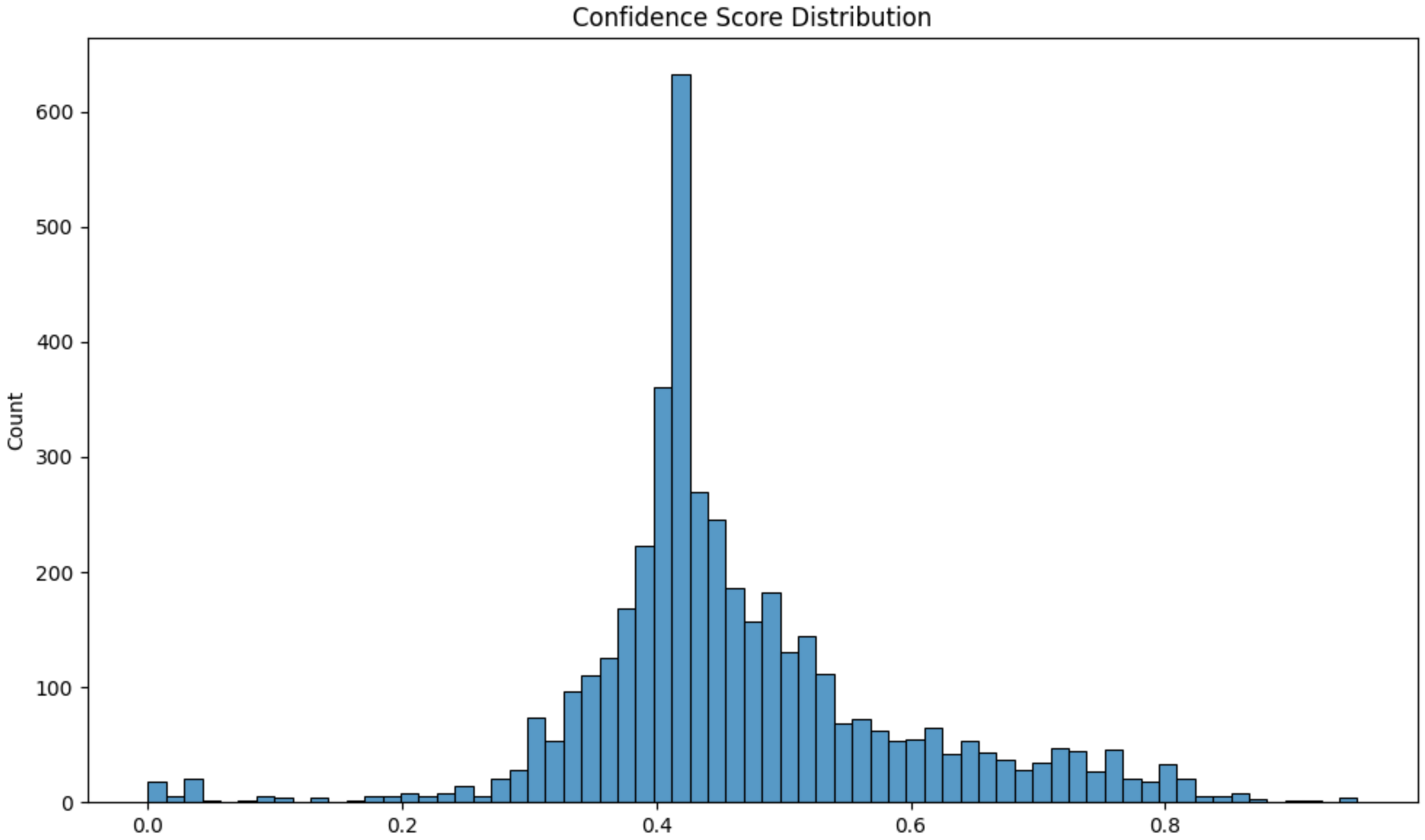
Results

Evaluation: Results



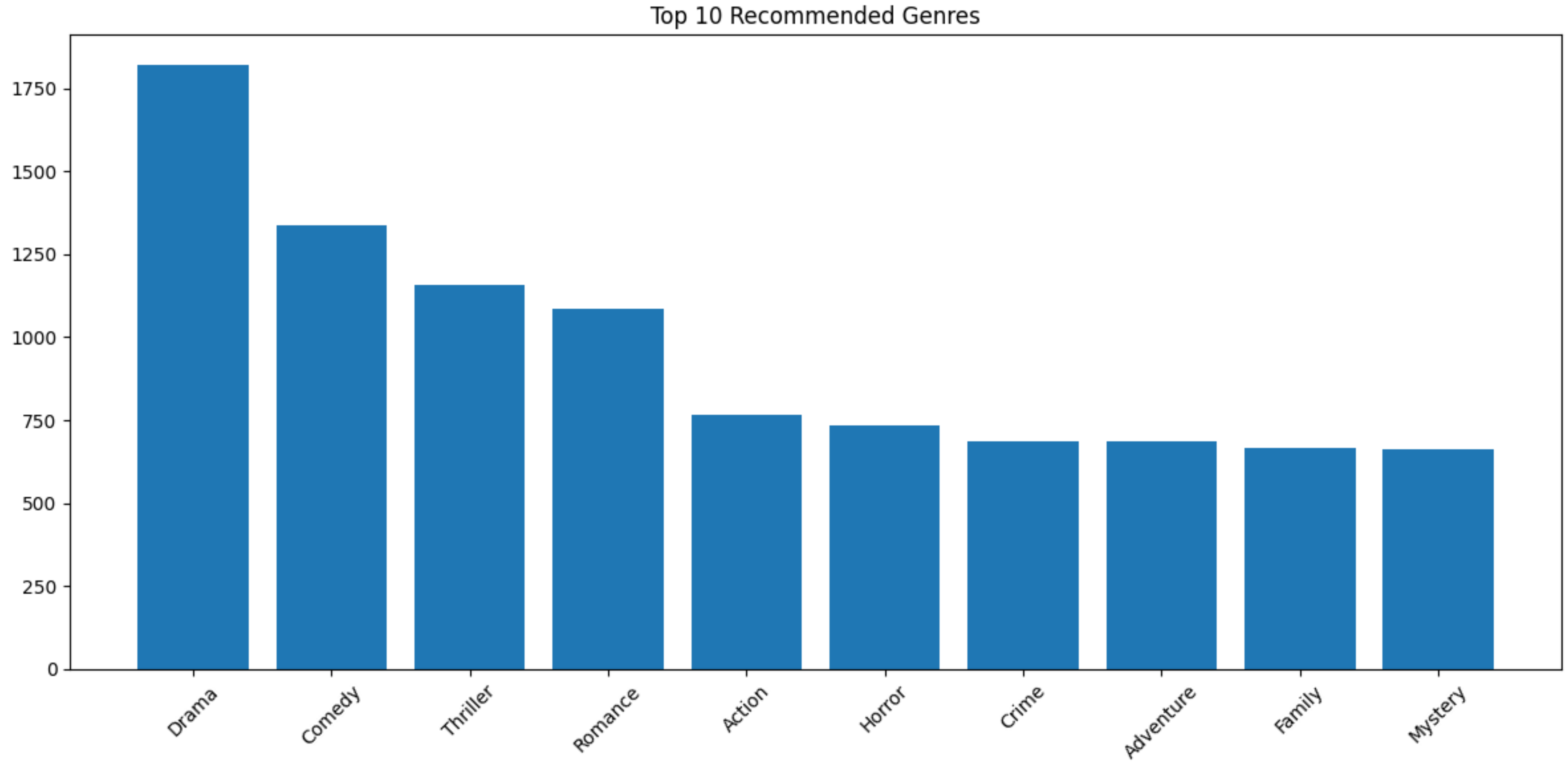
Results

Evaluation: Results



Results

Evaluation: Results



Results

Evaluation: Results

	Average	Standard Deviation	Min	Max	Median
Response Time (in seconds)	6.69	7.32	0.06	29.92	3.60
Genre Diversity	3.8	0.51	1	4	4
Confidence (in %)	46.55	12.94	0.00	95.13	43.36
Rating	6.84	0.81	3.00	9.75	9.60

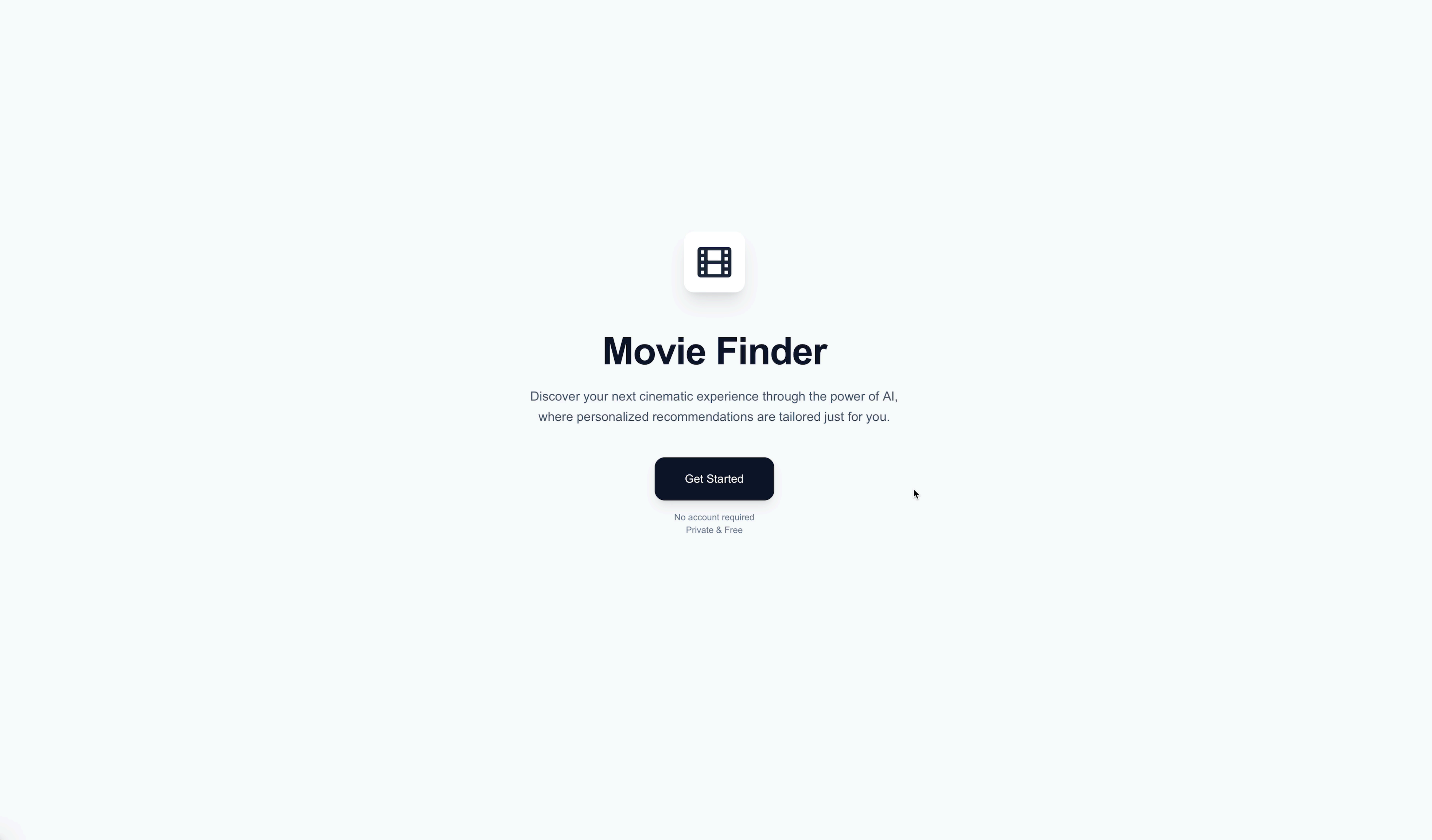
Results

Evaluation: Results

- High Precision: good recommendations for user
- Low Recall:
  - System may miss out on other movies
  - Could be due to sampling subset
- Low F1-Score: drag-down due to low recall

	Average
Average Precision (in %)	99.08
Average Recall (in %)	10.10
Average F1-Score (in %)	14.38

Showcase



Movie Finder

Discover your next cinematic experience through the power of AI,
where personalized recommendations are tailored just for you.

No account required
Private & Free

Conclusion

- Usable and efficient recommendations
- Tweaking and fine-tuning
- Minor tweaks lead to significant changes

Questions?