Building a Better Movie Recommender

With natural language processing of user generated queries

Caleb Stephenson, Data Scientist

Presentation Outline

01. The Problem

Why is this useful?

02. The Data

Where does the data come from?

03. The Models

How does this work?

04. Conclusions

Where are we now?

Types of Recommenders

The current state of recommender systems for movies

Collaborative

Recommend movies based on watch list / history and other users.

Drawbacks: Inflexible, may not reflect momentary desires well.

Feature-Based

User manually selects features such as genre, runtime, actors, or a list of predefined keywords.

Drawbacks: Limited scope, time-consuming

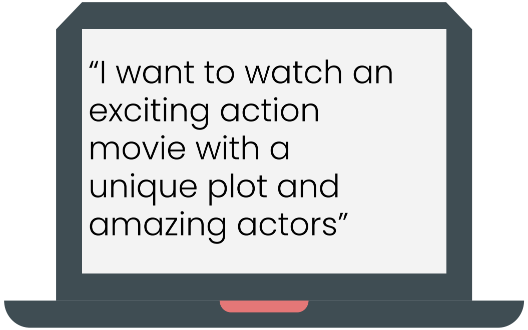
Problem Statement

I will use internet discussion posts on the topic of movie suggestions to build a features-based recommender that works with simple text input.

Problem Statement

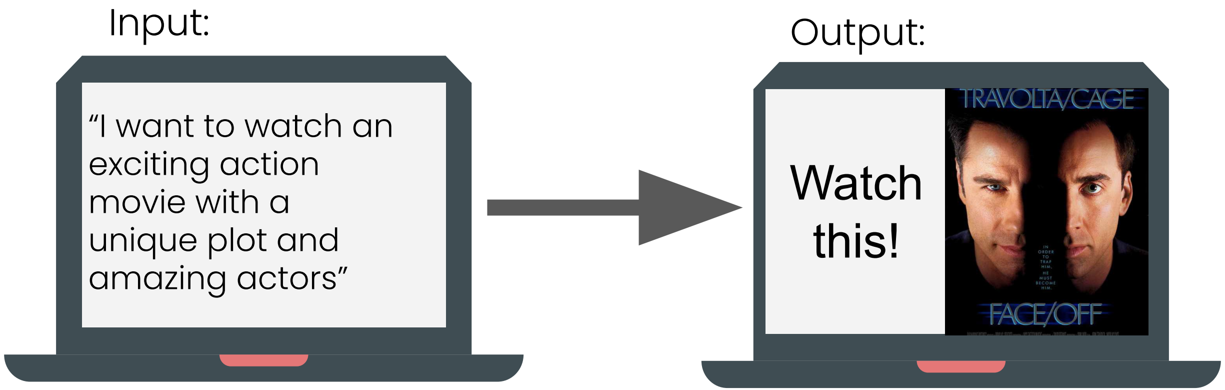
I will use internet discussion posts on the topic of movie suggestions to build a features-based recommender that works with simple text input.

Input:



Problem Statement

I will use internet discussion posts on the topic of movie suggestions to build a features-based recommender that works with simple text input.



Conceptual output. Not actual results.

Image source:
https://www.imdb.com/title/tt0119094/?ref_=tt_sims_tti

Data Sources

Discussion Posts:

www.reddit.com/r/moviesuggestions

- Users ask for movies or make suggestions
- Data scraped with PushShift API

Movies Data:

www.imdb.com/interfaces/

- Databases with basic information on hundreds of thousands of titles

Database

What this data looks like on the internet:

Request:	*Suggestion*:
"Cheesy old school kung fu movies…" Posted by: *iam4r33*	"Snake in Eagle's Shadow, Fearless Hyena, Master with Cracked Fingers, old school Jackie Chan could keep you occupied forever!" Posted by: *StinkyBrittches*

Database

Request:	*Suggestion*:
"Cheesy old school kung fu movies…" Posted by: *iam4r33*	"Snake in Eagle's Shadow, Fearless Hyena, Master with Cracked Fingers, ~~old school Jackie Chan could keep you occupied forever!~~" Posted by: *StinkyBrittches*

Find suggestions with spaCy and a filtered list of titles from IMDB

Request Text:	*Suggestion List*:
"Cheesy old school kung fu movies…"	Snake in Eagle's Shadow, Fearless Hyena, Master with Cracked Fingers

Features

Labels

Data Exploration

Requests

8,686 requests with suggestions

Suggestions

188,964 unique suggestions

21.8 per request

Titles

8,887 different titles were suggested at least once

Models and Evaluation

The data was split into training (80%) and testing (20%) data sets.

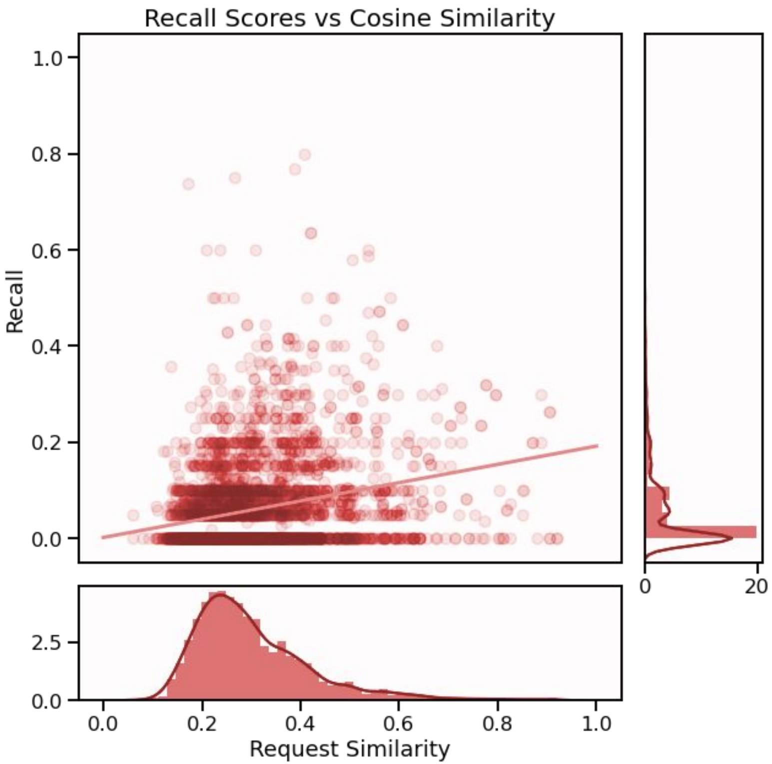
Model design was influenced by information retrieval systems.

- In these systems, a query is given, and documents matching the query are returned.
- This involves selecting a subset of data and returning ranked results.

Model performance was evaluated as the accuracy of a multi label classification when predicting from test data.

Baseline accuracy was measured by comparing the ten most recommended movies to the top ten suggestions for each request -- this was about 3.4%

What accuracy is achievable?

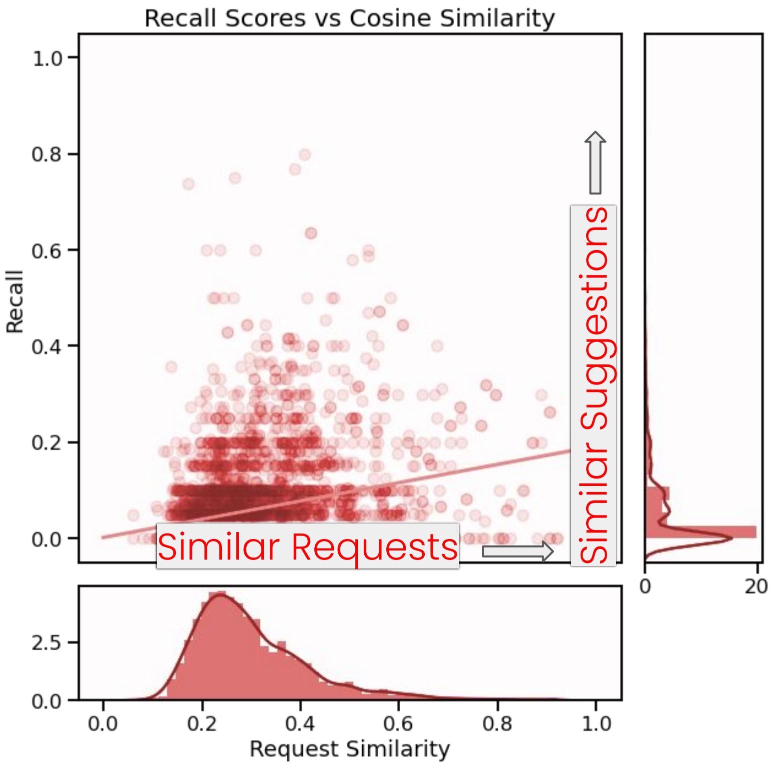


Humans are random and unpredictable.

This represents the accuracy and similarity scores when comparing each set of training document + labels with the most similar document in the training corpus.

The average "accuracy" is just under 5%, trending towards 20% for perfectly similar documents.

What accuracy is achievable?



Humans are random and unpredictable.

This represents the accuracy and similarity scores when comparing each set of training document + labels with the most similar document in the training corpus.

The average "accuracy" is just under 5%, trending towards 20% for perfectly similar documents.

TFIDF Vectorization and Cosine Similarity

Each request text is vectorized by TFIDF.

Titles are turned into documents by aggregating the vectors of the requests associated with that title.

A subset of the data is selected by choosing rows that have features in common with the vectorized queries.

This subset is ranked by cosine similarity to the query.

Average accuracy: 3.5%

Results of our Query:

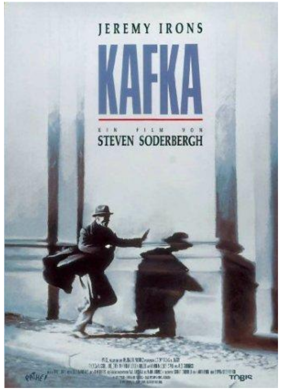
“I want to watch an exciting action movie with a unique plot and amazing actors”

Results of our Query:

"I want to watch an exciting action movie with a unique plot and amazing actors"

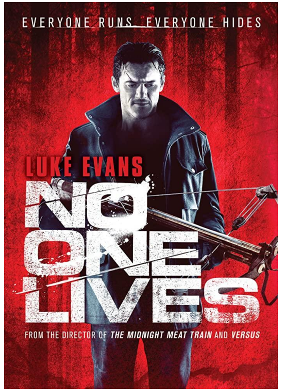
Watch These:

Kafka
Thriller/Mystery



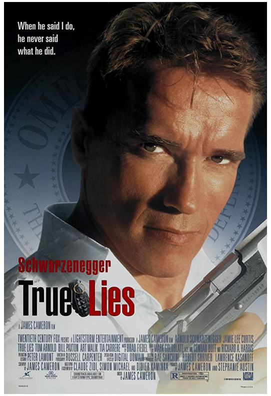
https://www.imdb.com/title/tt0102181/

No One Lives
Thriller/Horror



https://www.imdb.com/title/tt1763264/

True Lies
Action/Comedy



https://www.imdb.com/title/tt0111503/

Accuracy scores of different approaches:

TFIDF document similarity	Feedforward Neural Network	spaCy document similarity
3.5%	1.0%	0.0%

Accuracy scores of different approaches:

TFIDF document similarity	Feedforward Neural Network	spaCy document similarity
3.5%	1.0%	0.0%

With a model trained on a wide range of topics, document similarity did not perform well. This data requires models trained on the corpus. TFIDF works okay but can we do better?

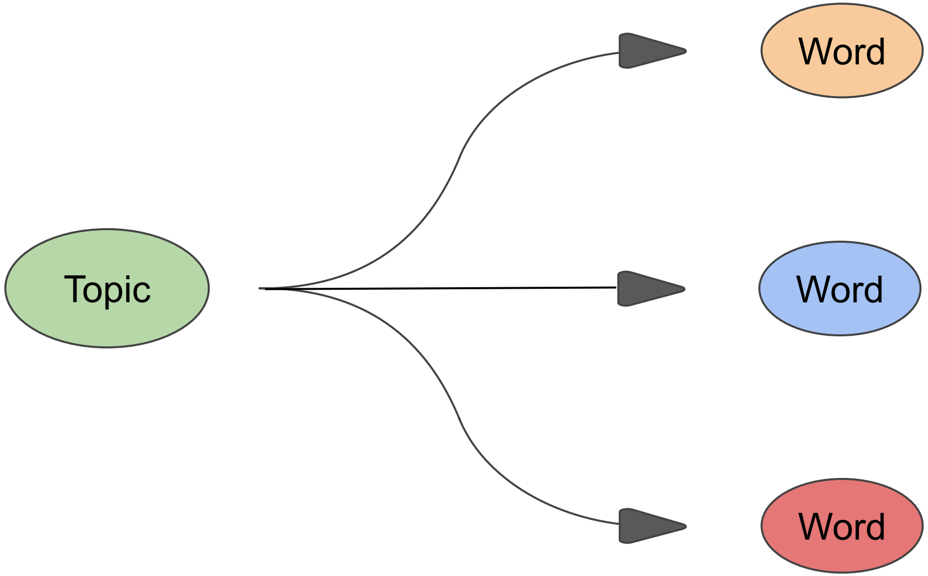
Latent Dirichlet Allocation

Unsupervised learning model with much in common with principal component analysis (PCA) and clustering.

LDA model trained with gensim with training data, preliminary tests done without any filtering of dataset.

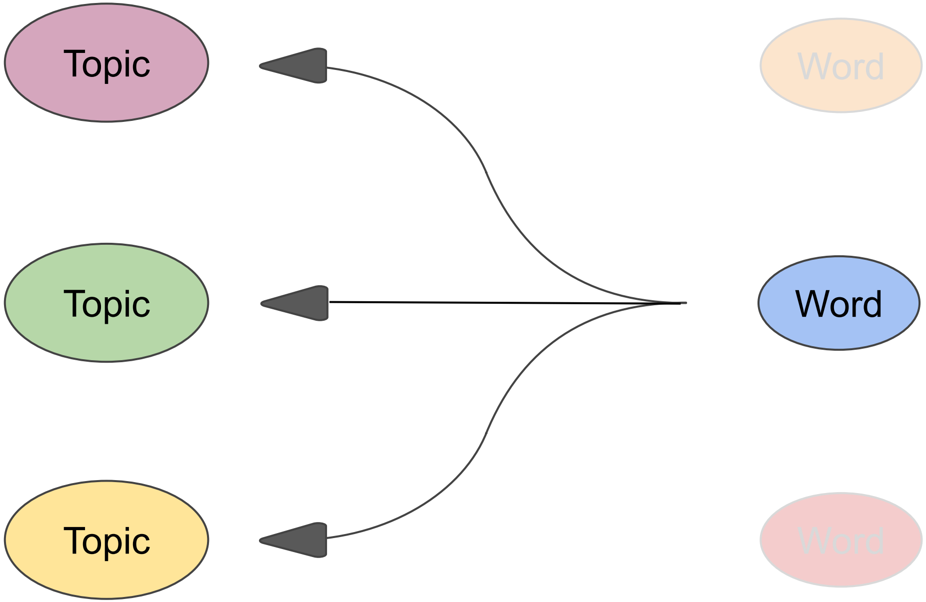
Latent Dirichlet Allocation

Unsupervised learning model with much in common with principal component analysis (PCA) and clustering.

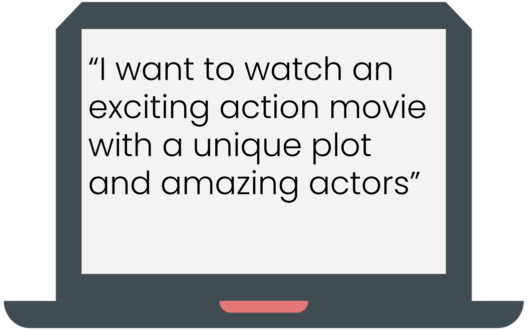


Latent Dirichlet Allocation

Unsupervised learning model with much in common with principal component analysis (PCA) and clustering.



Results of our Query:



Results of our Query:

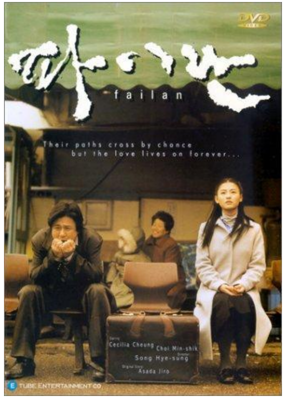
"I want to watch an exciting action movie with a unique plot and amazing actors"

Watch These:

House of 9
Horror/Mystery

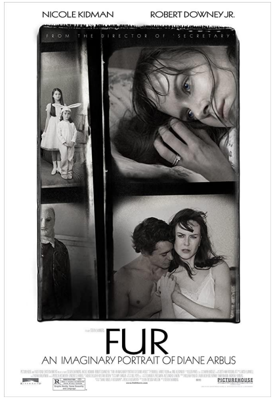
https://www.imdb.com/title/tt0395585/

Failan
Drama/Romance



https://www.imdb.com/title/tt0289181/

Fur
Drama/Romance



https://www.imdb.com/title/tt0422295/

Accuracy vs Baseline

At just under 1%, LDA model currently performs worse than baseline accuracy, but what exactly is the baseline predicting?

Accuracy vs Baseline

At just under 1%, LDA model currently performs worse than baseline accuracy, but what exactly is the baseline predicting?

TOP TEN MOVIES:

1. ***Up***
2. ***Star Trek Into Darkness***
3. ***Love***
4. ***Them!***
5. ***Life***
6. ***Her***
7. ***2012***
8. ***Toy Story 3***
9. ***After***
10. ***In Time***

Accuracy vs Baseline

At just under 1%, LDA model currently performs worse than baseline accuracy, but what exactly is the baseline predicting?

TOP TEN MOVIES:

1. ***Up***
2. ***Star Trek Into Darkness***
3. ***Love***
4. ***Them!***
5. ***Life***
6. ***Her***
7. ***2012***
8. ***Toy Story 3***
9. ***After***
10. ***In Time***

- Common words dominate this list
- These are mostly false positives
- The baseline is artificially high
- However, this does not result in these movies being recommended by the system.

Conclusions

- The data is unsurprisingly problematic. Much more cleaning and munging is needed.
- However, results are promising and sometimes provide interesting and relevant recommendations.
- There may not be enough data for neural networks.
- LDA, if performance improves, could be used for transfer learning with other models.

Thank you!

Questions?

Resources:

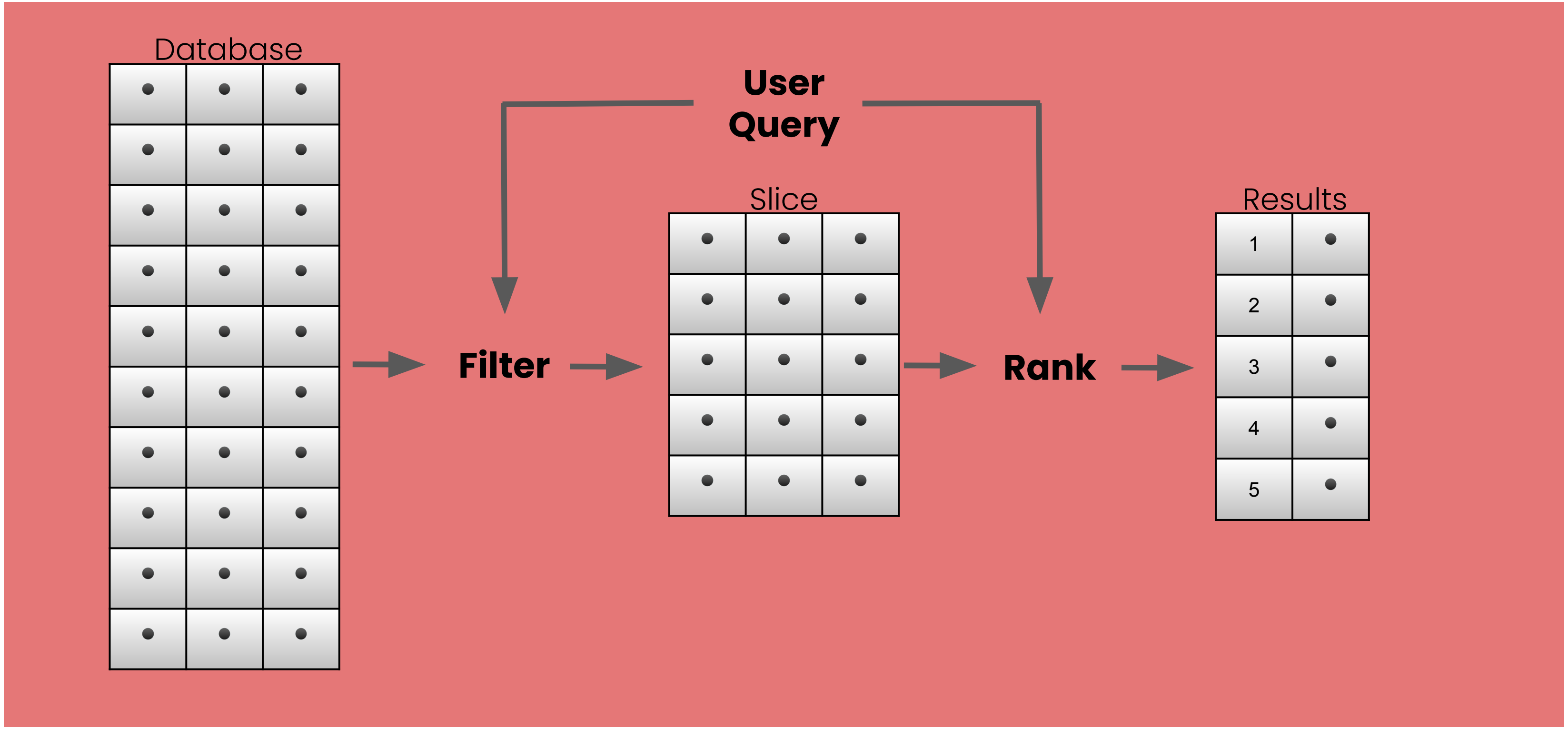
Slides:
www.slidesgo.com
www.freepik.com

Data:
www.reddit.com
www.imdb.com

Movie Posters:
www.imdb.com

Document Retrieval

A.k.a. Information Retrieval



Evaluation

Accuracy for Multi-Label Classification

True Labels:	✓	✗	✗	✓	✓	✗
Predictions:	✓	✓	✓	✗	✗	✗
Accurate?	Y	-	-	N	N	-

= Human-suggested Titles

= System-suggested Titles

Accuracy = 0.33

For development of the system, this metric is *informative*, but not *definitive*

LDA Topic Examples

Each topic is a collection of words with varying weights.

0.034*"sci_fi" + 0.015*"plot" + 0.013*"space" + 0.013*"ex" + 0.012*"men" + 0.012*"human" + 0.011*"protagonist" + 0.011*"examples" + 0.011*"interstellar" + 0.010*"death"

'0.019*"https_tt" + 0.012*"actors" + 0.011*"soundtrack" + 0.010*"series" + 0.009*"plot" + 0.008*"visually" + 0.008*"three" + 0.007*"make" + 0.007*"great" + 0.007*"rich"

'0.046*"war" + 0.027*"american" + 0.016*"country" + 0.013*"history" + 0.013*"us" + 0.012*"directors" + 0.011*"detective" + 0.010*"man" + 0.010*"small" + 0.009*"women"'),

May benefit from more topics (more granularity) and/or better stopwords