Movie Recommendation System

Team members:
Yunlu Liaozheng
Lingfeng Zhou
Cheng Jin

Goal

To answer the question “What movie should I watch tonight?”, our goal is to build a robust movie recommendation system, the users do search on movie A, then the system suggests movie B.

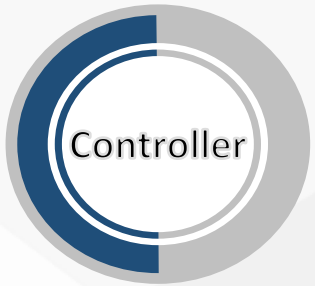
Use Cases

The hypothetical/typical user work flow.

Actor	Action
Administrator	Update the datacenter periodically , prepossessing raw data

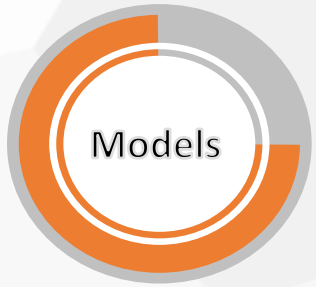
Actor	Action	Reaction
Users who want to know if this movie is worth watching, how's its reviews, rating of this movie and recommendation of related movies based on some conditions.	User inputs a movie's name in a web page.	**System** gives out a rating of the movie and a list of recommended movies. Finally, the user will give an evaluation of this recommendation as the feedback which will help to modulate the strategy of User Modeling.

Methodology



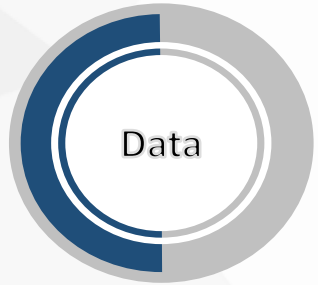
Controller is the distribution center of the system, it receives multiple users queries then distributes queries to Models nodes. And itself is mainly a stateless webserver.

Principal: Cheng Jin



The model level consists of different built rules and statistic models to be discussed. For example, Random recommendation, Item-based collaborative filtering, User-based collaborative filtering, singular vector decomposition based collaborative filtering, hybrid filtering etc. Built on akka actor cluster.

Principal: Yunlu Liaozheng



The data center is responsible for data storage, data processing and data management based on spark.

Principal: Lingfeng Zhou

Data sources

IMDB’s dataset

Subsets of IMDb data are available for access to customers for personal and non-commercial use. You can hold local copies of this data, and it is subject to our terms and conditions. Please refer to the Non-Commercial Licensing and copyright/license and verify compliance.
Size: 1,000,000
https://datasets.imdbws.com/

Milestones

1. week1, setup the dataset, build the framework of the system, confirm which models/rules will be used.
2. week2, complete each one's work.
3. week3, combine team members' work.
4. week4, test the system, improve UI

Scala part

1. Code in Scala: distributor controller, data management center, models

2. Code not in Scala: webpage will be written in js, c#, html, ts,
Python maybe

3. Code repository: Private repository on Github.

Acceptance criteria

95% requests are processed under 3 seconds, and at least 10 requests per second.

Thanks