MovieMatch Architecture

Amod Ghangurde, Luke Doolittle, and Cameron Bell

Final Project for W205 (section 5)

August 11, 2017

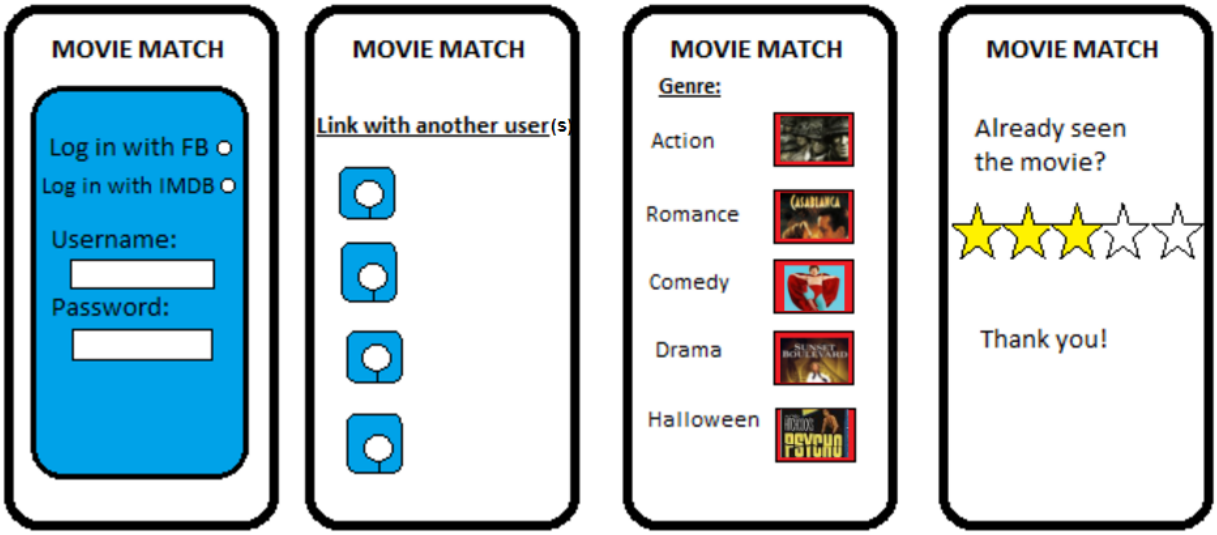
Table of Contents

Application Purpose

MovieMatch is a movie recommender system hosted on the web. Based on a user's existing ratings, the application suggests movies that similar users rated highly. This is done using a collaborative filtering algorithm in Spark's MLlib.

Ideally, the application would support the following features:

- Real-time updating when a user rates a movie within the application
- Suggestions to linked accounts, providing a movie that both (or more) users would enjoy
- Customized suggestions, within several genres (including seasonal/holiday categories)



Operating Instructions (README)

moviematch

Create EC2 Instance

Launch the vanilla UCB AMI `ami-a4c7edb2` from AWS (shortcut to east-1).

- The instance must have at least 8 GB of RAM
- The instance must have at least 20 GB of storage

Download and run the setup script then reboot

```
wget https://raw.githubusercontent.com/lukedoolittle/moviematch/master/setup.sh
chmod +x setup.sh
./setup.sh
sudo reboot
```

Run Website

Clone this repository and move into the directory

```
git clone https://github.com/lukedoolittle/moviematch.git
cd moviematch
```

Load the initial SMALL static set of data

```
./data/small_data_load.sh
```

Build and run webserver

```
./launch.sh
```

Get the Public DNS (IPv4) address from your EC2 instance and navigate to that uri in a browser

Load Additional Data

Load the initial static set of data

```
./data/initial_data_load.sh
```

Load the secondary set of data

```
./data/additional_data_load.sh
```

Dependencies

- Spark
- MongoDB
- Nodejs

Description of Application Architecture

1. The application starts by loading the MovieLens dataset (see https://movielens.org/) into Spark tables.
2. The Netflix data is then loaded into separate tables.
3. The tables are merged based on movie titles. This results in about 70 million merged ratings.
4. The user rates movies on the website hosted by nodejs.
5. Those ratings are then used to generate suggestions with Spark's MLlib.
6. The suggestions are displayed to the user.