Table of Contents

1. Introduction 3

2. Dataset 3

3. System Architecture 4

3.1 Recommendation Processor 4

3.2 Movie Recommendation System 4

3.3 Image Scrapper 5

4. Methodology 5

4.1 Singular Value Decomposition 5

4.2 Similar Movies Calculation 6

5. Technology 6

6. Screen shots 7

7. References 9

1. Introduction

In this busy world, people expect their purchase or browsing experience personalized to their needs. So many companies are constantly collecting the user behavior in order to provide them with a better experience. We could correlate this with our shopping experience. When we go to a shop to purchase things, we go to section by section and then rack by rack to find the items that we are interested in. But instead imagine if all our interesting things are in the first section and in the first rack we visit. It would make our shopping experience a delight and could save us a lot of time.

This strategy is being used by Amazon and other ecommerce websites to recommend products to the users based on their previous purchase history. This applies to the entertainment world as well. We have huge amount of movies/music to choose from. So, for a user its difficult to decide on which would suit their taste before watching the movie or listening to a music track. But if someone suggests us that we are likely to be interested in a set of movies/music tracks, we can easily choose one from such a small set. This is what companies like Netflix and Spotify does. This is a win-win strategy. The users are satisfied with their experience as well as the companies can make more revenue.

The goal of my project is to build a personalized movie recommendation system. The movie recommendation system is a web application connected to a database of 671 users with their previous movie watch history and their ratings for the movies. Based on their previous watch history, the system recommends new movies to the users to watch.

1. Dataset

The dataset (ml-latest-small) is downloaded from the site <https://grouplens.org/datasets/movielens/>. The dataset describes 5-star rating and free-text tagging activity from [MovieLens](http://movielens.org/), a movie recommendation service. It contains 100004 ratings and 1296 tag applications across 9125 movies. These data were created by 671 users between January 09, 1995 and October 16, 2016. This dataset was generated on October 17, 2016.

Users were selected at random for inclusion. All selected users had rated at least 20 movies. No demographic information is included. Each user is represented by an id. The data are contained in the files links.csv, movies.csv, ratings.csv and tags.csv.

1. System Architecture

IMBD Website

Dataset

My SQL Database

**Movie Recommender**

**Users**

**Movie Recommendation system**

**Figure: Movie Recommendation System Architecture**

The above diagram shows the architecture of the movie recommendation system. The different components in the system are

* 1. Recommendation Processor

This implements the Singular Value Decomposition algorithm and calculates top 10 movies that can be recommended for a user. The results are stored in MySQL database.

3.2 Movie Recommendation System

This is a web application for the user to login, check the system recommended movies, search for a movie in the database. This web application runs in flask. When a user logs in, it retrieves the watch history, recommended movies from the database. When user searches for a movie the system runs the Similar movies calculation dynamically for the best hit of the searched movie. Both the search result and similar movies are displayed for the user.

3.3 Image Scrapper

This is a utility program to get poster images for the movies in the data set by calling IMDB service. These poster images would be displayed in the web pages.

1. Methodology

The system uses Singular Value Decomposition algorithm to identify the recommendation for the users. When user performs a movie search, the system calculates similar movies in addition to the searched movie. This is obtained by running Similar Movies Calculation logic. These are explained below.

4.1 Singular Value Decomposition

Personalized recommendation engines help millions of people narrow the universe of potential films to fit their unique tastes. These services depend on a machine-learning strategy called singular value decomposition, which breaks down movies into long lists of attributes and matches these elements to a viewer's preferences.

First a recommendation engine takes a huge data set of films and viewer ratings. The recommendation engine then uses the collective ratings to break down individual movies into long lists of attributes. The resulting attributes may correspond to easily identifiable qualities such as “comedy” or “cult classic,” but they may not—the computer only knows them as X, Y and Z. Now recommendation is a simple task of decoding an individual’s tastes and matching those tastes to the relevant movies.

The essence of SVD is that it decomposes a matrix of any shape into a product of 3 matrices with nice mathematical properties: A=USV**T** , where

A: nm: number of records as rows and number of dimensions/features as columns.  
U: nn: orthogonal matrix containing eigenvectors of AAT.  
S: nm: ordered singular values in the diagonal. Square root of eigenvalues associated with AAT or ATA(it’s the same).  
V: mm: orthogonal matrix containing eigenvectors of ATA.

The result of the decomposition leaves us with an ordered matrix of singular values which encompass the variance associated with every direction. We assume that larger variances mean less redundancy and less correlation and encode more structure about the data. This allows us to use a representative subset of user rating directions or principal components to recommend movies.

4.2 Similar Movies Calculation

There is an option for users to search for a movie. User can type a string and the web application would fetch the movies matching the searched string from the movie dataset along with its movie posters and links. There is also recommendation of the movies which are similar to the searched movie based on its genres. For this functionality, I used Euclidean distance to calculate the movies which are closest to the searched movie. The mathematical formula for the Euclidean distance is given by, considering 2 points, A and B, with their associated coordinates, the distance is defined as:

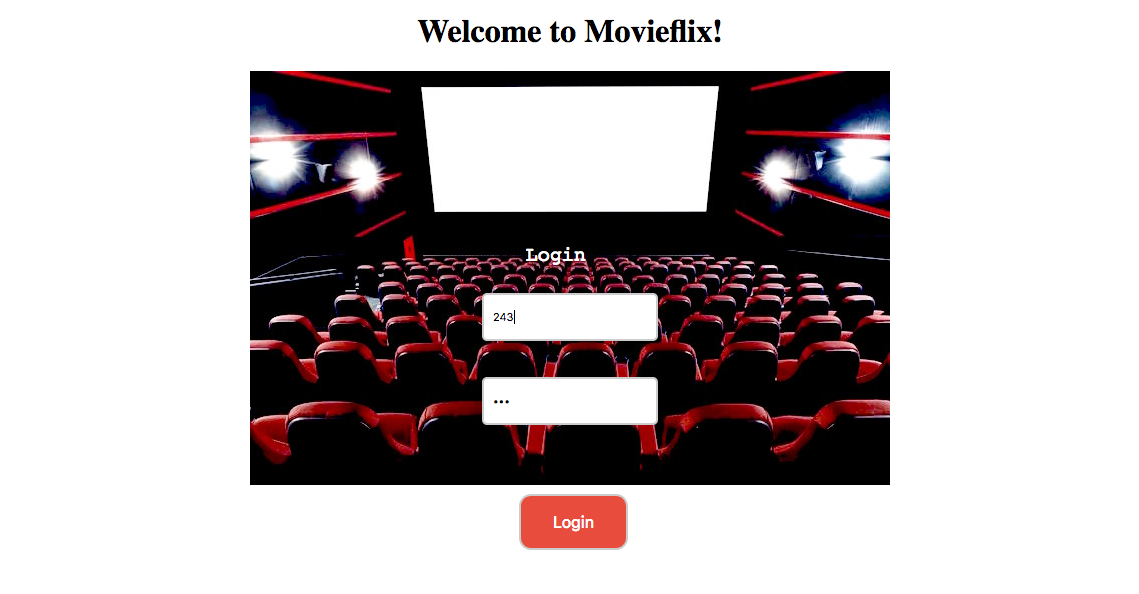


1. Technology

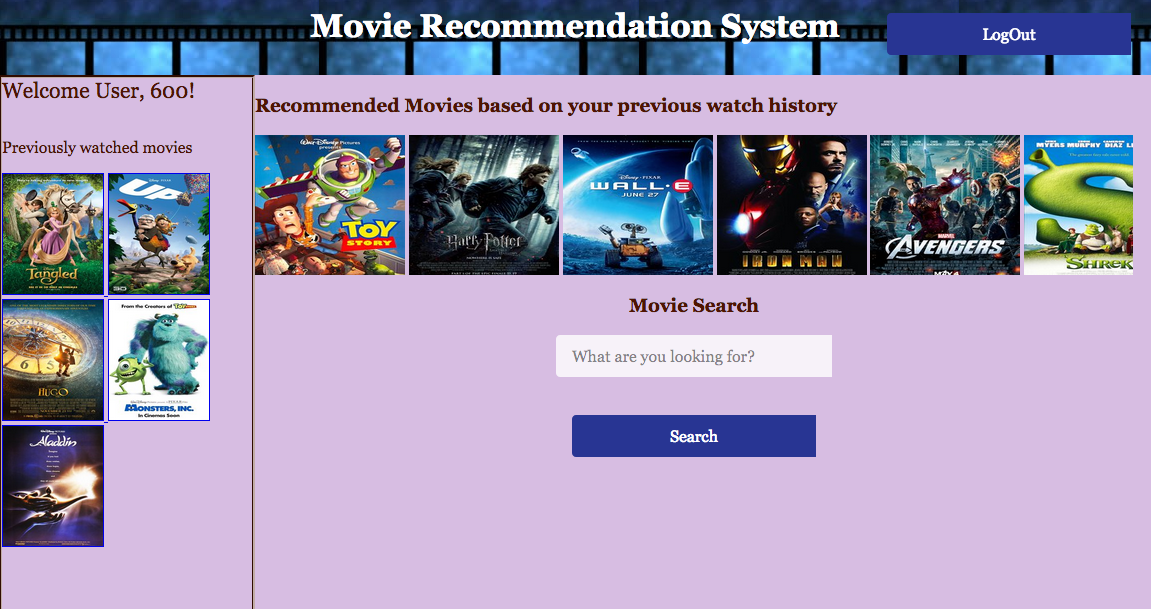
* Python 3 for the recommender system backend
* Flask, HTML and CSS for front end
* MySQL database

1. Screen shots

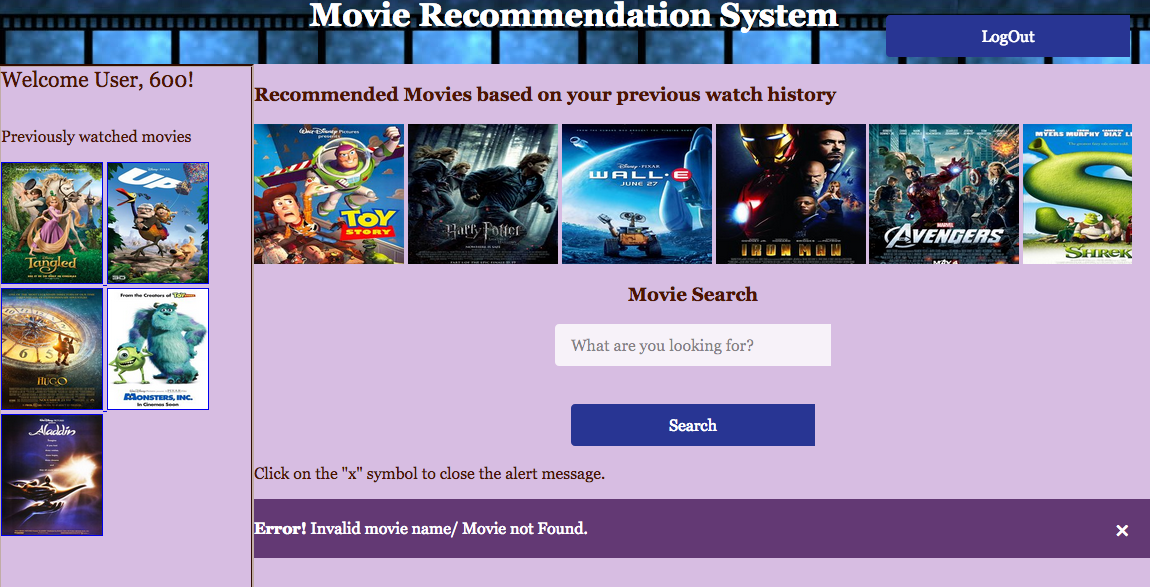
Login page



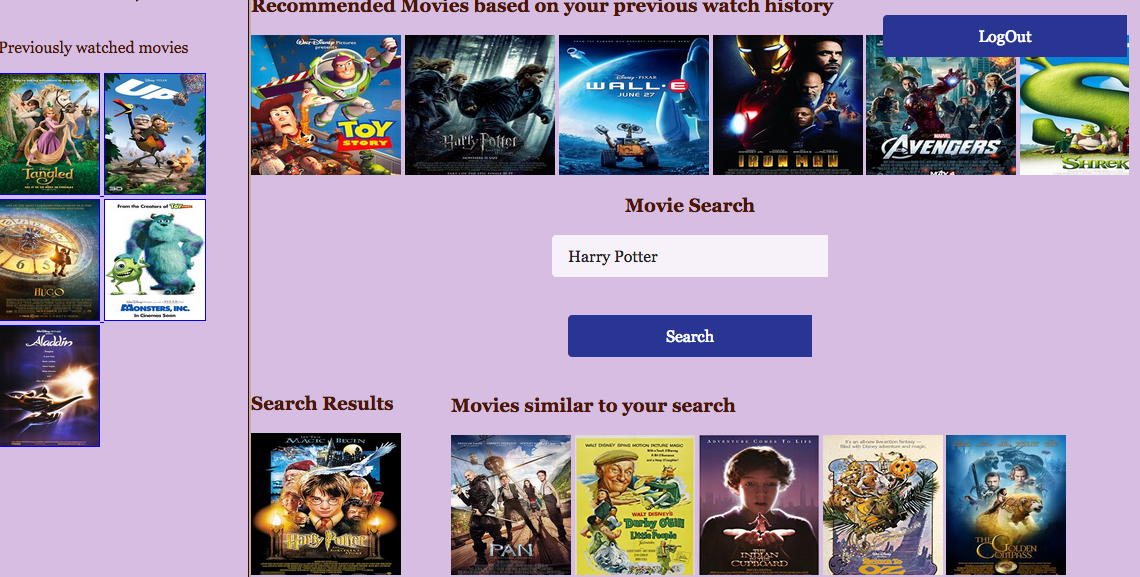
User Home Page (with movie recommendation)



Validation



Movie Search and Similar Movies Based on Search Result



1. References
2. https://www.slideshare.net/stanleywanguni/overview-of-recommender-system
3. https://www.slideshare.net/kerveros99/essir-2013-recsysfinal-25957057
4. https://www.tutorialspoint.com/flask/flask\_sending\_form\_data\_to\_template.htm
5. https://pdfs.semanticscholar.org/9e58/558a635a32f5a8bb77d3d79eac055b1d035a.pdf
6. https://www.ics.uci.edu/~welling/teaching/CS77Bwinter12/presentations/course\_Ricci/13-Item-to-Item-Matrix-CF.pdf
7. https://www.analyticsvidhya.com/blog/2015/10/recommendation-engines/
8. https://www.ics.uci.edu/~welling/teaching/CS77Bwinter12/presentations/course\_Ricci/14-ContentBased-Hybrid.pdf
9. http://mathworld.wolfram.com/SingularValueDecomposition.html
10. https://en.wikipedia.org/wiki/Collaborative\_filtering
11. https://arxiv.org/pdf/1404.1100.pdf
12. http://alyssaq.github.io/2015/singular-value-decomposition-visualisation/
13. https://www.dataquest.io/blog/k-nearest-neighbors-in-python/