Building an interactive recommendation system: MovieLens

La’Tisa Ward and Lesley Frew

CS 624 Fall 2024

Problem Statement

Users often face challenges in finding relevant content of interest amidst vast options.

Our project aims to develop a recommendation engine using the MovieLens dataset to analyze user viewing data and ratings, providing personalized movie suggestions to enhance user engagement and experience.

Dataset Description

The MovieLens dataset, available from TensorFlow Datasets, provides user-movie interaction data, including ratings, timestamps, and metadata like user demographics and movie genres. It is widely used for building and testing recommender systems.

The dataset is available in varying sizes, from 100K to millions of ratings, making it versatile for small-scale experiments or large-scale machine learning models. It's a foundational dataset for exploring collaborative filtering and personalization algorithms.

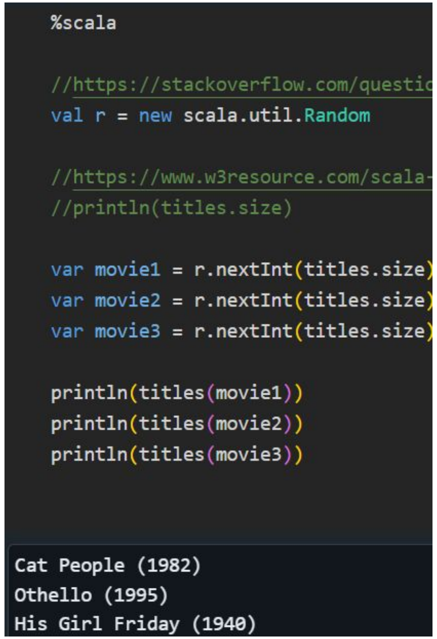
Reference- tensorflow.org/datasets/catalog/movielens

Proposed Solution

- Choose 3 random movies from the MovieLens dataset
- Use 3 widgets to accept user input ratings for those 3 movies
- Add the 3 new ratings to the ratings RDD
- Train the model using Alternating Least Squares
- Output the top 10 recommendations

Recommendation Engine Pipeline Code Walkthrough

First, choose 3 random movies from the dataset using the Random library.

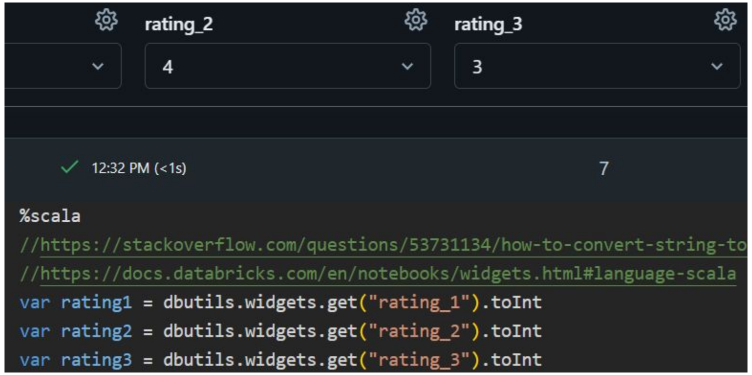


```
%scala

//https://stackoverflow.com/questic
val r = new scala.util.Random

//https://www.w3resource.com/scala-
//println(titles.size)

var movie1 = r.nextInt(titles.size)
var movie2 = r.nextInt(titles.size)
var movie3 = r.nextInt(titles.size)

println(titles(movie1))
println(titles(movie2))
println(titles(movie3))
```

```
Cat People (1982)
Othello (1995)
His Girl Friday (1940)
```

Next, use the widgets to accept user input.

Convert them from String to int.



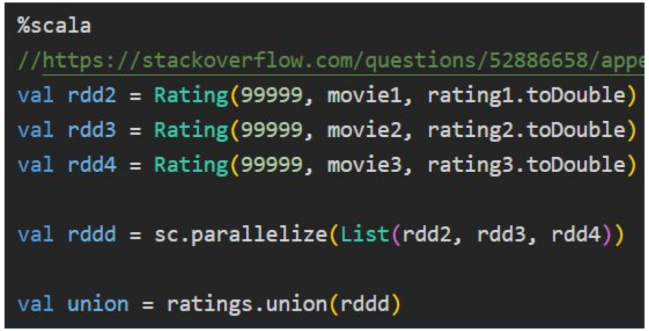
rating_2: 4

rating_3: 3

```
12:32 PM (<1s)                7
```

```
%scala
//https://stackoverflow.com/questions/53731134/how-to-convert-string-to
//https://docs.databricks.com/en/notebooks/widgets.html#language-scala
var rating1 = dbutils.widgets.get("rating_1").toInt
var rating2 = dbutils.widgets.get("rating_2").toInt
var rating3 = dbutils.widgets.get("rating_3").toInt
```

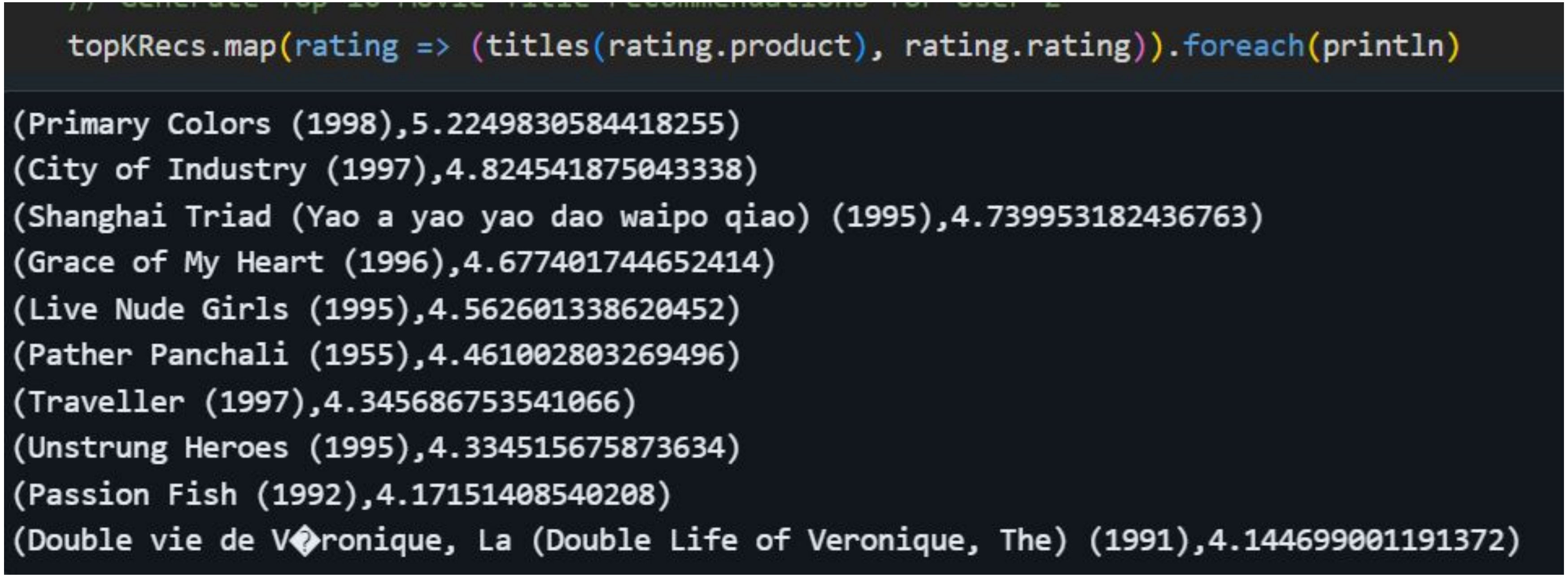
Finally, add the three new ratings for this user (“99999”) to the RDD.



```
%scala
//https://stackoverflow.com/questions/52886658/appe
val rdd2 = Rating(99999, movie1, rating1.toDouble)
val rdd3 = Rating(99999, movie2, rating2.toDouble)
val rdd4 = Rating(99999, movie3, rating3.toDouble)

val rddd = sc.parallelize(List(rdd2, rdd3, rdd4))

val union = ratings.union(rddd)
```

Recommendation Engine Pipeline Code Walkthrough

Use the ALS code from Lab 11 with the new union RDD to find the recommendations for user “99999”.



```
topKRecs.map(rating => (titles(rating.product), rating.rating)).foreach(println)
```

```
(Primary Colors (1998),5.2249830584418255)
(City of Industry (1997),4.824541875043338)
(Shanghai Triad (Yao a yao yao dao waipo qiao) (1995),4.739953182436763)
(Grace of My Heart (1996),4.677401744652414)
(Live Nude Girls (1995),4.562601338620452)
(Pather Panchali (1955),4.461002803269496)
(Traveller (1997),4.345686753541066)
(Unstrung Heroes (1995),4.334515675873634)
(Passion Fish (1992),4.17151408540208)
(Double vie de V�ronique, La (Double Life of Veronique, The) (1991),4.14469901191372)
```

Conclusion and Future Work

Strengths and limitations of models:

- Strength: The model is interactive which is usable and engaging
- Limitations: Relies on fairness in original dataset which is not only not guaranteed but shown to be particularly poor for MovieLens

Future work or improvements:

- Allow the user to skip movies they don’t know
- Allow the user to rate a variable number of movies