Project Extra

Film recommendations using MLIB

|  |  |
| --- | --- |
| **Prepared For:** | AcadGild |
|  |  |
| **Document Approval:** | **AcadGild** |
|  |  |
|  |  |
|  |  |
|  |  |
| **Project Title:** | Project Extra |
|  |  |
| **Prepared By:** | Duncan Burgess |
|  |  |
|  | dburgess@duncb.com |
|  |  |
| **Primary Engineer:** | Duncan Burgess |
|  |  |
| **Document Reference:** | **Project Extra** |
|  |  |
| **Start Date:** | 27/10/2017 |
|  |  |
|  |  |

[](https://www.google.co.uk/imgres?imgurl=https://i.ytimg.com/vi/6TTT1yKn4-o/maxresdefault.jpg&imgrefurl=https://www.youtube.com/watch?v%3D6TTT1yKn4-o&docid=tG8k9ly9l9la_M&tbnid=l3KYCh86GfUItM:&vet=10ahUKEwj4l8vmo5bXAhWDDMAKHS_lCFsQMwj9AShbMFs..i&w=1920&h=1080&bih=1284&biw=2560&q=movie%20rating&ved=0ahUKEwj4l8vmo5bXAhWDDMAKHS_lCFsQMwj9AShbMFs&iact=mrc&uact=8)

# 

# Contents

[Contents 2](#_Toc497061338)

[Change History 3](#_Toc497061339)

[1. Problem Statement 4](#_Toc497061340)

[2. Datasets 4](#_Toc497061341)

[3. Solution 6](#_Toc497061342)

[4. Results 7](#_Toc497061343)

# Change History

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Document Revision** | **Date** | **Authored By** | **Authorised By** | **Sections Affected** | **Reason for Change** |
| Rev 01 | 27/10/2017 | Duncan Burgess |  | All | Initial release. |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

# Problem Statement

Taking movie data and ratings find fil recommendations based on my selections using MLIB.

# Datasets

This show exerts from the 2 files u.data and u.item Film ID’s with ratings and the film titles.

0 516 5 881250949

0 140 5 881250949

0 201 1 881250949

196 242 3 881250949

186 302 3 891717742

22 377 1 878887116

244 51 2 880606923

166 346 1 886397596

298 474 4 884182806

115 265 2 881171488

253 465 5 891628467

305 451 3 886324817

6 86 3 883603013

62 257 2 879372434

286 1014 5 879781125

200 222 5 876042340

210 40 3 891035994

224 29 3 888104457

303 785 3 879485318

122 387 5 879270459

194 274 2 879539794

291 1042 4 874834944

234 1184 2 892079237

119 392 4 886176814

167 486 4 892738452

299 144 4 877881320

291 118 2 874833878

The 3 in red are my 2 favourite films and the 3rd my least favourite

**Local Hero**

**Homeward Bound: The Incredible Journey**

**Evil Dead II**

1|Toy Story (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Toy%20Story%20(1995)|0|0|0|1|1|1|0|0|0|0|0|0|0|0|0|0|0|0|0

2|GoldenEye (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?GoldenEye%20(1995)|0|1|1|0|0|0|0|0|0|0|0|0|0|0|0|0|1|0|0

3|Four Rooms (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Four%20Rooms%20(1995)|0|0|0|0|0|0|0|0|0|0|0|0|0|0|0|0|1|0|0

4|Get Shorty (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Get%20Shorty%20(1995)|0|1|0|0|0|1|0|0|1|0|0|0|0|0|0|0|0|0|0

5|Copycat (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Copycat%20(1995)|0|0|0|0|0|0|1|0|1|0|0|0|0|0|0|0|1|0|0

6|Shanghai Triad (Yao a yao yao dao waipo qiao) (1995)|01-Jan-1995||http://us.imdb.com/Title?Yao+a+yao+yao+dao+waipo+qiao+(1995)|0|0|0|0|0|0|0|0|1|0|0|0|0|0|0|0|0|0|0

7|Twelve Monkeys (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Twelve%20Monkeys%20(1995)|0|0|0|0|0|0|0|0|1|0|0|0|0|0|0|1|0|0|0

8|Babe (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Babe%20(1995)|0|0|0|0|1|1|0|0|1|0|0|0|0|0|0|0|0|0|0

9|Dead Man Walking (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Dead%20Man%20Walking%20(1995)|0|0|0|0|0|0|0|0|1|0|0|0|0|0|0|0|0|0|0

10|Richard III (1995)|22-Jan-1996||http://us.imdb.com/M/title-exact?Richard%20III%20(1995)|0|0|0|0|0|0|0|0|1|0|0|0|0|0|0|0|0|1|0

11|Seven (Se7en) (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Se7en%20(1995)|0|0|0|0|0|0|1|0|0|0|0|0|0|0|0|0|1|0|0

12|Usual Suspects, The (1995)|14-Aug-1995||http://us.imdb.com/M/title-exact?Usual%20Suspects,%20The%20(1995)|0|0|0|0|0|0|1|0|0|0|0|0|0|0|0|0|1|0|0

13|Mighty Aphrodite (1995)|30-Oct-1995||http://us.imdb.com/M/title-exact?Mighty%20Aphrodite%20(1995)|0|0|0|0|0|1|0|0|0|0|0|0|0|0|0|0|0|0|0

14|Postino, Il (1994)|01-Jan-1994||http://us.imdb.com/M/title-exact?Postino,%20Il%20(1994)|0|0|0|0|0|0|0|0|1|0|0|0|0|0|1|0|0|0|0

# Solution

**Code written**

**package** com.duncb.spark

**import** org.apache.spark.\_

**import** org.apache.spark.SparkContext.\_

**import** org.apache.log4j.\_

**import** scala.io.Source

**import** java.nio.charset.CodingErrorAction

**import** scala.io.Codec

**import** org.apache.spark.mllib.recommendation.\_

**object** MRALS {

/\*\* Load up a Map of movie IDs to movie names. \*/

**def** loadMovieNames() : *Map*[Int, *String*] = {

// Handle character encoding issues:

**implicit** **val** codec = Codec("UTF-8")

codec.onMalformedInput(CodingErrorAction.REPLACE)

codec.onUnmappableCharacter(CodingErrorAction.REPLACE)

// Create a Map of Ints to Strings, and populate it from u.item.

**var** movieNames:*Map*[Int, *String*] = Map()

**val** lines = Source.fromFile ("N:/Datasets/ml100k/u.item").getLines()

**for** (line <- lines) {

**var** fields = line.split('|')

**if** (fields.length > 1) {

movieNames += (fields(0).toInt -> fields(1))

}

}

**return** movieNames

}

/\*\* Our main function where the action happens \*/

**def** main(args: Array[*String*]) {

// Set the log level to only print errors

Logger.getLogger("org").setLevel(Level.ERROR)

// Create a SparkContext using every core of the local machine

**val** sc = **new** SparkContext("local[\*]", "MRALS")

println("Loading movie names...")

**val** nameDict = loadMovieNames()

**val** data = sc.textFile("file:///N:/Datasets/ml100k/u2.data")

**val** ratings = data.map( x => x.split('\t') ).map( x => **Rating**(x(0).toInt, x(1).toInt, x(2).toDouble) ).cache()

// Build the recommendation model using Alternating Least Squares

println("\nTraining recommendation model...")

**val** rank = 8

**val** numIterations = 20

**val** model = ALS.train(ratings, rank, numIterations)

**val** userID = args(0).toInt

println("\nRatings for user ID " + userID + ":")

**val** userRatings = ratings.filter(x => x.user == userID)

**val** myRatings = userRatings.collect()

**for** (rating <- myRatings) {

println(nameDict(rating.product.toInt) + ": " + rating.rating.toString)

}

println("\nTop 10 recommendations:")

**val** recommendations = model.recommendProducts(userID, 10)

**for** (recommendation <- recommendations) {

println( nameDict(recommendation.product.toInt) + " score " + recommendation.rating )

}

}

}

# Results

Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties

Loading movie names...

Training recommendation model...

Ratings for user ID 0:

Local Hero (1983): 5.0

Homeward Bound: The Incredible Journey (1993): 5.0

Evil Dead II (1987): 1.0

Top 10 recommendations:

Baby-Sitters Club, The (1995) score 7.249548022585525

A Chef in Love (1996) score 7.002818114762421

Waiting to Exhale (1995) score 6.92617068237527

Carpool (1996) score 6.914771511802729

Balto (1995) score 6.878934414450013

Old Lady Who Walked in the Sea, The (Vieille qui marchait dans la mer, La) (1991) score 6.822779340312573

Nina Takes a Lover (1994) score 6.80002059042393

Women, The (1939) score 6.703634740266624

Stalingrad (1993) score 6.6776256320168805

Sixth Man, The (1997) score 6.354280280575678