DATA-643 Assignment - 04

Mohamed Elmoudni
Shazia Khan
Senthil Dhanapal

Contents

Requirements:

Project 04

The goal of this assignment is to give you practice evaluating the performance of recommender systems. You only have to complete one of the two tasks.

1. Using one of the recommender systems you already delivered, analyze the performance of how it works and see if you can make significant improvements. You should consider the computational expense (how much data you have in memory, how long it takes to run) and how accurate it is (RMSE, MAE). If your system delivers Top-N recommendations, you may want to assess accuracy by surveying friends and classmates on their user experience.
2. Adapt one of your recommendation systems to work with Apache Spark and compare the performance with your previous iteration. Consider the efficiency of the system and the added complexity of using Apache.

Introduction

We used Alternating Least Square Formulation for Recommender Systems with and without Spark, using MovieLense Dataset. We first implemented ALS using recommenderlab function, Recommender(), with method = "ALS". Then we implemented ALS in Spark using ml_als_factorization(). To assess the performance of recommender systems, with and without Spark, we used both RMSE and time taken to run code to compare.

The root-mean-square error (RMSE) is a frequently used measure of the differences between values predicted by a model or an estimator and the values actually observed.

MovieLense Dataset

We will be using the MovieLense dataset that comes with the recommenderlab package.

Data Preparation

```
library(recommenderlab)
```

```
## Warning: package 'recommenderlab' was built under R version 3.3.2

## Loading required package: Matrix

## Loading required package: arules

## Warning: package 'arules' was built under R version 3.3.2

##
## Attaching package: 'arules'

## The following objects are masked from 'package:base':
##
##     abbreviate, write

## Loading required package: proxy

## Warning: package 'proxy' was built under R version 3.3.1

##
## Attaching package: 'proxy'

## The following object is masked from 'package:Matrix':
##
##     as.matrix

## The following objects are masked from 'package:stats':
##
##     as.dist, dist
```

```
## The following object is masked from 'package:base':
##
##     as.matrix

## Loading required package: registry

## Warning: package 'registry' was built under R version 3.3.2
```

```
library(reshape2)
```

```
## Warning: package 'reshape2' was built under R version 3.3.2
```

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 3.3.2

## Warning: failed to assign NativeSymbolInfo for lhs since lhs is already
## defined in the 'lazyeval' namespace

## Warning: failed to assign NativeSymbolInfo for rhs since rhs is already
## defined in the 'lazyeval' namespace
```

```
library(knitr)
```

```
## Warning: package 'knitr' was built under R version 3.3.1
```

```
library(recommenderlab)
library(recosystem)
```

```
## Warning: package 'recosystem' was built under R version 3.3.3
```

```
library(SlopeOne)
library(SVDApproximation)
```

```
## Warning: replacing previous import 'data.table::melt' by 'reshape2::melt'
## when loading 'SVDApproximation'

## Warning: replacing previous import 'data.table::dcast' by 'reshape2::dcast'
## when loading 'SVDApproximation'
```

```
library(data.table)
```

```
## Warning: package 'data.table' was built under R version 3.3.2

##
## Attaching package: 'data.table'

## The following objects are masked from 'package:reshape2':
##
##     dcast, melt
```

```
library(RColorBrewer)
library(ggplot2)
```

```
ratings_df <- read.csv("https://raw.githubusercontent.com/simonnyc/IS-643/master/train_v3.csv", head=T)
#ratings_df<- head(ratings_df, 2000)
head(ratings_df)
```

```
##       ID user movie rating
## 1 610739 3704  3784      3
## 2 324753 1924   802      3
## 3 808218 4837  1387      4
## 4 133808  867  1196      4
## 5 431858 2631  3072      5
## 6 895320 5410  2049      4
```

```
ratings_df <- ratings_df[,2:4]
colnames(ratings_df)[1:3] <- c("user","item","rating")
```

Data Exploration

```
#head(tr <- (as(MovieLense,'data.frame')))

tr <- ratings_df

kable(head(tr))
```

user	item	rating
3704	3784	3
1924	802	3
4837	1387	4
867	1196	4
2631	3072	5
5410	2049	4

```
kable(summary(tr), caption ="Data Summary")
```

Table 2: Data Summary

user	item	rating
Min. : 1	Min. : 1	Min. :1.000
1st Qu.:1509	1st Qu.:1030	1st Qu.:3.000
Median :3072	Median :1835	Median :4.000
Mean :3025	Mean :1865	Mean :3.582
3rd Qu.:4478	3rd Qu.:2770	3rd Qu.:4.000
Max. :6040	Max. :3952	Max. :5.000

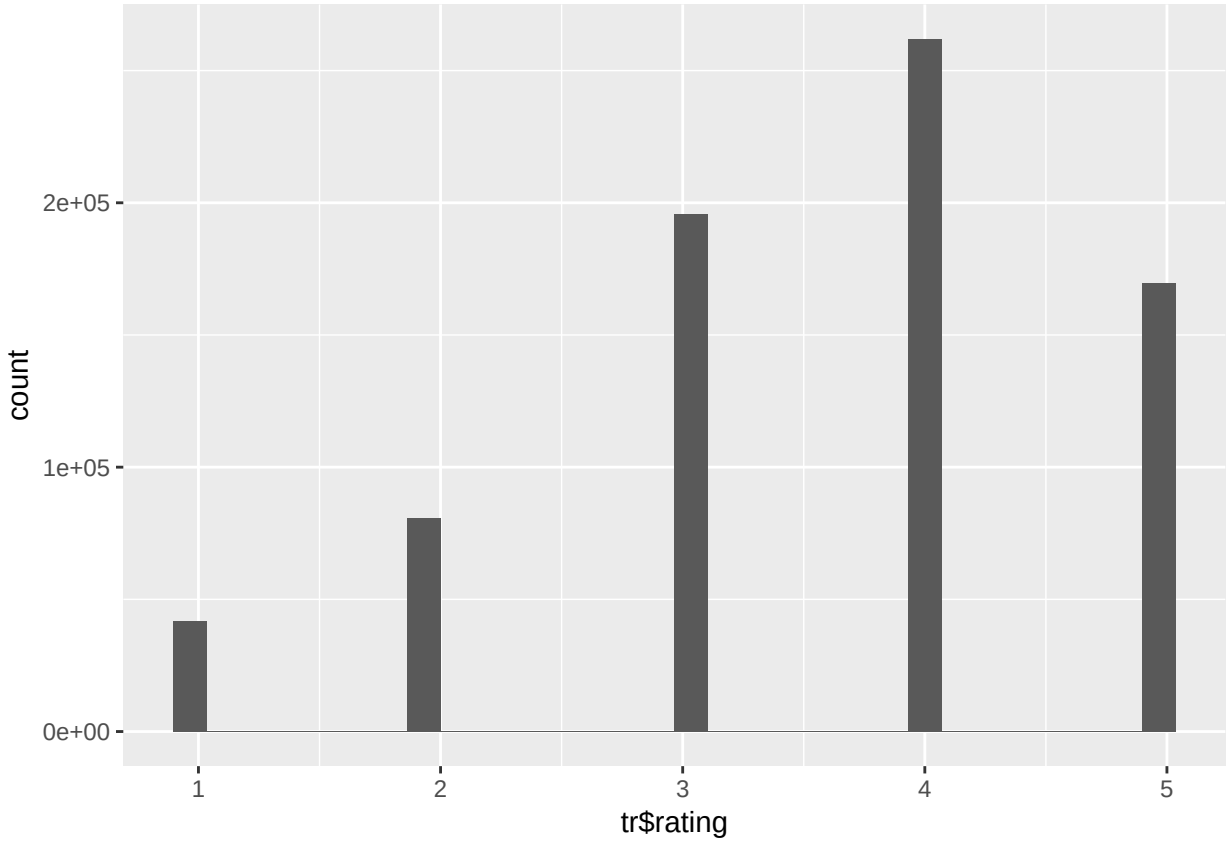
```
kable(as.data.frame(table(tr$rating)), caption = "Ratings Frequency Table")
```

Table 3: Ratings Frequency Table

Var1	Freq
1	41958
2	80862
3	195864
4	261916
5	169556

```
qplot(tr$rating)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Model 1 - Witout Spark ALS Recommender System

Description: Package: Recommenderlab, Function: recommender(), Method = "ALS"

Preparation for Model 1

```
library(recommenderlab)
library(reshape2)

ratings_wide <- dcast(ratings_df, user ~ item, value.var="rating")
ratingsM <- as.matrix(ratings_wide)
rownames(ratingsM) <- ratingsM[,1]
ratingsM <- ratingsM[,-1]

ratings_rRM <- as(ratingsM,'realRatingMatrix')
```

Creation of Model 1

```
start.time<- Sys.time()

r <- Recommender(data = ratings_rRM[1:1000,], method = "ALS",
                 parameter = list(normalize=NULL, lambda=0.1, n_factors=10,
                                  n_iterations=10, seed = NULL, verbose = FALSE))

recom_ratings <- predict(r, newdata = ratings_rRM[9:10,], type = "ratings")
#as(recom_ratings, "matrix")
recom_topNList <- predict(r, newdata = ratings_rRM[9:10,], type = "topNList", n = 5)
#as(recom_topNList, "list")

end.time<-  Sys.time()
```

RMSE for Model 1

```
scheme <- evaluationScheme( ratings_rRM, method="split", train=0.9, given=-5, goodRating=4)

#calcPredictionAccuracy(x, data, ...)
```

```
#e <- evaluationScheme(ratings_rRM, method="split", train=0.8, given=-5)
#5 ratings of 20% of users are excluded for testing
e<- scheme
model <- Recommender(getData(e, "train"), "ALS")
prediction <- predict(model, getData(e, "known"), type="ratings")

rmse_NoSpark <- calcPredictionAccuracy(prediction, getData(e, "unknown"))[1]
rmse_NoSpark
```

```
##      RMSE
## 0.9148424
```

Time Taken for Model 1

```
time.taken1 <- end.time - start.time
time.taken1
```

```
## Time difference of 11.68614 mins
```

Model 2 - With Spark ALS Recommender System

Description: Package: Sparklyr, Function: ml_als_factorization()

Preparation for Model 2

```
#install.packages('sparklyr')
#install.packages('dplyr')
#spark_install(version = "1.6.2")

library(sparklyr)
library(dplyr)
```

```
## ------------------------------------------------------------------------------
```

```
## data.table + dplyr code now lives in dtplyr.
## Please library(dtplyr)!
```

```
## ------------------------------------------------------------------------------
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:data.table':
##
##     between, first, last
```

```
## The following objects are masked from 'package:arules':
##
##     intersect, recode, setdiff, setequal, union
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(randomForest)
```

```
## randomForest 4.6-12

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:dplyr':
##
##     combine

## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
library(magrittr)
library(methods)
```

Creating Model 2

```
start.time<- Sys.time()

# Sys.setenv(SPARK_HOME='C:/spark-1.6.2-bin-hadoop2.6')
#
# Sys.unsetenv('SPARK_HOME')


# LOCAL CONNECTION:
#spark_install()
#spark_disconnect_all()

sc <- spark_connect(master = "local")

#dim(ratings_df)
#head(ratings_df)


movie_rating <- copy_to(sc, ratings_df, overwrite = T)
head(movie_rating)
```

```
## Source:   query [6 x 3]
## Database: spark connection master=local[2] app=sparklyr local=TRUE
##
##    user  item rating
##   <int> <int>  <int>
## 1  3704  3784      3
## 2  1924   802      3
```

```
## 3  4837  1387      4
## 4   867  1196      4
## 5  2631  3072      5
## 6  5410  2049      4
```

```
model <- ml_als_factorization(movie_rating, rating.column = "rating", user.column = "user",
                              item.column = "item", rank = 10L, regularization.parameter = 0.1,
                              iter.max = 10L, ml.options = ml_options())

summary(model)
```

```
##                   Length Class      Mode
## item.factors      11     data.frame list
## user.factors      11     data.frame list
## data               2     spark_jobj environment
## ml.options         6     ml_options list
## model.parameters   2     -none-     list
## .call              9     -none-     call
## .model             2     spark_jobj environment
```

```
predictions <- model$.model %>%
  invoke("transform", spark_dataframe(movie_rating)) %>%
  collect()


spark_disconnect(sc)

end.time<-  Sys.time()
```

RMSE for Model 2

```
sqrt(mean(with(predictions, prediction-rating)^2))
```

```
## [1] 0.8167405
```

Time Taken for Model 2

```
time.taken2 <- end.time - start.time
time.taken2
```

```
## Time difference of 1.560773 mins
```

Conclusion

From the results above, we clearly see that the performance of model1 using non-spark on the MovieLens dataset using ALS is worse than model2 using ALS with Spark. In other words, the running time of the ALS

without SPArk is 11.6861351 while the running time of RMSE with SPark is 1.5607726. it is three times faster than the one without Spark.

We also tested the RMSE in both cases (spark and non Spark). We noticed that the RMSE in Spark setting is lower than the one in non-Spark's.