Master Thesis
Recommender Systems Comparison

Vasileios Symeonidis

27-05-2017

Contents

Part I
Master Thesis

1 Intro

This is the intro suction for this master thesis. Why we need recommendation systems? Retailers can propose the right product to the right target group. User get advertisements the may be interested in.[2]

History, what has been tried so far?

2 Collaborative filtering

What is collaborative filtering. [2]

2.1 Content based

In content based recommender systems we try to recommend based on features we know. There are two types of content based recommender systems. On the one hand we have the user based recommendation. This recommendation is done by trying to match users profiles, in order to find which item the user i might like. But in real world we don't have the needed information to make the recommendation to the user. On the other hand we have the item-product based recommendation, in this case we are trying to find user that might like the given product. This is match easier due to the fact that you know more about a product than a user, and you can classify them easily.

In this case we have a matrix R that contains the rates given by users to items. This matrix most of the times will be low in density, this is because each user does not rate each product. The second matrix we come across is the M. This matrix contains all the movies with the their genres. Each characteristic is binary. For example, the movie with id i is both action and comedy and none of the other genres.

$$w = R^{-1}M^T \tag{1}$$

In order add an normalization factor to the above equation, we need to get it to the form below.

$$w = (\lambda I + R^T R)^{-1} R^T M \tag{2}$$

2.2 Latent Factors

In latent factors recommender systems we follow a similar approach but, in case of ALS(Alternating least squares), we are trying to find metrics that may lead us to the correct recommendation. Those metrics are not distinct, and may change in a number of iterations. Those metrics are inducted from the R matrix as we define it above. This makes this approach more tolerant to missing values, or wrong quality measures. Thus this metric as will be presented bellow is more efficient on prediction and time.

ALS explanation. ALS algorithm is based on the latent factors theory. This means that it is not going to use the attributes given by the dataset for the movies or the users. The algorithm is going to train it self based on the rating set only.

3 Our Experiment

3.1 Infrastructure

3.1.1 Apache Spark

Apache spark is the new trend on distributed computation and map-reduce. But first things first, what is map-reduce.

What is map reduce

Spark's predecessor, hadoop map reduce, was for a long time at its peak. Hadoop map reduce, is a distributed map-reduce system, this means that it has a mechanism to distribute work on nodes and a common interface for handling data. In hadoop's case this was able to happen due to Apache hadoop yarn and the HDFS (hadoop distributed file system). When a job was scheduled, data were loaded by the hdfs to a worker, then the worker was putting the result back to the hdfs.

Important note: cite apache hadoop yarn https://hadoop.apache.org/docs/current/hadoop-yarn/hadoop-yarn-site/YARN.html

Important note: add architecture diagrams for both hadoop and spark −¿ note the common parts of HDFS and YARN

But innovation knocked the door and resilient distributed datasets entered the room. In spark world, data are loaded to hdfs as before. Then spark loads them in an RDD, this means that data are now accessible on each machine's memory. Any transformation done to a RDD results a RDD, and so forth. After all the transformations are done, spark can transform the results to a file in hdfs.

How spark differentiates from its predecessors, hadoop yarn
Spark lightweight in memory data transformation Resilient Distributed Datasets (RDDs)
mllibs

add spark jira note

Important note: mention als distributed broadcasting implementation.

broadcasting rdd
//cite the mastering apache spark book [1]
a Spark cluster to be created on AWS EC2 storage.
New trends on spark https://github.com/apache-spark-on-k8s/spark cite this repository too.

3.2 Dataset

What is the dataset about. This dataset contains users, movies and the rating user made about the movies. This dataset is splited to multiple subsets of 80000 training sets and respective 20000 reviews. [3]

3.3 Metrics

3.3.1 Mean Absolute Error

As metrics are commonly used the MSE, RMSE and MAE. Due to the fact that the author prefers the last one, MAE was used in this experiment.

$$MAE = \frac{\sum_{i=1}^{n} |y_i - x_i|}{n} = \frac{\sum_{i=1}^{n} \sqrt{(y_i - x_i)^2}}{n} \tag{3}$$

3.3.2 Execution Time

Time is measured in milliseconds. Execution time is always a measure when we are comparing algorithms. Even more if those algorithms execution time is heavily dependent to their complexity.

4 Results

Table 1: **Content Based Algorithm Results**

Training Dataset	**Testing Dataset**	**Mean Absolute Error**	**Execution time (ms)**
u1.base	u1.test	1.6467431428213226	30514
u2.base	u2.test	1.6055222166704628	27714
u3.base	u3.test	1.608925907479106	27164
u4.base	u4.test	1.6259192043203685	26687
u5.base	u5.test	1.6284658627202895	27124
ua.base	ua.test	1.6425364580036836	26640
ub.base	ub.test	1.6357196576385744	26861

Table 2: **Latent Factors Algorithm Results**

Training Dataset	Testing Dataset	Mean Absolute Error	Execution time (ms)
u1.base	u1.test	1.1818684937209607	10195
u2.base	u2.test	1.1800652808093945	6517
u3.base	u3.test	1.1783366748334452	5377
u4.base	u4.test	1.1730543877181654	5433
u5.base	u5.test	1.1686585291940668	5217
ua.base	ua.test	1.2008035300836668	5214
ub.base	ub.test	1.2134460078406009	5083

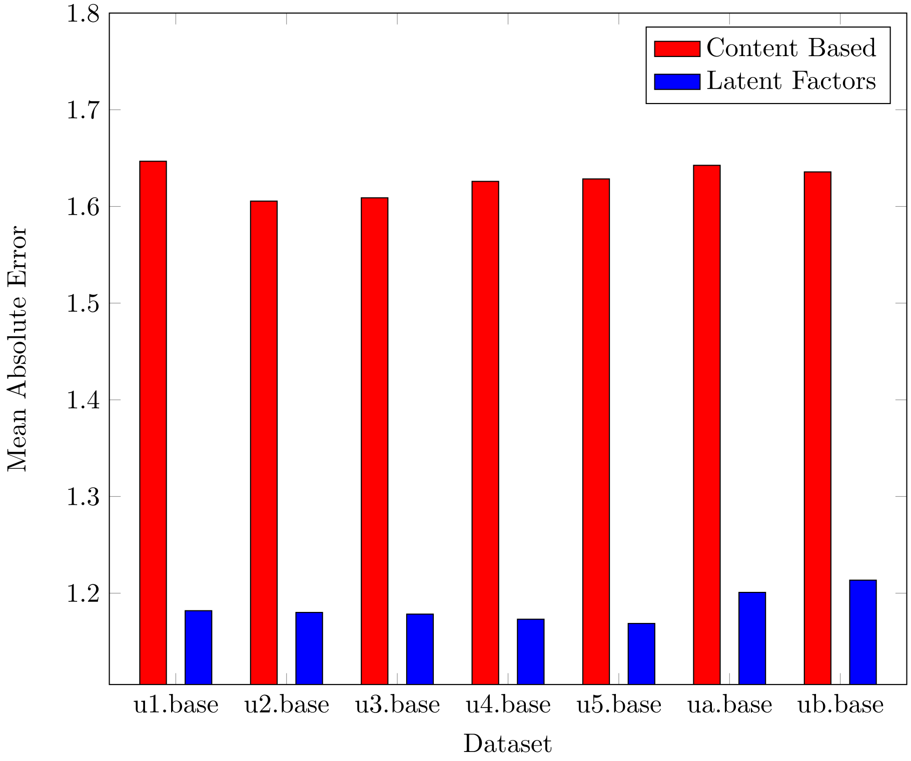

Figure 1: **Latent Factors vs Content Based on Mean Absolute Value**

husjdfsldfkjsaflkj

jhsdkfhksjhfkshf

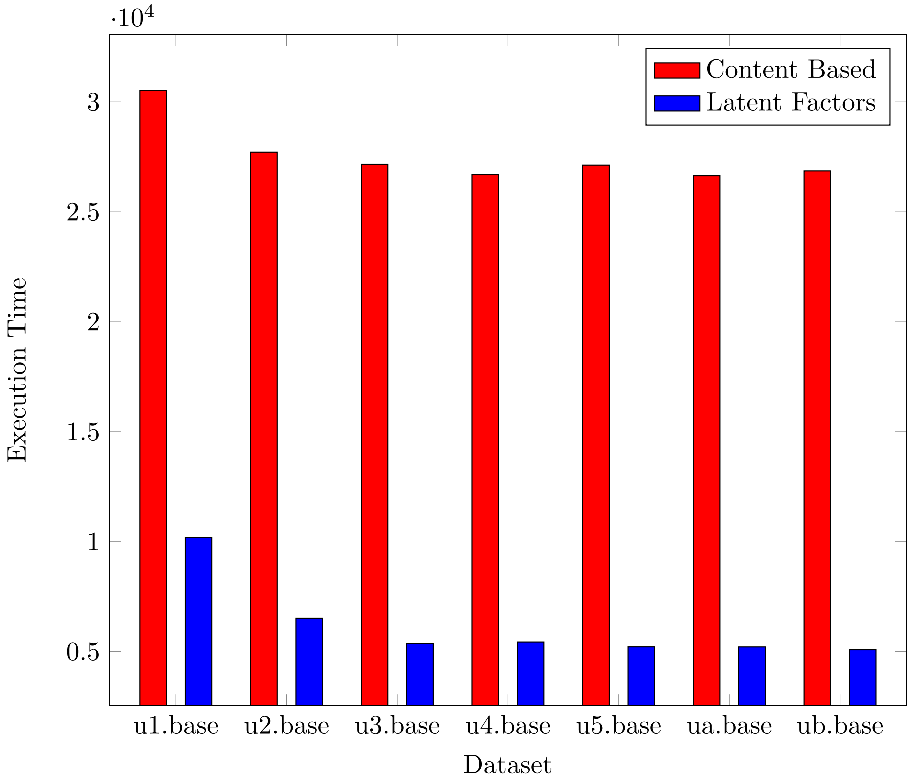

Figure 2: **Latent Factors vs Content Based on Execution Time**

5 Conclusion

As a conclusion we can see that als is better on both metrics from the content based.

6 References

References

[1] "Apache Spark lightning-fast cluster computing." `https://spark.apache.org/`. Accessed: 2017-05-21.

[2] P. Melville and V. Sindhwani, "Recommender systems," *Encyclopedia of Machine Learning and Data Mining*, pp. 1056–1066, 2017.

[3] "MovieLens grouplens." `https://grouplens.org/datasets/movielens/`. Accessed: 2017-05-22.

Part II

Appendices

A Code used

A.1 User Based Collaborative Filtering

A.2 Product Based Collaborative Filtering

A.3 Latent Factors

A.4 infra code

B Metrics

B.1 What is the mean absolute error

B.2 Time

List of Tables

List of Figures