

# **BÁO CÁO: BIG DATA**

## **Đề Tài: Amazon Product Recommendation System**

**Nhóm: Trần Văn Thành**

**Phan Xuân Đức**

**GVHD: Thầy Trần Vĩnh Đức**

# NỘI DUNG

1. Giới thiệu bài toán
2. Phương pháp học máy
3. Cài đặt
4. Kết quả

# AMAZON PRODUCT RECOMMENDATION SYSTEM

**amazon.com** [Help](#) | [Close window](#)

**Recommended for You**



**Inside Apple**  
Mark Zuckerberg

[Inside Apple: How America's Most Admired--and Secretive--Company Really Works](#)  
**Our Price: \$9.99**  
[Used & new](#) from **\$9.99**

[See all buying options](#)

Rate this item  
☒ ☆☆☆☆☆

☐ I own it  
☐ Not interested

**Because you purchased...**



**THE TOYOTA WAY**  
Jeffrey Liker

[The Toyota Way : 14 Management Principles from the World's Greatest Manufacturer](#)  
(Kindle Edition)

☒ ☆☆☆☆☆

☐ This was a gift  
☐ Don't use for recommendations

# AMAZON PRODUCT RECOMMENDATION SYSTEM

- Có 2 thực thể chính trong một hệ Recommendation system là user và item.
- User là người dùng.
- Item là sản phẩm: ví dụ như các bộ phim, bài hát, cuốn sách (amazon product RS), videos clip (youtube RS) hoặc cũng có thể là chính các user khác trong bài toán gợi ý kết bạn của Facebook.

# AMAZON PRODUCT RECOMMENDATION SYSTEM

## People You May Know

Add people you know as friends and connect with public profiles you like.



**Lala Lalabs** ✕  
Add as friend



**Brian Crecente** ✕  
Add as friend



**Brian Ashcraft** ✕  
Add as friend



**justin bieber** ✕  
Add as friend



**Camile Gozon** ✕  
Add as friend



**Karla Danielle Beger** ✕  
Add as friend



**Taylor-Alison Swifty** ✕  
Add as friend



**Adam Rifkin** ✕  
Add as friend



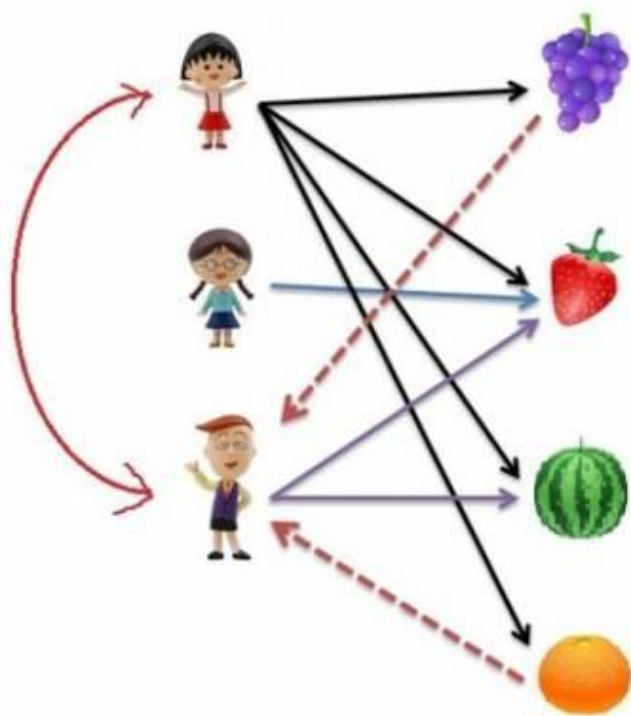
**Luke Plunkett** ✕  
Add as friend

# PHƯƠNG PHÁP HỌC MÁY

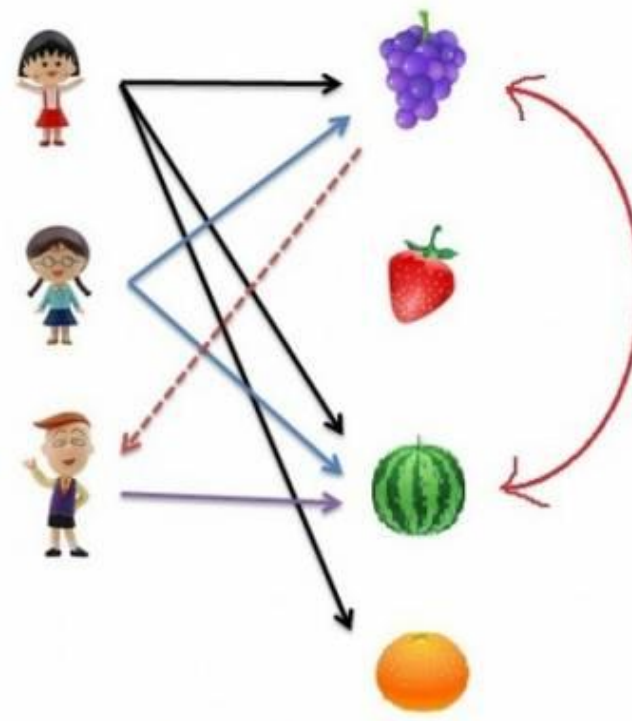
## ○ Collaborative Filtering (CF) – Lọc cộng tác

- Phương pháp này gợi ý các sản phẩm dựa trên sự tương quan giữa người dùng và/hoặc sản phẩm (*user-user, user-item, item-item*)
- Một sản phẩm được giới thiệu tới người dùng dựa trên những người dùng khác có hành vi tương tự.

# COLLABORATIVE FILTERING (CF)



User-based filtering



Item-based filtering

# UTILITY MATRIX

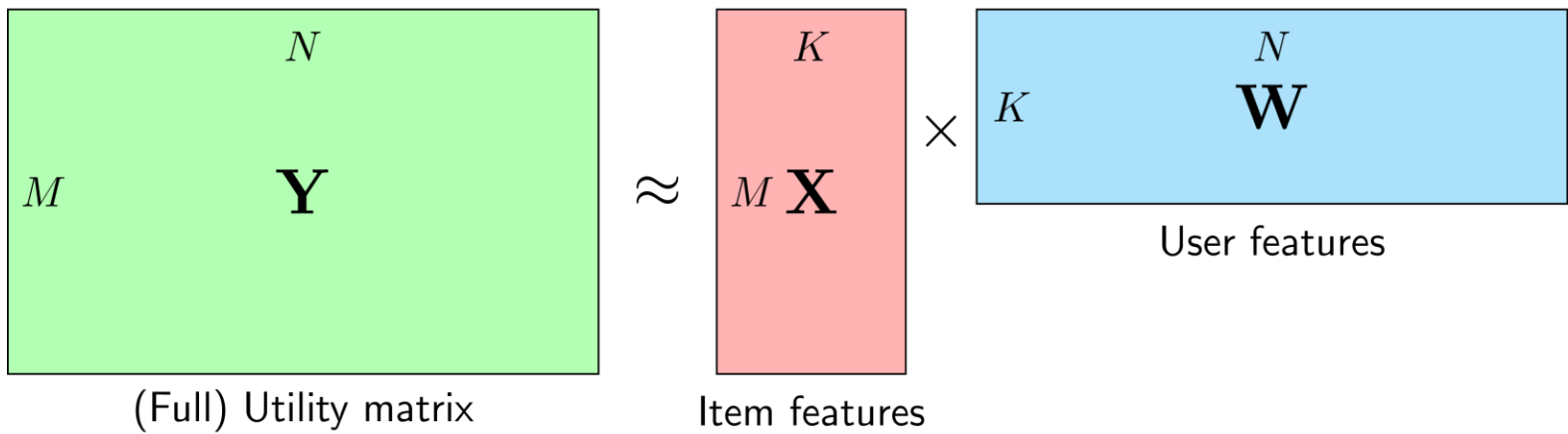
$$\mathbf{Y} \approx \begin{bmatrix} \mathbf{x}_1 \mathbf{w}_1 & \mathbf{x}_1 \mathbf{w}_2 & \dots & \mathbf{x}_1 \mathbf{w}_N \\ \mathbf{x}_2 \mathbf{w}_1 & \mathbf{x}_2 \mathbf{w}_2 & \dots & \mathbf{x}_2 \mathbf{w}_N \\ \dots & \dots & \ddots & \dots \\ \mathbf{x}_M \mathbf{w}_1 & \mathbf{x}_M \mathbf{w}_2 & \dots & \mathbf{x}_M \mathbf{w}_N \end{bmatrix} = \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \dots \\ \mathbf{x}_M \end{bmatrix} \begin{bmatrix} \mathbf{w}_1 & \mathbf{w}_2 & \dots & \mathbf{w}_N \end{bmatrix} = \mathbf{XW}$$

với  $M, N$  lần lượt là số *items* và số *users*.



# UTILITY MATRIX

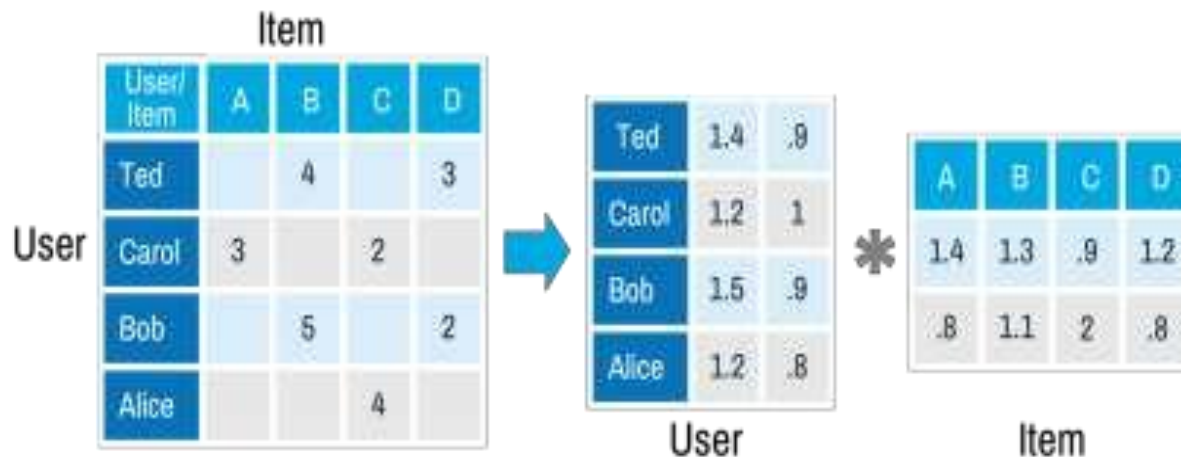
$$\mathbf{Y} \approx \hat{\mathbf{Y}} = \mathbf{X}\mathbf{W}$$



- Matrix Factorization. Utility matrix  $\mathbf{Y}$  được phân tích thành tích của hai ma trận low-rank  $\mathbf{X}$  và  $\mathbf{W}$
- Đánh giá của user  $n$  lên item có thể được xấp xỉ bởi  $y_{mn} = \mathbf{x}^m \mathbf{w}_n$

# MATRIX FACTORIZATION ALTERNATIVE LEAST SQUARE

- Chọn ngẫu nhiên giá trị U, V và điều chỉnh các giá trị của U, V để cho RMSE nhỏ hơn
  - Step1: Sửa V và Cập nhật tất cả các mục của U.
  - Step2: Sửa U và Cập nhật tất cả các mục của V.
  - Step3: Lặp lại bước đầu tiên và bước thứ hai cho đến khi đạt đến độ bão hòa ít nhất của giá trị RMSE.



# DATA SETS

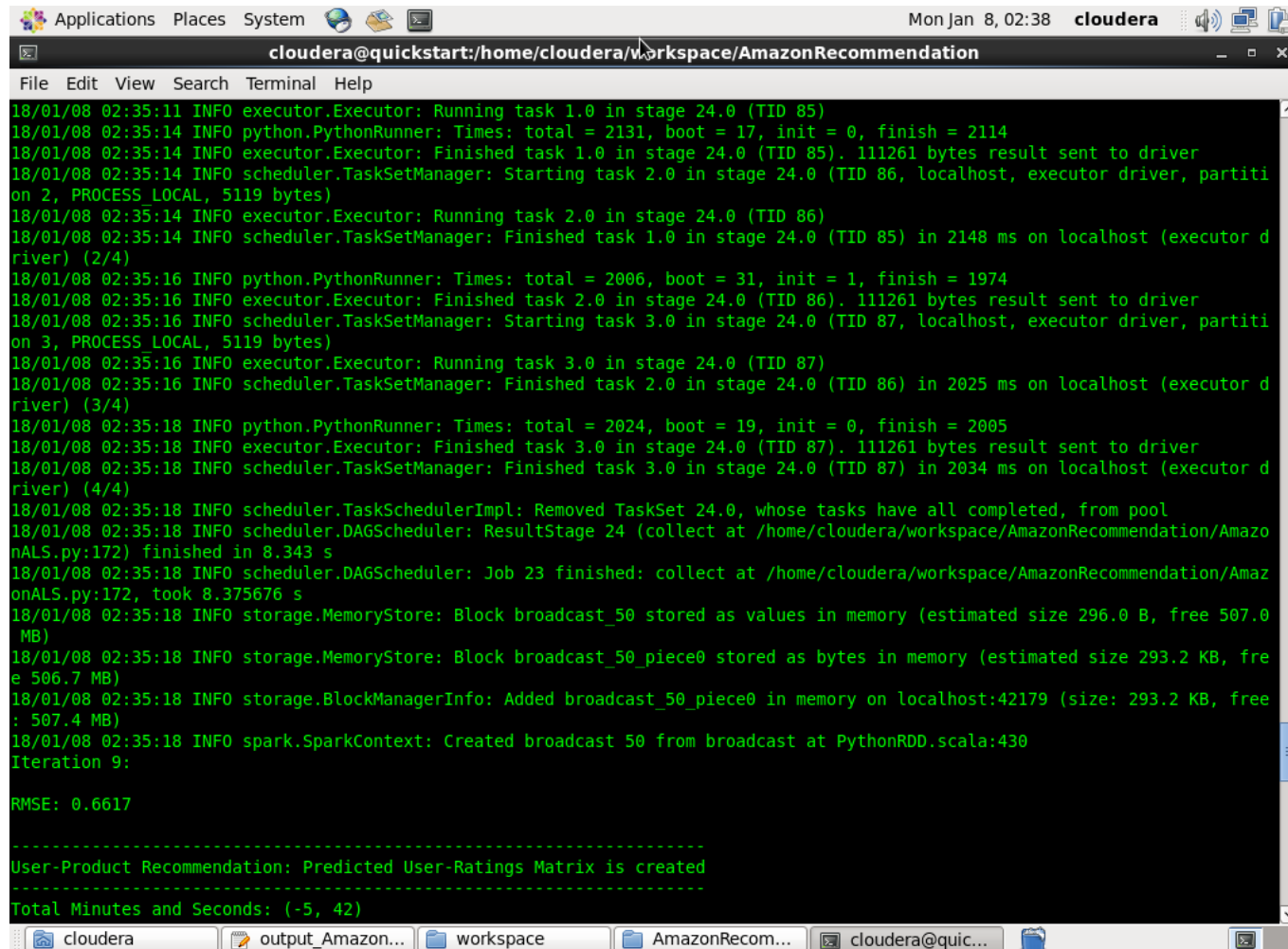
- Bộ dữ liệu được lấy từ dữ liệu sản phẩm của Amazon trong 18 năm qua của các loại sản phẩm khác nhau.
- Trong tập dữ liệu, chọn dữ liệu **rating** của sản phẩm Nhạc cụ (Musical Instruments) được bán trên Amazon.
- 500,176 ratings
- <http://jmcauley.ucsd.edu/data/amazon/links.html>

# CÀI ĐẶT

- Trên Hadoop đã cài sẵn môi trường spark 1.2.1 và python 2.7
- `$ spark-submit AmazonALS.py <InputDirectory>  
<OutputFileName> <Iterations> <Partitions>`

# KẾT QUẢ

- \$ spark-submit AmazonALS.py /user/AZ\_P/input outputAmazonReco.dat 10 4



```
cloudera@quickstart:/home/cloudera/workspace/AmazonRecommendation
File Edit View Search Terminal Help
18/01/08 02:35:11 INFO executor.Executor: Running task 1.0 in stage 24.0 (TID 85)
18/01/08 02:35:14 INFO python.PythonRunner: Times: total = 2131, boot = 17, init = 0, finish = 2114
18/01/08 02:35:14 INFO executor.Executor: Finished task 1.0 in stage 24.0 (TID 85). 111261 bytes result sent to driver
18/01/08 02:35:14 INFO scheduler.TaskSetManager: Starting task 2.0 in stage 24.0 (TID 86, localhost, executor driver, parti
on 2, PROCESS LOCAL, 5119 bytes)
18/01/08 02:35:14 INFO executor.Executor: Running task 2.0 in stage 24.0 (TID 86)
18/01/08 02:35:14 INFO scheduler.TaskSetManager: Finished task 1.0 in stage 24.0 (TID 85) in 2148 ms on localhost (executor d
river) (2/4)
18/01/08 02:35:16 INFO python.PythonRunner: Times: total = 2006, boot = 31, init = 1, finish = 1974
18/01/08 02:35:16 INFO executor.Executor: Finished task 2.0 in stage 24.0 (TID 86). 111261 bytes result sent to driver
18/01/08 02:35:16 INFO scheduler.TaskSetManager: Starting task 3.0 in stage 24.0 (TID 87, localhost, executor driver, parti
on 3, PROCESS LOCAL, 5119 bytes)
18/01/08 02:35:16 INFO executor.Executor: Running task 3.0 in stage 24.0 (TID 87)
18/01/08 02:35:16 INFO scheduler.TaskSetManager: Finished task 2.0 in stage 24.0 (TID 86) in 2025 ms on localhost (executor d
river) (3/4)
18/01/08 02:35:18 INFO python.PythonRunner: Times: total = 2024, boot = 19, init = 0, finish = 2005
18/01/08 02:35:18 INFO executor.Executor: Finished task 3.0 in stage 24.0 (TID 87). 111261 bytes result sent to driver
18/01/08 02:35:18 INFO scheduler.TaskSetManager: Finished task 3.0 in stage 24.0 (TID 87) in 2034 ms on localhost (executor d
river) (4/4)
18/01/08 02:35:18 INFO scheduler.TaskSchedulerImpl: Removed TaskSet 24.0, whose tasks have all completed, from pool
18/01/08 02:35:18 INFO scheduler.DAGScheduler: ResultStage 24 (collect at /home/cloudera/workspace/AmazonRecommendation/Amazo
nALS.py:172) finished in 8.343 s
18/01/08 02:35:18 INFO scheduler.DAGScheduler: Job 23 finished: collect at /home/cloudera/workspace/AmazonRecommendation/Amaz
onALS.py:172, took 8.375676 s
18/01/08 02:35:18 INFO storage.MemoryStore: Block broadcast_50 stored as values in memory (estimated size 296.0 B, free 507.0
MB)
18/01/08 02:35:18 INFO storage.MemoryStore: Block broadcast_50_piece0 stored as bytes in memory (estimated size 293.2 KB, fre
e 506.7 MB)
18/01/08 02:35:18 INFO storage.BlockManagerInfo: Added broadcast_50_piece0 in memory on localhost:42179 (size: 293.2 KB, free
: 507.4 MB)
18/01/08 02:35:18 INFO spark.SparkContext: Created broadcast 50 from broadcast at PythonRDD.scala:430
Iteration 9:
RMSE: 0.6617
-----
User-Product Recommendation: Predicted User-Ratings Matrix is created
-----
Total Minutes and Seconds: (-5, 42)
```

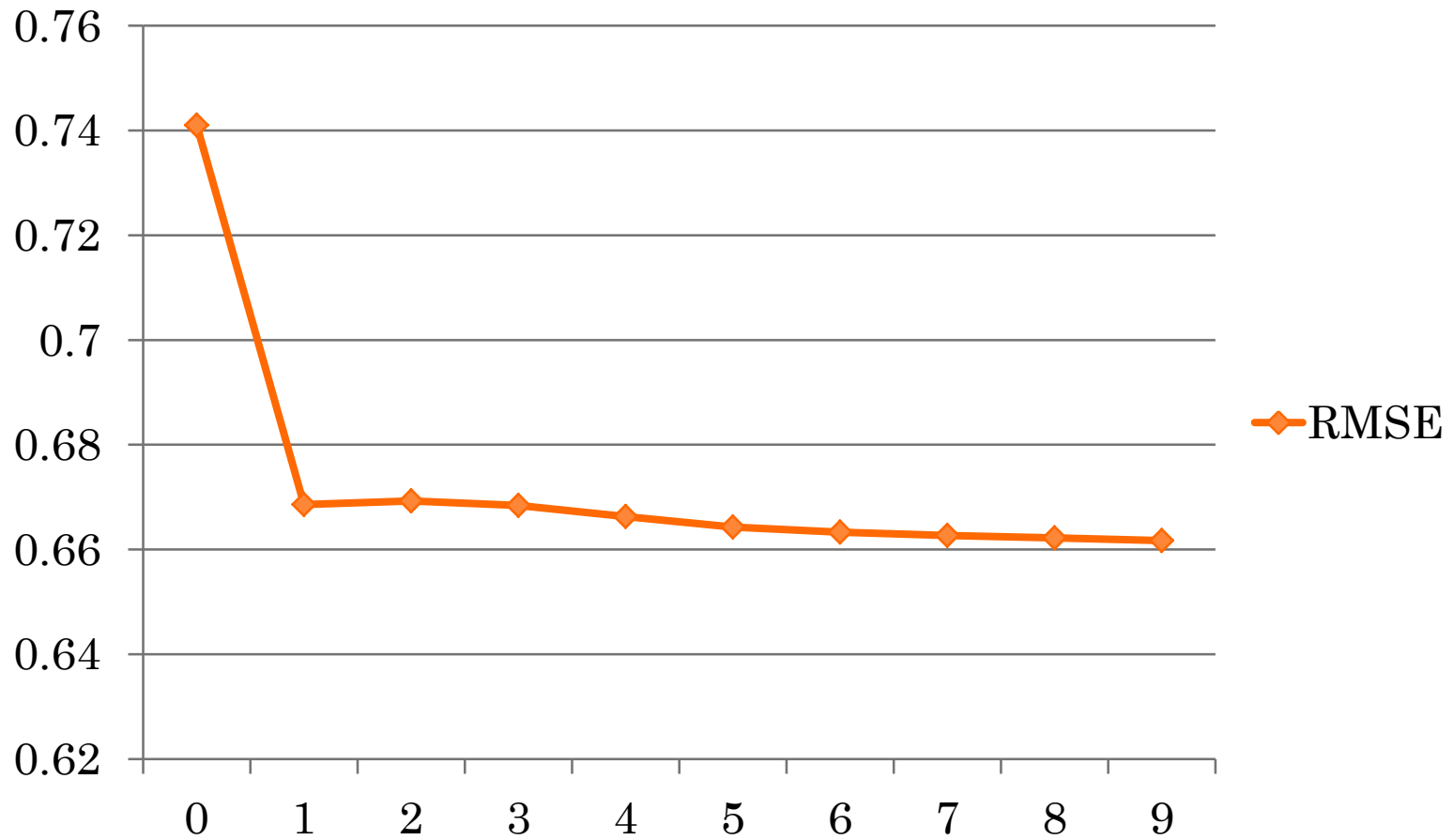
○ Đánh giá

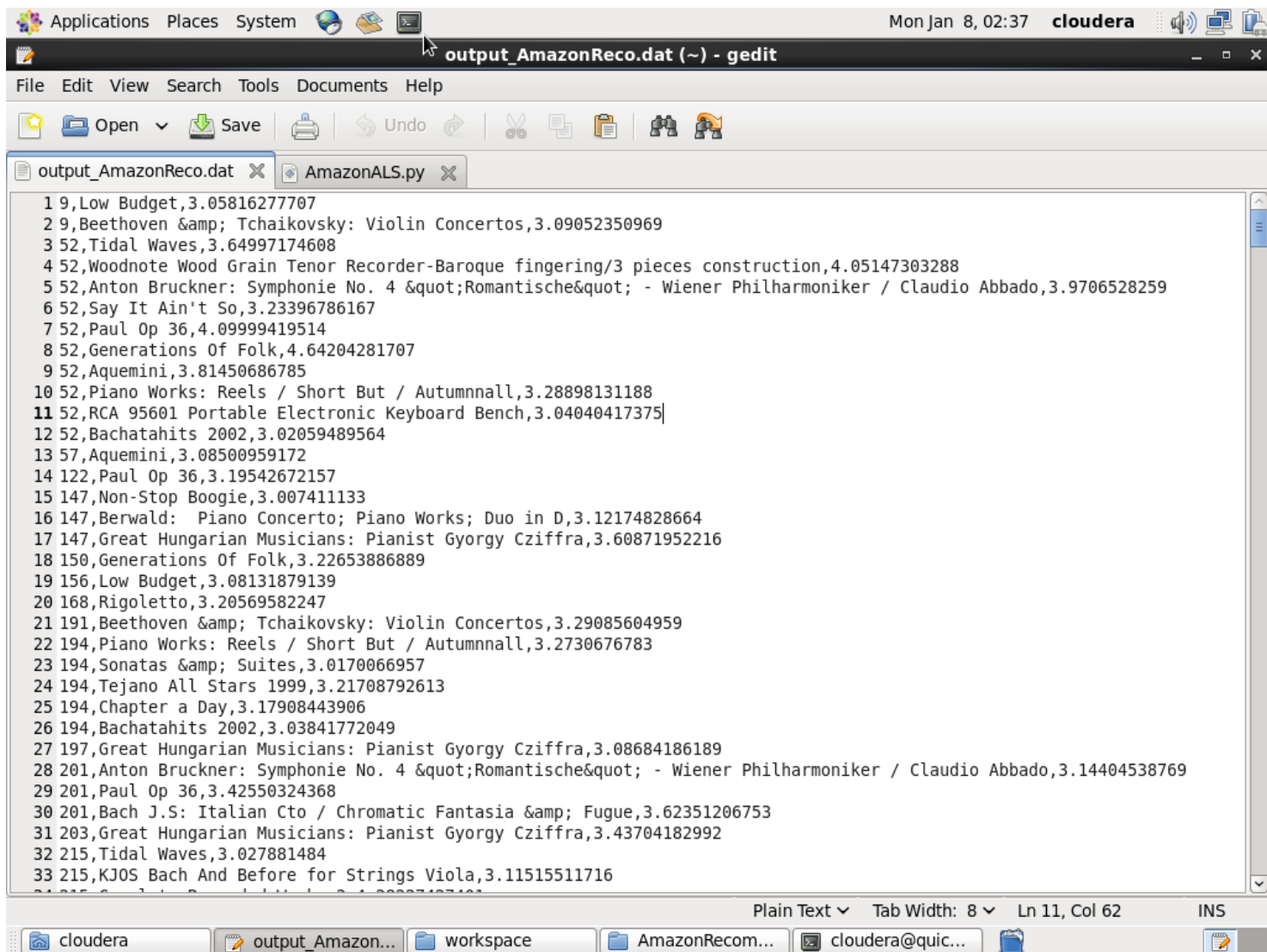
Loop	RMSE
0	0.7410
1	0.6686
2	0.6693
3	0.6684
4	0.6663
5	0.6643
6	0.6633
7	0.6627
8	0.6622
9	0.6617

$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2}$$

➤ Thời gian chạy: 5 phút 42 giây.

## RMSE





The screenshot shows a Gedit editor window titled "output\_AmazonReco.dat (~) - gedit". The window contains a list of music recommendations, each consisting of a user ID, a product name, and a predicted rating. The recommendations are as follows:

User ID	Product Name	Predicted Rating
19	Low Budget	3.05816277707
29	Beethoven & Tchaikovsky: Violin Concertos	3.09052350969
352	Tidal Waves	3.64997174608
452	Woodnote Wood Grain Tenor Recorder-Baroque fingering/3 pieces construction	4.05147303288
552	Anton Bruckner: Symphonie No. 4 "Romantische" - Wiener Philharmoniker / Claudio Abbado	3.9706528259
652	Say It Ain't So	3.23396786167
752	Paul Op 36	4.09999419514
852	Generations Of Folk	4.64204281707
952	Aquemini	3.81450686785
1052	Piano Works: Reels / Short But / Autumnall	3.28898131188
1152	RCA 95601 Portable Electronic Keyboard Bench	3.04040417375
1252	Bachatahits 2002	3.02059489564
1357	Aquemini	3.08500959172
14122	Paul Op 36	3.19542672157
15147	Non-Stop Boogie	3.007411133
16147	Berwald: Piano Concerto; Piano Works; Duo in D	3.12174828664
17147	Great Hungarian Musicians: Pianist Gyorgy Cziffra	3.60871952216
18150	Generations Of Folk	3.22653886889
19156	Low Budget	3.08131879139
20168	Rigoletto	3.20569582247
21191	Beethoven & Tchaikovsky: Violin Concertos	3.29085604959
22194	Piano Works: Reels / Short But / Autumnall	3.2730676783
23194	Sonatas & Suites	3.0170066957
24194	Tejano All Stars 1999	3.21708792613
25194	Chapter a Day	3.17908443906
26194	Bachatahits 2002	3.03841772049
27197	Great Hungarian Musicians: Pianist Gyorgy Cziffra	3.08684186189
28201	Anton Bruckner: Symphonie No. 4 "Romantische" - Wiener Philharmoniker / Claudio Abbado	3.14404538769
29201	Paul Op 36	3.42550324368
30201	Bach J.S: Italian Cto / Chromatic Fantasia & Fugue	3.62351206753
31203	Great Hungarian Musicians: Pianist Gyorgy Cziffra	3.43704182992
32215	Tidal Waves	3.027881484
33215	KJOS Bach And Before for Strings Viola	3.11515511716

<userID>, <recommendedProduct>, <predictedRating>



# REFERENCES

- <https://machinelearningcoban.com/>

Thanks!

A hand-drawn illustration of a smiling face with a hand reaching up towards the word 'Thanks!'. The face is simple, with a circle for the head, two dots for eyes, and a curved line for a smile. A hand with five fingers is reaching up from the bottom right towards the word. A horizontal line separates the word from the face. A small '©' symbol is at the bottom right of the face.