Verification and Validation Report: Movie Recommender

Seyed Ali Mousavi

April 16, 2024

1 Revision History

Date	Version	Notes
2024-04-15	1.0	Final Release

2 Symbols, Abbreviations and Acronyms

symbol	**description**
T	Test

Contents

List of Tables

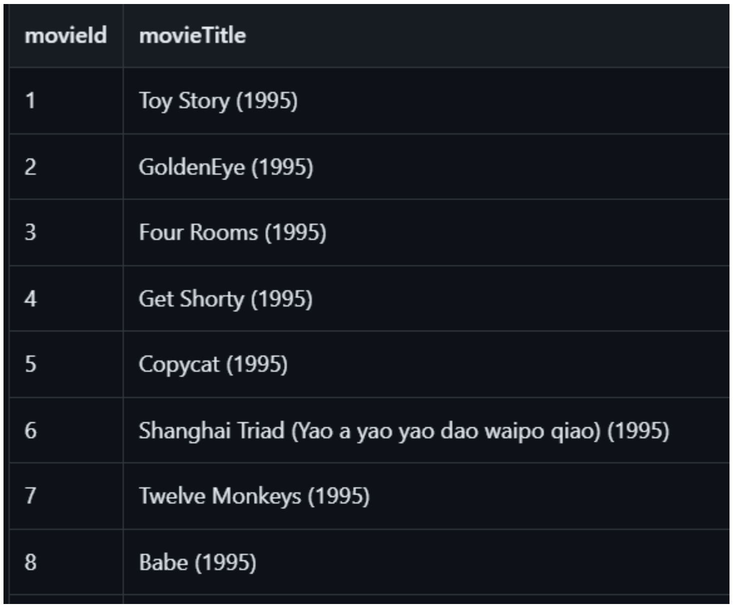
List of Figures

3 Functional Requirements Evaluation

In this Section, we report the outcome of tests for functional requirements. Please refer to the VnVPlane for the test details.

3.1 One Popular Movie Test

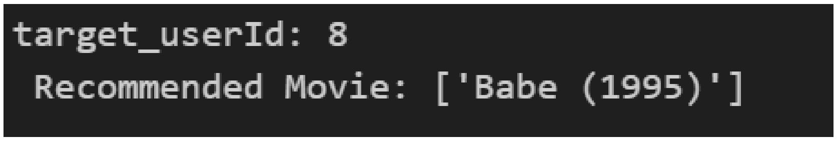
The figure below shows the movie IDs and titles to be used for the test:



movieId	movieTitle
1	Toy Story (1995)
2	GoldenEye (1995)
3	Four Rooms (1995)
4	Get Shorty (1995)
5	Copycat (1995)
6	Shanghai Triad (Yao a yao yao dao waipo qiao) (1995)
7	Twelve Monkeys (1995)
8	Babe (1995)

Figure 1: List of Movies

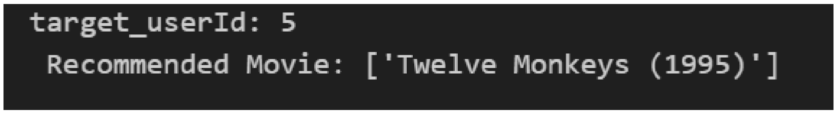
The outcome of the test is Movie 'Babe(1995)' (movieId = 8) as expected



```
target_userId: 8
 Recommended Movie: ['Babe (1995)']
```

Figure 2: The output of test-one-popular-movie-id1

The next test is to change the position of the target user and the best movie in the user-item interaction matrix (our target user is userId 5 this time and the best movie is movieId 7). We get the this movie as expected:

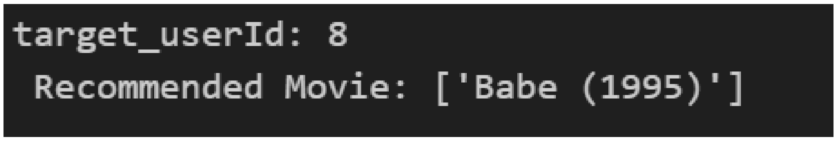


```
target_userId: 5
 Recommended Movie: ['Twelve Monkeys (1995)']
```

Figure 3: The output of test-one-popular-movie-id2

3.2 Artificial Close Users

Again we should get the last movieId as the best recommendation. The outcome is as expected:



```
target_userId: 8
 Recommended Movie: ['Babe (1995)']
```

Figure 4: The output of test-Artificial Close Users-id3

4 Nonfunctional Requirements Evaluation

4.1 Accuracy

Figures 5 and 6 on the next page show the result of Accuracy testing. As we can see the RMSE and MAE of our Movie Recommender rating prediction are significantly better than that of using a randomly generated ratings list:

4.2 Understandability

I made a diligent effort to thoroughly document and ensure the readability of all the code. However, I was unable to find the time to precisely follow the plan outlined in the VnVPlan.

4.3 Maintainability

I structured the code to be modular, testable, and readable. However, I didn't manage to allocate time to strictly adhere to the outlined plan in the VnVPlan.

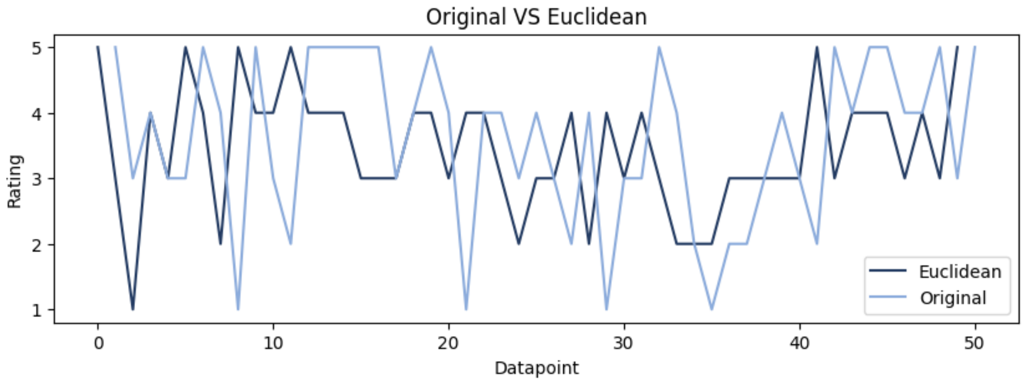
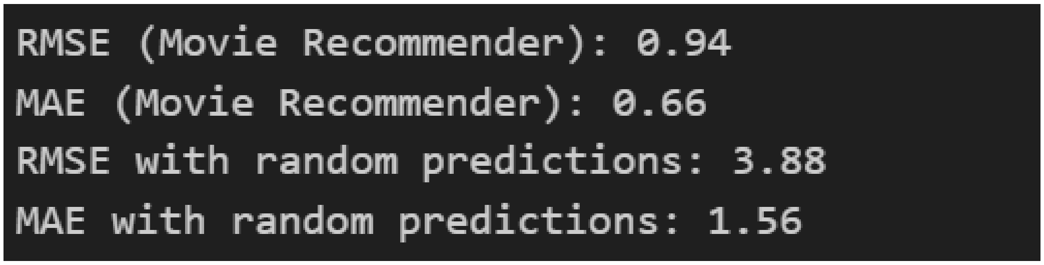


Figure 5: The output of test-accuracy-id4

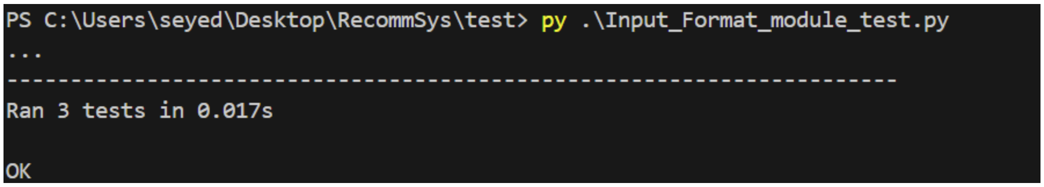


```
RMSE (Movie Recommender): 0.94
MAE (Movie Recommender): 0.66
RMSE with random predictions: 3.88
MAE with random predictions: 1.56
```

Figure 6: The output of test-accuracy-id4

5 Unit Testing

The figure below shows that unit testing for the Input Format Module has been done successfully.

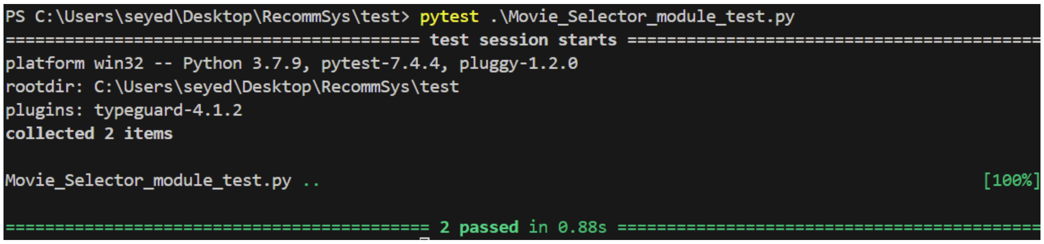


```
PS C:\Users\seyed\Desktop\RecommSys\test> py .\Input_Format_module_test.py
...
----------------------------------------------------------------------
Ran 3 tests in 0.017s

OK
```

Figure 7: The output of Input Format Module Test

The figure below shows that unit testing for the Movie Selector Module has been done successfully.



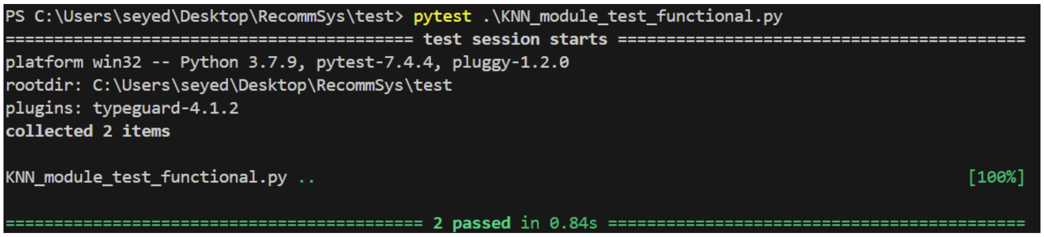
```
PS C:\Users\seyed\Desktop\RecommSys\test> pytest .\Movie_Selector_module_test.py
======================================== test session starts ========================================
platform win32 -- Python 3.7.9, pytest-7.4.4, pluggy-1.2.0
rootdir: C:\Users\seyed\Desktop\RecommSys\test
plugins: typeguard-4.1.2
collected 2 items

Movie_Selector_module_test.py ..                                                               [100%]

========================================= 2 passed in 0.88s =========================================
```

Figure 8: The output of Movie Selector Module Test

The figure below shows that unit testing for the KNN Module has been done successfully.



```
PS C:\Users\seyed\Desktop\RecommSys\test> pytest .\KNN_module_test_functional.py
======================================== test session starts ========================================
platform win32 -- Python 3.7.9, pytest-7.4.4, pluggy-1.2.0
rootdir: C:\Users\seyed\Desktop\RecommSys\test
plugins: typeguard-4.1.2
collected 2 items

KNN_module_test_functional.py ..                                                              [100%]

========================================= 2 passed in 0.84s =========================================
```

Figure 9: The output of KNN Module Test

6 Trace to Requirements

The following table shows which test cases are supporting which requirements.

	R1	R2	R3
test-id1		X	X
test-id2		X	X
test-id3		X	X
test-id4			

Table 1: Relation of Test Cases and Requirements.

7 Trace to Modules

In the Unit Testing section, it's evident which tests were executed for each module. (Please refer to the `test` folder within the project to access these tests.) Additionally, it's noteworthy that the tests described for nonfunctional testing can also be considered system testing, as they ensure the proper integration of modules and the overall performance of the code.

Appendix — Reflection

The information in this section will be used to evaluate the team members on the graduate attribute of Reflection. Please answer the following question:

1. In what ways was the Verification and Validation (VnV) Plan different from the activities that were actually conducted for VnV? If there were differences, what changes required the modification in the plan? Why did these changes occur? Would you be able to anticipate these changes in future projects? If there weren't any differences, how was your team able to clearly predict a feasible amount of effort and the right tasks needed to build the evidence that demonstrates the required quality? (It is expected that most teams will have had to deviate from their original VnV Plan.)

Hello