**CS673 Software Engineering**

**Team 4 - RecFlix**

**Software Test Document**

| Team Member | Role(s) | Signature | Date |
| --- | --- | --- | --- |
| Alex Kolbin | Configuration Leader | *AK* | 05/27/2022 |
| Eric Ashton | QA Leader | *EA* | 5/29/2022 |
| Ellie Nerney | Project Manager | *EN* | 5/30/2022 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Revision history**

| **Version** | **Author** | **Date** | **Change** |
| --- | --- | --- | --- |
| 1 | Eric | 5/29 | Added manual test for SQL server and automated test for control model |
| 1 | Alex | 5/29 | Automated CodeQL SAST Scan |

[Testing Summary](#_sm5odwyvuk3j)

[Manuel Tests Reports](#_pqso2mbjyzx4)

[Automated Testing Reports](#_mtfbusfb0eq3)

[Testing Metrics](#_rijyjeu2ojqa)

[References](#_15tmymhipvdv)

[Glossary](#_8n34lvocupub)

# Testing Summary

All team members in the project contribute to unit testing. Development is test-driven with each part of the code designed to meet the requirements of one or more tests.

The integration testing is the responsibility of the QA Leader (Eric). Eric is also responsible for testing that the project as a whole fulfills our main objective of creating a neural network that creates customized recommendations that would be more useful to users in real life than simply selecting the most popular movies.

# Manual Testing Report

**Test: Test new database by creating and deleting a table in SQL**

* Part of Pivotal Tracker item #182279461
* New or old: New
* Test items: Test logging into AWS server. Test creating a table and dropping it.
* Test priority: Medium
* Dependencies: Server setup
* Preconditions: Server is accessible by testers
* input data: Login information, host, port for server
* Test steps : 1) Install PGAdmin

2) Connect to server: cs673-team4-db.c5wjsyde0zav.us-east-1.rds.amazonaws.com Port: 5432

3) Make new Query

4) Enter code to create a table

5) Check that the table is created in the table list

6) Enter code to drop the table

7) Check that the table is dropped successfully

* Postconditions: After the test, the server should be in the same state as before as the table was dropped.
* Expected output: The table appears in the table list then is dropped successfully
* Actual output: same as expected
* Pass or Fail: Pass
* Bug id/link: n/a

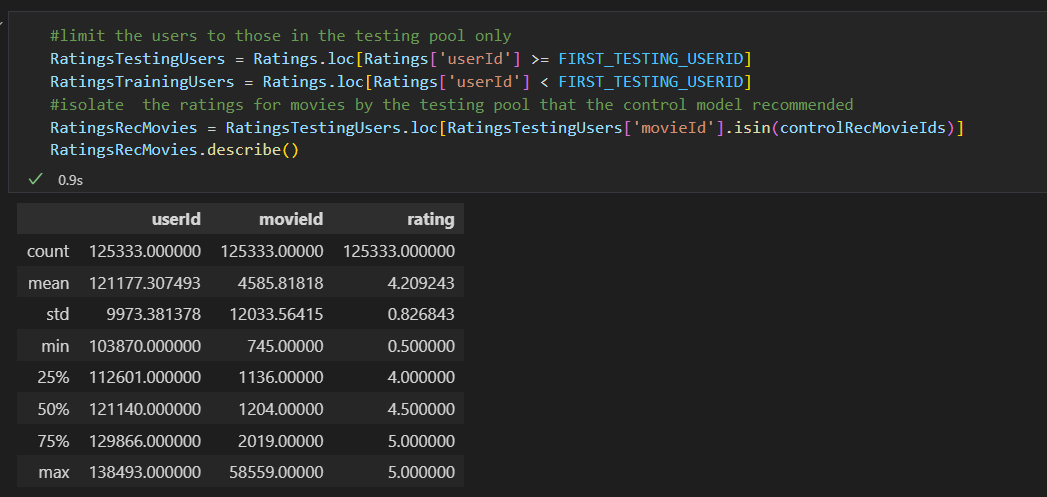
# Automated Testing Report

To test our neural network we will use a “control model” that is not customized for each user but simply recommends the 30 highest-rated eligible movies to each user. We expect that our neural network, using input data from testing users, will be able to predict a list of movies that they will like very well. We will measure our success by comparing the average rating of the control model to the average rating of the neural network model.

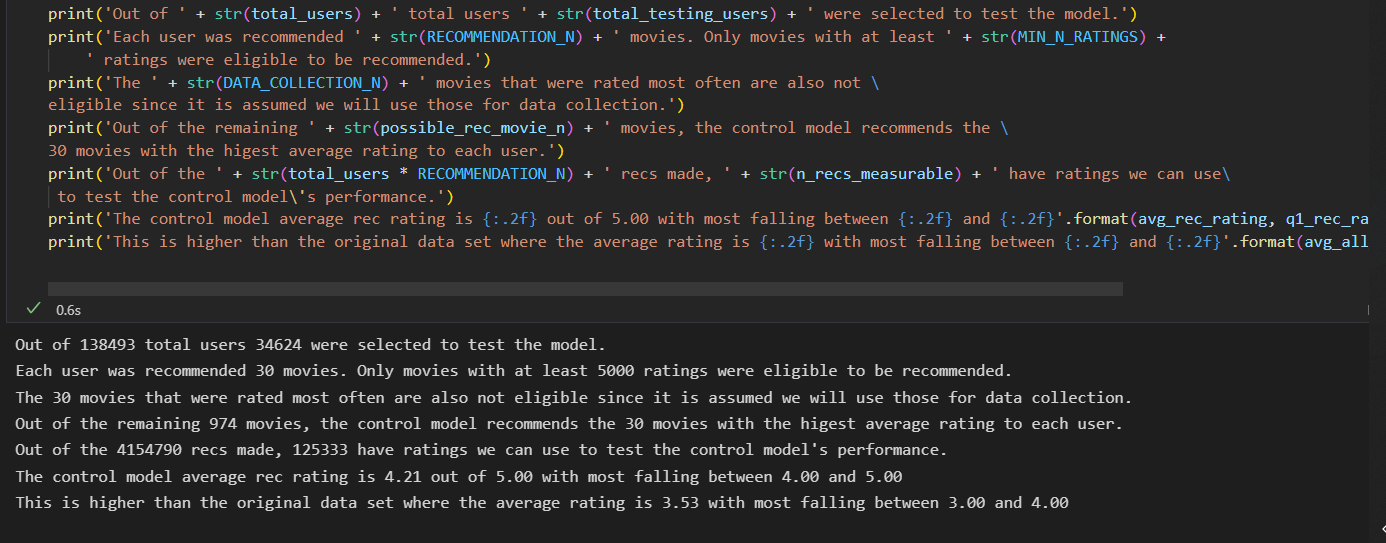
Currently (Iteration 1) we have created the control model and a prototype neural network but have not yet tested the neural network itself.

Screenshot showing the control model recommendation of 30 movies with the highest ratings with at least 5000 ratings (not all movies shown): 

Screenshot showing the ratings for the recommendations made by the control model that were available to evaluate. This shows there were 125,333 ratings that were testable and that they had an average rating of 4.209243 with most falling in the interquartile range of 4.0 to 5.0.

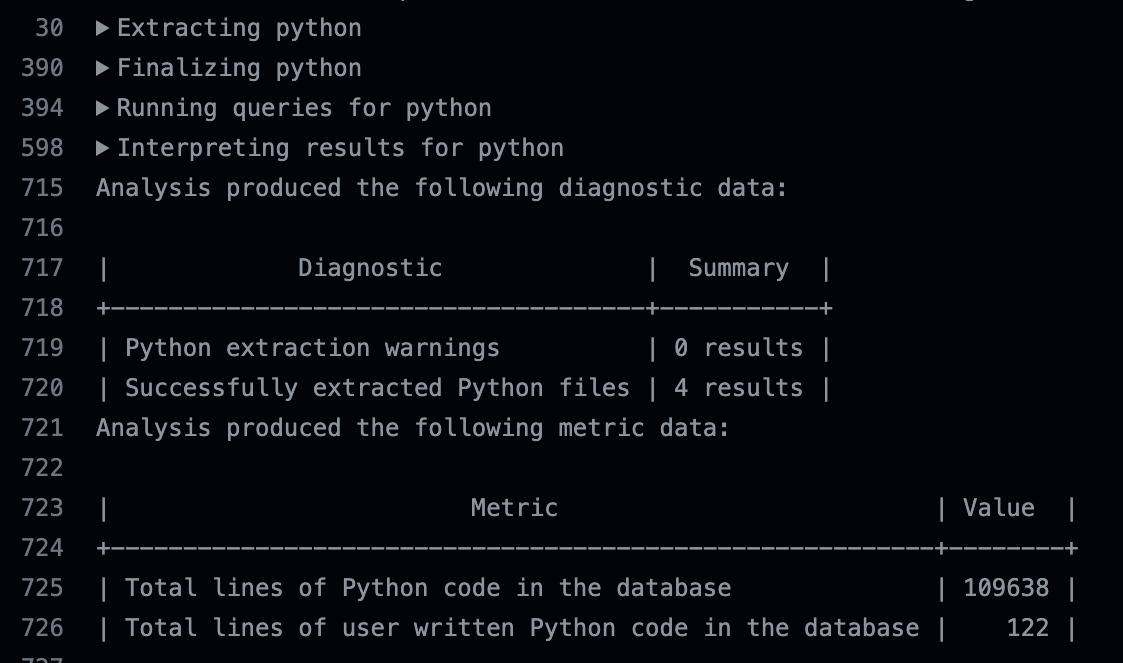


Full output of control model:



**SAST**

GitHub’s proprietary code analysis engine, CodeQL, has been integrated with our repository. CodeQL performs automated static security scans on the python and javascript/typescript code within the repo. The automation is configured using GitHub Actions (in the .github/workflows/ directory) such that anytime code is pushed to the main branch, or a PR is opened against the main branch, the scans will run. A report is generated in the workflow run itself, and any vulnerabilities will be reported in the “security” tab of our repository in the “Code scanning alerts” subsection. Here is an example of the generated report:



# Testing Metrics

Our main testing metric for our neural network will be the average rating of the movies recommended by the model. This assumes that

1. 25% of the original data will be reserved for testing and not used in training.
2. Predictions used in the model rating cannot have been used as inputs to the model (that is, we are recommending movies where the model is not using the user’s rating to know they will like it).

Our control model is based on recommending each person the 30 highest-rated movies not used in data collection that have a sufficient number of ratings to have likely been seen by the testing users. Our control model recommends movies rated 4.21 out of 5.00 which is significantly better than the moveal average in the data set of 3.53 out of 5.00.

# References

# Glossary