The two algorithms, *algorithmAforProject()* - uses the efficient way for solving the similarity between two users - and *algorithBforProject()* - uses the naive functions to solve the similarity - are implemented inside *RecommenderSystem.py* class. In both methods, similarity results were calculated according to the given formula 𝑠(𝑈1, 𝑈2) = (|𝐿1 ∩ 𝐿2| + |𝐷1 ∩ 𝐷2| − |𝐿1 ∩ 𝐷2| − |𝐿2 ∩ 𝐷1|) ÷ |𝐿1 ∪ 𝐿2 ∪ 𝐷1 ∪ 𝐷2|, yet differ in the way finding unions and intersections of users’ likes/dislikes (intersectionNaive(), intersectionEfficient() etc.)

The *test.py* class was used to test two algorithms by using the external data from [Amazon Review Data (2018)](https://nijianmo.github.io/amazon/index.html) , in JSON format as described in the report

|  |  |  |
| --- | --- | --- |
| Data Set Type | Total Rating Amount | Total Users |
| Magazine Subscriptions | 12,000 ratings | 11,000 |
| Patio, Lawn and Garden | 210,910 ratings | 137,662 |
| Musical Instruments | 304,000 ratings | 222,250 |
| Office Products | 606,000 ratings | 491,760 |
| Amazon Instant Video | 583,933 ratings | 439,287 |

The chart of running the tests on both algorithms which compared their processing times, in terms of seconds, is shown below. The *algorirthmA()* uses *getSimilarityEfficient()* method which is supposed to run in O(k) and *algorithmB()* uses *getSimilarityNaive()* method which runs in O(k2) where k is the average number of ratings in a user’s list. Note that in these trials we assumed that the targetUser in every data set is the first user given.