CSE 272 HW2 Report:

Ananya Das (adas13@ucsc.edu)

1. **Approach:**
    a. After analyzing the Amazon dataset I decided to build my algorithm using three columns - user-id(reviewerID), product-id(asin), and ratings(overall). This algorithm works on all the 5-core datasets mentioned on the website.
    b. **Processing the data:** I tried a few ways and finally, I converted the JSON file dataset into pandas data frame because of the below reasons:
        i. I dropped the unnecessary columns in one line
        ii. I grouped the data frame by user-id in one line
        iii. I sampled 80% of the dataframe thats grouped by user id to ensure that 80% of each user ratings in the training dataset in one line
        iv. I dropped the training dataset from the dataframe to keep the rest of the dataframe ,20% for testing in one line
    c. After going through the popular algorithms (user-based CF, item-based CF, Slope one, Matrix Factorization) I have implemented the user-based CF using implementations of nearest neighbours, pearson_correlation and some other functions from scratch
    d. I am using NearestNeighbours from sklearn.neighbours to speed up calculating nearest neighbours to items using item-item CF algorithm instead of using user-user CF algorithm
    e. I created a pivot table with ratings as values, user-ids as indices and items as columns of the table
    f. I created two dictionaries iterating through the table : the first one has users as keys and lists of items they rated as list

of values whereas the second one has keys of users and indices of items they rated as list of values

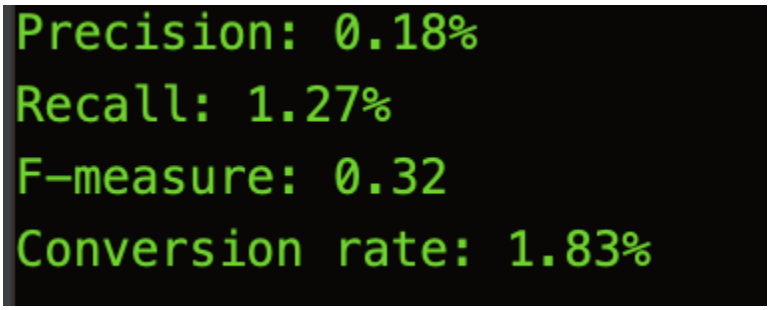
g. Using nearestneighbours function, I calculated the closest neighbours to an item
h. I used the brute force algorithm because it is fast and the cosine metric because it measures similarity between two vectors. The closest the vectors are the more correlated they are and the farther they are, the more uncorrelated they are. With prediction and ground truth I calculated the Mean Absolute Error(MAE) and the root mean square error (RMSE) to evaluate my predictions. Almost in all the runs I made the values of MAE and RMSE are around 1
i. My recommendation method uses the distances between the items and also the indices of the items and after combining both the indices and the items, I created a list of descendingly sorted similarities along with the items they refer to with excluding items that are already rated
   Below is an example of my recommendations:

   A0182108CPDLPRCXQUZQ - [('overall', 'B002IUNLLK'), ('overall', 'B008B68IUY'), ('overall', 'B00AFP86KG'), ('overall', 'B00BJT861Q'), ('overall', 'B00C6PXVEY'), ('overall', 'B00C6Q9Y2G')]
   ('overall', 'B00IVFCKEA') has similarity: 0.8333
   ('overall', 'B008A224EA') has similarity: 0.8333
   ('overall', 'B00IVFCGIA') has similarity: 0.8285
   ('overall', 'B008A22FR6') has similarity: 0.8174
   ('overall', 'B008B68IBI') has similarity: 0.8110
   ('overall', 'B00DQ9WQWM') has similarity: 0.8075
   ('overall', 'B008RO04WK') has similarity: 0.7959
   ('overall', 'B0073RN4PG') has similarity: 0.7959
   ('overall', 'B00C6Q3UNK') has similarity: 0.7892
   ('overall', 'B00C6PVN6C') has similarity: 0.7868

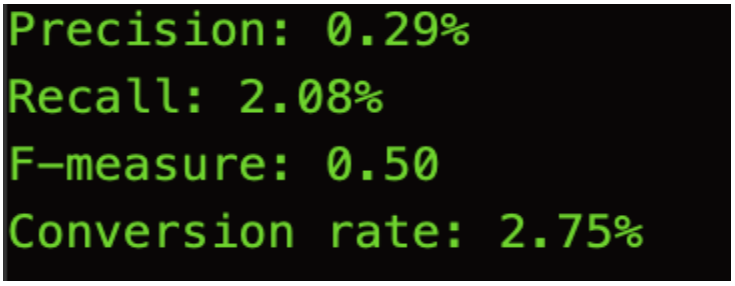
2. **Performance:**
    a. The performance of my algorithm varies drastically when I change the number of nearest neighbours
    b. I ran the algorithm against 2 different datasets and observed that the higher the number of nearest neighbours are the lower my metrics are
    c. Below are the examples of running the algorithm on different datasets:

Recommendation evaluation of Toys and Games:

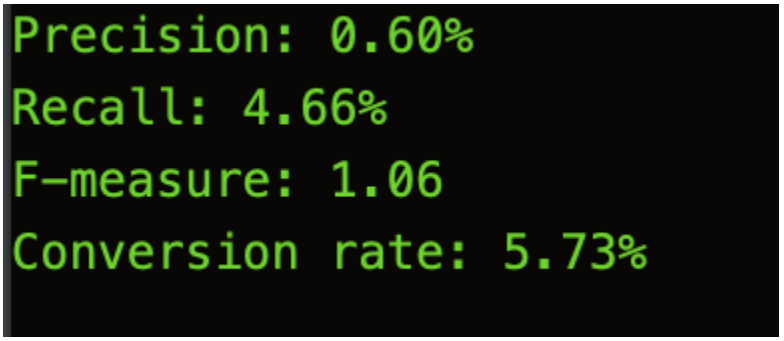
- **n-neighbours=15**



- **n-neighbours=10**



- **n-neighbours=5**



- n-neighbours=2

```
Precision: 0.60%
Recall: 4.05%
F-measure: 1.05
Conversion rate: 5.84%
```

<u>Recommendation evaluation of Health and Personal Care:</u>

- n-neighbours=15

```
Precision: 0.14%
Recall: 0.95%
F-measure: 0.24
Conversion rate: 1.34%
```

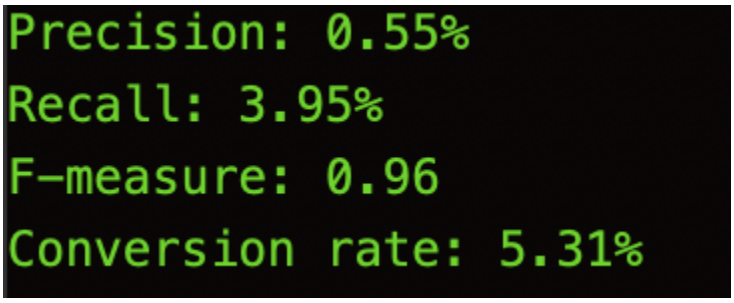
- n-neighbours=10

```
Precision: 0.21%
Recall: 1.47%
F-measure: 0.37
Conversion rate: 1.97%
```

- n-neighbours=5

```
Precision: 0.55%
Recall: 3.95%
F-measure: 0.96
Conversion rate: 5.31%
```

- **n-neighbours=2**



3. GitHub Link:

- Please find below the github link for the repository of assignment 2: https://github.com/ananyadas2607/CSE_272_HW2

- Also find a list of recommendations in the recommendations_for_users.txt file