Amazon Recc. System (CS439)

Project Repo Breakdown:

1. proccess.py
2. algorithm.py
3. algorithmCheck2.py
4. ratings.py
5. reccomendation.py

Steps for process.py:

1. hard code file paths from dhruv/dev machine for product jsons (instruct user to change file paths to where their data lives)
2. bring the dataset into polars??? (arrow tables are nicer to work with) :)
3. have python file split dataset into 80% training and 20% testing chunks
    1. These splits must be random (either bash command subprocess, hadoop, or smth else)
4. will confirm all column names exist in both datasets (we want all the column names from our original dataset)
5. have final func return both df's

Steps for algorithm.py:

1. Matrix Factorization --> SVD or Alternating Least Squares
    1. use Surprise or LightFM for this (python libraries to perform)
    2. scipy.linalg

return MSE and RMSE and return testing data accuracy/ratings scores comparing with what the model predicted.

Steps for algorithmCheck2.py:

we treat this solely as a check and hard code some value to see if the ratings are similar per user (we have the same testing set)

1. item based cf or user based cf
    1. KNN using scipy

return MSE and RMSE and return testing data accuracy/ratings scores comparing with what the model predicted.

Steps for ratings.py:

Ratings python file will house all the control flow for our ratings predictions systems

Steps for reccomendation.py

```
Apriori
```

- **Identify Candidate Items for Recommendation:**
  - For each user, filter out items they have already rated or purchased in the training set.
  - This leaves a set of items that the user has not interacted with, which can be considered as candidates for recommendation.
- **Predict Ratings for Candidate Items:**
  - Use your trained recommendation model to predict ratings for the candidate items for each user.
  - This can be done using your model's prediction function on all items in the candidate set for each user.
- **Rank the Items:**
  - Rank the candidate items for each user in descending order of the predicted ratings.
  - Select the top 10 items from the ranked list as the recommendation list for that user.

Precision:

- For each user, precision is calculated as:
  Precision=Number of recommended items that are in the testing set10\text{Precision} = \frac{\text{Number of recommended items that are in the testing set}}{10}Precision=10Number of recommended items that are in the testing set
- Compute this for each user and then average the results across all users.

2. Recall:

- For each user, recall is calculated as:
  Recall=Number of recommended items that are in the testing setTotal number of items in t he testing set for that user\text{Recall} = \frac{\text{Number of recommended items that are in the testing set}}{\text{Total number of items in the testing set for that

user}}Recall=Total number of items in the testing set for that userNumber of recommended items that are in the testing set

- Compute this for each user and average across all users.

3. F-Measure:

- The F-measure combines precision and recall as: F-measure=2×Precision×RecallPrecision+Recall\text{F-measure} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}F-measure=2×Precision+RecallPrecision×Recall

4. Normalized Discounted Cumulative Gain (NDCG):

- NDCG

print out all of the above
write training and testing as outfiles to a specified output directory as well as the predictions file