Documentation

Overview of Code

1. **app.py**

   This file contains functionality for the web app which serves recommendations to the user. See demo here.

2. **model.py**

   This file contains logic and algorithms powering the recommendations in the webapp.

3. **src/basic_approach/collaborative_filtering.py**

   This file contains an implementation of basic collaborative filtering as mention in class during week 6.

4. **notebooks/neural_network.ipynb**

   This notebook contains an approach using neural networks to predict the user's final rating.

5. **src/**/scraper.py**

   There are various scraper.py scripts used to scrape and assemble the dataset used.

Implementation details

This is our stack:

1. Data Wrangling: pandas, numpy
2. Web Scraping: beautifulsoup, requests, regex
   - Please refer to `scraper.py` for more details
2. Model: scikit-learn, scipy, tensorflow
   - See `requirements.txt` for more
3. Web Framework: flask
   - Run `app.py` on localhost:5000 ```
4. Front End: html & css

This is the data that we managed to scrape:

Content Data

- `all_recipes.csv`
- 1100+ Recipes from
- 460+ Cuisines & Categories

Content Data

- `all_users.csv`
- 55K Users
- 73K Ratings

The website we scraped this data from has much more users and ratings available but this is what we managed to collect with limited amount of time and compute.

We tried the following approaches for our recommender system:

1. Basic collaborative filtering
    a. Suggest recipes that similar users also liked. Similarity is based on what recipes the users have rated and calculated using cosine similarity.

2. Content based filtering
    a. Suggest recipes similar to the recipes that the user liked. Similarly is based on the categories of the recipe and other content based features and calculated using cosine similarity.

3. Matrix Factorization
    a. Use singular value decomposition to discover latent factors that describe users and items. The matrix can be reconstructed and used to predict unobserved user-recipe ratings.
    b. This approach performed the best in terms of test rmse and predicting the ratings the users give to recipes on a held out test set. It is a classic method that performs well on the dataset we scraped.

4. Neural Networks
    a. Use an embedding layer to learn embeddings for users and items. Multiple dense layers are built on top of the concatenation of the embeddings to learn a function that predicts the rating. In addition content based features can be concatenated to the embeddings and used for prediction.

b. It was found that adding further text based features from the title of the recipe slightly improved the test rmse. Overall the test rmse beats the basic collaborative filtering approach but loses to the Matrix Factorization approach.
c. The model was found to perform better when more data was scraped and added to the dataset. It is possible that it may beat the matrix factorization approach if even more data was scraped.

Usage of Software

Webapp

See demo here.

1. Clone the repo (https://github.com/nykznykz/CourseProject)
2. Install requirements.txt
   a. pip install -r requirements.txt

3. Run app.py
   a. python app.py

4. Navigate to localhost:5000 in the browser.

Basic collaborative filtering approach

See this readme for detailed documentation.

Neural Network approach

Run the jupyter notebook and follow along.

Contribution of Group Members

Tanmoy: Creating the webapp (app.py), modeling with SVD and other methods (model.py) and web scraping

Nikolas: Modeling using deep learning approach (notebooks/neural_network.ipynb), comparison of methods, reports & creation of video

Dikra: Web scraping, modeling with basic collaborative filtering approach used in class (src/basic_approach/collaborative_filtering.py)

Self evaluation

Overall we have completed the tasks we set out to do:

1. Scraping real world recipe data
2. Benchmarking what we learned in class with our own methods
3. Creating a webapp to serve actual recommendations to users

We think the following could have been done if more time were available:

1. Scraping more data and constructing a larger dataset
2. Engineering more user and item based features for use together with the neural network