Collaborative Filtering Model

Alexander Bonnet

Grand Canyon University

DSC - 530

Brian Stout

7/24/2024

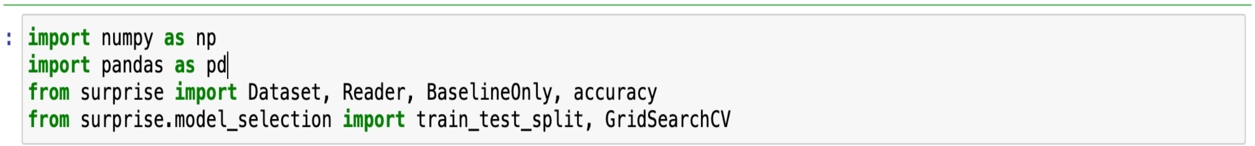
1. Formulate a prediction question that you want to answer by applying a collaborative filtering recommender algorithm.

How can we determine what movies are similar to each other and how can we predict what other users might like based on their previous watch history? How can we then measure the accuracy of these ratings.

2. Search and locate a dataset that is relevant to the question(s) you created in the previous step.

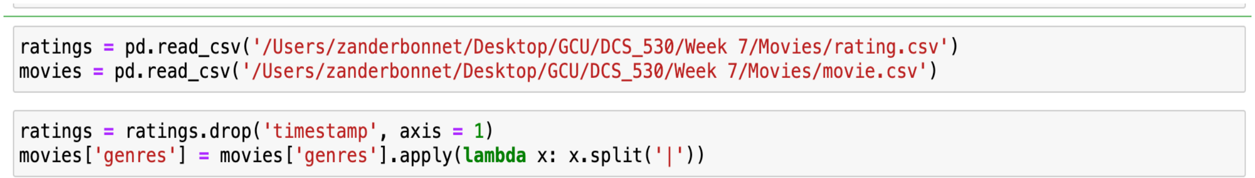
Movielens 20m dataset from Kaggle. Contains a variety of users that rate movies based on a scale of 1 to 5.

3. Import the necessary libraries.



```
import numpy as np
import pandas as pd
from surprise import Dataset, Reader, BaselineOnly, accuracy
from surprise.model_selection import train_test_split, GridSearchCV
```

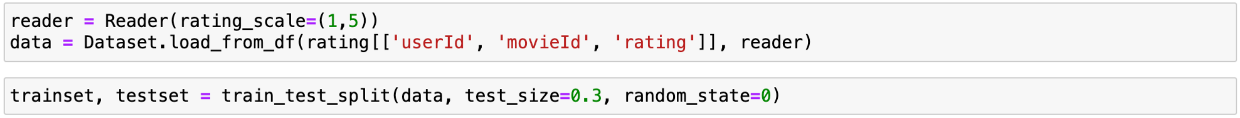
4. Import the dataset and conduct any necessary preprocessing.



```
ratings = pd.read_csv('/Users/zanderbonnet/Desktop/GCU/DCS_530/Week 7/Movies/rating.csv')
movies = pd.read_csv('/Users/zanderbonnet/Desktop/GCU/DCS_530/Week 7/Movies/movie.csv')
```

```
ratings = ratings.drop('timestamp', axis = 1)
movies['genres'] = movies['genres'].apply(lambda x: x.split('|'))
```

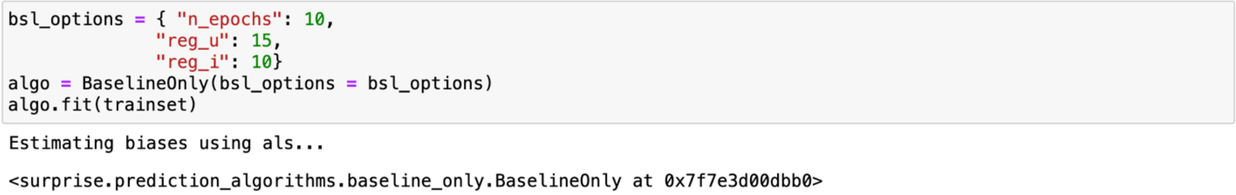
5. Split the data into a training and testing set.



```
reader = Reader(rating_scale=(1,5))
data = Dataset.load_from_df(rating[['userId', 'movieId', 'rating']], reader)
```

```
trainset, testset = train_test_split(data, test_size=0.3, random_state=0)
```

6. Fit the model



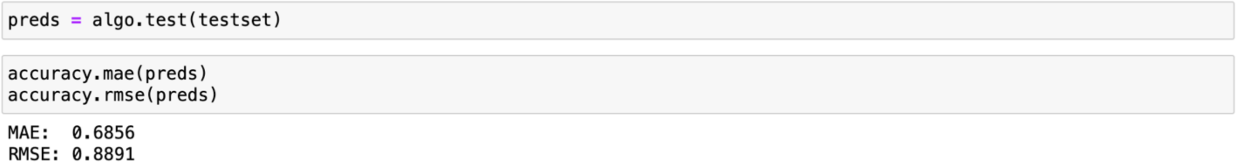
```
bsl_options = { "n_epochs": 10,
               "reg_u": 15,
               "reg_i": 10}
algo = BaselineOnly(bsl_options = bsl_options)
algo.fit(trainset)
```

```
Estimating biases using als...

<surprise.prediction_algorithms.baseline_only.BaselineOnly at 0x7f7e3d00dbb0>
```

I chose to use the default parameters to start.

7. Make predictions and measure the accuracy.



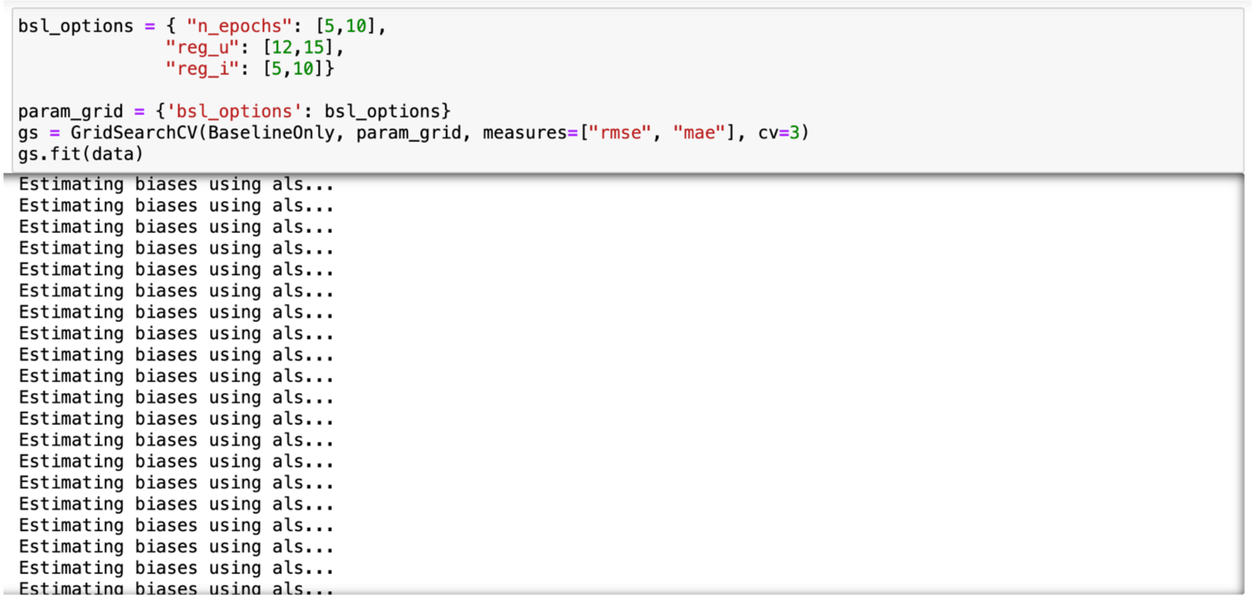
```
preds = algo.test(testset)
```

```
accuracy.mae(preds)
accuracy.rmse(preds)
```

```
MAE:  0.6856
RMSE: 0.8891
```

The model has a Mean Absolute Error of .68. This means that on average the prediction of rating is .68 away from the true rating. This is not great considering that the range of possible values is only from 1 to 5. The Root Mean Squared Error is .889. This does not mean a ton until we can compare this value to other models to evaluate the performance.

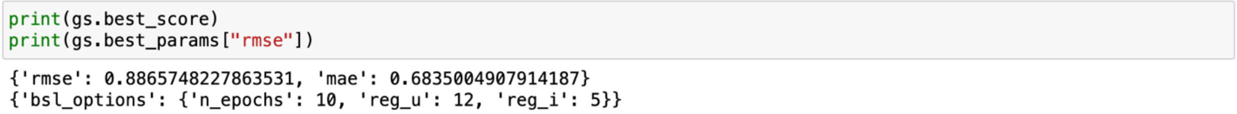
8. Use GridSearchCV to tune the parameters of the model.



```
bsl_options = { "n_epochs": [5,10],
              "reg_u": [12,15],
              "reg_i": [5,10]}

param_grid = {'bsl_options': bsl_options}
gs = GridSearchCV(BaselineOnly, param_grid, measures=["rmse", "mae"], cv=3)
gs.fit(data)
```

```
Estimating biases using als...
Estimating biases using als...
Estimating biases using als...
Estimating biases using als...
Estimating biases using als...
Estimating biases using als...
Estimating biases using als...
Estimating biases using als...
Estimating biases using als...
Estimating biases using als...
Estimating biases using als...
Estimating biases using als...
Estimating biases using als...
Estimating biases using als...
Estimating biases using als...
Estimating biases using als...
Estimating biases using als...
Estimating biases using als...
Estimating biases using als...
Estimating biases using als...
```



```
print(gs.best_score)
print(gs.best_params["rmse"])
```

```
{'rmse': 0.8865748227863531, 'mae': 0.68350049079141 87}
{'bsl_options': {'n_epochs': 10, 'reg_u': 12, 'reg_i': 5}}
```

The optimal parameters of the model were found to be n_epochs = 10, reg_u = 12, and reg_i = 5. These parameters improved our RMSE ever so slightly. The MAE was also very slightly improved. It would be very interesting to explore more values, but the data set is humongous and takes a lot of computational power to run these optimizations.

Reference

GroupLens. (2018, August 15). Movielens 20m dataset. Kaggle. https://www.kaggle.com/datasets/grouplens/movielens-20m-dataset/data