DePaul CSC577

Homework 3

Brian Craft

Github: [https://github.com/priorfire4411/Craft-Repo/blob/master/recommender\_systems/Homework\_3](https://github.com/priorfire4411/Craft-Repo/blob/master/recommender_systems/Homework_3/NaiveBayesRecommender.java)

Youtube: <https://www.youtube.com/watch?v=-byPeCigRmc&feature=youtu.be>

**Naïve Bayes Implementation**

The train method is a for loop that for each user finds:

* Probability of liking an item
* Probability of disliking an item
* Probability of a feature given like
* Probability of a feature given not like

The prediction method takes a user and an item and calculates:

* The constant K
* Probability of like given feature
* Probability of dislike given a feature
* Logliklihood
* Probability of like using the logliklihood
* Predicted rating

**Hybrid Implementation**

I chose to use User KNN as my collaborative filtering recommender with a k of 150. I didn’t get much variance when testing values of k so there isn’t much discussion to be had. The Hybrid Recommender performs the following:

* Setup User KNN and NB
* Train User KNN and NB
* Predict User KNN and NB
* Return weighted average of the predictions

This setup was quite simplistic and the only major adjustments that had to be made to the User KNN were to make the train and predict methods public.

**Hybrid Evaluation**

The results were quite poor. The NDCG@10 never got above .1 and while there was some variance, I wouldn’t consider it anything of note. As the CF ratio increased, the scoring didn’t necessarily do so. By association there wasn’t a clear trend with respect to the Bayes system ratio. At an aggregate, I think the implementation of the hybrid was overly simplistic and made it difficult to access the ratios and the influence they may have.

|  |  |  |
| --- | --- | --- |
| NB Ratio | CF Ratio | NDCG@10 |
| 1 | 0 | .049 |
| .75 | .25 | .030 |
| .5 | .5 | .034 |
| .25 | .75 | .035 |
| 0 | 1 | .045 |