Data Mining Lab 2 – Recommendation

INTRODUCTION

Recommendation is a core task in data mining particularly well suited for making decisions based on an overwhelming amount past decisions, typically made by other people. Typical examples are the buying recommendations at large web shops, or what music to listen to. The typical approach is to look for past decisions made by people that have similar preferences to the decision maker. This is called collaborative filtering and is what you will be doing in this lab.

The data comes from Breeze, a local startup that has their own dating platform. We have pairs of "like"s/"don't like"s and optional matches. A "like" means that a person liked a date suggestion, "don't like" means the opposite. A match means that a like is both ways. Your job is to predict likes/don't likes for new possible matches. Some points about the data:

- It is real data, but anonymized.
- A match is very rare; hence we will only consider the like/don't like feature.

It is your job to analyze this data, find some interesting patterns, and build pipelines for collaborative filtering. We will use two algorithms for this filtering:

- Matrix-based – via non-negative matrix factorization
- Distance-based – via min-hashing and Jaccard distance

You will first implement and test these two algorithms in WebLab, and then apply them to the data from Breeze.

A note on min-hashing: although the data is sufficiently small and it is not necessary to hash the data-rows to speed up computation, we do require that you implement and apply min-hashing. The real value of min-hashing will be apparent when the data becomes too large for your computer's memory. However, for obvious reasons we do not include such a dataset in our labs.

LEARNING OUTCOMES

After completing this assignment, you will be able to:

1. Implement non-negative matrix factorization.
2. Implement min-hashing for quick Jaccard distance computation.
3. Make different recommendations for different types of users.
4. Build a recommendation system pipeline and evaluate its performance.

INSTRUCTIONS

Non-negative Matrix Factorization (NMF) – (25 points)

Implement the **nmf()** subroutine in the provided code base. This function takes as input a matrix X, the number of required components n ("number of features" from the lecture), a maximum number of iterations, and an error tolerance threshold. It returns two matrices A and B (with width/height n) such that AB approximates X. Test your implementation in WebLab. Look in WebLab for further instructions.

Min-hashing – (25 points)

Implement the **compute_signature()** subroutine in the provided code base. This function takes as input a list of *k* hash functions and a list of lists of integers. It should return the minhash signature for the given lists of integer values (the rows/columns corresponding to these values have a value of 1, the other rows/columns have a value of 0), when applying the provided hash functions. The signature should be of size *k*. Test your implementation in WebLab. Look in WebLab for further instructions.

Familiarization (10 points)

Load the data (train with training data, test with test data) into a Jupyter Notebook and understand the data using visualizations. Answer the following questions:

1. What properties of the data do you think are important for your pipeline design? Think of the data sparsity and distribution of labels.
2. Do you see different types of people (in terms of both which id pairs are present and what they like/don't like)?

Make visualizations that support your answers.

NMF-based recommender system – (15 points)

You must apply your NMF algorithm to the Breeze data. Decide what preprocessing to perform such as the removal of outliers, users with very few "like"s or "don't like"s. Decide if you need normalization based on your initialization scheme for matrices A and B in the **nmf()** subroutine. Decide on a recommendation threshold based on the NMF reconstruction and explain your decision.

Importantly: Decide how the loss function (reconstruction error) in the **nmf()** subroutine should treat the "missing" values during optimization. For a recommender system, the "missing" values in X should not be treated as zero entries!

Distance-based recommender system – (15 points)

You must apply min-hashing to build a user-user and/or item-item based recommender system for the Breeze data. Use min-hashing to find the nearest neighbors in terms of Jaccard distance for both rows and columns. The easiest setup is to simply return the rows/columns with equal hash signatures. An optional second step is to compute Jaccard distance for the columns to retrieve a top-k. Use these (user-user, item-item, or both) to find a meaningful estimate (your choice of aggregation) for a new row-column pair. Play with the number of hash functions and neighbors and report your findings.

Bonus – (10 points)

Try to outperform our baseline on the Kaggle competition! Feel free to use a distance-based method, a factorization-based method, or a combination. You are not allowed to use scikit-learn (or any other machine learning) libraries for your submission. Your solution has to run your own code for making its predictions.

RESOURCES

See Brightspace for details.

PRODUCTS

A Jupyter Python notebook for all parts of the assignment. The word count should not exceed **1600 words**. You are not allowed to include libraries other than numpy, scipy, and pandas. **Do not include the data with your notebook!** The data will be available on the evaluation machine.

The notebooks will be assessed using the below criteria.

ASSESSMENT CRITERIA

The assignment will be reviewed by your peers, and you are expected to individually review 2 reports. The estimated time you should spend on a review (including code review) is 1 hour. The login details will be provided in the week of the deadline.

Knockout criteria (will not be evaluated if unsatisfied):

Your code needs to execute successfully on computers/laptops of your fellow students (who will assess your work). You may assume the availability of 4GB RAM. Please test your code before submitting. In addition, the flow from data to prediction must be highlighted, e.g., using inline comments.

Submissions submitted after the deadline will not be graded, **deadlines are strict!**

The report/code will be assessed using these criteria:

Criteria	*Description*	*Evaluation*
NMF	*Test suite score on WebLab.*	*0-25 points*
Min-hashing	*Test suite score on WebLab.*	*0-25 points*
Familiarization	*Shows the behavior of the Breeze data to answer the questions in the instructions. Provides useful input for further tasks.*	*0-10 points*
NMF-based pipeline	*NMF is used correctly, with the correct loss function and converges. Explanations or cross-validation is presented for the choices of number of components, max iterations and error tolerance.*	*0-15 points*
Distance-based pipeline	*MinHashing is used correctly, the performance is analyzed, a sensible aggregation is used and supported by experiments. The number of hash-functions and neighbors are set sensibly.*	*0-15 points*
Report and code	*The recommender prediction flow is clearly described, including preprocessing and post-processing steps.*	*0-10 points*
Bonus	*Performance on Kaggle.*	*0-10 points*

Your total score will be determined by summing up the points assigned to the individual criteria. Your report and code will be graded by the teacher and assistants, and the peer reviews are used as guidance.

110 points (including bonus) can be obtained in each lab assignment. 330 points (including bonus) can be obtained in the 3 lab assignments. The total number of obtained points will be divided by 30 to determine the final lab grade.

The lab grade counts for 30% of the total graded for the course.

You will receive a penalty of 10 points for each peer review not performed. Significantly different reviews will be subject to investigation. If deemed badly done by the teacher or TA, you will also receive 10 penalty points.

SUPERVISION AND HELP

We use Mattermost for this assignment. Under channel Questions Lab2, you may ask questions to the teacher, TAs, and fellow students. It is wise to ask for help when encountering start-up problems related to loading the data or getting the expected output from Numpy. Experience teaches that students typically answer within an hour, TAs within a day, and the teacher the next working day. Important questions and issues may lead to discussions in class.

Lab sessions are Friday's 13:45-17:45 physically in different locations at campus (check mytimetable for locations). Please see Brightspace for details.

SUBMISSION AND FEEDBACK

Submit your work in Brightspace, under assignments. Also submit it on peer.tudelft.nl. Within a day after the deadline, you will receive several (typically two) reports to grade for peer review as well as access to the online peer review form. You have 5 days to complete these reviews. You will then receive the anonymous review forms for your groups report and code.

There is the possibility to question the review of your work, up to 3 days after receiving the completed forms. You should do so via the response function on peer.tudelft.nl.

In case of a failing grade for a lab assignment, you have the opportunity to resubmit your work on Brightspace until one week after grade notification.