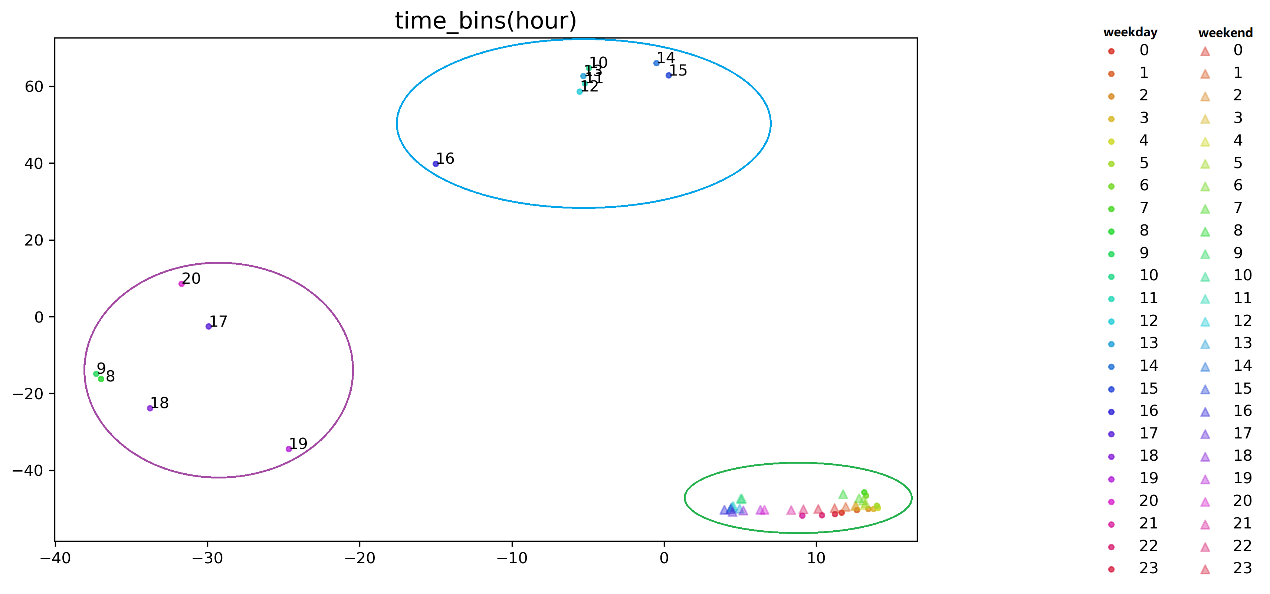
Homework 3

Representation Learning & Recommender Systems

1. **Experiment-1 Representation Learning Reconstruction embedding**
2. Algorithm

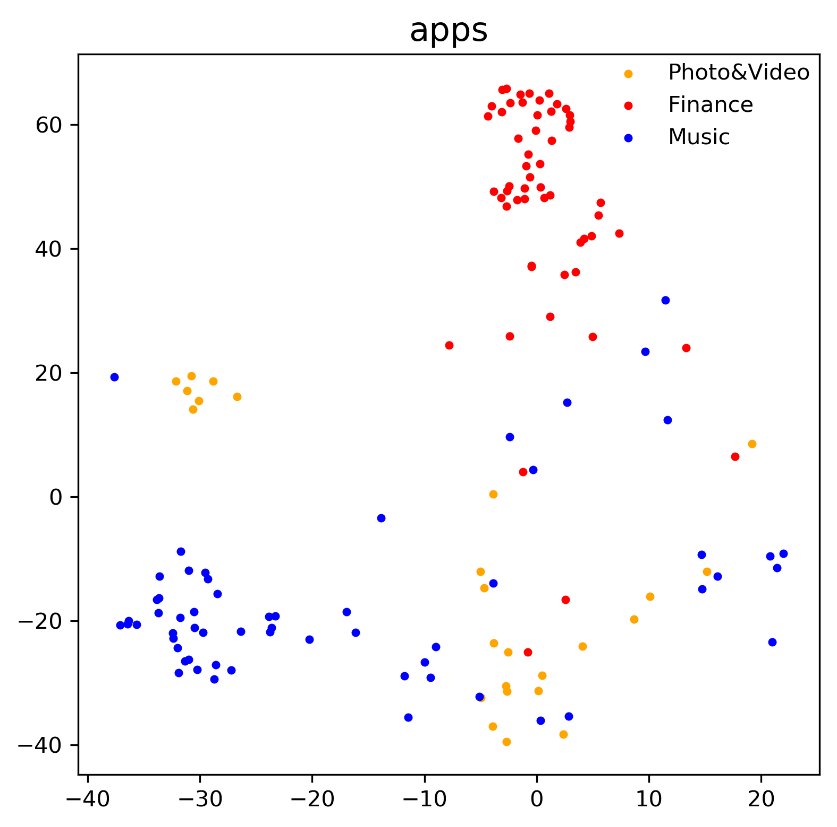
The optimization target is . R is the set of all app usage records. Each record *r* contains 3 units *e*, the time-bin, the base station, and the used app. And the . We approximate with . And . ,. is the D-dimensional embedding of unit 2. And we can use stochastic gradient descent to update each .

1. Virtualization results
2. Plot the embeddings of different time bins (hours).



Interpretation /discussion: as seen from the plot, time-bins are perfectly separated into 3 clusters after embeddings according apps usages. The purple cluster includes morning rush hours (8-9) and evening rush hours (18-20) in weekday, which may represent rush time; The blue cluster includes working hours in weekday, which may represent working time; The green cluster includers all weekend hours and night hours in weekday, which may represent leisure time.

1. Plot the embeddings of apps of the category “Video”, “Finance”, “ Music.



Interpretation /discussion : as seen from the plot, three kinds of apps are well separated, especially “Finance” to “Music” and “Photo&Video” to “Music”. But “Photo&Video” and “Music” are separated not so perfectly, which may due to “Photo&Video” and “Music” are similar, both using for entertainment.

1. Classification results

Focus on apps of “Photo&Video” and “Finance” and use SVM to divide them based on their embeddings.

The obtained accuracy: **0.679**

Interpretation /discussion：the reasons why obtained accuracy is not high are 1) the number of training samples are only 92 and 2) “Photo&Video” samples are much less than “ Finance”.

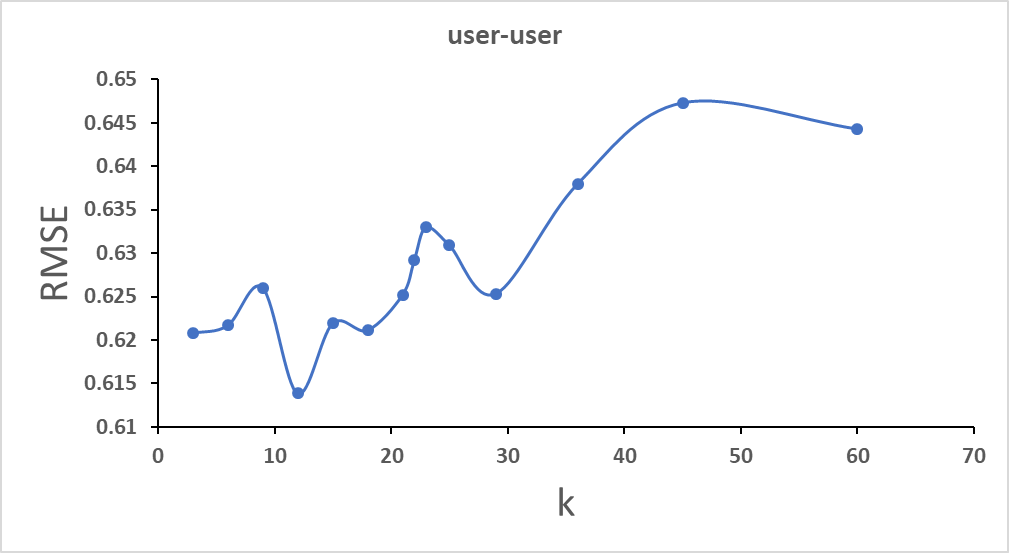
1. **Experiment-2 Use Collaborative Filtering to Recommend Apps to Users**
2. User-User Collaborative Filtering
3. Algorithm

we construct a matrix *r* for user(*u*)-app(*l*), and predict , , is the set of k users most similar to *u* .

1. results

Calculate the RMSE RMSE=, is the predicted value, 𝑛 is the number of predicted user app pair. Due to the time , I only use 0.2% user-app pair for plotting, then according the curve, I set k=12 and use 2% user-app pair to predict, and get the RMSE: **0.617**.

Plot the RMSE-𝑘 curve.



1. Interpretation /discussion :

as seen from RMSE-𝑘 curve , when k =12, RMSE has a minimum , and RMSE increases with the increasing of 𝑘 when k more than 12.

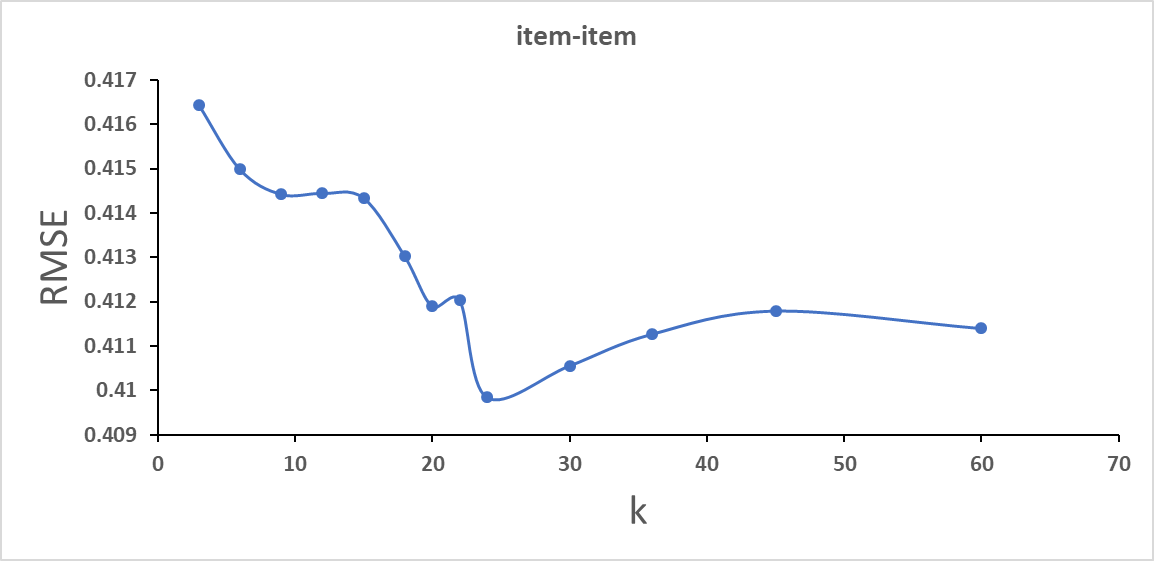
1. Item-Item Collaborative Filtering
2. Algorithm

we construct a matrix *r* for user(*u*)-app(*l*), and predict , , is the set of k apps most similar to *l* .

1. results

Calculate the RMSE RMSE=, is the predicted value, 𝑛 is the number of predicted user app pair. Due to the time , I only use 0.02% user-app pair for plotting, then according the curve, I set k=24 and use 0.2% user-app pair to predict, and get the RMSE: **0.297**

Plot the RMSE-𝑘 curve.



1. Interpretation /discussion : as seen from RMSE-𝑘 curve , RMSE decreases a little with the increasing of k ; When k=24, RMSE has a minimum.
2. Matrix Factorization (MF) based Collaborative Filtering
3. Algorithm

we construct a matrix *r* for user(*u*)-app(*l*), and predict by ,

are 𝑘 dimensional latent vectors for user 𝑢 and app 𝑙 . are first initialized from SVD, and then updated by solving the optimization problem: . We can use stochastic gradient descent to update .

1. results

Calculate the RMSE RMSE=, is the predicted value, 𝑛 is the number of predicted user app pair. Due to “MemoryError” , I test 2% user-app pair in the dataset and get RMSE：**190.46 ,** elapsed time: ~2000s

•Data processing code

train.py ; trace\_handler.py ; embed.py

•Collaborative filtering code

recommender\_sysytem\_CF.py

•Other analysis code

DimReduce.py ; classify.py ; visualizing.py