Technical Review

Recommendation tool of dining places

Group Members: June Yang, Quoc Dung (Daniel) Cao, Runqiu Hu, Zichen Liu

Background

- What is a recommendation system: A recommendation system suggests relevant items to users (vice versa). Two aspects of designing a recommendation system are speed and accuracy.
- What we are doing: Build a recommendation system based on Yelp! Dataset.
    - Recommend dining places to *known* users
    - Recommend dining places to *new* users
    - (vice versa for businesses)
- Modules we are comparing:
    - LightFM
    - Scikit Learn
    - Self written module
    - Surprise

Recommendation system

scikit learn

Machine Learning with Scikit-Learn

light fm

surprise

LightFM: A hybrid recommendation algorithm

- A hybrid recommendation system that can use both collaborative filtering and content-based filtering for making recommendations.
- Advantages:
  - Can target *known* users as well as *new* users - enabled by the hybrid algorithm
  - Uses matrix factorization, powerful and fast
  - Streamlined, easy to implement
- Performance of collaborative filtering using LightFM:
  - Training set AUC: 0.992; test set AUC: 0.946
  - Train precision at 1: 0.30; test precision at 1: 0.05
  - Train recall: 0.24; test recall: 0.11
- More to be done with hybrid: add user/business features

scikit learn:

`sklearn.feature_extraction.text.`CountVectorizer

The **sklearn.feature_extraction.text** submodule gathers utilities to build feature vectors from text documents.

CountVectorizer package Convert a collection of text documents to a matrix of token counts.

scikit learn:

`sklearn.feature_extraction.text.`CountVectorizer

Example:

List of words: ['and', 'document', 'first', 'is', 'one', 'second', 'the', 'third', 'this']

List of sentence : [

'This is the first document.',

'This document is the second document.',
'And this is the third one.',

'Is this the first document?'

]

Return:

```
[[0 1 1 1 0 0 1 0 1]

 [0 2 0 1 0 1 1 0 1]

 [1 0 0 1 1 0 1 1 1]

 [0 1 1 1 0 0 1 0 1]]
```

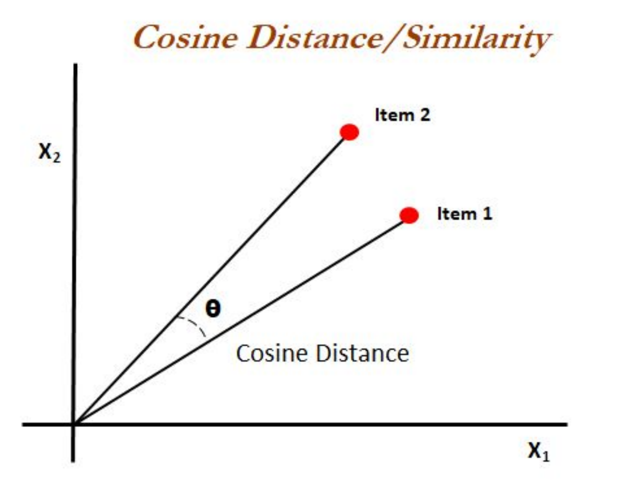
scikit learn

`sklearn.metrics.pairwise`.cosine_similarity

The **sklearn.metrics.pairwise** submodule implements utilities to evaluate pairwise distances or affinity of sets of samples.

Cosine_similarity package compute cosine similarity between samples in X and Y.

$$k(x, y) = \frac{xy^\top}{\|x\| \|y\|}$$



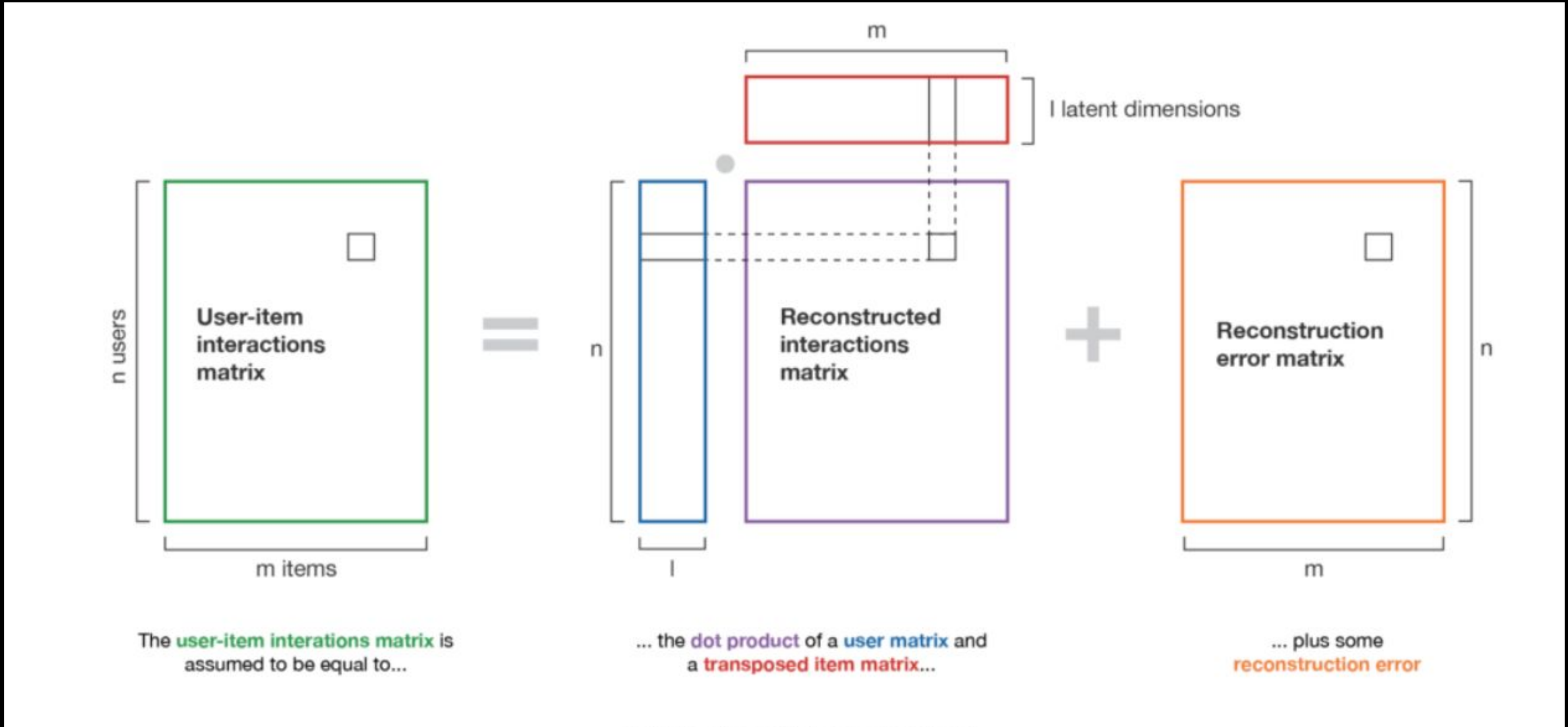
Customized module: Matrix Factorization with Stochastic Gradient Descent

- Implement matrix completion with Alternative Least Square algorithm.
- Advantages:
  - Transparency to understand the algorithm
  - Ability to extract the intermediate information, e.g. loss
  - Flexibility to tune hyper-parameters, e.g. learning rate, regularization
- Disadvantages:
  - Time consuming to implement the algorithm
  - Resource consuming: RAM and CPU
  - Performance may be not as stable as pre-built modules

Why we picked LightFM:

- To deal with the cold-start problem.
- scikit learn better with content-based filtering; surprise better with collaborative filtering. LightFM easy to do both.
- Nice performance.
- Less coding work...

Questions


m
l latent dimensions
n users
User-item interactions matrix
m items
n
l
Reconstructed interactions matrix
Reconstruction error matrix
n
m

The user-item interations matrix is assumed to be equal to...

... the dot product of a user matrix and a transposed item matrix...

... plus some reconstruction error

Illustration of the matrix factorization method.

From: https://towardsdatascience.com/introduction-to-recommender-systems-6c66cf15ada

Go Back