Introduction to Recommender Systems, with an Emphasis on Collaborative Filtering Methods

James Henegan
Machine Learning Final Project

Fall Semester, 2019

Overview

Terminology: User and Item

Recommender systems are based on the assumption that it is possible to infer a consumer's interests by utilizing various sources of data related to the consumer (e.g., purchasing history, preferences, etc.).

Once the interests of the consumer have been estimated, recommendations can be given.

We say that the user is the person who receives the recommendation.

We say that the item is the product that is being recommended.

Examples

User	*Item*
Spotify Account Holder	Song
YouTube Account Holder	YouTube Video

Terminology: Feedback and Ratings

As noted on the previous slide, recommender systems use various sources of data to infer user interests.

These sources of data are often given to the recommender system by the user in the form of feedback.

For example, a user may indicate their attitude towards an item by providing a rating for that item.

Intuitively, if a user gives an item a high rating, then this indicates that the item is of high value to the user.

The Utility of Recommender Systems

For the Business

A good recommender system can help drive sales for a business.

For the User

Ideally, a recommender system helps a user find (heretofore unknown) items that will be of high value to the user.

Formulation of the Problem: Prediction

There are two main ways of formulating the recommendation problem.

In the prediction version of the problem, the goal is to predict the rating value for a user-item combination.

Here, we assume that some training data is available, indicating user preferences for (past, previously encountered) items.

Example. Given the following ratings matrix, can we predict User 5's rating for Item 4?

	Item 1	Item 2	Item 3	Item 4
User 1	5	0	3	4
User 2	1	4	?	1
User 3	4	1	4	3
User 4	0	?	1	0
User 5	5	1	3	?

Tthis problem is sometimes referred to as the *matrix completion problem.*

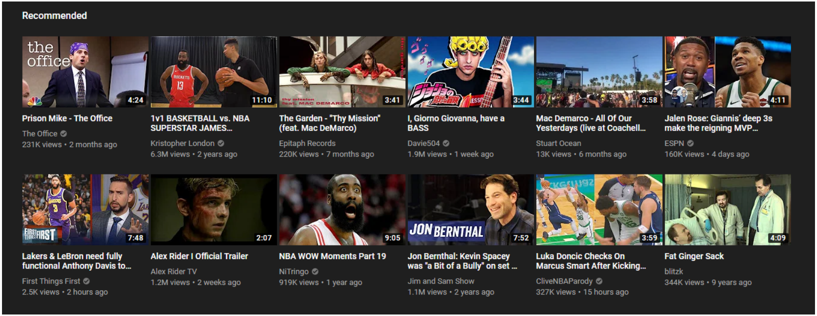
Formulation of the Problem: Ranking

Often times, it is neither necessary nor practical to predict ratings for all of the possible user-item combinations.

Instead, a recommender system may recommend the top-k items for a particular user.

This version of the problem is called the top-k recommendation problem.

Example. Here, YouTube has recommended 12 videos to me, based on my past behavior.



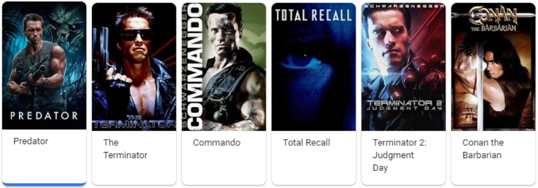
Recommendation Criteria

Broadly speaking, top-k recommendations may be based on the following concepts.

- **Relevance**. Is the item relevant/interesting to the user?
- **Novelty**. Is the item new to the user?
    - *Example.* If one of my favorite musicians releases a new song, then Spotify might recommend this song to me.
- **Serendipity**. To what extent would a user be surprised by a recommendation for a particular item?
    - *Example.* Spotify might recommend a song by an musician that I've never heard of before, but one who is somehow similar to one of my favorite musicians.
- **Increasing Recommendation Diversity**. The user may get bored if similar items are recommended again and again.
    - *Example.* On the previous slide, YouTube suggested several basketball videos to me, but they also suggested non-basketball videos.

Basic Models of Recommender Systems

- **Collaborative Filtering Methods**. Based on information provided by several users. We will discuss these methods in more detail shortly.
- **Content-Based Recommender Systems**. Based on descriptive attributes of items. *Example.* If a user likes the movie *Predator* [attributes: ("action", "Schwarzenegger")], then the user might also like *The Terminator*.



- **Knowledge-Based Methods**. These are often based on *user requirements*. Useful for items that are not bought very often. *Example.* I need a heating technician in the Jackson-metro area who can work with floor furnaces.
- **Hybrid and Ensemble-Based Recommender Systems**. The methods mentioned above can be combined.

Collaborative Filtering Methods

For the rest of the presentation, we will focus on Collaborative Filtering Methods. These methods are based on data involving user-item interactions. The data may be represented in the form of a ratings matrix.

	Item 1	Item 2	Item 3	Item 4
User 1	5	0	3	4
User 2	1	4	?	1
User 3	4	1	4	3
User 4	0	?	1	0
User 5	5	1	3	?

These may be difficult to implement because, often times, the underlying ratings matrices are sparse.

The ratings which have been specified are called observed ratings.

Meanwhile, we can use the terms unobserved or missing to refer to the the unspecified ratings.

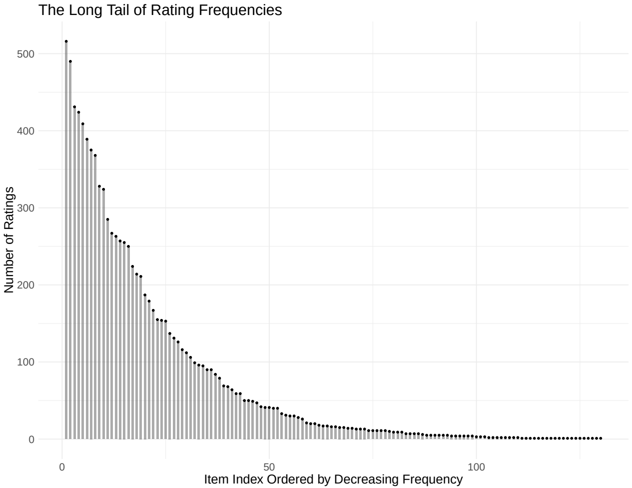
Types of Ratings

The data in a ratings matrix can be continuous, interval-based, ordinal, or binary.

In addition to these types of data, recommender systems can also work with unary ratings. Here, a user has the ability to indicate a “positive” response to an item, but there is no mechanism in place that would allow the user to indicate a “negative” response. Unary ratings can be given either explicitly or implicitly.

Long-Tail Property

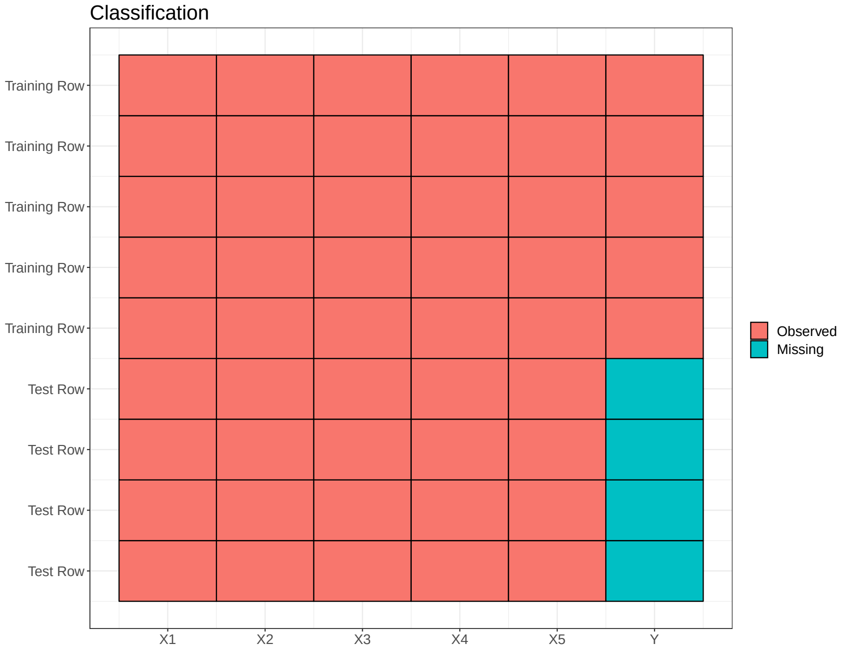
Often times, the distribution of ratings in a real-world ratings matrix has what is known as a long-tail property. Here, only a small fraction of the items are rated frequently. The vast majority of items are rated rarely.

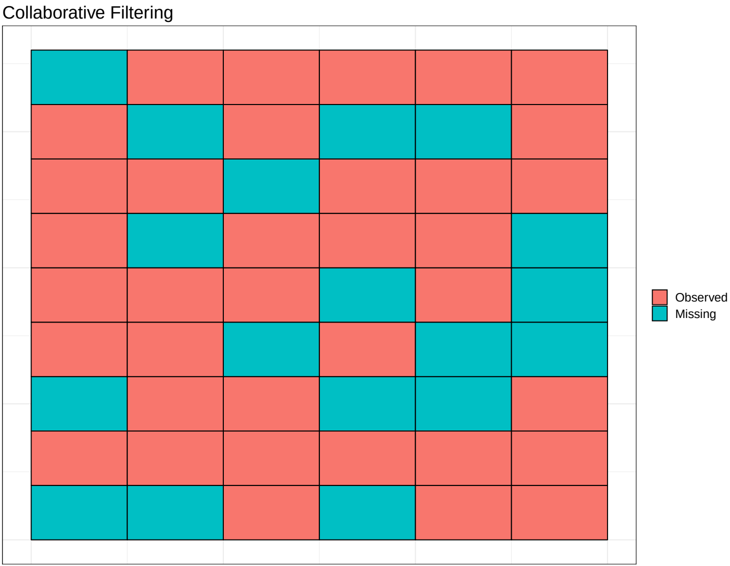


Collaborative Filtering and Data Science / Biostatistics

Missing Value Analysis. There is a close relationship between Collaborative Filtering Models and Missing Value analysis. Traditionally, Missing Value analysis deals with the imputation of entries in an incomplete matrix. We can view Collaborative Filtering as a Missing Value problem, one in which the data matrix is very large and sparse.

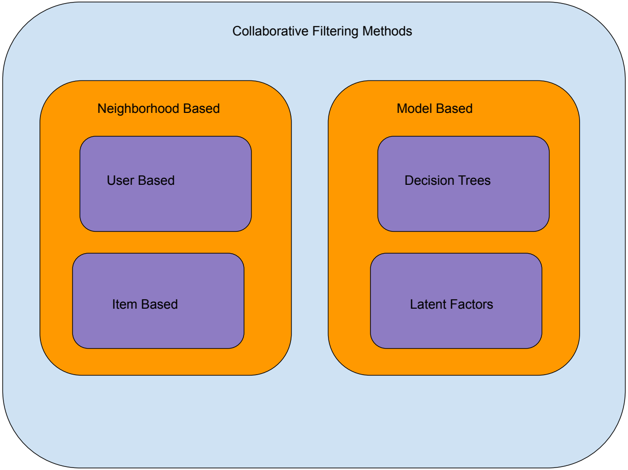
Regression and Classification. It is possible to view the Collaborative Filtering problem as a generalization of a regression/classification problem. For example, in a classification problem, there is a clear distinction between the "training rows" and the "testing rows", just as there is a clear distinction between the "independent variable" and the "dependent variables." In a Collaborative Filtering problem, these distinctions don't exist.


Classification
Training Row
Training Row
Training Row
Training Row
Training Row
Test Row
Test Row
Test Row
Test Row
X1
X2
X3
X4
X5
Y
Observed
Missing


Collaborative Filtering
Observed
Missing

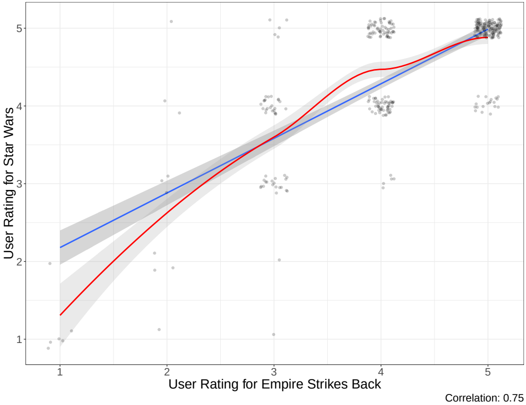
Collaborative Filtering Methods

Broadly speaking, Collaborative Filtering Methods can either be *Neighborhood-based* or *Model-based*.



Collaborative Filtering Methods

Whether neigborhood based or model based, the main idea behind collaborative filtering methods is that the unobserved ratings can be predicted by taking advantage of the fact that the observed ratings are often highly correlated across various users and items.



Neighborhood-Based Collaborative Filtering Problems

Here, the ratings of user-item combinations are predicted on the basis of their so-called neighborhoods.

These neighborhoods could be constructed in terms of either users or items.

- **User-based collaborative filtering**. To predict ratings for a *target user* A, we would first gather up all the ratings provided by a *peer group*: users who are "similar" to A. We would then use these ratings to predict the ratings for A.
- **Item-based collaborative filtering**. Suppose we wanted to predict the rating of item B by user A. Then we would gather up all the ratings that A gave for items that were similar to B. We would then use these ratings to predict the ratings for A.

Model-Based Collaborative Filtering Problems

We can apply decision trees to the collaborative filtering problem, but, when we do so, we are faced with two issues.

- There is no clear distinction between the dependent and independent variables.
- The ratings matrix is extremely sparse.

It is easy to work around the first issue by constructing a decisions tree for each item (or for each user).

It is possible to work around the second issue by taking advantage of certain dimensionality reduction methods. These can take a sparse ratings matrix M and return a dense, low-dimensional representation of M in terms of latent factors. This low-dimensional representation can be computed with either Principal Component Axis-type methods or Singular Value Decomposition-type methods.

References

- Aggarwal, Charu C. *Recommender Systems*. Springer, 2016.
- Kane, Frank. *Machine Learning, Data Science and Deep Learning with Python.* Udemy Course.