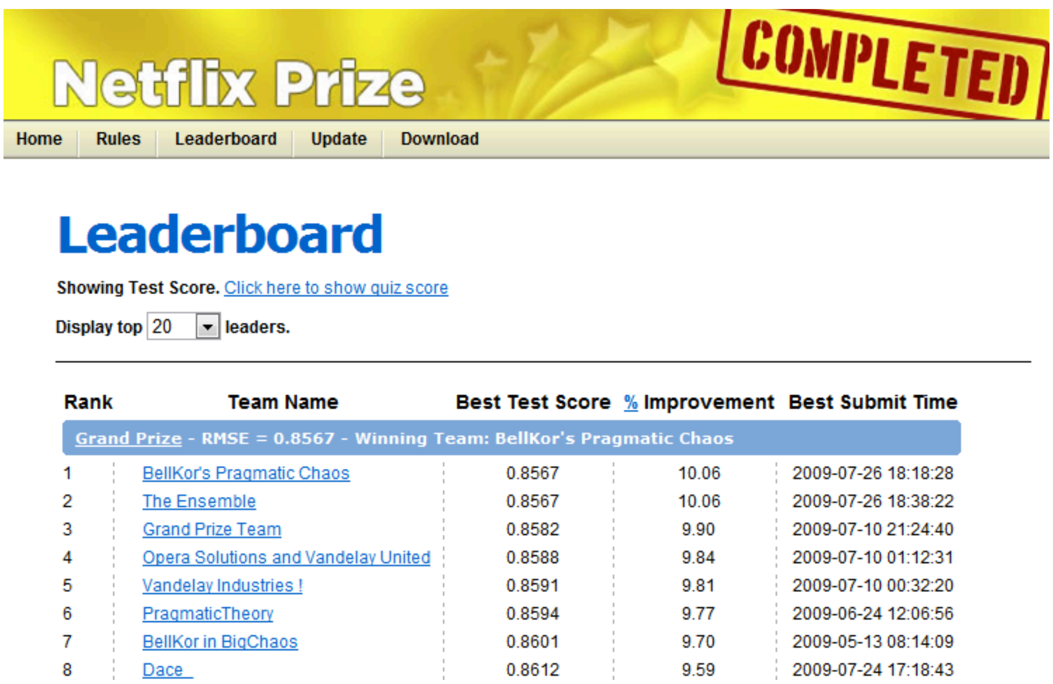
Week 6

From the Expert: Recommender Systems

The objective of recommendation systems or recommender systems (RS) is to assist or guide the users to items (movies, books, songs, jobs, friends, etc.) according to each individual preference. RS can help users or customers to find interesting things, reduce information overload, explore the options, discover new things, and increase customer loyalty and satisfaction. For providers, benefits of utilizing RS include increasing sales, profit, conversion rates, click through rates, promotion opportunities, persuasive opportunities, and even learn more about users in Jannach & Friedrich's (2013) *Tutorial: Recommender Systems*. Information retrieval can be viewed as the initiation of the RS.

The well-known recommendation systems are Amazon and Netflix. Especially, Netflix offered a million dollars to the team that can improve the performance of their Netflix movie recommendation systems called, Cinematch, by 10% root mean squared error (RMSE). Koren (2009) provides the algorithm details for winning the Netflix grand prize.



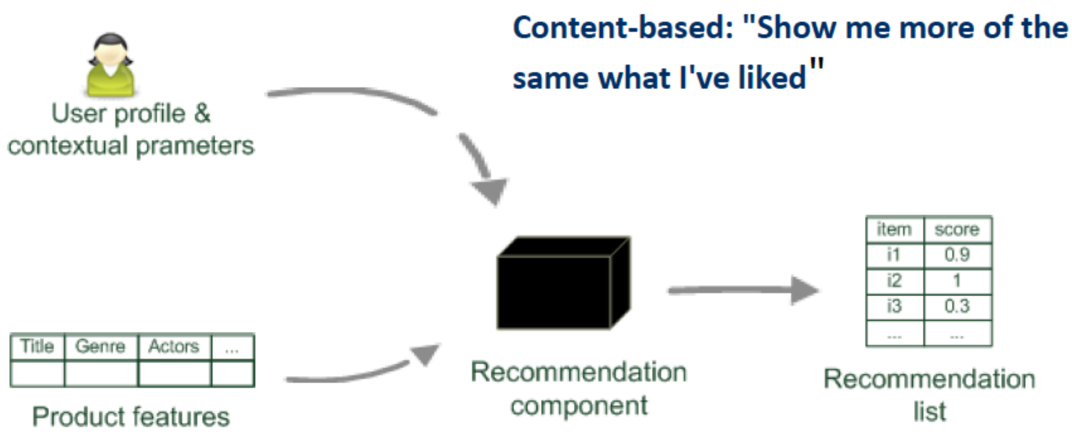
Rank	Team Name	Best Test Score	% Improvement	Best Submit Time
1	BellKor's Pragmatic Chaos	0.8567	10.06	2009-07-26 18:18:28
2	The Ensemble	0.8567	10.06	2009-07-26 18:38:22
3	Grand Prize Team	0.8582	9.90	2009-07-10 21:24:40
4	Opera Solutions and Vandelay United	0.8588	9.84	2009-07-10 01:12:31
5	Vandelay Industries !	0.8591	9.81	2009-07-10 00:32:20
6	PragmaticTheory	0.8594	9.77	2009-06-24 12:06:56
7	BellKor in BigChaos	0.8601	9.70	2009-05-13 08:14:09
8	Dace	0.8612	9.59	2009-07-24 17:18:43

Source: http://techblog.netflix.com/2012/04/netflix-recommendations-beyond-5-stars.html

The ideal RS should be able to recommend items, which users truly like but not even necessarily knowing about the items' existence.

Two main types of RS are content-based and collaborative filtering (CF).

In *Content-based*, items are recommended to users based on information (attributes) about items. For example, movie attributes consist of title, genre, actors, producer, etc. In addition, the user profile that described the user preferences (value and weight for each attribute) will be exploited. There is no need for community data only the comparison between items. The drawbacks of this method are comprised of cold start (no information) for new items and being domain specific.



Source: Content-based paradigm (Jannach & Friedrich, 2013)

Typically, content-based aims to find and rank text documents using extracted words (keywords). Term frequency-Inverse document frequency (TF-IDF) is the standard measure for evaluating how important a word is to a document in a collection or corpus. The importance of word increases as the number of times a word appears in the documents; nonetheless it is offset by the frequency of the word in the corpus. This purpose of the offset is that some words are more common than others. TF-IDF and its variation are commonly employed in search engines for scoring and ranking a document relevancy to the user query.

For a given keyword i and a document j

$$TFIDF(i,j) = TF(i,j) * IDF(i)$$

Let $freq\ (i,j)$ be the number of occurrences of keyword i in document j
$\max\ \_occ\ (i,j)$ be the maximum number of occurrences of another keyword of j

$$TF(i,j) = \frac{freq(i,j)}{\max\ \_occ(i,j)}$$

Let N be the number of all recommended documents
$n(i)$ be the number of documents in which keyword i appears

$$IDF(i) = \log\frac{N}{n(i)}$$

Collaborative filtering (CF) is one of the RS techniques and is utilized widely on e-commerce web sites. The basic idea is that users rate the items explicitly (ask users to rate or rank items) or implicitly (observe users' behaviors) and these ratings will be used to recommend items assuming that the users' tastes remain unchanged over time. In other words, CF utilizes the preferences of similar users to recommend items. Since CF is not domain specific, any product types can use the same algorithm.

There are 2 main types of CF: User-based CF and Item-based CF.

User-based CF relies on other users who have similar preferences. Items are recommended based on users common interest. For example, look for items that users similar to you like (but is a new item to you). This approach utilizes preference data as input. Then, similar users are identified based on similarity metrics. Finally, a list of items, together with the predicted rating is produced as output.

Item-based CF uses item contents to find the similarity between items. Items are recommended based on similarity of those to the item that a user likes. For example, look for items that are similar to item X. Amazon is a well-recognized service provider for the item-based approach (i.e. what would you like to buy given previous purchases)

The limitations of CF can be summarized as follows:

- The "cold start problem"
  This usually happens with new service/start-up companies where there is no information of similar users to begin with. One alternative is utilized the content-based filtering first. At the end, there will be enough data for CF.
- Sparse matrix
  The matrix of user and items can be tremendously huge. In reality, customers do not rate them all since they have seen only a small fraction of the movies. As a result, we will get a sparse matrix.
- People are not always good at rating things
  Each person gives rating differently. Some like to give high rate, neutral rate or low rate. The comparisons of these ratings can be difficult and/or misleading. Instead of ask for rating directly, observing the user behavior (i.e. click, time spent on page, download) can be used as an alternative.
- A single account can relate to many users. For instance, a single father's account is available for a daughter, himself and his wife.

Practically, good recommendations may measured by total sale, click through rates, customer return rates, customer satisfaction and loyalty, etc.

Different recommendation approaches (i.e. content-based, CF) can be combined to create a hybrid recommendation system by average the rating of all the approaches. In addition, one of the most powerful algorithms is known as the *ensemble method*, in which a number of independent predictive models are combined by voting or averaging among the models. The goal is to build an effective learning that improves the overall performance. For instance, the team Bellcore, who won NetFlix grand prize, employed machine learning techniques and statistics (i.e. restricted Boltzmann Machines, and Singular Value Decomposition) in their ensemble recommendation system.

To measure the similarity between objects (i.e. users, items), these similarity metrics are applied.

Similarity Metrics (Tan, 2006)
Given d : distance between two objects
$x_{k,}y_k$: objects with k attributes (components)
n : number of dimensions
r : parameter

- *Minkowski* distance can be formulated as

$$d(x,y) = \left(\sum_{k=1}^{n} |x_k - y_k|^r \right)^{1/r}$$

- *Hamming Distance* (or city block) is the Minkowski distance where $r = 1$.

$$d(x,y) = \sum_{k=1}^{n} |x_k - y_k|$$

- *Euclidean Distance* (for numeric data) is the Minknowski distance where $r = 2$. The lower distance number signifies the stronger similarity.

$$d(x,y) = \sqrt{\sum_{k=1}^{n} (x_k - y_k)^2}$$

- *Cosine Similarity (vector-based similarity)*

This is one of the most common measures for document similarity. The similarity between documents is measured using the angle between them. If the angle between x and y is zero degree, the value of cosine would be 1. This means that x and y are similar except the length. However, if the angle between them is 90 degree, the value of cosine would be 0. This means that x and y do not share any word terms. Here, the documents are represented as vectors and each attribute is referred to the term (word) frequency appeared in the document. The essential point is that this measure ignores 0-0 matches (no shared term) and can handle non-binary vectors.

Given two document-vectors x and y, the Cosine Similarity can be formulated as:

$$\cos(x,y) = \frac{x.y}{||x||||y||}$$

The numerator is the vector dot product (x.y). The denominator $||x||||y||$ are the length of vector x, y consecutively. Therefore, we get:

$$\cos(x,y) = \frac{\sum_{k=1}^{n} x_k\, y_k}{\sqrt{\sum_{k=1}^{n} {x_k}^2}\sqrt{\sum_{k=1}^{n} {y_k}^2}}$$

- *Pearson's correlation*
This is the measure of linear relationship between the attributes of objects. The objects can be binary or continuous variable. Let x and y are two data objects. Pearson's correlation can be defined as:

$$corr(x,y) = \frac{covariance(x,y)}{standard\_deviation(x) * \ standard\_deviation(y)}$$

$$corr(x,y) = \frac{s_{xy}}{s_x s_y}$$

where,

$$s_{xy} = \frac{1}{n-1}\sum_{i=1}^{n}(x_k - \overline{x})(y_k - \overline{y})$$

$$s_x = \sqrt{\frac{1}{n}\sum_{k=1}^{n}(x_k - \overline{x}\,)^2}$$

$$s_{xy} = \sqrt{\frac{1}{n}\sum_{k=1}^{n}(y_k - \overline{y}\,)^2}$$

The Pearson's correlation is in the range -1 and 1. A correlation of 1 indicates a positive linear relationship between x and y. The negative relationship has the correlation value of -1. Pearson does not consider number of shared items so it is ideal when many ratings are overlap.

- *Tanimoto* coefficient is the number of features in common for both x and y to total number of features. The value is between 0.0 and 1.0. The value of 1.0 signifies match, and the value moves close to 0.0 as the common items decrease.

$$Tanimoto\ coefficient = \frac{x\ \cap y}{x + y - (x \cap y)}$$

To measure similarity for binary attributes, Simple matching coefficient and Jaccard coefficient are common:

- *Simple matching coefficient (SMC)*
  Let x and y be two objects with n ***binary*** attributes. Thus, we have these 4 frequencies:

f_{00} is the number of attributes where x is 0 and y is 0
f_{01} is the number of attributes where x is 0 and y is 1
f_{10} is the number of attributes where x is 1 and y is 0
f_{11} is the number of attributes where x is 1 and y is 1

$$SMC = \frac{number\ of\ matching\ attribute\ value}{number\ of\ attributes}$$

$$= \frac{f_{11} + f_{00}}{f_{01}+f_{10}+f_{11}+f_{00}}$$

SMC can be used to find students who have similar answers on the True/False questions.

- *Jaccard Coefficient* (J) is usually applied for asymmetric binary attributes (i.e. a lot more 0 than 1). When 1 denotes match and 0 denotes not match

$$J = \frac{number\ of\ matching}{number\ of\ attributes\ without\ 00\ matches}$$

$$= \frac{f_{11}}{f_{01}+f_{10}+f_{11}}$$

Jaccard is often used to handle asymmetric binary attributes such as the transactions from buying at a store. Where 1 indicates an item was purchased, while 0 denotes an item was not purchased. The number of products not purchased by any customer surpasses the products that were purchased (i.e. the number of 0 is far more than 1). As a result, SMC will interpret that all transactions are similar. Therefore, 0-0 matches should be ignored.

Defining Effectiveness in RS
Generally, accuracy can be used to define effectiveness. Accuracy is measured by comparing the predicted rating with actual ratings. The difference between these two ratings is called error. To get the predicted and actual ratings, the input data must be divided into two different sets. The first set is the training set, which is used to generate the model of RS. The second set is the test set, which is used to measure the performance of the RS. From the test set, the actual results (a_i) will compare with the corresponding predicted results (p_i). The most common types of error are:

- Root mean squared error (RMSE) is the sum of squared difference between predicted rating and actual rating

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(p_i - a_i)^2}$$

Some drawbacks of using RMSE are: 1) it does not consider the order of the recommendations 2) it is not appropriate for binary (Boolean) preference data, and 3) it is sensitive to outliers.

- Mean absolute error (MAE) is the absolute deviation value between predicted ratings and actual rating

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|p_i - a_i|$$

Defining quality in RS
RS is considered as a task in information retrieval. The quality is measured by recommended items, which are predicted to be relevant (or good) to users.

Utility / Confusion matrix:

	Reality: Rate good	Reality: Rate bad
Recommend (Prediction) Rate good	True positive (TP)	False positive (FP)
Recommend (Prediction) Rate Bad	False negative (FN)	True negative (TN)

Metrics: Precision and Recall

- Precision is the proportion of recommended items that are actually good. Precision is the measure of exactness.

$$Precision\ (p) = \frac{TP}{TP + FP}$$

- Recall is the proportion of all good items recommended. Recall is the measure of completeness.

$$Recall\ (r) = \frac{TP}{TP + FN}$$

- F_1 measure is the summarization of precision and recall into a single metric

$$F_1 = \frac{2rp}{r + p}$$

References

Amatriain, X., & Basilico, J. (April 6, 2012). *Nextflix recommendations: Beyond the 5 stars (part1)*. The Netflix Tech Blog. Retrieved from http://techblog.netflix.com/2012/04/netflix-recommendations-beyond-5-stars.html.

Anderson, C. (October 2004). *The long tail.* Wired Magazine. Retrieved from http://archive.wired.com/wired/archive/12.10/tail.html.

Cohen, W. W. (n.d.). *Collaborative filtering*. Carnegie Mellon University. Retrieved from http://www.ccs.neu.edu/home/jaa/CSG339.06F/Lectures/collab_filter.pdf.

Jain, A. (n.d.). *Lecture 12 – Analyzing big data with Twitter*. UC Berkeley School of Information. [Video file] (close captioned, 1:15:04 minutes duration) Retrieved from http://www.ischool.berkeley.edu/node/22627.

Jannach, D., & Friedrich, G. (Aug 4, 2013). *Tutorial: Recommender systems.* International Joint Conference on Artificial Intelligence: Bejing. Retrieved from http://ijcai13.org/files/tutorial_slides/td3.pdf.

Jannach, D. Zanker, D., Felfernig, A., & Friedrich, G. (2010). *Recommender systems: An introduction*. Cambridge University Press. Retrieved from http://www.recommenderbook.net/recommender-systems-introduction.

Koren, Y. (August 2009). *The BellKor Solution to the Netflix Grand Prize.* Retrieved from http://www.netflixprize.com/assets/GrandPrize2009_BPC_BellKor.pdf.

Linden, G., Smith, B., & York, J. (January/February 2003). *Amazon.com recommendations: Item-to-item collaborative filtering*: IEEE Computer Society. Retrieved from http://www.cs.umd.edu/~samir/498/Amazon-Recommendations.pdf.

Mackey, L. (October 18, 2009). *Collaborative filtering: Practical machine learning, CS 294-34*. CU Berkeley. Retrieved from http://www.cs.berkeley.edu/~jordan/courses/294-fall09/lectures/collaborative/slides.pdf.

Tan, P., Steinbach, M., & Kumar, V. (2005). *Introduction to data mining*. Addison-Wesley Longman Publishing: Boston, MA. Retrieved from http://dl.acm.org/citation.cfm?id=1095618.

Ullman, D. F. (n.d.). *Recommendation systems, chapter 9*. Stanford University Info Lab. Retrieved from http://infolab.stanford.edu/~ullman/mmds/ch9.pdf.

University of Aberdeen (n.d.). *Lecture 6: Recommender Systems 2 Collaborative Filtering*. Retrieved from http://homepages.abdn.ac.uk/advaith/pages/teaching/abdn.only/AIS/lectures/abdn.only/CollaborativeFiltering.pdf.

Zacharski, R. (2012) *A programmer's guide to data mining: Collaborative filtering, chapter 2*. Retrieved from http://guidetodatamining.com/guide/ch2/DataMining-ch2.pdf.