Recommender Systems
Harnessing the Power of Personalization

Michael Hahsler

Engineering Management, Information, and Systems (EMIS)
Intelligent Data Analysis Lab (IDA@SMU)
Bobby B. Lyle School of Engineering, Southern Methodist University
# Table of Contents

1. **Motivation**
2. **What are Recommender Systems?**
3. **Content-based Approach**
4. **Collaborative Filtering (CF)**
   - Memory-based CF
   - Model-based CF
5. **Further Considerations**
   - Explaining Recommendations
   - Strategies for the Cold Start Problem
   - Security and Recommender Systems
   - Recommender Systems and Revenue Management
6. **Implementation**
   - Open-Source Tools
   - An Example using recommenderlab
   - Deployment
Kristina, Welcome to Your Amazon.com

Today's Recommendations For You

Here's a daily sample of items recommended for you. Click here to see all recommendations.

- **Street Food of India: The 50...** (Hardcover) by Sephi Bergerson
  - Rating: ★★★★★ (4) $19.17
  - Fix this recommendation

- **Lavazza Tierra 100% Arabica Whole Bean Espresso...**
  - Rating: ★★★★★ (36) $34.41
  - Fix this recommendation

- **Entourage: The Complete Fou...** DVD ~ Adrian Grenier
  - Rating: ★★★★☆ (44) $16.49
  - Fix this recommendation

New For You®

- **The Race (Isaac Bell)** Clive Cussler, Justin Scott
  - Hardcover $27.95 $14.97
  - Fix this recommendation

- **Multisensory Teaching of Basic...** Judith R. Birsh, Sally E. Shaywitz
  - Hardcover $79.95 $44.99
  - Fix this recommendation

- **Kill Shot (Mitch Rapp)** Vince Flynn
  - Hardcover $27.99 $16.62
  - Fix this recommendation

- **Limitless (Unrated Extended Cut)** Bradley Cooper, Anna Friel, Abbie...
  - DVD $29.99 $15.19
  - Fix this recommendation
**The Fugitive (1993)**

Wrongfully convicted of murdering his wife, Dr. Richard Kimble (Harrison Ford) escapes custody after a ferocious train accident (one of the most thrilling wrecks ever filmed). While Kimble tries to find the true murderer, gung-ho U.S. Marshal Samuel Gerard (Tommy Lee Jones, in an Oscar-winning performance) is hot on Kimble’s trail, pulling out all stops to put him back behind bars.

**Starring:** Harrison Ford, Tommy Lee Jones  
**Director:** Andrew Davis  
**Genre:** Action & Adventure  
**MPAA:** PG-13

4.7 Our best guess for Michael  
4.1 Customer Average

Recommended based on 8 ratings
A Great Infographic

Can I recommend anything else?

http://www.columnfivemedia.com/work-items/infographic-can-i-recommend anything-else
Table of Contents

1 Motivation
2 What are Recommender Systems?
3 Content-based Approach
4 Collaborative Filtering (CF)
   • Memory-based CF
   • Model-based CF
5 Further Considerations
   • Explaining Recommendations
   • Strategies for the Cold Start Problem
   • Security and Recommender Systems
   • Recommender Systems and Revenue Management
6 Implementation
   • Open-Source Tools
   • An Example using recommenderlab
   • Deployment
Recommender Systems

Original Definition

Recommender systems apply statistical and knowledge discovery techniques to the problem of making product recommendations.

Sarwar et al. (2000)

Advantages of recommender systems (e.g., Schafer et al., 2001):

- Improve conversion rate: Help customers find a product she/he wants to buy.
- Cross-selling: Suggest additional and more diverse products.
- Up-selling: Suggest premium products.
- Improve customer satisfaction/loyalty: Create a value-added relationship.
- Better understand what users want: Knowledge can be reused.
A recommender system is a **fully automatic system** to provide (near) personalized decision support **given limited information** while optimizing a set of potentially conflicting **objective functions**.
A recommender system is a **fully automatic system** to provide (near) personalized decision support **given limited information** while optimizing a set of potentially conflicting **objective functions**.

**Design Space:**
- **Domain** - What are the recommended items? Products, info, etc.
- **Purpose** - Why recommendations? Sales, building a community, etc.
- **Recommendation context** - What is the user doing?
- **Whose opinions** - Available data, incentives, quality.
- **Personalization level** - From non-personalized to persistent.
- **Privacy and trust** - Are the recommendations biased?
- **Interfaces** - Data collection and presenting recommendations.
- **Used algorithms** - Quality and speed.
What Items to Recommend?

MovieLense100k Data

Increase diversity by recommending less well known items.
Recommender System Architecture

Source: Recommender Systems - An Introduction
Common Approaches

- **Non-Personalized recommendations**: Recommendations by experts or summary of community ratings.

Personalized Recommendations

- **Content-based filtering**: Use consumer preferences for product attributes.
- **Collaborative filtering**: Mimics word-of-mouth based on analysis of rating/usage/sales data from many users.
- **Hybrid recommender systems**: Incorporate content, collaborative filtering, expert information and contextual information.
Table of Contents

1 Motivation

2 What are Recommender Systems?

3 Content-based Approach

4 Collaborative Filtering (CF)
   • Memory-based CF
   • Model-based CF

5 Further Considerations
   • Explaining Recommendations
   • Strategies for the Cold Start Problem
   • Security and Recommender Systems
   • Recommender Systems and Revenue Management

6 Implementation
   • Open-Source Tools
   • An Example using recommenderlab
   • Deployment
Content-based Approach

1. Analyze the objects (documents, video, music, etc.) and extract attributes/features (e.g., words, phrases, actors, genre).
2. Recommend objects that match the user profile (e.g., with similar attributes to an object the user likes).
“The Music Genome Project is an effort to capture the essence of music at the fundamental level using almost 400 attributes to describe songs and a complex mathematical algorithm to organize them.”

http://en.wikipedia.org/wiki/Music_Genome_Project
Content can be dynamic...
An issue with content based filtering?
An issue with content based filtering?

Missing diversity!
# Table of Contents

1. Motivation
2. What are Recommender Systems?
3. Content-based Approach
4. Collaborative Filtering (CF)
   - Memory-based CF
   - Model-based CF
5. Further Considerations
   - Explaining Recommendations
   - Strategies for the Cold Start Problem
   - Security and Recommender Systems
   - Recommender Systems and Revenue Management
6. Implementation
   - Open-Source Tools
   - An Example using recommenderlab
   - Deployment
Collaborative Filtering (CF)

Make automatic predictions (filtering) about the interests of a user by collecting preferences or taste information from many other users (collaboration).

Assumption: those who agreed in the past tend to agree again in the future.
Data Collection

- **Explicit**: ask the user for ratings, rankings, list of favorites, etc.
- **Observed behavior (Implicit)**: clicks, page impressions, purchase, uses, downloads, posts, tweets, etc.

What is the incentive structure?
Output of a Recommender System

- Predict ratings of unrated movies (Breese et al., 1998).
- Top-$N$ lists of unrated (unknown) movies ordered by predicted rating/score (Deshpande and Karypis, 2004).
- Annotation in context (e.g., in a electronic guide).
- Recommend as sequence or a bundle.

How do you explain the recommendation to the user? → Trust
Types of CF Algorithms

- **Memory-based**: Find similar users (user-based CF) or items (item-based CF) to predict missing ratings.
- **Model-based**: Build a model from the rating data (clustering, latent structure, etc.) and then use this model to predict missing ratings.
Table of Contents

1 Motivation
2 What are Recommender Systems?
3 Content-based Approach
4 Collaborative Filtering (CF)
   • Memory-based CF
   • Model-based CF
5 Further Considerations
   • Explaining Recommendations
   • Strategies for the Cold Start Problem
   • Security and Recommender Systems
   • Recommender Systems and Revenue Management
6 Implementation
   • Open-Source Tools
   • An Example using recommenderlab
   • Deployment
User-based vs. Item-based CF

User-based filtering

Active User
Tim
Lisa
Sara

Item-based filtering

User-based CF

Produce recommendations based on the preferences of similar users (Goldberg et al., 1992; Resnick et al., 1994; Mild and Reutterer, 2001).

1. Find $k$ nearest neighbors for the user in the user-item matrix.
2. Generate recommendation based on the items liked by the $k$ nearest neighbors. E.g., average ratings or use a weighting scheme.
User-based CF II

- **Pearson correlation coefficient:**
  \[
  \text{sim}_{\text{Pearson}}(x, y) = \frac{\sum_{i \in I} x_i y_i - I \bar{x} \bar{y}}{(I-1)s_x s_y}
  \]

- **Cosine similarity:**
  \[
  \text{sim}_{\text{Cosine}}(x, y) = \frac{x \cdot y}{\|x\|_2 \|y\|_2}
  \]

- **Jaccard index** (only binary data):
  \[
  \text{sim}_{\text{Jaccard}}(X, Y) = \frac{|X \cap Y|}{|X \cup Y|}
  \]

where \(x = b_{ux}\), and \(y = b_{uy}\), represent the user’s profile vectors and \(X\) and \(Y\) are the sets of the items with a 1 in the respective profile.

**Problem**

Memory-based. Expensive online similarity computation.
Item-based CF

Produce recommendations based on the relationship between items in the user-item matrix (Kitts et al., 2000; Sarwar et al., 2001)

<table>
<thead>
<tr>
<th></th>
<th>i₁</th>
<th>i₂</th>
<th>i₃</th>
<th>i₄</th>
<th>i₅</th>
<th>i₆</th>
<th>i₇</th>
<th>i₈</th>
</tr>
</thead>
<tbody>
<tr>
<td>i₁</td>
<td>-</td>
<td>0.1</td>
<td>0</td>
<td>0.3</td>
<td>0.2</td>
<td>0</td>
<td>0</td>
<td>0.1</td>
</tr>
<tr>
<td>i₂</td>
<td>0.1</td>
<td>-</td>
<td>0.8</td>
<td>0.9</td>
<td>0</td>
<td>0</td>
<td>0.2</td>
<td>0</td>
</tr>
<tr>
<td>i₃</td>
<td>0</td>
<td>0.8</td>
<td>-</td>
<td>0</td>
<td>0.4</td>
<td>0</td>
<td>0.1</td>
<td>0.3</td>
</tr>
<tr>
<td>i₄</td>
<td>0.3</td>
<td>0.9</td>
<td>0</td>
<td>-</td>
<td>0</td>
<td>0</td>
<td>0.3</td>
<td>0</td>
</tr>
<tr>
<td>i₅</td>
<td>0.2</td>
<td>0</td>
<td>0.7</td>
<td>0</td>
<td>-</td>
<td>0</td>
<td>0.2</td>
<td>0</td>
</tr>
<tr>
<td>i₆</td>
<td>0.4</td>
<td>0.2</td>
<td>0.1</td>
<td>0.3</td>
<td>0</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>i₇</td>
<td>0</td>
<td>0.1</td>
<td>0.3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-</td>
<td>0</td>
</tr>
<tr>
<td>i₈</td>
<td>0.1</td>
<td>0</td>
<td>0.9</td>
<td>0.1</td>
<td>0</td>
<td>0</td>
<td>0.1</td>
<td>0</td>
</tr>
</tbody>
</table>

Calculate similarities between items and keep for each item only the values for the $k$ most similar items.

$u_a = \{i_1, i_5, i_8\}$

$u_{ua} = \{?, ?, ?, 4, ?, ?, 5\}$

$S_r = \{2, ?, ?, ?, 4, ?, ?, 5\}$

Recommendation: $i_3$

1. Calculate similarities between items and keep for each item only the values for the $k$ most similar items.
2. Use the similarities to calculate a weighted sum of the user's ratings for related items.

$$\hat{r}_{ui} = \frac{\sum_{j \in s_i} s_{ij} r_{uj}}{\sum_{j \in s_i} |s_{ij}|}$$

Regression can also be used to create the prediction.
Item-based CF II

Similarity measures:

- Pearson correlation coefficient, cosine similarity, Jaccard index
- Conditional probability-based similarity (Deshpande and Karypis, 2004):

$\text{sim}_{\text{Conditional}}(x, y) = \frac{\text{Freq}(xy)}{\text{Freq}(x)} = \hat{P}(y|x)$

where $x$ and $y$ are two items, $\text{Freq}(\cdot)$ is the number of users with the given item in their profile.

Properties

- Model (reduced similarity matrix) is relatively small ($N \times k$) and can be fully precomputed.
- Item-based CF was reported to only produce slightly inferior results compared to user-based CF (Deshpande and Karypis, 2004).
- Higher order models which take the joint distribution of sets of items into account are possible (Deshpande and Karypis, 2004).
- Successful application in large scale systems (e.g., Amazon.com)
Table of Contents

1 Motivation
2 What are Recommender Systems?
3 Content-based Approach
4 Collaborative Filtering (CF)
   • Memory-based CF
   • Model-based CF
5 Further Considerations
   • Explaining Recommendations
   • Strategies for the Cold Start Problem
   • Security and Recommender Systems
   • Recommender Systems and Revenue Management
6 Implementation
   • Open-Source Tools
   • An Example using recommenderlab
   • Deployment
Different Model-based CF Techniques

There are many techniques:

- **Cluster users** (i.e., customer segmentation) and then recommend items the users in the cluster closest to the active user like.

- **Mine association rules** (if-then rules) and then use the rules to recommend items.

- Define a null-model (a stochastic process which models usage of independent items) and then find significant deviation from the null-model.

- **Learning to rank**: Logistic regression, neural networks (deep learning) and many other machine learning methods.

- Learn a **latent factor model** from the data and then use the discovered factors to find items with high expected ratings.
Latent Factor Approach

Latent semantic indexing (LSI) developed by the IR community (late 80s) addresses sparsity, scalability and can handle synonyms

⇒ Dimensionality reduction.
Matrix Factorization

Given a user-item (rating) matrix $M = (r_{ui})$, map users and items on a joint latent factor space of dimensionality $k$.

- Each item $i$ is modeled by a vector $q_i \in \mathbb{R}^k$.
- Each user $u$ is modeled by a vector $p_u \in \mathbb{R}^k$.

such that a value close to the actual rating $r_{ui}$ can be computed (e.g., by the dot product also known as the cosine similarity)

$$r_{ui} \approx \hat{r}_{ui} = q_i^T p_u$$

The hard part is to find a suitable latent factor space!
Singular Value Decomposition (Matrix Fact.)

Linear algebra: **Singular Value Decomposition (SVD)** to factorizes $M$

\[ M = U \Sigma V^T \]

$M$ is the $m \times n$ (users $\times$ items) rating matrix of rank $r$. Columns of $U$ and $V$ are the left and right singular vectors. Diagonal of $\Sigma$ contains the $r$ singular values.
Linear algebra: Singular Value Decomposition (SVD) to factorizes $M$

\[ M = U \Sigma V^T \]

$M$ is the $m \times n$ (users $\times$ items) rating matrix of rank $r$. Columns of $U$ and $V$ are the left and right singular vectors. Diagonal of $\Sigma$ contains the $r$ singular values.

Best rank-$k$ approximation minimizes error $\| M - M_k \|_F$ (Frobenius norm).

Approximation of $M$ using $k$ Factors
Challenges (Matrix Fact.)

- **Missing values**: Imputation using column means (mean item ratings). For centered columns the mean is zero.

- **SVD is $O(m^3)$**: Use incremental SVD to 'fold in' new users/items without recomputing the whole SVD (Sarwar et al., 2002).

1. Calculate user-feature vector from imputed ratings $m_a$.
   \[ u_a = m_a V_k^T \Sigma_k^{-1} \]
   
2. Predict ratings
   \[ \hat{m}_a = u_a \Sigma_k V_k^T \]
   
Works similarly for new items.
Challenges (Matrix Fact.)

Too many missing values are a problem. SVD with missing values by minimizing the square error on only known ratings (regularized to avoid overfitting).

\[
\arg\min_{p^*, q^*} \sum_{(u,i) \in \kappa} (r_{ui} - q_i^T p_u)^2 + \lambda(||q_i||^2 + ||p_u||^2)
\]

where \(\kappa\) are the \((u, i)\) pairs for which \(r\) is known.

Good solutions can be found by stochastic gradient descent or alternating least squares (Koren et al., 2009).

1. For new user (item) compute \(q_i (p_u)\).
2. After all \(q_i\) and \(p_u\) are known, prediction is very fast:

\[\hat{r}_{ui} = q_i^T p_u\]
# Table of Contents

1. Motivation
2. What are Recommender Systems?
3. Content-based Approach
4. Collaborative Filtering (CF)
   - Memory-based CF
   - Model-based CF
5. Further Considerations
   - Explaining Recommendations
   - Strategies for the Cold Start Problem
   - Security and Recommender Systems
   - Recommender Systems and Revenue Management
6. Implementation
   - Open-Source Tools
   - An Example using recommenderlab
   - Deployment
Explaining Recommendations

WHAT IS THE TOMATOMETER™?

The Tomatometer rating – based on the published opinions of hundreds of film and television critics – is a trusted measurement of movie and TV programming quality for millions of moviegoers. It represents the percentage of professional critic reviews that are positive for a given film or television show.

FROM THE CRITICS

**Fresh**
The Tomatometer is 60% or higher.

**Rotten**
The Tomatometer is 59% or lower.

FROM RT USERS LIKE YOU!

**Certified Fresh**
Movies and TV shows are Certified Fresh with a steady Tomatometer of 75% or higher after a set amount of reviews (80 for wide-release movies, 40 for limited-release movies, 20 for TV shows), including 5 reviews from Top Critics.

**Audience Score**
Percentage of users who rate a movie or TV show positively.

Learn More ➤
Cold Start Problem

What do we recommend to new users for whom we have no ratings yet?
Cold Start Problem

What do we recommend to new users for whom we have no ratings yet?

- Recommend popular items
- Have some start-up questions (e.g., "What are your 10 favorite movies?")
- Obtain/purchase personal information
Cold Start Problem

What do we recommend to new users for whom we have no ratings yet?
- Recommend popular items
- Have some start-up questions (e.g., "What are your 10 favorite movies?")
- Obtain/purchase personal information

What do we do with new items?
Cold Start Problem

What do we recommend to new users for whom we have no ratings yet?

- Recommend popular items
- Have some start-up questions (e.g., "What are your 10 favorite movies?")
- Obtain/purchase personal information

What do we do with new items?

- Content-based filtering techniques.
- Use expert/domain knowledge.
- Pay a focus group to rate new items.
# Table of Contents

1. Motivation
2. What are Recommender Systems?
3. Content-based Approach
4. Collaborative Filtering (CF)
   - Memory-based CF
   - Model-based CF
5. Further Considerations
   - Explaining Recommendations
   - Strategies for the Cold Start Problem
   - Security and Recommender Systems
   - Recommender Systems and Revenue Management
6. Implementation
   - Open-Source Tools
   - An Example using recommenderlab
   - Deployment
Protect recommender neutrality
From malicious users who want to push their product and can create fake accounts
Possible solutions: prevent account creation or detect and remove
Security and Recommender Systems

- **Protect recommender neutrality**
  From malicious users who want to push their product and can create fake accounts
  Possible solutions: prevent account creation or detect and remove

- **Protect recommender accuracy**
  From users who give low-quality, inconsistent ratings.
  Possible solutions: Normal de-noising problem
Security and Recommender Systems

- **Protect recommender neutrality**
  From malicious users who want to push their product and can create fake accounts
  
  Possible solutions: prevent account creation or detect and remove

- **Protect recommender accuracy**
  From users who give low-quality, inconsistent ratings.
  
  Possible solutions: Normal de-noising problem

- **Protect user data (privacy)**
  From other users and from the service provider
  
  Possible solutions: Use trusted computing infrastructure, pool ratings, add noise
# Table of Contents

1 Motivation
2 What are Recommender Systems?
3 Content-based Approach
4 Collaborative Filtering (CF)
   - Memory-based CF
   - Model-based CF
5 Further Considerations
   - Explaining Recommendations
   - Strategies for the Cold Start Problem
   - Security and Recommender Systems
   - Recommender Systems and Revenue Management
6 Implementation
   - Open-Source Tools
   - An Example using recommenderlab
   - Deployment
Revenue Management

Recommender systems have the potential to increase revenue
- cross-selling
- up-selling

How about influencing which items are recommended using revenue considerations?
Revenue Management

Recommender systems have the potential to increase revenue

- cross-selling
- up-selling

How about influencing which items are recommended using revenue considerations?

What about trust + incentive to share information?
Table of Contents

1 Motivation
2 What are Recommender Systems?
3 Content-based Approach
4 Collaborative Filtering (CF)
   - Memory-based CF
   - Model-based CF
5 Further Considerations
   - Explaining Recommendations
   - Strategies for the Cold Start Problem
   - Security and Recommender Systems
   - Recommender Systems and Revenue Management
6 Implementation
   - Open-Source Tools
   - An Example using recommenderlab
   - Deployment
Table of Contents

1 Motivation
2 What are Recommender Systems?
3 Content-based Approach
4 Collaborative Filtering (CF)
   • Memory-based CF
   • Model-based CF
5 Further Considerations
   • Explaining Recommendations
   • Strategies for the Cold Start Problem
   • Security and Recommender Systems
   • Recommender Systems and Revenue Management
6 Implementation
   • Open-Source Tools
   • An Example using recommenderlab
   • Deployment
Open-Source Implementations

- **Apache Mahout**: ML library including collaborative filtering (Java)
- **C/Matlab Toolkit for Collaborative Filtering**: (C/Matlab)
- **Cofi**: Collaborative Filtering Library (Java)
- **Crab**: Components for recommender systems (Python)
- **easyrec**: Recommender for Web pages (Java)
- **LensKit**: CF algorithms from GroupLens Research (Java)
- **MyMediaLite**: Recommender system algorithms. (C# / Mono)
- **RACOFI**: A rule-applying collaborative filtering system
- **Rating-based item-to-item recommender system**: (PHP / SQL)
- **recommenderlab**: Infrastructure to test and develop recommender algorithms (R)

Table of Contents

1. Motivation
2. What are Recommender Systems?
3. Content-based Approach
4. Collaborative Filtering (CF)
   - Memory-based CF
   - Model-based CF
5. Further Considerations
   - Explaining Recommendations
   - Strategies for the Cold Start Problem
   - Security and Recommender Systems
   - Recommender Systems and Revenue Management
6. Implementation
   - Open-Source Tools
   - An Example using recommenderlab
   - Deployment
100k MovieLense ratings data set: The data was collected through movielens.umn.edu from 9/1997 to 4/1998. The data set contains about 100,000 ratings (1-5) from 943 users on 1664 movies.

```r
R> library("recommenderlab")
R> data(MovieLense)
R> MovieLense
943 x 1664 rating matrix of class ‘realRatingMatrix’ with 99392 ratings.
R> train <- MovieLense[1:900]
R> u <- MovieLense[901]
R> u
1 x 1664 rating matrix of class ‘realRatingMatrix’ with 124 ratings.
R> as(u, "list")[[1]][1:5]
   5 3 5 1 5
```
R> r <- Recommender(train, method = "UBCF")
R> r
Recommender of type ‘UBCF’ for ‘realRatingMatrix’
learned using 900 users.
R> recom <- predict(r, u, n = 5)
R> recom
Recommendations as ‘topNList’ with n = 5 for 1 users.
R> as(recom, "list")[[1]]
[3] "It's a Wonderful Life (1946)"
R> scheme <- evaluationScheme(train, method = "cross", k = 4,
+       given = 10, goodRating = 3)
R> algorithms <- list(
+   `random items` = list(name = "RANDOM", param = NULL),
+   `popular items` = list(name = "POPULAR", param = NULL),
+   `user-based CF` = list(name = "UBCF",
+     param = list(method = "Cosine", nn = 50)),
+   `item-based CF` = list(name = "IBCF",
+     param = list(method = "Cosine", k = 50)))
R> results <- evaluate(scheme, algorithms,
+   n = c(1, 3, 5, 10, 15, 20, 50))
R> plot(results, annotate = c(1, 3), legend = "right")
Table of Contents

1 Motivation
2 What are Recommender Systems?
3 Content-based Approach
4 Collaborative Filtering (CF)
   • Memory-based CF
   • Model-based CF
5 Further Considerations
   • Explaining Recommendations
   • Strategies for the Cold Start Problem
   • Security and Recommender Systems
   • Recommender Systems and Revenue Management
6 Implementation
   • Open-Source Tools
   • An Example using recommenderlab
   • Deployment
Technology

Java

Cassandra

Hadoop

Amazon Web Services

HTML5

Linux

Hive

Teradata

Jenkins

http://techblog.netflix.com
References


Thank you!

This presentation can be downloaded from http://michael.hahsler.net/ (under publications/talks)

For questions, please contact the author at mhahsler@lyle.smu.edu

recommenderlab is available in R from CRAN. An introduction can be found at https://cran.r-project.org/web/packages/recommenderlab/vignettes/recommenderlab.pdf