Package 'CFilt'

May 30, 2026

Title Collaborative Filtering Models for Recommendation Systems

Version 1.0.1

Description Implements collaborative filtering methods for recommendation systems based on user-item interaction data. Supports both explicit feedback (ratings) and implicit feedback (consumption). The package uses efficient sparse matrix representations and provides incremental updates for users, items, and similarity structures through an R6 class-based architecture. See Aggarwal (2016) <doi:10.1007/978-3-319-29659-3> for an overview.

License MIT + file LICENSE

Encoding UTF-8

Imports Matrix, R6

Config/roxygen2/version 8.0.0

NeedsCompilation no

Author Jessica Kubrusly [aut, cre] (ORCID:
<https://orcid.org/0000-0003-0465-4629>),
Thiago Lima [ctb],
Lucas Oliveira [ctb],
Caio Salviano [ctb]

Maintainer Jessica Kubrusly <jessicakubrusly@id.uff.br>

Depends R (>= 3.5.0)

Repository CRAN

Date/Publication 2026-05-30 20:10:02 UTC

Contents

CF *Recommendation by Collaborative Filtering*

Description

CF is a class of objects that stores information about a recommendation system. This information includes the consumption or rating of each (user, item) pair in the utility matrix MU, the similarities between each pair of users in the similarity matrix SU, the similarities between each pair of items in the similarity matrix SI, the number of items consumed and/or rated by each user in the vector n_aval_u, the number of users who consumed and/or rated each item in the vector n_aval_i, the average rating value of each user in the vector averages_u, the average rating value received by each item in the vector averages_i, the number of items consumed in common by each pair of users in the matrix Int_U, and the number of users in common for each pair of items in the matrix Int_I. The class contains methods such as addNewUser, addNewEmptyUser, deleteUser, addNewItem, addNewEmptyItem, deleteItem, newRating and deleteRating, which modify the object's structure by altering users, items, or consumption data. The class also includes functions such as kClosestItems, topKUsers, and topKItems, which return items to recommend to a user or users to whom an item should be recommended. An object of the CF class is created using the CFBuilder function.

Details

This class implements a collaborative filtering system supporting both explicit (ratings) and implicit (consumption) feedback. The internal state is updated incrementally after each operation.

Public fields

`MU` The Utility Matrix, a matrix that contains all the users' ratings. The rows comprise users and the columns, items.

`SU` The user similarity matrix.

`SI` The item similarity matrix

`IntU` A symmetric matrix that records the number of items in common between pairs of users.

`IntI` A symmetric matrix that records the number of users in common between pairs of items.umber of items in common that

`averages_u` A vector that contains the averages of users' ratings.

`averages_i` A vector that contains the averages of items' ratings.

`n_aval_u` A vector that stores the number of items rated by each user.

`n_aval_i` A vector that stores the number of users who consumed each item.

`datatype` A character that indicates the type of data, which can be either "consumption" or "rating".

`similarity` A character string indicating the similarity measure used. It can be "pearson" or "cosine" for rating data, and "jaccard" for consumption data.

`data_0` A data.frame containing the original dataset used to build the object. This corresponds to the input data provided to CFbuilder and is stored for reference.

Methods

Public methods:

- CF$addnewemptyuser()
- CF$addnewemptyitem()
- CF$newrating()
- CF$deleteuser()
- CF$deleteitem()
- CF$deleterating()
- CF$clone()

`CF$addnewemptyuser()`: Add a new empty user to the system. This method creates a new user with no interactions.

Usage:

```
CF$addnewemptyuser(Id_u)
```

Arguments:

`Id_u` A character string (or a list of strings) representing user ID(s).

Returns: Invisibly returns the updated object.

`CF$addnewemptyitem()`: Add a new empty item to the system.

Usage:

```
CF$addnewemptyitem(Id_i)
```

Arguments:

`Id_i` A character string (or list of strings) representing item ID(s).

`CF$newrating()`: Add a new rating or consumption.

Usage:

```
CF$newrating(Id_u, Id_i, r = NULL)
```

Arguments:

`Id_u` A character string (or a list of strings) representing user ID(s).

`Id_i` A character string (or a list of strings) representing item ID(s).

`r` A numeric (or a list of numeric) for rating value(s) (only for rating data)

`CF$deleteuser()`: Delete a user from the system.

Usage:

```
CF$deleteuser(Id_u)
```

Arguments:

`Id_u` A character string (or a list of strings) representing user ID(s).

`CF$deleteitem()`: Delete an item from the system.

Usage:

```
CF$deleteitem(Id_i)
```

Arguments:

Id_i A character string (or a list of strings) representing item ID(s).

`CF$deleterating()`: Delete a rating or consumption.

Usage:

```
CF$deleterating(Id_u, Id_i)
```

Arguments:

Id_u A character string (or a list of strings) representing user ID(s).

Id_i A character string (or a list of strings) representing item ID(s).

`CF$clone()`: The objects of this class are cloneable with this method.

Usage:

```
CF$clone(deep = FALSE)
```

Arguments:

deep Whether to make a deep clone.

Author(s)

Jessica Kubrusly

References

- LINDEN, G.; SMITH, B.; YORK, J. Amazon. com recommendations: Item-to-item collaborative filtering. Internet Computing, IEEE, v. 7, n. 1, p. 76-80,2003
- Aggarwal, C. C. (2016). Recommender systems (Vol. 1). Cham: Springer International Publishing.
- Leskovec, J., Rajaraman, A., & Ullman, J. D. (2020). Mining of massive data sets. Cambridge university press.

See Also

CFbuilder

Examples

```
data(movies, package = "CFilt")
# --- Rating data ---
objectCF_r <- CFbuilder(Data = movies[1:500,],Datatype = "rating",
similarity = "pearson")
dim(objectCF_r$MU)
colnames(objectCF_r$MU) #movies Id
rownames(objectCF_r$MU) #users Id
dim(objectCF_r$SU)
dim(objectCF_r$SI)
objectCF_r$averages_u
objectCF_r$averages_i
objectCF_r$n_aval_u
objectCF_r$n_aval_i
objectCF_r$addnewemptyuser(Id_u = "newuser1")
```

```
objectCF_r$newrating(Id_u = "newuser1",Id_i = "Frozen",r = 5)
objectCF_r$MU["newuser1","Frozen"]
objectCF_r$newrating(Id_u = list("newuser1","newuser1","newuser1"),
Id_i = list("Thor: The Dark World","The Lego Movie","Despicable Me 2"),
r = list(2,3,4))
objectCF_r$n_aval_u["newuser1"]
objectCF_r$averages_u["newuser1"]
objectCF_r$addnewemptyuser(Id_u = "newuser2")
objectCF_r$newrating(Id_u = list("newuser2","newuser1"),
Id_i = list("Frozen","Her"),r = c(2,1))
objectCF_r$addnewemptyuser(Id_u = list("newuser3","newuser4"))
objectCF_r$newrating(Id_u = list("newuser3","newuser3","newuser4","newuser4"),
Id_i = list("The Lego Movie","Wreck-It Ralph","Fast & Furious 6",
"12 Years a Slave"),r = list(4,5,4,2))
objectCF_r$addnewemptyitem(Id_i = list("movie1","movie2","movie3"))
objectCF_r$newrating(
Id_u = list("newuser1","newuser1","newuser1",
            "newuser2","newuser2","newuser2",
            "newuser3","newuser3","newuser3",
            "newuser4","newuser4","newuser4"),
Id_i = list("movie1","movie2","movie3",
            "movie1","movie2","movie3",
            "movie1","movie2","movie3",
            "movie1","movie2","movie3"),
r = list(4,5,4,2,
         1,2,1,1,
         4,3,1,2))
objectCF_r$MU[,"movie1"]
objectCF_r$SI["movie1","movie2"]
objectCF_r$SU["newuser2","newuser4"]
# --- Consumption data ---
objectCF_c <- CFbuilder(Data = movies[1:300,-3],Datatype = "consumption",
similarity = "jaccard")
objectCF_c$addnewemptyuser(Id_u = list("newuser1","newuser2","newuser3"))
objectCF_c$newrating(Id_u = list("newuser1","newuser2","newuser3"),
Id_i = list("Frozen","Frozen","Frozen"))
objectCF_c$newrating(Id_u = list("newuser1","newuser1","newuser1"),
Id_i = list("Gravity","The Wolverine","Iron Man 3"))
objectCF_c$addnewemptyitem(Id_i = list("movie1","movie2","movie3"))
objectCF_c$newrating(Id_u = list("newuser1","newuser1","newuser2","newuser2",
"newuser3"),Id_i = list("movie1","movie2","movie1","movie3","movie3"))
objectCF_c$MU[,"movie1"]
objectCF_c$SI["movie1","movie2"]
objectCF_c$SI["movie1","movie3"]
objectCF_c$SI["movie2","movie3"]
objectCF_c$SU["newuser1","newuser2"]
objectCF_c$SU["newuser2","newuser3"]
```

CFbuilder *Build a Collaborative Filtering Object*

Description

Creates an object of class CF from a dataset of user-item interactions. The dataset can represent either explicit ratings or implicit consumption.

Usage

```
CFbuilder(
  Data,
  Datatype = ifelse(ncol(Data) == 2, "consumption", "rating"),
  similarity = ifelse(Datatype == "consumption", "jaccard", "pearson")
)
```

Arguments

Data	A data.frame containing: • 2 columns: (user, item) for consumption data • 3 columns: (user, item, rating) for rating data
Datatype	A character string indicating the type of data: "consumption" or "rating". Default is inferred from number of columns in Data.
similarity	A character string indicating the similarity measure: • "jaccard" for consumption data • "pearson" or "cosine" for rating data Default is chosen based on Datatype.

Value

An object of class CF.

Examples

```
data(movies, package = "CFilt")
# --- Rating data ---
CF1 <- CFbuilder(Data = movies[1:300,],Datatype = "rating",
similarity = "pearson")
CF1_ <- CFbuilder(Data = movies[1:300,])
CF2 <- CFbuilder(Data = movies[1:300,],Datatype = "rating",
similarity = "cosine")
CF2_ <- CFbuilder(Data = movies[1:300,],similarity = "cosine")
# --- Consumption data ---
CF3 <- CFbuilder(Data = movies[1:300,-3],Datatype = "consumption",
similarity = "jaccard")
CF3_ <- CFbuilder(Data = movies[1:300,-3])
```

kclosestitems *K Closest Items*

Description

Returns the k most similar items to a given item based on the item-item similarity matrix of a collaborative filtering model.

Usage

```
kclosestitems(CF, Id_i, k = 10)
```

Arguments

CF	An object of class CF created by CFbuilder.
Id_i	A character string representing the item ID.
k	A positive integer indicating the number of similar items to return. Default is 10.

Details

The similarity between items is obtained from the item similarity matrix stored in CF$SI. The item itself is excluded from the result.

Value

A character vector containing the IDs of the k most similar items.

See Also

CFbuilder, topkitems, topkusers

Examples

```
data(movies, package = "CFilt")

CF1 <- CFbuilder(movies[1:200, ], Datatype = "rating")

# Find the 5 items most similar to a given item
kclosestitems(CF1, Id_i = "Frozen", k = 5)
```

movies *Movie ratings by users*

Description

A dataset containing 7276 ratings for 50 movies by 526 users.

Format

A data frame with 7276 rows and 3 variables:

IdUsers Users identifier. Numbers 1 to 526.

IdItems Movies identifier. Movies names.

Ratings Movie ratings by users. Ratings from 1 to 5 (Likert scale).

Source

Giglio (2014)

topkitems *Top-K Item Recommendation*

Description

Returns the top-k items to recommend for a given user based on a collaborative filtering model.

Usage

```
topkitems(CF, Id_u, k = 10, type = "user")
```

Arguments

`CF`	An object of class `CF`.
`Id_u`	A character string representing the user ID.
`k`	A positive integer indicating the number of items to recommend. Default is 10.
`type`	A character string indicating the recommendation strategy: • `"user"`: user-based collaborative filtering • `"item"`: item-based collaborative filtering

Details

For `type = "user"`, recommendations are computed based on similarities between users. For `type = "item"`, recommendations are computed based on similarities between items.

Only items not yet consumed/rated by the user are considered.

Value

A character vector containing the IDs of the top-k recommended items.

See Also

CFbuilder, topkitems

Examples

```
data(movies, package = "CFilt")

CF1 <- CFbuilder(movies[1:200, ], Datatype = "rating")

# Recommend users for an item using user-based CF
topkitems(CF1, Id_u = "1", k = 5, type = "user")

# Recommend users for an item using item-based CF
topkitems(CF1, Id_u = "1", k = 3, type = "item")

CF2 <- CFbuilder(movies[1:200,-3])

# Recommend users for an item using user-based CF
topkitems(CF2, Id_u = "1", k = 5, type = "user")

# Recommend users for an item using item-based CF
topkitems(CF2, Id_u = "1", k = 3, type = "item")
```

---

topkusers *Top-K User Recommendation for an Item*

Description

Returns the top-k users to whom a given item should be recommended, based on a collaborative filtering model.

Usage

```
topkusers(CF, Id_i, k = 10, type = "user")
```

Arguments

CF	An object of class CF created by CFbuilder.
Id_i	A character string representing the item ID.
k	A positive integer indicating the number of users to return. Default is 10.
type	A character string indicating the recommendation strategy: • "user": user-based collaborative filtering • "item": item-based collaborative filtering

Details

For `type = "user"`, recommendations are based on similarities between users. For `type = "item"`, recommendations are based on similarities between items.

Only users who have not yet consumed/rated the item are considered.

Value

A character vector containing the IDs of the top-k users for whom the item is recommended.

See Also

`CFbuilder`, `topkitems`

Examples

```
data(movies, package = "CFilt")

CF1 <- CFbuilder(movies[1:200, ], Datatype = "rating")

# Recommend users for an item using user-based CF
topkusers(CF1, Id_i = "Frozen", k = 5, type = "user")

# Recommend users for an item using item-based CF
topkusers(CF1, Id_i = "Frozen", k = 5, type = "item")

CF2 <- CFbuilder(movies[1:200,-3])

# Recommend users for an item using user-based CF
topkusers(CF2, Id_i = "Frozen", k = 5, type = "user")

# Recommend users for an item using item-based CF
topkusers(CF2, Id_i = "Frozen", k = 3, type = "item")
```

Index