Cornac Documentation

Release 0.1.0

Cornac Contributors

Feb 15, 2019

Getting Started

Cornac is python recommender systems library for **easy**, **effective** and **efficient** experiments. Cornac is **simple** and **handy**. It is designed from the ground-up to faithfully reflect the standard steps taken by researchers to implement and evaluate personalized recommendation models. Moreover, contributing new recommender models, evaluation metrics, etc., to Cornac is very easy and smooth. For instance, if you already have a python implementation of your model, e.g., PMF, you will need to spend less than 5 minutes in average to integrate it to Cornac.

CHAPTER 1

Installation

Currently, we are supporting Python 3 (version 3.6 is recommended). There are several ways to install Cornac:

- **From PyPI (you may need a C compiler)**:

```
pip3 install cornac
```

- **From Anaconda**:

```
conda install cornac -c qttruong -c pytorch
```

- **From the GitHub source (for latest updates)**:

```
# Optional: install Cython
pip3 install cython

# Clone Cornac from the main repository
git clone https://github.com/PreferredAI/cornac.git
cd cornac

# You will need a C compiler
python3 setup.py install
```

Note:

Some installed dependencies are CPU versions. If you want to utilize your GPU, you might consider:

- TensorFlow installation instructions: https://www.tensorflow.org/install/.
- PyTorch installation instructions: https://pytorch.org/get-started/locally/.
- cuDNN: https://docs.nvidia.com/deeplearning/sdk/cudnn-install/ (for Nvidia GPUs).

CHAPTER 2

First example

This example will show you how to run your very first experiment using Cornac. It consists in training and evaluating the Probabilistic Matrix Factorization (PMF) recommender model.

```
# Importing required modules from Cornac.
from cornac.models import PMF
from cornac import Experiment
from cornac.eval_methods import RatioSplit
from cornac.datasets import MovieLens100K
from cornac import metrics


# Load the MovieLens 100K dataset
ml_100k = MovieLens100K.load_data()

# Instantiate an evaluation strategy.
ratio_split = RatioSplit(data=ml_100k, test_size=0.2, rating_threshold=4.0, exclude_
↪unknowns=False)

# Instantiate a PMF recommender model.
pmf = PMF(k=10, max_iter=100, learning_rate=0.001, lamda=0.001)

# Instantiate evaluation metrics.
mae = metrics.MAE()
rmse = metrics.RMSE()
rec_20 = metrics.Recall(k=20)
pre_20 = metrics.Precision(k=20)

# Instantiate and then run an experiment.
exp = Experiment(eval_strategy=ratio_split,
                 models=[pmf],
                 metrics=[mae, rmse, rec_20, pre_20],
                 user_based=True)
exp.run()
```

CHAPTER 3

models

3.1 Probabilistic Collaborative Representation Learning (PCRL)

@author: Aghiles Salah <asalah@smu.edu.sg>

class cornac.models.pcrl.recom_pcrl.**PCRL**(*k=100, z_dims=[300], max_iter=300, batch_size=300, learning_rate=0.001, aux_info=None, name='pcrl', trainable=True, w_determinist=True, init_params={'G_r': None, 'G_s': None, 'L_r': None, 'L_s': None}*)

Probabilistic Collaborative Representation Learning.

Parameters

- **k** (*int, optional, default: 100*) – The dimension of the latent factors.
- **z_dims** (*Numpy 1d array, optional, default: [300]*) – The dimensions of the hidden intermdiate layers 'z' in the order [dim(z_L), ...,dim(z_1)], please refer to Figure 1 in the orginal paper for more details.
- **max_iter** (*int, optional, default: 300*) – Maximum number of iterations (number of epochs) for variational PCRL.
- **batch_size** (*int, optional, default: 300*) – The batch size for SGD.
- **learning_rate** (*float, optional, default: 0.001*) – The learning rate for SGD.
- **aux_info** (*csc sparse matrix, required*) – The item auxiliary information matrix, item-context in the PCRL's paper, in the scipy csc sparse format.
- **name** (*string, optional, default: 'PCRL'*) – The name of the recommender model.
- **trainable** (*boolean, optional, default: True*) – When False, the model is not trained and Cornac assumes that the model already pre-trained (Theta, Beta and Xi are not None).

- **w_determinist** (*boolean, optional, default: True*) – When True, determinist wheights "W" are used for the generator network, otherwise "W" is stochastic as in the original paper.
- **init_params** (*dictionary, optional, default: {'G_s':None, 'G_r':None, 'L_s':None, 'L_r':None}*) – List of initial parameters, e.g., init_params = {'G_s':G_s, 'G_r':G_r, 'L_s':L_s, 'L_r':L_r}, where G_s and G_r are of type csc_matrix or np.array with the same shape as Theta, see below). They represent respectively the "shape" and "rate" parameters of Gamma distribution over Theta. It is the same for L_s, L_r and Beta.
- **Theta** (*csc_matrix, shape (n_users,k)*) – The expected user latent factors.
- **Beta** (*csc_matrix, shape (n_items,k)*) – The expected item latent factors.

References

- Salah, Aghiles, and Hady W. Lauw. Probabilistic Collaborative Representation Learning for Personalized Item Recommendation. In UAI 2018.

fit(*X*)

Fit the model to observations.

Parameters **X** (*scipy sparse matrix, required*) – the user-item preference matrix (traning data), in a scipy sparse format (e.g., csc_matrix).

rank(*user_index*, *known_items=None*)

Rank all test items for a given user.

Parameters

- **user_index** (*int, required*) – The index of the user for whom to perform item raking.
- **known_items** (*1d array, optional, default: None*) – A list of item indices already known by the user

Returns Array of item indices sorted (in decreasing order) relative to some user preference scores.

Return type Numpy 1d array

score(*user_index*, *item_indexes=None*)

Predict the scores/ratings of a user for a list of items.

Parameters

- **user_index** (*int, required*) – The index of the user for whom to perform score predictions.
- **item_indexes** (*1d array, optional, default: None*) – A list of item indexes for which to predict the rating score. When "None", score prediction is performed for all test items of the given user.

Returns Array containing the predicted values for the items of interest

Return type Numpy 1d array

3.2 Collaborative Context Poisson Factorization (C2PF)

@author: Aghiles Salah <asalah@smu.edu.sg>

class cornac.models.c2pf.recom_c2pf.**C2PF**(*k=100*, *max_iter=100*, *aux_info=None*, *variant='c2pf'*, *name=None*, *trainable=True*, *init_params={'G_r': None, 'G_s': None, 'L2_r': None, 'L2_s': None, 'L3_r': None, 'L3_s': None, 'L_r': None, 'L_s': None}*)

Collaborative Context Poisson Factorization.

Parameters

- **k** (*int, optional, default: 100*) – The dimension of the latent factors.
- **max_iter** (*int, optional, default: 100*) – Maximum number of iterations for variational C2PF.
- **aux_info** (*array, required, shape (n_context_items,3)*) – The item-context matrix, noted C in the original paper, in the triplet sparse format: (row_id, col_id, value).
- **variant** (*string, optional, default: 'c2pf'*) – C2pf's variant: c2pf: 'c2pf', 'tc2pf' (tied-c2pf) or 'rc2pf' (reduced-c2pf). Please refer to the original paper for details.
- **name** (*string, optional, default: None*) – The name of the recommender model. If None, then "variant" is used as the default name of the model.
- **trainable** (*boolean, optional, default: True*) – When False, the model is not trained and Cornac assumes that the model already pre-trained (Theta, Beta and Xi are not None).
- **init_params** (*dictionary, optional, default: {'G_s':None, 'G_r':None, 'L_s':None, 'L_r':None, 'L2_s':None, 'L2_r':None, 'L3_s':None, 'L3_r':None}*) – List of initial parameters, e.g., init_params = {'G_s':G_s, 'G_r':G_r, 'L_s':L_s, 'L_r':L_r, 'L2_s':L2_s, 'L2_r':L2_r, 'L3_s':L3_s, 'L3_r':L3_r}, where G_s and G_r are of type csc_matrix or np.array with the same shape as Theta, see below). They represent respectively the "shape" and "rate" parameters of Gamma distribution over Theta. It is the same for L_s, L_r and Beta, L2_s, L2_r and Xi, L3_s, L3_r and Kappa.
- **Theta** (*csc_matrix, shape (n_users,k)*) – The expected user latent factors.
- **Beta** (*csc_matrix, shape (n_items,k)*) – The expected item latent factors.
- **Xi** (*csc_matrix, shape (n_items,k)*) – The expected context item latent factors multiplied by context effects Kappa, please refer to the paper below for details.

References

- Salah, Aghiles, and Hady W. Lauw. A Bayesian Latent Variable Model of User Preferences with Item Context. In IJCAI, pp. 2667-2674. 2018.

fit(*X*)

Fit the model to observations.

Parameters **X** (*scipy sparse matrix, required*) – the user-item preference matrix (traning data), in a scipy sparse format (e.g., csc_matrix).

rank (*user_index*, *known_items=None*)
Rank all test items for a given user.

Parameters

- **user_index** (*int, required*) – The index of the user for whom to perform item raking.
- **known_items** (*1d array, optional, default: None*) – A list of item indices already known by the user

Returns Array of item indices sorted (in decreasing order) relative to some user preference scores.

Return type Numpy 1d array

score (*user_index*, *item_indexes=None*)
Predict the scores/ratings of a user for a list of items.

Parameters

- **user_index** (*int, required*) – The index of the user for whom to perform score predictions.
- **item_indexes** (*1d array, optional, default: None*) – A list of item indexes for which to predict the rating score. When "None", score prediction is performed for all test items of the given user.

Returns Array containing the predicted values for the items of interest

Return type Numpy 1d array

3.3 Indexable Bayesian Personalized Ranking (IBPR)

@author: Dung D. Le (Andrew) <ddle.2015@smu.edu.sg>

class cornac.models.ibpr.recom_ibpr.**IBPR**(*k=20*, *max_iter=100*, *learning_rate=0.05*, *lamda=0.001*, *batch_size=100*, *name='ibpr'*, *trainable=True*, *verbose=False*, *init_params=None*)

Indexable Bayesian Personalized Ranking.

Parameters

- **k** (*int, optional, default: 20*) – The dimension of the latent factors.
- **max_iter** (*int, optional, default: 100*) – Maximum number of iterations or the number of epochs for SGD.
- **learning_rate** (*float, optional, default: 0.05*) – The learning rate for SGD.
- **lamda** (*float, optional, default: 0.001*) – The regularization parameter.
- **batch_size** (*int, optional, default: 100*) – The batch size for SGD.
- **name** (*string, optional, default: 'IBRP'*) – The name of the recommender model.
- **trainable** (*boolean, optional, default: True*) – When False, the model is not trained and Cornac assumes that the model already pre-trained (U and V are not None).

- **verbose** (*boolean, optional, default: False*) – When True, some running logs are displayed.
- **init_params** (*dictionary, optional, default: None*) – List of initial parameters, e.g., init_params = {'U':U, 'V':V} please see below the definition of U and V.
- **U** (*csc_matrix, shape (n_users,k)*) – The user latent factors, optional initialization via init_params.
- **V** (*csc_matrix, shape (n_items,k)*) – The item latent factors, optional initialization via init_params.

References

- Le, D. D., & Lauw, H. W. (2017, November). Indexable Bayesian personalized ranking for efficient top-k recommendation. In Proceedings of the 2017 ACM on Conference on Information and Knowledge Management (pp. 1389-1398). ACM.

fit (*train_set*)

Fit the model to observations.

Parameters **train_set** (*object of type TrainSet, required*) – An object contraining the user-item preference in csr scipy sparse format, as well as some useful attributes such as mappings to the original user/item ids. Please refer to the class TrainSet in the "data" module for details.

rank (*user_id*, *candidate_item_ids=None*)

Rank all test items for a given user.

Parameters

- **user_id** (*int, required*) – The index of the user for whom to perform item raking.
- **candidate_item_ids** (*1d array, optional, default: None*) – A list of item indices to be ranked by the user. If *None*, list of ranked known item indices will be returned

Returns Array of item indices sorted (in decreasing order) relative to some user preference scores.

Return type Numpy 1d array

score (*user_id*, *item_id*)

Predict the scores/ratings of a user for a list of items.

Parameters

- **user_id** (*int, required*) – The index of the user for whom to perform score predictions.
- **item_id** (*int, required*) – The index of the item to be scored by the user.

Returns The estimated score (e.g., rating) for the user and item of interest

Return type A scalar

3.4 Online Indexable Bayesian Personalized Ranking (OIBPR)

@author: Dung D. Le (Andrew) <ddle.2015@smu.edu.sg>

class cornac.models.online_ibpr.recom_online_ibpr.**OnlineIBPR**(*k=20*, *max_iter=100*, *learning_rate=0.05*, *lamda=0.001*, *batch_size=100*, *name='online_ibpr'*, *trainable=True*, *verbose=False*, *init_params=None*)

Online Indexable Bayesian Personalized Ranking.

Parameters

- **k** (*int, optional, default: 20*) – The dimension of the latent factors.
- **max_iter** (*int, optional, default: 100*) – Maximum number of iterations or the number of epochs for SGD.
- **learning_rate** (*float, optional, default: 0.05*) – The learning rate for SGD.
- **lamda** (*float, optional, default: 0.001*) – The regularization parameter.
- **batch_size** (*int, optional, default: 100*) – The batch size for SGD.
- **name** (*string, optional, default: 'IBRP'*) – The name of the recommender model.
- **trainable** (*boolean, optional, default: True*) – When False, the model is not trained and Cornac assumes that the model already pre-trained (U and V are not None).
- **verbose** (*boolean, optional, default: False*) – When True, some running logs are displayed.
- **init_params** (*dictionary, optional, default: None*) – List of initial parameters, e.g., init_params = {'U':U, 'V':V} please see below the definition of U and V.
- **U** (*csc_matrix, shape (n_users,k)*) – The user latent factors, optional initialization via init_params.
- **V** (*csc_matrix, shape (n_items,k)*) – The item latent factors, optional initialization via init_params.

References

- Le, D. D., & Lauw, H. W. (2017, November). Indexable Bayesian personalized ranking for efficient top-k recommendation. In Proceedings of the 2017 ACM on Conference on Information and Knowledge Management (pp. 1389-1398). ACM.

fit(*train_set*)

Fit the model to observations.

Parameters **train_set** (*object of type TrainSet, required*) – An object containing the user-item preference in csr scipy sparse format, as well as some useful attributes such as mappings to the original user/item ids. Please refer to the class TrainSet in the "data" module for details.

rank(*user_id*, *candidate_item_ids=None*)
Rank all test items for a given user.

Parameters

- **user_id** (*int, required*) – The index of the user for whom to perform item raking.
- **candidate_item_ids** (*1d array, optional, default: None*) – A list of item indices to be ranked by the user. If *None*, list of ranked known item indices will be returned

Returns Array of item indices sorted (in decreasing order) relative to some user preference scores.

Return type Numpy 1d array

score(*user_id*, *item_id*)
Predict the scores/ratings of a user for a list of items.

Parameters

- **user_id** (*int, required*) – The index of the user for whom to perform score predictions.
- **item_id** (*int, required*) – The index of the item to be scored by the user.

Returns The estimated score (e.g., rating) for the user and item of interest

Return type A scalar

3.5 Collaborative Ordinal Embedding (COE)

@author: Dung D. Le (Andrew) <ddle.2015@smu.edu.sg>

class cornac.models.coe.recom_coe.**COE**(*k=20*, *max_iter=100*, *learning_rate=0.05*, *lamda=0.001*, *batch_size=1000*, *name='coe'*, *trainable=True*, *verbose=False*, *init_params=None*)
Collaborative Ordinal Embedding.

Parameters

- **k** (*int, optional, default: 20*) – The dimension of the latent factors.
- **max_iter** (*int, optional, default: 100*) – Maximum number of iterations or the number of epochs for SGD.
- **learning_rate** (*float, optional, default: 0.05*) – The learning rate for SGD.
- **lamda** (*float, optional, default: 0.001*) – The regularization parameter.
- **batch_size** (*int, optional, default: 100*) – The batch size for SGD.
- **name** (*string, optional, default: 'IBRP'*) – The name of the recommender model.

- **trainable** (*boolean, optional, default: True*) – When False, the model is not trained and Cornac assumes that the model already pre-trained (U and V are not None).
- **verbose** (*boolean, optional, default: False*) – When True, some running logs are displayed.
- **init_params** (*dictionary, optional, default: None*) – List of initial parameters, e.g., init_params = {'U':U, 'V':V} please see below the definition of U and V.
- **U** (*csc_matrix, shape (n_users,k)*) – The user latent factors, optional initialization via init_params.
- **V** (*csc_matrix, shape (n_items,k)*) – The item latent factors, optional initialization via init_params.

References

- Le, D. D., & Lauw, H. W. (2016, June). Euclidean co-embedding of ordinal data for multi-type visualization. In Proceedings of the 2016 SIAM International Conference on Data Mining (pp. 396-404). Society for Industrial and Applied Mathematics.

fit (*train_set*)
Fit the model to observations.

Parameters **train_set** (*object of type TrainSet, required*) – An object containing the user-item preference in csr scipy sparse format, as well as some useful attributes such as mappings to the original user/item ids. Please refer to the class TrainSet in the "data" module for details.

rank (*user_id*, *candidate_item_ids=None*)
Rank all test items for a given user.

Parameters

- **user_id** (*int, required*) – The index of the user for whom to perform item raking.
- **candidate_item_ids** (*1d array, optional, default: None*) – A list of item indices to be ranked by the user. If *None*, list of ranked known item indices will be returned

Returns Array of item indices sorted (in decreasing order) relative to some user preference scores.

Return type Numpy 1d array

score (*user_id*, *item_id*)
Predict the scores/ratings of a user for a list of items.

Parameters

- **user_id** (*int, required*) – The index of the user for whom to perform score predictions.
- **item_id** (*int, required*) – The index of the item to be scored by the user.

Returns The estimated score (e.g., rating) for the user and item of interest

Return type A scalar

3.6 Visual Bayesian Personalized Ranking (VBPR)

@author: Guo Jingyao <jyguo@smu.edu.sg>

class cornac.models.vbpr.recom_vbpr.**VBPR**(*k=10*, *d=10*, *max_iter=100*, *aux_info=None*, *learning_rate=0.001*, *lamda=0.01*, *batch_size=100*, *name='vbpr'*, *trainable=True*, *init_params=None*)

Visual Bayesian Personalized Ranking.

Parameters

- **k** (*int, optional, default: 5*) – The dimension of the latent factors.
- **d** (*int, optional, default: 5*) – The dimension of the latent factors.
- **max_iter** (*int, optional, default: 100*) – Maximum number of iterations or the number of epochs for SGD.
- **shape (n_items, feature dimension), optional, default** (*aux_infondarray,*) – Image features of items
- **learning_rate** (*float, optional, default: 0.001*) – The learning rate for SGD.
- **lamda** (*float, optional, default: 0.01*) – The regularization parameter.
- **batch_size** (*int, optional, default: 100*) – The batch size for SGD.
- **name** (*string, optional, default: 'BRP'*) – The name of the recommender model.
- **trainable** (*boolean, optional, default: True*) – When False, the model is not trained and Cornac assumes that the model already pre-trained (U and V are not None).
- **init_params** (*dictionary, optional, default: None*) – List of initial parameters, e.g., init_params = {'U':U, 'V':V} please see below the definition of U and V.
- **U** (*ndarray, shape (n_users,k)*) – The user latent factors, optional initialization via init_params.
- **V** (*ndarray, shape (n_items,k)*) – The item latent factors, optional initialization via init_params.
- **E** (*ndarray, shape (d, feature dimension)*) – The matrix embedding deep CNN feature, optional initialization via init_params.
- **Ue** (*ndarray, shape (n_users, d)*) – The visual factors of users, optional initialization via init_params.

References

- HE, Ruining et MCAULEY, Julian. VBPR: Visual Bayesian Personalized Ranking from Implicit Feedback. In : AAAI. 2016. p. 144-150.

fit(*X*)

Fit the model to observations.

Parameters **X** (*scipy sparse matrix, required*) – the user-item preference matrix (traning data), in a scipy sparse format (e.g., csc_matrix).

rank(*user_index*, *known_items=None*)
Rank all test items for a given user.

Parameters

- **user_index** (*int, required*) – The index of the user for whom to perform item raking.
- **known_items** (*1d array, optional, default: None*) – A list of item indices already known by the user

Returns Array of item indices sorted (in decreasing order) relative to some user preference scores.

Return type Numpy 1d array

score(*user_index*, *item_indexes=None*)
Predict the scores/ratings of a user for a list of items.

Parameters

- **user_index** (*int, required*) – The index of the user for whom to perform score predictions.
- **item_indexes** (*1d array, optional, default: None*) – A list of item indexes for which to predict the rating score. When "None", score prediction is performed for all test items of the given user.

Returns Array containing the predicted values for the items of interest

Return type Numpy 1d array

3.7 Spherical k-means (Skmeans)

@author: Aghiles Salah <asalah@smu.edu.sg>

class cornac.models.skm.recom_skmeans.**SKMeans**(*k=5*, *max_iter=100*, *name='Skmeans'*, *trainable=True*, *tol=1e-06*, *verbose=True*, *init_par=None*)

Spherical k-means based recommender.

Parameters

- **k** (*int, optional, default: 5*) – The number of clusters.
- **max_iter** (*int, optional, default: 100*) – Maximum number of iterations.
- **name** (*string, optional, default: 'Skmeans'*) – The name of the recommender model.
- **trainable** (*boolean, optional, default: True*) – When False, the model is not trained and Cornac assumes that the model is already trained.
- **tol** (*float, optional, default: 1e-6*) – Relative tolerance with regards to skmeans' criterion to declare convergence.
- **verbose** (*boolean, optional, default: False*) – When True, some running logs are displayed.

- **init_par** (*numpy 1d array, optional, default: None*) – The initial object parition, 1d array contaning the cluster label (int type starting from 0) of each object (user). If par = None, then skmeans is initialized randomly.
- **centroids** (*csc_matrix, shape (k,n_users)*) – The maxtrix of cluster centroids.

References

- Salah, Aghiles, Nicoleta Rogovschi, and Mohamed Nadif. "A dynamic collaborative filtering system via a weighted clustering approach." Neurocomputing 175 (2016): 206-215.

fit (*train_set*)
Fit the model to observations.

Parameters **train_set** (*object of type TrainSet, required*) – An object contraining the user-item preference in csr scipy sparse format, as well as some useful attributes such as mappings to the original user/item ids. Please refer to the class TrainSet in the "data" module for details.

rank (*user_id*, *candidate_item_ids=None*)
Rank all test items for a given user.

Parameters

- **user_id** (*int, required*) – The index of the user for whom to perform item raking.
- **candidate_item_ids** (*1d array, optional, default: None*) – A list of item indices to be ranked by the user. If *None*, list of ranked known item indices will be returned

Returns Array of item indices sorted (in decreasing order) relative to some user preference scores.

Return type Numpy 1d array

score (*user_id*, *item_id*)
Predict the scores/ratings of a user for a list of items.

Parameters

- **user_id** (*int, required*) – The index of the user for whom to perform score predictions.
- **item_id** (*int, required*) – The index of the item to be scored by the user.

Returns The estimated score (e.g., rating) for the user and item of interest

Return type A scalar

3.8 Collaborative Deep Learning (CDL)

@author: Trieu Thi Ly Ly

class cornac.models.cdl.recom_cdl.**CDL**(*k=50*, *text_information=None*, *autoencoder_structure=None*, *lambda_u=0.1*, *lambda_v=0.01*, *lambda_w=0.01*, *lambda_n=0.01*, *a=1*, *b=0.01*, *autoencoder_corruption=0.3*, *learning_rate=0.001*, *keep_prob=1.0*, *batch_size=100*, *max_iter=100*, *name='CDL'*, *trainable=True*, *init_params=None*)

Collaborative Deep Learning.

Parameters

- **k** (*int, optional, default: 50*) – The dimension of the latent factors.
- **max_iter** (*int, optional, default: 100*) – Maximum number of iterations or the number of epochs for SGD.
- **shape (n_items, n_vocabularies), optional, default** (*text_informationndarray,*) – Bag-of-words features of items
- **optional, default** (*autoencoder_structurearray,*) – The number of neurons of encoder/ decoder layer for SDAE
- **learning_rate** (*float, optional, default: 0.001*) – The learning rate for AdamOptimizer.
- **lambda_u** (*float, optional, default: 0.1*) – The regularization parameter for users.
- **lambda_v** (*float, optional, default: 10*) – The regularization parameter for items.
- **lambda_w** (*float, optional, default: 0.1*) – The regularization parameter for SDAE weights.
- **lambda_n** (*float, optional, default: 1000*) – The regularization parameter for SDAE output.
- **a** (*float, optional, default: 1*) – The confidence of observed ratings.
- **b** (*float, optional, default: 0.01*) – The confidence of unseen ratings.
- **autoencoder_corruption** (*float, optional, default: 0.3*) – The corruption ratio for SDAE.
- **keep_prob** (*float, optional, default: 1.0*) – The probability that each element is kept in dropout of SDAE.
- **batch_size** (*int, optional, default: 100*) – The batch size for SGD.
- **name** (*string, optional, default: 'CDL'*) – The name of the recommender model.
- **trainable** (*boolean, optional, default: True*) – When False, the model is not trained and Cornac assumes that the model already pre-trained (U and V are not None).
- **init_params** (*dictionary, optional, default: None*) – List of initial parameters, e.g., init_params = {'U':U, 'V':V} please see below the definition of U and V.
- **U** (*ndarray, shape (n_users,k)*) – The user latent factors, optional initialization via init_params.

- **V** (*ndarray, shape (n_items,k)*) – The item latent factors, optional initialization via init_params.

References

- Hao Wang, Naiyan Wang, Dit-Yan Yeung. CDL: Collaborative Deep Learning for Recommender Systems. In : SIGKDD. 2015. p. 1235-1244.

fit (*X*)
Fit the model to observations.

Parameters

- **X** (*scipy sparse matrix, required*) – the user-item preference matrix (training data), in a scipy sparse format (e.g., csc_matrix).
- **csc_matrix)** (*(e.g.,)* –

rank (*user_index*, *known_items=None*)
Rank all test items for a given user.

Parameters

- **user_index** (*int, required*) – The index of the user for whom to perform item raking.
- **known_items** (*1d array, optional, default: None*) – A list of item indices already known by the user

Returns Array of item indices sorted (in decreasing order) relative to some user preference scores.

Return type Numpy 1d array

score (*user_index*, *item_indexes=None*)
Predict the scores/ratings of a user for a list of items.

Parameters

- **user_index** (*int, required*) – The index of the user for whom to perform score predictions.
- **item_indexes** (*1d array, optional, default: None*) – A list of item indexes for which to predict the rating score. When "None", score prediction is performed for all test items of the given user.

Returns Array containing the predicted values for the items of interest

Return type Numpy 1d array

3.9 Hierarchical Poisson Factorization (HPF)

@author: Aghiles Salah <asalah@smu.edu.sg>

class cornac.models.hpf.recom_hpf.**HPF** (*k=5*, *max_iter=100*, *name='HPF'*, *trainable=True*, *verbose=False*, *hierarchical=True*, *init_params={'G_r': None*, *'G_s': None*, *'L_r': None*, *'L_s': None}*)
Hierarchical Poisson Factorization.

Parameters

- **k** (*int, optional, default: 5*) – The dimension of the latent factors.
- **max_iter** (*int, optional, default: 100*) – Maximum number of iterations.
- **name** (*string, optional, default: 'HPF'*) – The name of the recommender model.
- **trainable** (*boolean, optional, default: True*) – When False, the model is not trained and Cornac assumes that the model is already pre-trained (Theta and Beta are not None).
- **verbose** (*boolean, optional, default: False*) – When True, some running logs are displayed.
- **hierarchical** (*boolean, optional, default: True*) – When False, PF is used instead of HPF.
- **init_params** (*dictionary, optional, default: {'G_s':None, 'G_r':None, 'L_s':None, 'L_r':None}*) – List of initial parameters, e.g., init_params = {'G_s':G_s, 'G_r':G_r, 'L_s':L_s, 'L_r':L_r}, where G_s and G_r are of type csc_matrix or np.array with the same shape as Theta, see below). They represent respectively the "shape" and "rate" parameters of Gamma distribution over Theta. Similarly, L_s, L_r are the shape and rate parameters of the Gamma over Beta.
- **Theta** (*csc_matrix, shape (n_users,k)*) – The expected user latent factors.
- **Beta** (*csc_matrix, shape (n_items,k)*) – The expected item latent factors.

References

- Gopalan, Prem, Jake M. Hofman, and David M. Blei. Scalable Recommendation with Hierarchical Poisson Factorization. In UAI, pp. 326-335. 2015.

fit (*train_set*)

Fit the model to observations.

Parameters **train_set** (*object of type TrainSet, required*) – An object containing the user-item preference in csr scipy sparse format, as well as some useful attributes such as mappings to the original user/item ids. Please refer to the class TrainSet in the "data" module for details.

rank (*user_id*, *candidate_item_ids=None*)

Rank all test items for a given user.

Parameters

- **user_id** (*int, required*) – The index of the user for whom to perform item raking.
- **candidate_item_ids** (*1d array, optional, default: None*) – A list of item indices to be ranked by the user. If *None*, list of ranked known item indices will be returned

Returns Array of item indices sorted (in decreasing order) relative to some user preference scores.

Return type Numpy 1d array

score (*user_id*, *item_id*)

Predict the scores/ratings of a user for a list of items.

Parameters

- **user_id** (*int, required*) – The index of the user for whom to perform score predictions.
- **item_id** (*int, required*) – The index of the item to be scored by the user.

Returns The estimated score (e.g., rating) for the user and item of interest

Return type A scalar

3.10 Bayesian Personalized Ranking (BPR)

@author: Guo Jingyao <jyguo@smu.edu.sg>

class cornac.models.bpr.recom_bpr.**BPR**(*k=5, max_iter=100, learning_rate=0.001, lamda=0.001, batch_size=100, name='bpr', trainable=True, verbose=False, init_params={'U': None, 'V': None}*)

Bayesian Personalized Ranking.

Parameters

- **k** (*int, optional, default: 5*) – The dimension of the latent factors.
- **max_iter** (*int, optional, default: 100*) – Maximum number of iterations or the number of epochs for SGD.
- **learning_rate** (*float, optional, default: 0.001*) – The learning rate for SGD.
- **lamda** (*float, optional, default: 0.001*) – The regularization parameter.
- **batch_size** (*int, optional, default: 100*) – The batch size for SGD.
- **name** (*string, optional, default: 'BRP'*) – The name of the recommender model.
- **trainable** (*boolean, optional, default: True*) – When False, the model is not trained and Cornac assumes that the model already pre-trained (U and V are not None).
- **verbose** (*boolean, optional, default: False*) – When True, some running logs are displayed.
- **init_params** (*dictionary, optional, default: {'U':None, 'V':None}*) – List of initial parameters, e.g., init_params = {'U':U, 'V':V}. U: a csc_matrix of shape (n_users,k), containing the user latent factors. V: a csc_matrix of shape (n_items,k), containing the item latent factors.

References

- Rendle, Steffen, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. BPR: Bayesian personalized ranking from implicit feedback. In UAI, pp. 452-461. 2009.

fit(*train_set*)

Fit the model to observations.

Parameters **train_set** (*object of type TrainSet, required*) – An object containing the user-item preference in csr scipy sparse format, as well as some useful attributes such as mappings to the original user/item ids. Please refer to the class TrainSet in the "data" module for details.

rank(*user_id*, *candidate_item_ids=None*)
Rank all test items for a given user.

Parameters

- **user_id** (*int, required*) – The index of the user for whom to perform item raking.
- **candidate_item_ids** (*1d array, optional, default: None*) – A list of item indices to be ranked by the user. If *None*, list of ranked known item indices will be returned

Returns Array of item indices sorted (in decreasing order) relative to some user preference scores.

Return type Numpy 1d array

score(*user_id*, *item_id*)
Predict the scores/ratings of a user for a list of items.

Parameters

- **user_id** (*int, required*) – The index of the user for whom to perform score predictions.
- **item_id** (*int, required*) – The index of the item to be scored by the user.

Returns The estimated score (e.g., rating) for the user and item of interest

Return type A scalar

3.11 Probabilitic Matrix Factorization (PMF)

@author: Aghiles Salah

class cornac.models.pmf.recom_pmf.**PMF**(*k=5*, *max_iter=100*, *learning_rate=0.001*, *gamma=0.9*, *lamda=0.001*, *name='PMF'*, *variant='non_linear'*, *trainable=True*, *verbose=False*, *init_params={'U': None, 'V': None}*)
Probabilistic Matrix Factorization.

Parameters

- **k** (*int, optional, default: 5*) – The dimension of the latent factors.
- **max_iter** (*int, optional, default: 100*) – Maximum number of iterations or the number of epochs for SGD.
- **learning_rate** (*float, optional, default: 0.001*) – The learning rate for SGD_RMSProp.
- **gamma** (*float, optional, default: 0.9*) – The weight for previous/current gradient in RMSProp.
- **lamda** (*float, optional, default: 0.001*) – The regularization parameter.
- **name** (*string, optional, default: 'PMF'*) – The name of the recommender model.

- **variant** (*{"linear","non_linear"}, optional, default: 'non_linear'*) – Pmf variant. If 'non_linear', the Gaussian mean is the output of a Sigmoid function. If 'linear' the Gaussian mean is the output of the identity function.
- **trainable** (*boolean, optional, default: True*) – When False, the model is not trained and Cornac assumes that the model already pre-trained (U and V are not None).
- **verbose** (*boolean, optional, default: False*) – When True, some running logs are displayed.
- **init_params** (*dictionary, optional, default: {'U':None, 'V':None}*) – List of initial parameters, e.g., init_params = {'U':U, 'V':V}. U: a csc_matrix of shape (n_users,k), containing the user latent factors. V: a csc_matrix of shape (n_items,k), containing the item latent factors.

References

- Mnih, Andriy, and Ruslan R. Salakhutdinov. Probabilistic matrix factorization. In NIPS, pp. 1257-1264. 2008.

fit (*train_set*)

Fit the model to observations.

Parameters **train_set** (*object of type TrainSet, required*) – An object containing the user-item preference in csr scipy sparse format, as well as some useful attributes such as mappings to the original user/item ids. Please refer to the class TrainSet in the "data" module for details.

rank (*user_id*, *candidate_item_ids=None*)

Rank all test items for a given user.

Parameters

- **user_id** (*int, required*) – The index of the user for whom to perform item raking.
- **candidate_item_ids** (*1d array, optional, default: None*) – A list of item indices to be ranked by the user. If *None*, list of ranked known item indices will be returned

Returns Array of item indices sorted (in decreasing order) relative to some user preference scores.

Return type Numpy 1d array

score (*user_id*, *item_id*)

Predict the scores/ratings of a user for a list of items.

Parameters

- **user_id** (*int, required*) – The index of the user for whom to perform score predictions.
- **item_id** (*int, required*) – The index of the item to be scored by the user.

Returns The estimated score (e.g., rating) for the user and item of interest

Return type A scalar

CHAPTER 4

metrics

4.1 Normalized Discount Cumulative Gain (NDCG)

class cornac.metrics.**NDCG**(*k=-1*)

Normalized Discount Cumulative Gain.

Parameters

- **k** (*int, optional, default: -1 (all)*) – The number of items in the top@k list, if None then all items are considered to compute NDCG.
- **name** (*string, value: 'NDCG'*) – Name of the measure.
- **type** (*string, value: 'ranking'*) – Type of the metric, e.g., "ranking".

References

https://en.wikipedia.org/wiki/Discounted_cumulative_gain

4.2 Normalized Cumulative Reciprocal Rank (NCRR)

class cornac.metrics.**NCRR**(*k=-1*)

Normalized Cumulative Reciprocal Rank.

Parameters

- **k** (*int, optional, default: -1 (all)*) – The number of items in the top@k list, if None then all items are considered to compute NDCG.
- **name** (*string, value: 'NCRR'*) – Name of the measure.
- **type** (*string, value: 'ranking'*) – Type of the metric, e.g., "ranking".

4.3 Mean Reciprocal Rank (MRR)

class cornac.metrics.**MRR**
Mean Reciprocal Rank.

Parameters

- **name** (*string, value: 'MRR'*) – Name of the measure.
- **type** (*string, value: 'ranking'*) – Type of the metric, e.g., "ranking".

References

https://en.wikipedia.org/wiki/Mean_reciprocal_rank

4.4 Precision

class cornac.metrics.**Precision**(*k=-1*)
Precision@K.

Parameters

- **k** (*int, optional, default: -1 (all)*) – The number of items in the top@k list.
- **name** (*string, value: 'Precision@k'*) – Name of the measure.
- **type** (*string, value: 'ranking'*) – Type of the metric, e.g., "ranking".

4.5 Recall

class cornac.metrics.**Recall**(*k=-1*)
Recall@K.

Parameters

- **k** (*int, optional, default: -1 (all)*) – The number of items in the top@k list.
- **name** (*string, value: 'Recall@k'*) – Name of the measure.
- **type** (*string, value: 'ranking'*) – Type of the metric, e.g., "ranking".

4.6 Fmeasure (F1)

class cornac.metrics.**FMeasure**(*k=-1*)
F-measure@K@.

Parameters

- **k** (*int, optional, default: -1 (all)*) – The number of items in the top@k list.
- **name** (*string, value: 'F1@k'*) – Name of the measure.

- **type** (*string, value: 'ranking'*) – Type of the metric, e.g., "ranking".

4.7 Mean Absolute Error (MAE)

class cornac.metrics.**MAE**
Mean Absolute Error.

Parameters **name** (*string, value: 'MAE'*) – Name of the measure.

4.8 Root Mean Squared Error (RMSE)

class cornac.metrics.**RMSE**
Root Mean Squared Error.

Parameters **name** (*string, value: 'RMSE'*) – Name of the measure.

CHAPTER 5

evaluation methods

5.1 Base Method

@author: Quoc-Tuan Truong <tuantq.vnu@gmail.com>

class cornac.eval_methods.base_method.**BaseMethod**(*data=None*, *data_format='UIR'*, *train_set=None*, *test_set=None*, *total_users=None*, *total_items=None*, *rating_threshold=1.0*, *exclude_unknowns=False*, *verbose=False*)

Base Evaluation Method

Parameters

- **train_set** (TrainSet, optional, default: None) – The training data.
- **test_set** (TestSet, optional, default: None) – The test data.
- **total_users** (*int, optional, default: None*) – Total number of unique users in the data including train, val, and test sets
- **total_users** – Total number of unique items in the data including train, val, and test sets
- **rating_threshold** (*float, optional, default: 1*) – The minimum value that is considered to be a good rating used for ranking, e.g, if the ratings are in {1, . . . , 5}, then good_rating = 4.
- **exclude_unknowns** (*bool, optional, default: False*) – Ignore unknown users and items (cold-start) during evaluation and testing
- **verbose** (*bool, optional, default: False*) – Output running log

5.2 Ratio Split

@author: Quoc-Tuan Truong <tuantq.vnu@gmail.com>

class cornac.eval_methods.ratio_split.**RatioSplit**(*data*, *data_format='UIR'*, *test_size=0.2*, *val_size=0.0*, *rating_threshold=1.0*, *shuffle=True*, *random_state=None*, *exclude_unknowns=False*, *verbose=False*)

Train-Test Split Evaluation Method.

Parameters

- **data** (*.., required*) – The input data in the form of triplets (user, item, rating).
- **data_format** (*str, optional, default: "UIR"*) – The format of input data: - UIR: (user, item, rating) triplet data - UIRT: (user, item , rating, timestamp) quadruplet data
- **test_size** (*float, optional, default: 0.2*) – The proportion of the test set, if > 1 then it is treated as the size of the test set.
- **val_size** (*float, optional, default: 0.0*) – The proportion of the validation set, if > 1 then it is treated as the size of the validation set.
- **rating_threshold** (*float, optional, default: 1.*) – The minimum value that is considered to be a good rating used for ranking, e.g, if the ratings are in {1, . . . , 5}, then rating_threshold = 4.
- **shuffle** (*bool, optional, default: True*) – Shuffle the data before splitting.
- **exclude_unknowns** (*bool, optional, default: False*) – Ignore unknown users and items (cold-start) during evaluation and testing
- **verbose** (*bool, optional, default: False*) – Output running log

5.3 Cross Validation

@author: Aghiles Salah

class cornac.eval_methods.cross_validation.**CrossValidation**(*data*, *data_format='UIR'*, *n_folds=5*, *rating_threshold=1.0*, *partition=None*, *exclude_unknowns=True*, *verbose=False*)

Cross Validation Evaluation Method.

Parameters

- **data** (*.. , required*) – Input data in the triplet format (user_id, item_id, rating_val).
- **n_folds** (*int, optional, default: 5*) – The number of folds for cross validation.

- **rating_threshold** (*float, optional, default: 1.*) – The minimum value that is considered to be a good rating, e.g, if the ratings are in {1, ... ,5}, then rating_threshold = 4.
- **partition** (*array-like, shape (n_observed_ratings,), optional, default: None*) – The partition of ratings into n_folds (fold label of each rating) If None, random partitioning is performed to assign each rating into a fold.
- **rating_threshold** – The minimum value that is considered to be a good rating used for ranking, e.g, if the ratings are in {1, ..., 5}, then rating_threshold = 4.
- **exclude_unknowns** (*bool, optional, default: False*) – Ignore unknown users and items (cold-start) during evaluation and testing
- **verbose** (*bool, optional, default: False*) – Output running log

CHAPTER 6

experiment

class cornac.experiment.**Experiment**(*eval_method*, *models*, *metrics*, *user_based=True*, *verbose=False*)

Experiment Class

Parameters

- **eval_method** (*BaseMethod object, required*) – The evaluation method (e.g., RatioSplit).
- **models** (*array of objects Recommender, required*) – A collection of recommender models to evaluate, e.g., [C2pf, Hpf, Pmf].
- **metrics** (*array of object metrics, required*) – A collection of metrics to use to evaluate the recommender models, e.g., [Ndcg, Mrr, Recall].
- **user_based** (*bool, optional, default: True*) – Performance will be averaged based on number of users for rating metrics. If *False*, results will be averaged over number of ratings.
- **avg_results** (*DataFrame, default: None*) – The average result per model.
- **user_results** (*dictionary, default: {}*) – Results per user for each model. Result of user u, of metric m, of model d will be user_results[d][m][u]

Python Module Index

C

Index

S

V