Package 'ptf'

October 14, 2022

Type Package

Title Probit Tensor Factorization

Version 0.0.1

Date 2021-06-13

Author Ye Liu

Maintainer Ye Liu <yliu87@ncsu.edu>

Description Efficient algorithms to implement Probit Tensor Factorization (PTF) model for statistical relational learning, which not only inherits the computation efficiency from the classic tensor factorization model but also accounts for the binary nature of relational data. The methodology is based on Ye Liu (2021) <https://repository.lib.ncsu.edu/bitstream/handle/1840.20/37507/etd.pdf?sequence=1> ``Computational Methods for Complex Models with Latent Structure".

License GPL

Imports Rcpp (>= 0.12.9), Matrix(>= 1.2), rARPACK (>= 0.11), plyr (>= 1.8.4)

LinkingTo Rcpp, RcppArmadillo

RoxygenNote 6.1.1

Encoding UTF-8

NeedsCompilation yes

Repository CRAN

Date/Publication 2021-06-15 14:50:15 UTC

R topics documented:

package-ptf *probit tensor factorization*

Description

Package:	ptf
Type:	Package
Version:	0.0.1
Date:	2020-09-30
License:	GPL
LazyLoad:	yes
LazyData:	yes

Author(s)

Ye Liu Maintainer: Ye Liu <yeliu.nikki@gmail.com>

ptf *Fit a Probit Tensor Factorization Model*

Description

Fit a Probit Tensor Factorization Model

Usage

```
ptf(X, k, n, r = 0, max_iter = 1000, tol = 1e-08, tol_M = 1e-05,
  iter_M_max = 2, print_option = TRUE)
```

Arguments

X	response data, which is a three-way array of size n by n by k
k	number of relations
n	number of entities
r	decomposition rank
max_iter	max number of iterations
tol	tolerance of absolute change in likelihood
tol_M	tolerance of absolute change in the M step
iter_M_max	max number of iterations for M step
print_option	whether print loss for each iteration or not

Value

fitted parameters

References

@references Ye Liu, 2021. Computational Methods for Complex Models with Latent Structure. PhD thesis with link at https://repository.lib.ncsu.edu/bitstream/handle/1840.20/37507/etd.pdf?sequence=1

Examples

```
n <- 20
k <- 10
r <- 3
p <- c(n, n, k)
X <- array(rnorm(prod(p)),dim=p)
X_binary <- ifelse(X < -1.5,1,0)
X_binary_with_missing <- X_binary
num_missing <- 200
missing_index <- data.frame(x1=sample(1:n,num_missing,replace=TRUE),
                            x2=sample(1:n,num_missing,replace=TRUE),
                            x3=sample(1:k,num_missing,replace=TRUE))
for(i in 1:num_missing){
 X_binary[missing_index[i,1],
          missing_index[i,2],
          missing_index[i,3]] <- NA
}
result <- ptf(X_binary_with_missing,k,n,r,print_option=FALSE)
```

---

ptf_sparse *Fit a Probit RESCAL model (sparse representation).*

Description

Fit a Probit RESCAL model (sparse representation).

Usage

```
ptf_sparse(df, n, k, r = 0, max_iter = 500, tol = 1e-08,
  tol_M = 1e-05, iter_M_max = 2, print_option = TRUE)
```

Arguments

df	a four-column dataframe with columns (1) ent1 (2) ent2 (3) relation (4) true, which is an indicator of whether there is such a relation from ent1 to ent2
n	number of entities
k	number of relations
r	decomposition rank

max_iter	max number of iterations
tol	tolerance of absolute change in ,elihood
tol_M	tolerance of absolute change in the M step
iter_M_max	max number of iterations for M step
print_option	whether print log-likelihood for each iteration or not

Value

fitted parameters

References

Ye Liu, 2021. Computational Methods for Complex Models with Latent Structure. PhD thesis with link at https://repository.lib.ncsu.edu/bitstream/handle/1840.20/37507/etd.pdf?sequence=1

Examples

```
n <- 20
k <- 10
r <- 3
A <- matrix(runif(n*r,0,1),n,r)
mean.k <- runif(k,-0.1,-0.01)
W <- lapply(1:k, function(i) matrix(rnorm(r*r,mean.k[i],1),r,r))
Z <- lapply(1:k, function(i) A%*%W[[i]]%*%t(A))
df <- data.frame(expand.grid(0:(n-1),0:(n-1),0:(k-1)))
df <-  df[sample(nrow(df),2000),]
colnames(df) <- c('ent1','ent2','relation')
awa <- function(row,A,W){
  inx1 <- row[1] + 1
  inx2 <- row[2] + 1
  rel <- row[3] + 1
  out <- A[inx1,] %*% W[[rel]] %*% A[inx2,]
}
z <- apply(df,1,function(row) awa(row,A,W))
df$true <- ifelse(z>0,1,0)
result2 <- ptf_sparse(df,n,k,r)
```

Rescal *RESCAL Model*

Description

RESCAL Model

Usage

```
Rescal(X, k, n, r, max_iter = 500)
```

Arguments

X	response data, which is a three-way array of size n by n by k
k	number of relations
n	number of entities
r	decomposition rank
max_iter	max number of iterations

Value

fitted parameters

References

Nickel, Maximilian, Volker Tresp, and Hans-Peter Kriegel. "A three-way model for collective learning on multi-relational data.", Icml. 2011.

Index