Package: JQL (via r-universe)

October 26, 2024

Type Package

Title Jump Q-Learning for Individualized Interval-Valued Dose Rule

Version 3.6.9

Depends caret, pdist, stats, randomForest

Author Hengrui Cai <hcai5@ncsu.edu>, Chengchun shi <cshi4@ncsu.edu>, Rui Song <rsong@ncsu.edu>, Wenbin Lu <wlu4@ncsu.edu>

Maintainer Hengrui Cai <hcai5@ncsu.edu>

Description We provide tools to estimate the individualized interval-valued dose rule (I2DR) that maximizes the expected beneficial clinical outcome for each individual and returns an optimal interval-valued dose, by using the jump Q-learning (JQL) method. The jump Q-learning method directly models the conditional mean of the response given the dose level and the baseline covariates via jump penalized least squares regression under the framework of Q learning. We develop a searching algorithm by dynamic programming in order to find the optimal I2DR with the time complexity O(n2) and spatial complexity O(n). To alleviate the effects of misspecification of the Q-function, a residual jump Q-learning is further proposed to estimate the optimal I2DR. The outcome of interest includes the best partition of the entire dosage of interest, the regression coefficients of each partition, and the value function under the estimated I2DR as well as the Wald-type confidence interval of value function constructed through the Bootstrap.

License LGPL-3

Encoding UTF-8

LazyData false

NeedsCompilation no

Date/Publication 2019-11-15 17:40:06 UTC

Repository https://anoce-cvae.r-universe.dev

RemoteUrl https://github.com/cran/JQL

RemoteRef HEAD

RemoteSha 81242e031811640a1dd5b8fbdb005250f0311387

Contents

`find.I2DR` *Estimating the Individualized Interval-valued Dose Rule via (Residual) Jump Q-learning.*

Description

This function estimates the optimal Individualized Interval-valued Dose Rule (I2DR), and calculates a Wald-type confidence interval for the value function under the estimated optimal I2DR via Bootstrap.

Usage

```
find.I2DR(Y,A,X,cm=6,method='JQL',Gamma.list=seq(from=1,to=20,by=2)/5,
Lambda.list=seq(from=1,to=20,by=2)/5,RF_A.list=c(0,0.25,0.5,0.75,1),
folds_num=5,alpha=0.95,nboots=500)
```

Arguments

`Y`	The patient's associated response/outcome, the larger the better by convention.
`A`	The dose level received by each patient, should be continuous.
`X`	The patient's baseline covariates, could be a matrix, including continous or discrete covariates.
`cm`	The constent cm in m=n/cm, where m is the number of total subinterval that diverges with sample size n. The default value is 6.
`method`	Two methods are available, Jump Q-learning ('JQL') and Residual Jump Q-learning ('RJQL'). The default method is 'JQL'.
`Gamma.list`	The candidate tuning paramter space for c1 in penalty term gamma=c1 log(n)/n. The default value is seq(from=1,to=20,by=2)/5. If the length of Gamma.list is 1, then the tuning process will be skipped.
`Lambda.list`	The candidate tuning paramter space for c2 in penalty term lambda=c2 log(n)/n. The default value is seq(from=1,to=20,by=2)/5. If the length of Lambda.list is 1, then the tuning process will be skipped.
`RF_A.list`	The candidate tuning paramter space for A in fitted E(Y|A=a,X) by Random Forest Regression for method 'RJQL' only. The default value is c(0,0.25,0.5,0.75,1). If the length of RF_A.list is 1, then the tuning process will be skipped.
`folds_num`	The number of the folds in the cross-validation process. The default value is 5.
`alpha`	The Confidence level. The default level is 0.95.
`nboots`	The number of Bootstrap. The default number is 500.

Value

An object of class "I2DR" which is a list with components:

Partition	A partition of the entire dose range.
Beta	The regression coefficients for each partition.
Value	The estimated value function under our proposed I2DR.
low_bd	The lower bound of the confidence interval.
up_bd	The upper bound of the confidence interval.
method	The method used to find the I2DR.

References

Jump Q-learning for Individualized Interval-valued Dose Rule.

Examples

```
n=50
d=4
x=matrix(runif(n*(d-1),-1,1),nrow=n,ncol=d-1)
a=runif(n,0,1)
y=(1+x[,1])*(a>=0&a<0.35)+(x[,1]-x[,2])*(a>=0.35&a<0.65)+(1-x[,2])*(a>=0.65&a<=1)+rnorm(n,0,1)
find.I2DR(Y=y,A=a,X=x)
```

---

opt.dose — *Optimal Interval-valued Dose under the Individualized Interval-valued Dose Rule via (Residual) Jump Q-learning.*

Description

This function assigns each individual to one of the subintervals of the entire dosage according to his/her baseline covariates under the estimated I2DR.

Usage

```
opt.dose(X,I2DR)
```

Arguments

X	The patient's baseline covariates, coule be a matrix, including continous or discrete covariates.
I2DR	The Individualized Interval-valued Dose Rule found by the function "JQL" or "RJQL".

Value

opt.dose	The optimal Interval-valued dosage for each individual.

References

Jump Q-learning for Individualized Interval-valued Dose Rule.

Examples

```
n=50
d=4
x=matrix(runif(n*(d-1),-1,1),nrow=n,ncol=d-1)
a=runif(n,0,1)
y=(1+x[,1])*(a>=0&a<0.35)+(x[,1]-x[,2])*(a>=0.35&a<0.65)+(1-x[,2])*(a>=0.65&a<=1)+rnorm(n,0,1)
rule=find.I2DR(Y=y,A=a,X=x)
n0=10
xnew=matrix(runif(n0*(d-1),-1,1),nrow=n0,ncol=d-1)
opt.dose(X=xnew,I2DR=rule)
```

---

tune.JQL *Tuning function via k-fold cross vaidation for Jump Q-learning.*

Description

This function uses the cross-validation to train the best tuning parameters lambda_n and gamma_n for Jump Q-learning.

Usage

```
tune.JQL(sample,cm=6,Gamma.list=seq(from=1,to=20,by=2)/5,
Lambda.list=seq(from=1,to=20,by=2)/5,folds_num=5)
```

Arguments

sample	The training dataset (Y,A,X), where Y is the patient's associated response/outcome, A is the dose level received by each patient, and X is the patient's baseline covariates.
cm	The constent cm in m=n/cm, where m is the number of total subinterval that diverges with sample size n. The default value is 6.
Gamma.list	The candidate tuning paramter space for c1 in penalty term gamma=c1 log(n)/n. The default value is seq(from=1,to=20,by=2)/5.
Lambda.list	The candidate tuning paramter space for c2 in penalty term lambda=c2 log(n)/n. The default value is seq(from=1,to=20,by=2)/5.
folds_num	The number of the folds in the cross-validation process. The default value is 5.

Value

best_gamma	The best tuning parameter gamma by minimuming the least square loss function.
best_lambda	The best tuning parameter lambda by minimuming the least square loss function.

References

Jump Q-learning for Individualized Interval-valued Dose Rule.

Examples

```
n=50
d=4
x=matrix(runif(n*(d-1),-1,1),nrow=n,ncol=d-1)
a=runif(n,0,1)
y=(1+x[,1])*(a>=0&a<0.35)+(x[,1]-x[,2])*(a>=0.35&a<0.65)+(1-x[,2])*(a>=0.65&a<=1)+rnorm(n,0,1)
sample=data.frame(y=y,a=a,x=x)
tune.JQL(sample)
```

`tune.RJQL` *Tuning function via k-fold cross vaidation for Residual Jump Q-learning.*

Description

This function uses the cross-validation to train the best tuning parameters lambda_n and gamma_n for Residual Jump Q-learning.

Usage

```
tune.RJQL(sample,cm=6,Gamma.list=seq(from=1,to=20,by=2)/5,
Lambda.list=seq(from=1,to=20,by=2)/5,RF_A.list=c(0,0.25,0.5,0.75,1),folds_num=5)
```

Arguments

`sample`	The training dataset (Y,A,X), where Y is the patient's associated response/outcome, A is the dose level received by each patient, and X is the patient's baseline covariates.
`cm`	The constent cm in m=n/cm, where m is the number of total subinterval that diverges with sample size n. The default value is 6.
`Gamma.list`	The candidate tuning paramter space for c1 in penalty term gamma=c1 log(n)/n. The default value is seq(from=1,to=20,by=2)/5.
`Lambda.list`	The candidate tuning paramter space for c2 in penalty term lambda=c2 log(n)/n. The default value is seq(from=1,to=20,by=2)/5.
`RF_A.list`	The candidate tuning paramter space for A in fitted E(Y|A=a,X) by Random Forest Regression for method 'RJQL' only. The default value is c(0,0.25,0.5,0.75,1).
`folds_num`	The number of the folds in the cross-validation process. The default value is 5.

Value

`best_gamma`	The best tuning parameter gamma by minimuming the least square loss function.
`best_lambda`	The best tuning parameter lambda by minimuming the least square loss function.
`best_a`	The best tuning parameter a to fit random forest by minimuming the least square loss function.

References

Jump Q-learning for Individualized Interval-valued Dose Rule.

Examples

```
n=50
d=4
x=matrix(runif(n*(d-1),-1,1),nrow=n,ncol=d-1)
a=runif(n,0,1)
y=(1+x[,1])*(a>=0&a<0.35)+(x[,1]-x[,2])*(a>=0.35&a<0.65)+(1-x[,2])*(a>=0.65&a<=1)+rnorm(n,0,1)
sample=data.frame(y=y,a=a,x=x)
tune.RJQL(sample)
```

Index