Package 'AddInfo'

August 17, 2016

Title Performs the adaptive external information estimator from Tarima and Pavlov (2006)

Version 0.1

Description Tarima and Pavlov (2006) prevents a method of estimating a parameter of interest while using information provided from an external source such as previous publications. The external information need not be the same parameter as the one of interest, as long as the parameter (s) from the external information can be calculated using data you have it can be used to decrease the variance in estimating your parameter of interest. The paper also describes an adaptive method of doing this, which is exactly what is implemented here.

LazyData TRUE

RoxygenNote 5.0.1

Depends Matrix

R topics documented:

Ext.est 1

Index **5**

`Ext.est` *Ext.est*

Description

This function allows you to perform the external data adpative estimation described in "Using auxilary information in statistical function estimate", ESAIM (2006)

Usage

```
Ext.est(data, func.theta, means, vars, funcs, B = 500, eig.keep = 1)
```

Arguments

`data`	Dataset to be used
`func.theta`	The function you wished to estimate, must return an S by 1 vector
`means`	A list of mean vectors
`vars`	A list of covariance matrices (of the estimators, not the population)
`funcs`	A list of functions, to return vectors of size of vectors in means
`B`	Number of bootstrap samples, defaults to 500
`eig.keep`	Keep eig.keep*100 percent of eigen values when inverting a matrix, defaults to 1

References

- Tarima S, Pavlov D. Using auxilary information in probability estimation. *ESAIM: Probability and Statistics* 2006; 10: 11-23.
- Tarima S, Slavova S, Fritsch T, Hall, L. Probability estimation when some observations are grouped. *Statistics in Medicine* 2007; 26: 1745-61.

Examples

```
###  This example will walk the user through how to use the Ext.est function.
###  It will deal with estimating the survival probability of those
###  with a certain form of cancer using Kaplain Meier estimation.

require(survival)

###  Use Survival information from Klein and Moeschberger (2003)
deaths = c(1,3,3,4,10,13,13,16,16,24,26,27,
     28,30,30,32,41,51,65,67,70,72,73,77,91,93,96,
     100,104,157,167,1,3,4,5,5,8,12,13,18,23,26,
     27,30,42,56,62,69,104,104,112,129,181)

cens = c(61,74,79,80,81,87,87,88,
     89,93,97,101,104,108,109,120,131,150,231,240,400,
     8,67,76,104,176,231)

times = c(deaths, cens)

status = c(rep(1, length(deaths)), rep(0, length(cens)))

data = data.frame(times, status)

###  For basic KM estimate
km = survfit(Surv(times, status) ~ 1, data = data)

###  But according to
###  http://www.cancerresearchuk.org/cancer-help/type/mouth-cancer/treatment/
###  statistics-and-outlook-for-mouth-cancers
###  About 50% will be alive after 5 years
###  The Ext.est function allows us to incorporate this information into
###  our own survival estimates using our own data

###  Let's estimate 32 month survival. We need to create a
###  function that takes in only a dataset like ours and
### returns an estimate of the 32 month survival.
### Note the only argument should be a dataframe.
func.theta = function(d) {
     km = survfit(Surv(times, status) ~ 1, data = d)
     kmest = stepfun(km$time, c(1, km$surv))
     kmest(32)
}

func.theta(data) # = 0.634 is our estimate of survival based on just our data

### With approx 95% CI: 0.5363 to 0.750

### Next we need to create a means list that is our external estimate (at 5 years)
means = list()

means[[1]] = 0.5

### Next we need a variance list, for our estimate at 5 years.
### In this example, we'll treat 0.5 as exact, perfect, information
### So it has 0 variance
vars = list()

vars[[1]] = 0

### Lastly, we need to write a list of functions, that will estimate the
### outside data estimate on our own data
funcs = list()

funcs[[1]] = function(d) {
     km = survfit(Surv(times, status) ~ 1, data = d)
     kmest = stepfun(km$time, c(1, km$surv))
     kmest(60)
}

### Now we can use the function to add this outside information
### to our estimate of 32 month survival
res = Ext.est(data, func.theta, means, vars, funcs, B = 500)

### The first element of the returned list is the
### adapted estimate of 32 month survival.
t = res[[1]]

### And the second is it's variance.
v = res[[2]]

lower = t - 1.96*sqrt(v)
upper = t + 1.96*sqrt(v)

### Now our estimate, using the external data is 0.56, with interval
### 0.51 to 0.61, a much smaller interval than with just our data

### Using exact information isn't really realistic, let's suppose the 0.5 estimate
### at 5 years was due to 20 patients. All we need to change is the vars.
### So the variance of this estimated proportion is p(1-p)/n.
vars[[1]] = (0.5*0.5)/20

### Then we can run Ext.est as before and get a new estimate and CI.
```

```
res = Ext.est(data, func.theta, means, vars, funcs, B = 500)

t = res[[1]]

v = res[[2]]

lower = t - 1.96*sqrt(v)
upper = t + 1.96*sqrt(v)

###  Now the estimate, using the uncertain external data, is still aobut 0.62,
###  which shows that the external data is having less of an influence,
###  with interval 0.52 to 0.72

###  The Ext.est function also allows for multidimensional estimation
###  as well as multidimensional outside information.

###  Let's suppose now you want to jointly estimate the survival at 32 and 60 months,
###  and you had external information on 24 and 60 months with survivals
###  0.25 and 0.5 based on 20 patients.

### The func.theta function should return a vector of the estimates.
func.theta = function(d){
     km = survfit(Surv(times, status) ~ 1, data = d)
     kmest = stepfun(km$time, c(1, km$surv))
     c(kmest(32), kmest(60))
}

###   And we add the outside estimates in our mean list as a vector.
means[[1]] = c(0.25, 0.5)

###   The vars should now be a covariance matrix, let's guess
###   the covariance between the two estimates is 0.5.
vars[[1]] = matrix(c(0.25*0.75/20, 0.5, 0.5, 0.5*0.5/20), 2)

###   The funcs list should return the outside estimates as a vector.
funcs[[1]] = function(d) {
     km = survfit(Surv(times, status) ~ 1, data = d)
     kmest = stepfun(km$time, c(1, km$surv))
     c(kmest(24), kmest(60))
}

###  Now again, we can run everything as before.
Ext.est(data, func.theta, means, vars, funcs, B = 500)
```

Index

`Ext.est`, 1