Package 'entropy'

February 15, 2013

Version 1.1.7

Date 2012-01-20

Title Entropy and Mutual Information Estimation

Author Jean Hausser and Korbinian Strimmer

Maintainer Korbinian Strimmer <strimmer@uni-leipzig.de>

Depends R (>= 2.10.0)

Suggests

Description This package implements various estimators of entropy,such as the shrinkage estimator by Hausser and Strimmer, the maximum likelihood and the Millow-Madow estimator, various Bayesian estimators, and the Chao-Shen estimator. It also offers an R interface to the NSB estimator. Furthermore, it provides functions for estimating mutual information.

URL http://strimmerlab.org/software/entropy/

Repository CRAN

Date/Publication 2012-01-20 06:49:53

NeedsCompilation no

R topics documented:

---

entropy-package *The entropy Package*

Description

This package implements various estimators of the Shannon entropy. Most estimators in this package can be applied in "small n, large p" situations, i.e. when there are many more bins than counts.

The main function of this package is `entropy`, which provides a unified interface to various entropy estimators.

If you use this package please cite: Jean Hausser and Korbinian Strimmer. 2009. Entropy inference and the James-Stein estimator, with application to nonlinear gene association networks. J. Mach. Learn. Res. **10**: 1469-1484. Available online from http://jmlr.csail.mit.edu/papers/v10/hausser09a.html.

This paper contains a detailed statistical comparison of the estimators available in this package. It also describes the shrinkage entropy estimator `entropy.shrink`.

Author(s)

Jean Hausser and Korbinian Strimmer (http://strimmerlab.org/)

References

See website: http://strimmerlab.org/software/entropy/

See Also

`entropy`

---

entropy *Estimating Entropy From Observed Counts*

Description

`entropy` estimates the Shannon entropy H of the random variable Y from the corresponding observed counts `y`.

`freqs` estimates bin frequencies from the counts `y`.

Usage

```
entropy(y, method=c("ML", "MM", "Jeffreys", "Laplace", "SG", "minimax", "CS", "NSB", "shrink"),
                   unit=c("log", "log2", "log10"), target=1/length(y), verbose=TRUE, ...)
freqs(y, method=c("ML", "MM", "Jeffreys", "Laplace", "SG", "minimax", "CS", "NSB", "shrink"), targe
```

Arguments

y	vector of counts.
method	the method employed to estimate entropy (see Details).
unit	the unit in which entropy is measured. The default is "nats" (natural units). For computing entropy in "bits" set `unit="log2"`.
target	shrinkage target (for "shrink" option).
verbose	verbose option (for "shrink" option).
...	option passed on to `entropy.NSB`.

Details

The `entropy` function allows to estimate entropy from observed counts by a variety of methods:

- `method="ML"`:maximum likelihood, see `entropy.empirical`
- `method="MM"`:bias-corrected maximum likelihood, see `entropy.MillerMadow`
- `method="Jeffreys"`:`entropy.Dirichlet` with `a=1/2`
- `method="Laplace"`:`entropy.Dirichlet` with `a=1`
- `method="SG"`:`entropy.Dirichlet` with `a=a=1/length(y)`
- `method="minimax"`:`entropy.Dirichlet` with `a=sqrt(sum(y))/length(y`
- `method="CS"`:see `entropy.ChaoShen`
- `method="NSB"`:see `entropy.NSB`
- `method="shrink"`:see `entropy.shrink`

The `freqs` function estimates the underlying bin frequencies. Note that estimated frequencies are not available for `method="MM"`, `method="CS"` and `method="NSB"`. In these instances a vector containing NAs is returned.

Value

`entropy` returns an estimate of the Shannon entropy.

`freqs` returns a vector with estimated bin frequencies (if available).

Author(s)

Korbinian Strimmer (`http://strimmerlab.org`).

See Also

`entropy-package`

Examples

```
# load entropy library
library("entropy")

# observed counts for each bin
y = c(4, 2, 3, 0, 2, 4, 0, 0, 2, 1, 1)

entropy(y, method="ML")
entropy(y, method="MM")
entropy(y, method="Jeffreys")
entropy(y, method="Laplace")
entropy(y, method="SG")
entropy(y, method="minimax")
entropy(y, method="CS")
#entropy(y, method="NSB")
entropy(y, method="shrink")
```

---

entropy.ChaoShen *Chao-Shen Entropy Estimator*

Description

`entropy.ChaoShen` estimates the Shannon entropy H of the random variable Y from the corresponding observed counts `y` using the method of Chao and Shen (2003).

Usage

```
entropy.ChaoShen(y, unit=c("log", "log2", "log10"))
```

Arguments

`y`	vector of counts.
`unit`	the unit in which entropy is measured. The default is "nats" (natural units). For computing entropy in "bits" set `unit="log2"`.

Details

The Chao-Shen entropy estimator (2003) is a Horvitz-Thompson (1952) estimator applied to the problem of entropy estimation, with additional coverage correction as proposed by Good (1953).

Note that the Chao-Shen estimator is not a plug-in estimator, hence there are no explicit underlying bin frequencies.

Value

`entropy.ChaoShen` returns an estimate of the Shannon entropy.

Author(s)

Korbinian Strimmer (http://strimmerlab.org).

References

Chao, A., and T.-J. Shen. 2003. Nonparametric estimation of Shannon's index of diversity when there are unseen species in sample. Environ. Ecol. Stat. **10**:429-443.

Good, I. J. 1953. The population frequencies of species and the estimation of population parameters. Biometrika **40**:237-264.

Horvitz, D.G., and D. J. Thompson. 1952. A generalization of sampling without replacement from a finite universe. J. Am. Stat. Assoc. **47**:663-685.

See Also

entropy, entropy.shrink, entropy.Dirichlet, entropy.NSB.

Examples

```
# load entropy library
library("entropy")

# observed counts for each bin
y = c(4, 2, 3, 0, 2, 4, 0, 0, 2, 1, 1)

# estimate entropy using Chao-Shen method
entropy.ChaoShen(y)

# compare to empirical estimate
entropy.empirical(y)
```

---

entropy.Dirichlet *Family of Dirichlet Entropy and Mutual Information Estimators*

Description

`entropy.Dirichlet` estimates the Shannon entropy H of the random variable Y from the corresponding observed counts `y` by plug-in of Bayesian estimates of the bin frequencies using the Dirichlet-multinomial pseudocount model.

`mi.Dirichlet` estimates the corresponding mutual information of two random variables.

`freqs.Dirichlet` computes the Bayesian estimates of the bin frequencies using the Dirichlet-multinomial pseudocount model.

Usage

```
entropy.Dirichlet(y, a, unit=c("log", "log2", "log10"))
mi.Dirichlet(y, a, unit=c("log", "log2", "log10"))
freqs.Dirichlet(y, a)
```

Arguments

`y`	vector or matrix of counts.
`a`	pseudocount per bin.
`unit`	the unit in which entropy is measured. The default is "nats" (natural units). For computing entropy in "bits" set `unit="log2"`.

Details

The Dirichlet-multinomial pseudocount entropy estimator is a Bayesian plug-in estimator: in the definition of the Shannon entropy the bin probabilities are replaced by the respective Bayesian estimates of the frequencies, using a model with a Dirichlet prior and a multinomial likelihood.

The parameter `a` is a parameter of the Dirichlet prior, and in effect specifies the pseudocount per bin. Popular choices of `a` are:

- a=0:maximum likelihood estimator (see `entropy.empirical`)
- a=1/2:Jeffreys' prior; Krichevsky-Trovimov (1991) entropy estimator
- a=1:Laplace's prior
- a=1/length(y):Schurmann-Grassberger (1996) entropy estimator
- a=sqrt(sum(y))/length(y):minimax prior

The pseudocount `a` can also be a vector so that for each bin an individual pseudocount is added.

Value

`entropy.Dirichlet` returns an estimate of the Shannon entropy.

`mi.Dirichlet` returns an estimate of the mutual information.

`freqs.Dirichlet` returns the underlying frequencies.

Author(s)

Korbinian Strimmer (`http://strimmerlab.org`).

References

Agresti, A., and D. B. Hitchcock. 2005. Bayesian inference for categorical data analysis. Stat. Methods. Appl. **14**:297–330.

Krichevsky, R. E., and V. K. Trofimov. 1981. The performance of universal encoding. IEEE Trans. Inf. Theory **27**: 199-207.

Schurmann, T., and P. Grassberger. 1996. Entropy estimation of symbol sequences. Chaos **6**:41-427.

See Also

`entropy`, `entropy.shrink`, `entropy.NSB`, `entropy.ChaoShen`, `entropy.empirical`, `entropy.plugin`, `mi.plugin`.

Examples

```
# load entropy library
library("entropy")

# observed counts for each bin
y = c(4, 2, 3, 0, 2, 4, 0, 0, 2, 1, 1)

# Dirichlet estimate with a=0
entropy.Dirichlet(y, a=0)

# compare to empirical estimate
entropy.empirical(y)

# Dirichlet estimate with a=1/2
entropy.Dirichlet(y, a=1/2)

# Dirichlet estimate with a=1
entropy.Dirichlet(y, a=1)

# Dirichlet estimate with a=1/length(y)
entropy.Dirichlet(y, a=1/length(y))

# Dirichlet estimate with a=sqrt(sum(y))/length(y)
entropy.Dirichlet(y, a=sqrt(sum(y))/length(y))


# contigency table with counts for two discrete variables
y = rbind( c(1,2,3), c(6,5,4) )

# Dirichlet estimate of mutual information (with a=1/2)
mi.Dirichlet(y, a=1/2)
```

`entropy.empirical` *Empirical Entropy and Mutual Information Estimator*

Description

`entropy.empirical` estimates the Shannon entropy H of the random variable Y from the corresponding observed counts `y` by plug-in of the empirical frequencies.

`mi.empirical` computes the empirical mutual information from counts `y`.

`freqs.empirical` computes the empirical frequencies from counts `y`.

Usage

```
entropy.empirical(y, unit=c("log", "log2", "log10"))
mi.empirical(y, unit=c("log", "log2", "log10"))
freqs.empirical(y)
```

Arguments

y	vector or matrix of counts.
unit	the unit in which entropy is measured. The default is "nats" (natural units). For computing entropy in "bits" set `unit="log2"`.

Details

The empirical entropy estimator is a plug-in estimator: in the definition of the Shannon entropy the bin probabilities are replaced by the respective empirical frequencies.

The empirical entropy estimator is the maximum likelihood estimator. If there are many zero counts and the sample size is small it is very inefficient and also strongly biased.

Value

`entropy.empirical` returns an estimate of the Shannon entropy.

`mi.empirical` returns an estimate of the mutual information.

`freqs.empirical` returns the underlying frequencies.

Author(s)

Korbinian Strimmer (`http://strimmerlab.org`).

See Also

`entropy`, `entropy.MillerMadow`, `entropy.plugin`, `mi.plugin`.

Examples

```
# load entropy library
library("entropy")

# observed counts for each bin
y = c(4, 2, 3, 0, 2, 4, 0, 0, 2, 1, 1)

# empirical estimate of entropy
entropy.empirical(y)

# contigency table with counts for two discrete variables
y = rbind( c(1,2,3), c(6,5,4) )

# empirical estimate of mutual information
mi.empirical(y)
```

`entropy.MillerMadow` *Miller-Madow Entropy Estimator*

Description

`entropy.MillerMadow` estimates the Shannon entropy H of the random variable Y from the corresponding observed counts `y` using the Miller-Madow correction to the empirical entropy).

Usage

```
entropy.MillerMadow(y, unit=c("log", "log2", "log10"))
```

Arguments

`y`	vector of counts.
`unit`	the unit in which entropy is measured. The default is "nats" (natural units). For computing entropy in "bits" set `unit="log2"`.

Details

The Miller-Madow entropy estimator (1955) is the bias-corrected empirical entropy estimate.

Note that the Miller-Madow estimator is not a plug-in estimator, hence there are no explicit underlying bin frequencies.

Value

`entropy.MillerMadow` returns an estimate of the Shannon entropy.

Author(s)

Korbinian Strimmer (`http://strimmerlab.org`).

References

Miller, G. 1955. Note on the bias of information estimates. Info. Theory Psychol. Prob. Methods **II-B**:95-100.

See Also

`entropy.empirical`

Examples

```
# load entropy library
library("entropy")

# observed counts for each bin
y = c(4, 2, 3, 0, 2, 4, 0, 0, 2, 1, 1)

# estimate entropy using Miller-Madow method
entropy.MillerMadow(y)

# compare to empirical estimate
entropy.empirical(y)
```

---

entropy.NSB *R Interface to NSB Entropy Estimator*

Description

`entropy.NSB` estimates the Shannon entropy H of the random variable Y from the corresponding observed counts `y` using the method of Nemenman, Shafee and Bialek (2002).

Note that this function is an R interface to the "nsb-entropy" program. Hence, this needs to be installed separately from http://nsb-entropy.sourceforge.net/.

Usage

```
entropy.NSB(y, unit=c("log", "log2", "log10"), CMD="nsb-entropy")
```

Arguments

y	vector of counts.
unit	the unit in which entropy is measured. The default is "nats" (natural units). For computing entropy in "bits" set `unit="log2"`.
CMD	path to the "nsb-entropy" executable.

Details

The NSB estimator is due to Nemenman, Shafee and Bialek (2002). It is a Dirichlet-multinomial entropy estimator, with a hierarchical prior over the Dirichlet pseudocount parameters.

Note that the NSB estimator is not a plug-in estimator, hence there are no explicit underlying bin frequencies.

Value

`entropy.NSB` returns an estimate of the Shannon entropy.

Author(s)

Jean Hausser.

References

Nemenman, I., F. Shafee, and W. Bialek. 2002. Entropy and inference, revisited. In: Dietterich, T., S. Becker, Z. Gharamani, eds. Advances in Neural Information Processing Systems 14: 471-478. Cambridge (Massachusetts): MIT Press.

See Also

entropy, entropy.shrink, entropy.Dirichlet, entropy.ChaoShen.

Examples

```
# load entropy library
library("entropy")

# observed counts for each bin
y = c(4, 2, 3, 0, 2, 4, 0, 0, 2, 1, 1)

## Not run:
# estimate entropy using the NSB method
entropy.NSB(y) # 2.187774

## End(Not run)

# compare to empirical estimate
entropy.empirical(y)
```

---

entropy.plugin *Plug-In Entropy Estimator*

Description

entropy.plugin computes the Shannon entropy H of a discrete random variable from the specified bin frequencies.

Usage

```
entropy.plugin(freqs, unit=c("log", "log2", "log10"))
```

Arguments

freqs	bin frequencies.
unit	the unit in which entropy is measured. The default is "nats" (natural units). For computing entropy in "bits" set unit="log2".

Details

The Shannon entropy of a discrete random variable is defined as $H = -\sum p(x_i) \log(p(x_i))$, where $p(x_i)$ are the bin probabilities.

Value

entropy.plugin returns the Shannon entropy.

Author(s)

Korbinian Strimmer (http://strimmerlab.org).

See Also

entropy, entropy.empirical, mi.shrink.

Examples

```
# load entropy library
library("entropy")

# some frequencies
freqs = c(0.2, 0.1, 0.15, 0.05, 0, 0.3, 0.2)

# and corresponding entropy
entropy.plugin(freqs)
```

---

entropy.shrink *Shrinkage Entropy and Mutual Information Estimator*

Description

entropy.shrink estimates the Shannon entropy H of the random variable Y from the corresponding observed counts y by plug-in of shrinkage estimate of the bin frequencies.

mi.shrink estimates the corresponding mutual information of two random variables.

freq.shrink estimates the bin frequencies from the counts y using a James-Stein-type shrinkage estimator. The default shrinkage target is the uniform, unless otherwise specified.

Usage

```
entropy.shrink(y, unit=c("log", "log2", "log10"), target=1/length(y), verbose=TRUE)
mi.shrink(y, unit=c("log", "log2", "log10"), target=1/length(y), verbose=TRUE)
freqs.shrink(y, target=1/length(y), verbose=TRUE)
```

Arguments

y	vector or matrix of counts.
unit	the unit in which entropy is measured. The default is "nats" (natural units). For computing entropy in "bits" set unit="log2".
target	the shrinkage target for the frequencies (default: uniform distribution).
verbose	report shrinkage intensity.

Details

The shrinkage estimator is a James-Stein-type estimator. It is essentially a `entropy.Dirichlet` estimator, where the pseudocount is estimated from the data.

For details see Hausser and Strimmer (2009).

Value

`entropy.shrink` returns an estimate of the Shannon entropy.

`mi.shrink` returns an estimate of mutual information.

`freqs.shrink` returns the underlying frequencies.

In all instances the estimated shrinkage intensity is attached to the returned value in the attribute `lambda.freqs`.

Author(s)

Korbinian Strimmer (http://strimmerlab.org).

References

Hausser, J., and K. Strimmer. 2009. Entropy inference and the James-Stein estimator, with application to nonlinear gene association networks. J. Mach. Learn. Res. **10**: 1469-1484. Available online from http://jmlr.csail.mit.edu/papers/v10/hausser09a.html.

See Also

`entropy`, `entropy.Dirichlet`, `entropy.NSB`, `entropy.ChaoShen`, `entropy.plugin`, `mi.plugin`.

Examples

```
# load entropy library
library("entropy")

# observed counts for each bin
y = c(4, 2, 3, 0, 2, 4, 0, 0, 2, 1, 1)

# shrinkage estimate
entropy.shrink(y)

# contigency table with counts for two discrete variables
y = rbind( c(1,2,3), c(6,5,4) )

# shrinkage estimate of mutual information
mi.shrink(y)
```

mi.plugin *Plug-In Mutual Information Estimator*

Description

mi.plugin computes the mutual information of of two discrete random variables from the specified bin frequencies.

Usage

```
mi.plugin(freqs2d, unit=c("log", "log2", "log10"))
```

Arguments

freqs2d	matrix of bin frequencies.
unit	the unit in which entropy is measured. The default is "nats" (natural units). For computing entropy in "bits" set unit="log2".

Details

The mutual information of two discrete random variables X and Y is defined as $MI = H(X) + H(Y) - H(X, Y)$.

Value

mi.plugin returns the mutual information.

Author(s)

Korbinian Strimmer (http://strimmerlab.org).

See Also

entropy.plugin.

Examples

```
# load entropy library
library("entropy")

# joint distribution of two discrete variables
freqs2d = rbind( c(0.2, 0.1, 0.15), c(0, 0.3, 0.25) )

# and corresponding mutual information
mi.plugin(freqs2d)
```

Index