Package ‘dbnlearn’

July 30, 2020

Type Package
Title Dynamic Bayesian Network Structure Learning, Parameter Learning and Forecasting
Version 0.1.0
Depends R (>= 3.4)
License MIT + file LICENSE
Encoding UTF-8
LazyData true
RoxygenNote 7.1.1
Repository CRAN
Imports bnlearn, bnviewer, ggplot2
NeedsCompilation no
Author Robson Fernandes [aut, cre, cph]
Maintainer Robson Fernandes <robson.fernandes@usp.br>
Date/Publication 2020-07-30 10:10:02 UTC

R topics documented:

  dbnlearn-package .................................................. 2
dbn.fit ............................................................. 4
dbn.learn ........................................................... 5
dbn.predict ......................................................... 7
dbn.preprocessing .................................................. 9

Index 11
**dbnlearn-package**

*Dynamic Bayesian Network Structure Learning, Parameter Learning and Forecasting*

**Description**

Dynamic Bayesian Network Structure Learning, Parameter Learning and Forecasting. This package implements a model of Gaussian Dynamic Bayesian Networks with temporal windows, based on collections of linear regressors for Gaussian nodes. The package allows learning the structure of univariate time series, learning parameters and forecasting.

**Details**

<table>
<thead>
<tr>
<th>Package</th>
<th>dbnlearn-package</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Package</td>
</tr>
<tr>
<td>Version</td>
<td>0.1.0</td>
</tr>
<tr>
<td>Date</td>
<td>2020-07-17</td>
</tr>
<tr>
<td>License</td>
<td>MIT + file LICENSE</td>
</tr>
</tbody>
</table>

**Author(s)**

Robson Fernandes  
Institute of Mathematical and Computer Sciences  
University of Sao Paulo - ICMC-USP  
Maintainer: Robson Fernandes <robson.fernandes@usp.br>

**References**


**Examples**

```r
library(dbnlearn)  
library(bnviewer)  
library(ggplot2)  

#Time Series AirPassengers
```
ts <- AirPassengers

# Time Series Preprocessing with time window = 12
X.ts = dbn.preprocessing(ts, window = 12)

# Define 70% Train and 30% Test Data Set
percent = 0.7
n = nrow(X.ts)

trainIndex <- seq_len(length.out = floor(x = percent * n))
X.ts.train <- X.ts[trainIndex,]
X.ts.test <- X.ts[-trainIndex,]

# Dynamic Bayesian Network Structure Learning
ts.learning = dbn.learn(X.ts.train)

# Viewer Dynamic Bayesian Network
viewer(ts.learning, edges.smooth = TRUE, bayesianNetwork.height = "400px", node.colors = list(background = "#f4bafd", border = "#2b7ce9", highlight = list(background = "#97c2fc", border = "#2b7ce9")), bayesianNetwork.layout = "layout_with_sugiyama")

# Dynamic Bayesian Network Fit
ts.fit = dbn.fit(ts.learning, X.ts.train)

# Predict values
prediction = dbn.predict(ts.fit, X.ts.test)

# Plot Real vs Predict
real = X.ts.test[, "X_t"]
prediction = prediction

df.validation = data.frame(list(real = real, prediction = prediction))

ggplot(df.validation, aes(seq(1:nrow(df.validation)))) + geom_line(aes(y = real, colour="real")) + geom_line(aes(y = prediction, colour="prediction")) + scale_color_manual(values = c("real" = 'deepskyblue', 'prediction' = 'maroon1')) + labs(title = "Dynamic Bayesian Network", subtitle = "AirPassengers Time Series", colour = "Legend", x = "Time Index", y = "Values") + theme_minimal()
Dynamic Bayesian Network Fit

Usage

dbn.fit(dbn.learn = NULL, ts = NULL)

Arguments

dbn.learn Dynamic Bayesian Network Learning.
ts Time Series.

Value

Dynamic Bayesian Network Fit

Author(s)

Robson Fernandes

Examples

library(dbnlearn)
library(bnviewer)
library(ggplot2)

# Time Series AirPassengers
ts <- AirPassengers

# Time Series Preprocessing with time window = 12
X.ts = dbn.preprocessing(ts, window = 12)

# Define 70\% Train and 30\% Test Data Set
percent = 0.7
n = nrow(X.ts)

trainIndex <- seq_len(length.out = floor(x = percent * n))
X.ts.train <- X.ts[trainIndex,]
X.ts.test <- X.ts[-trainIndex,]

# Dynamic Bayesian Network Structure Learning
ts.learning = dbn.learn(X.ts.train)

# Viewer Dynamic Bayesian Network
viewer(ts.learning,
   edges.smooth = TRUE,
   bayesianNetwork.height = "400px",
   node.colors = list(background = "#f4bafd",
                       border = "#2b7ce9",
                       highlight = list(background = "#97c2fc",
                                        border = "#2b7ce9")),
   bayesianNetwork.layout = "layout_with_sugiyama")

#Dynamic Bayesian Network Fit
ts.fit = dbn.fit(ts.learning, X.ts.train)

#Predict values
prediction = dbn.predict(ts.fit, X.ts.test)

#Plot Real vs Predict
real = X.ts.test[, "X_t"]
prediction = prediction
df.validation = data.frame(list(real = real, prediction = prediction))

ggplot(df.validation, aes(seq(1:nrow(df.validation)))) +
   geom_line(aes(y = real, colour="real")) +
   geom_line(aes(y = prediction, colour="prediction")) +
   scale_color_manual(values = c(
                           'real' = 'deepskyblue',
                           'prediction' = 'maroon1')) +
   labs(title = "Dynamic Bayesian Network",
        subtitle = "AirPassengers Time Series",
        colour = "Legend",
        x = "Time Index",
        y = "Values") + theme_minimal()
Value
Dynamic Bayesian Network Structure Learning

Author(s)
Robson Fernandes

Examples

library(dbnlearn)
library(bnviewer)
library(ggplot2)

#Time Series AirPassengers
ts <- AirPassengers

#Time Series Preprocessing with time window = 12
X.ts = dbn.preprocessing(ts, window = 12)

#Define 70\% Train and 30\% Test Data Set
percent = 0.7
n = nrow(X.ts)
trainIndex <- seq_len(length.out = floor(x = percent * n))
X.ts.train <- X.ts[trainIndex,]
X.ts.test <- X.ts[-trainIndex,]

#Dynamic Bayesian Network Structure Learning
ts.learning = dbn.learn(X.ts.train)

#Viewer Dynamic Bayesian Network
viewer(ts.learning,
edges.smooth = TRUE,
bayesianNetwork.height = "400px",
node.colors = list(background = "#f4bafd",
border = "#2b7ce9",
highlight = list(background = "#97c2fc",
border = "#2b7ce9")),
bayesianNetwork.layout = "layout_with_sugiyama")

#Dynamic Bayesian Network Fit
ts.fit = dbn.fit(ts.learning, X.ts.train)

#Predict values
prediction = dbn.predict(ts.fit, X.ts.test)

#Plot Real vs Predict
real = X.ts.test[, "X_t"]
prediction = prediction

df.validation = data.frame(list(real = real, prediction = prediction))

ggplot(df.validation, aes(seq(1:nrow(df.validation)))) +
  geom_line(aes(y = real, colour="real")) +
  geom_line(aes(y = prediction, colour="prediction")) +
  scale_color_manual(values = c(
    'real' = 'deepskyblue',
    'prediction' = 'maroon1')) +
  labs(title = "Dynamic Bayesian Network",
       subtitle = "AirPassengers Time Series",
       colour = "Legend",
       x = "Time Index",
       y = "Values") + theme_minimal()

---

dbn.predict  Dynamic Bayesian Network Predict

Description
Dynamic Bayesian Network Predict

Usage

dbn.predict(dbn.fit = NULL, ts = NULL)

Arguments

dbn.fit Dynamic Bayesian Network Fit
ts Time Series.

Value
Dynamic Bayesian Network Predict

Author(s)
Robson Fernandes

Examples

library(dbnlearn)
library(bnviewer)
library(ggplot2)

#Time Series AirPassengers
ts <- AirPassengers

# Time Series Preprocessing with time window = 12
X.ts = dbn.preprocessing(ts, window = 12)

# Define 70% Train and 30% Test Data Set
percent = 0.7
n = nrow(X.ts)

trainIndex <- seq_len(length.out = floor(x = percent * n))
X.ts.train <- X.ts[trainIndex,]
X.ts.test <- X.ts[-trainIndex,]

# Dynamic Bayesian Network Structure Learning
ts.learning = dbn.learn(X.ts.train)

# Viewer Dynamic Bayesian Network
viewer(ts.learning,
      edges.smooth = TRUE,
      bayesianNetwork.height = "400px",
      node.colors = list(background = "#f4bafd",
                          border = "#2b7ce9",
                          highlight = list(background = "#97c2fc",
                                             border = "#2b7ce9")),
      bayesianNetwork.layout = "layout_with_sugiyama")

# Dynamic Bayesian Network Fit
ts.fit = dbn.fit(ts.learning, X.ts.train)

# Predict values
prediction = dbn.predict(ts.fit, X.ts.test)

# Plot Real vs Predict
real = X.ts.test[, "X_t"]
prediction = prediction
df.validation = data.frame(list(real = real, prediction = prediction))

ggplot(df.validation, aes(seq(1:nrow(df.validation)))) +
  geom_line(aes(y = real, colour="real")) +
  geom_line(aes(y = prediction, colour="prediction")) +
  scale_color_manual(values = c(
    'real' = 'deepskyblue',
    'prediction' = 'maroon1')) +
  labs(title = "Dynamic Bayesian Network",
       subtitle = "AirPassengers Time Series",
       colour = "Legend",
       x = "Time Index",
       y = "Values") + theme_minimal()
**Description**

Time Series Preprocessing with time window.

**Usage**

```r
dbn.preprocessing(ts = NULL, window = 12)
```

**Arguments**

- `ts` Time Series.
- `window` Number of steps in the time window.

**Value**

Time Series Preprocessing

**Author(s)**

Robson Fernandes

**Examples**

```r
library(dbnlearn)
library(bnviewer)
library(ggplot2)

#Time Series AirPassengers
ts <- AirPassengers

#Time Series Preprocessing with time window = 12
X.ts = dbn.preprocessing(ts, window = 12)

#Define 70\% Train and 30\% Test Data Set
percent = 0.7
n = nrow(X.ts)

trainIndex <- seq_len(length.out = floor(x = percent * n))
X.ts.train <- X.ts[trainIndex,]
X.ts.test <- X.ts[-trainIndex,]

#Dynamic Bayesian Network Structure Learning
ts.learning = dbn.learn(X.ts.train)

#Viewer Dynamic Bayesian Network
```
viewer(ts.learning, 
   edges.smooth = TRUE, 
   bayesianNetwork.height = "400px", 
   node.colors = list(background = "#f4bafd", border = "#2b7ce9", highlight = list(background = "#97c2fc", border = "#2b7ce9")), 
   bayesianNetwork.layout = "layout_with_sugiyama")

#Dynamic Bayesian Network Fit 
ts.fit = dbn.fit(ts.learning, X.ts.train)

#Predict values 
prediction = dbn.predict(ts.fit, X.ts.test)

#Plot Real vs Predict 
real = X.ts.test[, "X_t"] 
prediction = prediction 

df.validation = data.frame(list(real = real, prediction = prediction))

ggplot(df.validation, aes(seq(1:nrow(df.validation)))) + 
geom_line(aes(y = real, colour="real")) + 
geom_line(aes(y = prediction, colour="prediction")) + 
scale_color_manual(values = c('real' = 'deepskyblue', 'prediction' = 'maroon1')) + 
labs(title = "Dynamic Bayesian Network", subtitle = "AirPassengers Time Series", colour = "Legend", x = "Time Index", y = "Values") + theme_minimal()
Index

* **package**
  - dbnlearn-package, 2

  dbn.fit, 4
  dbn.learn, 5
  dbn.predict, 7
  dbn.preprocessing, 9
  dbnlearn (dbnlearn-package), 2
  dbnlearn-package, 2