Package 'fastTS'

December 2, 2024

Type Package

Title Fast Time Series Modeling for Seasonal Series with Exogenous Variables

Version 1.0.2

Description An implementation of sparsity-ranked lasso and related methods for time series data. This methodology is especially useful for large time series with exogenous features and/or complex seasonality. Originally described in Peterson and Cavanaugh (2022) <doi:10.1007/s10182-021-00431-7> in the context of variable selection with interactions and/or polynomials, ranked sparsity is a philosophy with methods useful for variable selection in the presence of prior informational asymmetry. This situation exists for time series data with complex seasonality, as shown in Peterson and Cavanaugh (2024) <doi:10.1177/1471082X231225307>, which also describes this package in greater detail. The sparsity-ranked penalization methods for time series implemented in 'fastTS' can fit large/complex/high-frequency time series quickly, even with a high-dimensional exogenous feature set. The method is considerably faster than its competitors, while often producing more accurate predictions. Also included is a long hourly series of arrivals into the University of Iowa Emergency Department with concurrent local temperature.

Suggests covr, kableExtra, knitr, magrittr, rmarkdown, testthat (>= 3.0.0), tibble

Imports dplyr, methods, ncvreg, RcppRoll, rlang, yardstick

Depends R (>= 3.5)

License GPL (>= 3)

Encoding UTF-8

LazyData true

RoxygenNote 7.3.1

Config/testthat/edition 3

VignetteBuilder knitr

URL https://petersonr.github.io/fastTS/,
https://github.com/petersonR/fastTS/

BugReports https://github.com/petersonR/fastTS/issues

Date 2024-12-01

NeedsCompilation no

Author Ryan Andrew Peterson [aut, cre, cph]
(<https://orcid.org/0000-0002-4650-5798>)

Maintainer Ryan Andrew Peterson <ryan.a.peterson@cuanschutz.edu>

Repository CRAN

Date/Publication 2024-12-01 23:10:08 UTC

Contents

`AICc` *internal AICc function for lasso models*

Description

internal AICc function for lasso models

Internal function for obtaining oos results

Internal function for converting time series into model matrix of lags

Usage

```
AICc(fit, eps = 1)

get_oos_results(fits, ytest, Xtest)

get_model_matrix(y, X = NULL, n_lags_max)
```

Arguments

`fit`	an object with logLik method,
`eps`	minimum df used in computation
`fits`	a list of fits with different tuning parameters
`ytest`	validation data
`Xtest`	new X data, including lags
`y`	time series vector
`X`	Additional exogenous features
`n_lags_max`	Maximum number of lags to add

`fastTS` *Fast time series modeling with ranked sparsity*

Description

Uses penalized regression to quickly fit time series models with potentially complex seasonal patterns and exogenous variables. Based on methods described in Peterson & Cavanaugh (2024).

Usage

```
fastTS(
  y,
  X = NULL,
  n_lags_max,
  gamma = c(0, 2^(-2:4)),
  ptrain = 0.8,
  pf_eps = 0.01,
  w_endo,
  w_exo,
  weight_type = c("pacf", "parametric"),
  m = NULL,
  r = c(rep(0.1, length(m)), 0.01),
  plot = FALSE,
  ncvreg_args = list(penalty = "lasso", returnX = FALSE, lambda.min = 0.001)
)

## S3 method for class 'fastTS'
plot(x, log.l = TRUE, ...)

## S3 method for class 'fastTS'
coef(object, choose = c("AICc", "BIC"), ...)

## S3 method for class 'fastTS'
print(x, ...)
```

```
## S3 method for class 'fastTS'
summary(object, choose = c("AICc", "BIC"), ...)
```

Arguments

y	univariate time series outcome
X	matrix of predictors (no intercept)
n_lags_max	maximum number of lags to consider
gamma	vector of exponent for weights
ptrain	prop. to leave out for test data
pf_eps	penalty factors below this will be set to zero
w_endo	optional pre-specified weights for endogenous terms
w_exo	optional pre-specified weights for exogenous terms (details)
weight_type	type of weights to use for endogenous terms
m	mode(s) for seasonal lags (used if weight_type = "parametric")
r	penalty factors for seasonal + local scaling functions (used if weight_type = "parametric")
plot	logical; whether to plot the penalty functions
ncvreg_args	additional args to pass through to ncvreg
x	a fastTS object
log.l	Should the x-axis (lambda) be logged?
...	passed to downstream functions
object	a fastTS object
choose	which criterion to use for lambda selection (AICc or BIC)

Details

The default weights for exogenous features will be chosen based on a similar approach to the adaptive lasso (using bivariate OLS estimates). For lower dimensional X, it's advised to set `w_exo="unpenalized"`, because this allows for statistical inference on exogenous variable coefficients via the `summary` function.

By default, a seasonal frequency `m` must not be specified and the PACF is used to estimate the weights for endogenous terms. A parametric version is also available, which allows for a penalty scaling function that penalizes seasonal and recent lags less according to the penalty scaling functions described in Peterson & Cavanaugh (2024). See the `penalty_scaler` function for more details, and to plot the penalty function for various values of `m` and `r`.

Value

A list of class `fastTS` with elements

fits	a list of lasso fits

`ncvreg_args`	arguments passed to ncvreg
`gamma`	the (negative) exponent on the penalty weights, one for each fit
`n_lags_max`	the maximum number of lags
`y`	the time series
`X`	the utilized matrix of exogenous features
`oos_results`	results on test data using best of fits
`train_idx`	index of observations used in training data
`weight_type`	the type of weights used for endogenous terms
`m`	the mode(s) for seasonal lags (used if weight_type = "parametric")
`r`	penalty factors for seasonal + local scaling functions
`ptrain`	the proportion used to train the model

x invisibly

a vector of model coefficients

x (invisibly)

the summary object produced by ncvreg evaluated at the best tuning parameter combination (best AICc).

References

Breheny, P. and Huang, J. (2011) Coordinate descent algorithms for nonconvex penalized regression, with applications to biological feature selection. Ann. Appl. Statist., 5: 232-253.

Peterson, R.A., Cavanaugh, J.E. (2022) Ranked sparsity: a cogent regularization framework for selecting and estimating feature interactions and polynomials. AStA Adv Stat Anal. https://doi.org/10.1007/s10182-021-00431-7

Peterson, R.A., Cavanaugh, J.E. (2024). Fast, effective, and coherent time series modeling using the sparsity-ranked lasso. Statistical Modelling (accepted). DOI: https://doi.org/10.48550/arXiv.2211.01492

See Also

predict.fastTS

Examples

```
data("LakeHuron")
fit_LH <- fastTS(LakeHuron)
fit_LH
coef(fit_LH)
plot(fit_LH)
```

penalty_scaler *Penalty Scaling Function for parametric penalty weights*

Description

Penalty Scaling Function for parametric penalty weights

Usage

```
penalty_scaler(lag, m, r, plot = TRUE, log = TRUE)
```

Arguments

lag	a vector of lags for which to calculate the penalty function
m	a vector of seasonality modes
r	a vector of dim (m + 1) for the factor penalties on c(m, time)
plot	logical; whether to plot the penalty function
log	logical; whether to return the log of the penalty function

predict.fastTS *Predict function for fastTS object*

Description

Predict function for fastTS object

Usage

```
## S3 method for class 'fastTS'
predict(
  object,
  n_ahead = 1,
  X_test,
  y_test,
  cumulative = FALSE,
  forecast_ahead = FALSE,
  return_intermediate = FALSE,
  ...
)
```

Arguments

`object`	an fastTS object
`n_ahead`	the look-ahead period for predictions
`X_test`	a matrix exogenous features for future predictions (optional)
`y_test`	the test series for future predictions (optional)
`cumulative`	cumulative (rolling) sums of 1-, 2-, 3-, ..., k-step-ahead predictions.
`forecast_ahead`	returns forecasted values for end of training series
`return_intermediate`	if TRUE, returns the intermediate predictions between the 1st and n_ahead predictions, as data frame.
`...`	currently unused

Details

The 'y_test' argument must be supplied if predictions are desired or if 'n_ahead' < 'nrow(X_test)'. This is because in order to obtain 1-step forecast for, say, the 10th observation in the test data set, the 9th observation of 'y_test' is required.

Forecasts for the first 'n_ahead' observations after the training set can be obtained by setting 'forecast_ahead' to TRUE, which will return the forecasted values at the end of the training data. it produces the 1-step-ahead prediction, the 2-step-ahead prediction, ... through the 'n_ahead'-step prediction. The 'cumulative' argument is similar but will return the cumulative (rolling) sums of 1-, 2-, 3=, ..., 'n_ahead'-step-ahead predictions.

Value

a vector of predictions, or a matrix of 1- through n_ahead predictions.

Examples

```
data("LakeHuron")
fit_LH <- fastTS(LakeHuron)
predict(fit_LH)
```

---

uihc_ed_arrivals *Hourly arrivals into the University of Iowa Hospital Emergency Department*

Description

A data set containing the 17 columns described below. There are 41640 observations running from 2013 to 2018. Data set are already sorted by time.

Usage

```
uihc_ed_arrivals
```

Format

a data frame with 17 columns and 41640 rows:

Year Calendar year

Quarter Fiscal year quarter

Month Integer for month of year

Day Integer for day of month

Hour Integer for hour of day

Arrivals Number of arrivals into the ED (outcome)

Date Date

Weekday Indicator for day of week

temp hourly concurrent temperature

xmas Christmas day indicator

xmas2 Day after Christmas

nye New Years Eve indicator

nyd New Years Day indicator

thx Thanksgiving day indicator

thx Thanksgiving day (after) indicator

ind Independence day indicator

game_Day Hawkeye football game day indicator

Source

UIHC Emergency Department.

Index