Package 'prophet'

April 19, 2017

Title Automatic Forecasting Procedure

Version 0.1.1

Date 2017-04-17

Description Implements a procedure for forecasting time series data based on an additive model where non-linear trends are fit with yearly and weekly seasonality, plus holidays. It works best with daily periodicity data with at least one year of historical data. Prophet is robust to missing data, shifts in the trend, and large outliers.

Depends R (>= 3.2.3), Rcpp (>= 0.12.0)

Imports dplyr (>= 0.5.0), extraDistr, ggplot2, grid, rstan (>= 2.14.0), scales, stats, tidyr (>= 0.6.1), utils, zoo

Suggests knitr, testthat, readr

License BSD_3_clause + file LICENSE

LazyData true

RoxygenNote 5.0.1

VignetteBuilder knitr

NeedsCompilation yes

Author Sean Taylor [cre, aut], Ben Letham [aut]

Maintainer Sean Taylor <sjt@fb.com>

Repository CRAN

Date/Publication 2017-04-19 08:47:01 UTC

R topics documented:

`compile_stan_model` *Compile Stan model*

Description

Compile Stan model

Usage

```
compile_stan_model(model)
```

Arguments

`model`	String 'linear' or 'logistic' to specify a linear or logistic trend.

Value

Stan model.

df_for_plotting *Merge history and forecast for plotting.*

Description

Merge history and forecast for plotting.

Usage

```
df_for_plotting(m, fcst)
```

Arguments

m	Prophet object.
fcst	Data frame returned by prophet predict.

fit.prophet *Fit the prophet model.*

Description

This sets m$params to contain the fitted model parameters. It is a list with the following elements: k (M array): M posterior samples of the initial slope. m (M array): The initial intercept. delta (MxN matrix): The slope change at each of N changepoints. beta (MxK matrix): Coefficients for K seasonality features. sigma_obs (M array): Noise level. Note that M=1 if MAP estimation.

Usage

```
fit.prophet(m, df, ...)
```

Arguments

m	Prophet object.
df	Data frame.
...	Additional arguments passed to the optimizing or sampling functions in Stan.

`fourier_series` *Provides Fourier series components with the specified frequency and order.*

Description

Provides Fourier series components with the specified frequency and order.

Usage

```
fourier_series(dates, period, series.order)
```

Arguments

`dates`	Vector of dates.
`period`	Number of days of the period.
`series.order`	Number of components.

Value

Matrix with seasonality features.

`get_changepoint_matrix` *Gets changepoint matrix for history dataframe.*

Description

Gets changepoint matrix for history dataframe.

Usage

```
get_changepoint_matrix(m)
```

Arguments

`m`	Prophet object.

Value

array of indexes.

get_prophet_stan_model *Load compiled Stan model*

Description

Load compiled Stan model

Usage

```
get_prophet_stan_model(model)
```

Arguments

model	String 'linear' or 'logistic' to specify a linear or logistic trend.

Value

Stan model.

linear_growth_init *Initialize linear growth.*

Description

Provides a strong initialization for linear growth by calculating the growth and offset parameters that pass the function through the first and last points in the time series.

Usage

```
linear_growth_init(df)
```

Arguments

df	Data frame with columns ds (date), y_scaled (scaled time series), and t (scaled time).

Value

A vector (k, m) with the rate (k) and offset (m) of the linear growth function.

logistic_growth_init *Initialize logistic growth.*

Description

Provides a strong initialization for logistic growth by calculating the growth and offset parameters that pass the function through the first and last points in the time series.

Usage

```
logistic_growth_init(df)
```

Arguments

df	Data frame with columns ds (date), cap_scaled (scaled capacity), y_scaled (scaled time series), and t (scaled time).

Value

A vector (k, m) with the rate (k) and offset (m) of the logistic growth function.

make_all_seasonality_features *Dataframe with seasonality features.*

Description

Dataframe with seasonality features.

Usage

```
make_all_seasonality_features(m, df)
```

Arguments

m	Prophet object.
df	Dataframe with dates for computing seasonality features.

Value

Dataframe with seasonality.

make_future_dataframe *Make dataframe with future dates for forecasting.*

Description

Make dataframe with future dates for forecasting.

Usage

```
make_future_dataframe(m, periods, freq = "d", include_history = TRUE)
```

Arguments

m	Prophet model object.
periods	Int number of periods to forecast forward.
freq	'day', 'week', 'month', 'quarter', or 'year'.
include_history	Boolean to include the historical dates in the data frame for predictions.

Value

Dataframe that extends forward from the end of m$history for the requested number of periods.

make_holiday_features *Construct a matrix of holiday features.*

Description

Construct a matrix of holiday features.

Usage

```
make_holiday_features(m, dates)
```

Arguments

m	Prophet object.
dates	Vector with dates used for computing seasonality.

Value

A dataframe with a column for each holiday.

`make_seasonality_features` *Data frame with seasonality features.*

Description

Data frame with seasonality features.

Usage

```
make_seasonality_features(dates, period, series.order, prefix)
```

Arguments

`dates`	Vector of dates.
`period`	Number of days of the period.
`series.order`	Number of components.
`prefix`	Column name prefix.

Value

Dataframe with seasonality.

`piecewise_linear` *Evaluate the piecewise linear function.*

Description

Evaluate the piecewise linear function.

Usage

```
piecewise_linear(t, deltas, k, m, changepoint.ts)
```

Arguments

`t`	Vector of times on which the function is evaluated.
`deltas`	Vector of rate changes at each changepoint.
`k`	Float initial rate.
`m`	Float initial offset.
`changepoint.ts`	Vector of changepoint times.

Value

Vector y(t).

piecewise_logistic *Evaluate the piecewise logistic function.*

Description

Evaluate the piecewise logistic function.

Usage

```
piecewise_logistic(t, cap, deltas, k, m, changepoint.ts)
```

Arguments

t	Vector of times on which the function is evaluated.
cap	Vector of capacities at each t.
deltas	Vector of rate changes at each changepoint.
k	Float initial rate.
m	Float initial offset.
changepoint.ts	Vector of changepoint times.

Value

Vector y(t).

plot.prophet *Plot the prophet forecast.*

Description

Plot the prophet forecast.

Usage

```
## S3 method for class 'prophet'
plot(x, fcst, uncertainty = TRUE, plot_cap = TRUE,
  xlabel = "ds", ylabel = "y", ...)
```

Arguments

x	Prophet object.
fcst	Data frame returned by predict(m, df).
uncertainty	Boolean indicating if the uncertainty interval for yhat should be plotted. Must be present in fcst as yhat_lower and yhat_upper.
plot_cap	Boolean indicating if the capacity should be shown in the figure, if available.
xlabel	Optional label for x-axis
ylabel	Optional label for y-axis
...	additional arguments

Value

A ggplot2 plot.

Examples

```
## Not run:
history <- data.frame(ds = seq(as.Date('2015-01-01'), as.Date('2016-01-01'), by = 'd'),
                      y = sin(1:366/200) + rnorm(366)/10)
m <- prophet(history)
future <- make_future_dataframe(m, periods = 365)
forecast <- predict(m, future)
plot(m, forecast)

## End(Not run)
```

---

plot_holidays *Plot the holidays component of the forecast.*

Description

Plot the holidays component of the forecast.

Usage

```
plot_holidays(m, df, uncertainty = TRUE)
```

Arguments

m	Prophet model
df	Forecast dataframe for plotting.
uncertainty	Boolean to plot uncertainty intervals.

Value

A ggplot2 plot.

plot_trend *Plot the prophet trend.*

Description

Plot the prophet trend.

Usage

```
plot_trend(df, uncertainty = TRUE, plot_cap = TRUE)
```

Arguments

df	Forecast dataframe for plotting.
uncertainty	Boolean to plot uncertainty intervals.
plot_cap	Boolean indicating if the capacity should be shown in the figure, if available.

Value

A ggplot2 plot.

plot_weekly *Plot the weekly component of the forecast.*

Description

Plot the weekly component of the forecast.

Usage

```
plot_weekly(m, uncertainty = TRUE, weekly_start = 0)
```

Arguments

m	Prophet model object
uncertainty	Boolean to plot uncertainty intervals.
weekly_start	Integer specifying the start day of the weekly seasonality plot. 0 (default) starts the week on Sunday. 1 shifts by 1 day to Monday, and so on.

Value

A ggplot2 plot.

plot_yearly *Plot the yearly component of the forecast.*

Description

Plot the yearly component of the forecast.

Usage

```
plot_yearly(m, uncertainty = TRUE, yearly_start = 0)
```

Arguments

m	Prophet model object.
uncertainty	Boolean to plot uncertainty intervals.
yearly_start	Integer specifying the start day of the yearly seasonality plot. 0 (default) starts the year on Jan 1. 1 shifts by 1 day to Jan 2, and so on.

Value

A ggplot2 plot.

predict.prophet *Predict using the prophet model.*

Description

Predict using the prophet model.

Usage

```
## S3 method for class 'prophet'
predict(object, df = NULL, ...)
```

Arguments

object	Prophet object.
df	Dataframe with dates for predictions (column ds), and capacity (column cap) if logistic growth. If not provided, predictions are made on the history.
...	additional arguments.

Value

A dataframe with the forecast components.

Examples

```
## Not run:
history <- data.frame(ds = seq(as.Date('2015-01-01'), as.Date('2016-01-01'), by = 'd'),
                      y = sin(1:366/200) + rnorm(366)/10)
m <- prophet(history)
future <- make_future_dataframe(m, periods = 365)
forecast <- predict(m, future)
plot(m, forecast)

## End(Not run)
```

`predict_seasonal_components` *Predict seasonality broken down into components.*

Description

Predict seasonality broken down into components.

Usage

```
predict_seasonal_components(m, df)
```

Arguments

`m`	Prophet object.
`df`	Prediction dataframe.

Value

Dataframe with seasonal components.

`predict_trend` *Predict trend using the prophet model.*

Description

Predict trend using the prophet model.

Usage

```
predict_trend(model, df)
```

Arguments

model	Prophet object.
df	Prediction dataframe.

Value

Vector with trend on prediction dates.

---

predict_uncertainty *Prophet uncertainty intervals.*

Description

Prophet uncertainty intervals.

Usage

```
predict_uncertainty(m, df)
```

Arguments

m	Prophet object.
df	Prediction dataframe.

Value

Dataframe with uncertainty intervals.

---

prophet *Prophet forecaster.*

Description

Prophet forecaster.

Usage

```
prophet(df = df, growth = "linear", changepoints = NULL,
  n.changepoints = 25, yearly.seasonality = "auto",
  weekly.seasonality = "auto", holidays = NULL,
  seasonality.prior.scale = 10, holidays.prior.scale = 10,
  changepoint.prior.scale = 0.05, mcmc.samples = 0, interval.width = 0.8,
  uncertainty.samples = 1000, fit = TRUE, ...)
```

Arguments

df	Dataframe containing the history. Must have columns ds (date type) and y, the time series. If growth is logistic, then df must also have a column cap that specifies the capacity at each ds.
growth	String 'linear' or 'logistic' to specify a linear or logistic trend.
changepoints	Vector of dates at which to include potential changepoints. If not specified, potential changepoints are selected automatically.
n.changepoints	Number of potential changepoints to include. Not used if input 'changepoints' is supplied. If 'changepoints' is not supplied, then n.changepoints potential changepoints are selected uniformly from the first 80 percent of df$ds.
yearly.seasonality	Fit yearly seasonality; 'auto', TRUE, or FALSE.
weekly.seasonality	Fit weekly seasonality; 'auto', TRUE, or FALSE.
holidays	data frame with columns holiday (character) and ds (date type)and optionally columns lower_window and upper_window which specify a range of days around the date to be included as holidays. lower_window=-2 will include 2 days prior to the date as holidays.
seasonality.prior.scale	Parameter modulating the strength of the seasonality model. Larger values allow the model to fit larger seasonal fluctuations, smaller values dampen the seasonality.
holidays.prior.scale	Parameter modulating the strength of the holiday components model.
changepoint.prior.scale	Parameter modulating the flexibility of the automatic changepoint selection. Large values will allow many changepoints, small values will allow few changepoints.
mcmc.samples	Integer, if greater than 0, will do full Bayesian inference with the specified number of MCMC samples. If 0, will do MAP estimation.
interval.width	Numeric, width of the uncertainty intervals provided for the forecast. If mcmc.samples=0, this will be only the uncertainty in the trend using the MAP estimate of the extrapolated generative model. If mcmc.samples>0, this will be integrated over all model parameters, which will include uncertainty in seasonality.
uncertainty.samples	Number of simulated draws used to estimate uncertainty intervals.
fit	Boolean, if FALSE the model is initialized but not fit.
...	Additional arguments, passed to fit.prophet

Value

A prophet model.

Examples

```
## Not run:
history <- data.frame(ds = seq(as.Date('2015-01-01'), as.Date('2016-01-01'), by = 'd'),
                      y = sin(1:366/200) + rnorm(366)/10)
m <- prophet(history)

## End(Not run)
```

`prophet_plot_components` *Plot the components of a prophet forecast. Prints a ggplot2 with panels for trend, weekly and yearly seasonalities if present, and holidays if present.*

Description

Plot the components of a prophet forecast. Prints a ggplot2 with panels for trend, weekly and yearly seasonalities if present, and holidays if present.

Usage

```
prophet_plot_components(m, fcst, uncertainty = TRUE, plot_cap = TRUE,
  weekly_start = 0, yearly_start = 0)
```

Arguments

`m`	Prophet object.
`fcst`	Data frame returned by predict(m, df).
`uncertainty`	Boolean indicating if the uncertainty interval should be plotted for the trend, from fcst columns trend_lower and trend_upper.
`plot_cap`	Boolean indicating if the capacity should be shown in the figure, if available.
`weekly_start`	Integer specifying the start day of the weekly seasonality plot. 0 (default) starts the week on Sunday. 1 shifts by 1 day to Monday, and so on.
`yearly_start`	Integer specifying the start day of the yearly seasonality plot. 0 (default) starts the year on Jan 1. 1 shifts by 1 day to Jan 2, and so on.

Value

Invisibly return a list containing the plotted ggplot objects

sample_model *Simulate observations from the extrapolated generative model.*

Description

Simulate observations from the extrapolated generative model.

Usage

```
sample_model(m, df, seasonal.features, iteration)
```

Arguments

m	Prophet object.
df	Prediction dataframe.
seasonal.features	Data frame of seasonal features
iteration	Int sampling iteration to use parameters from.

Value

List of trend, seasonality, and yhat, each a vector like df$t.

sample_predictive_trend *Simulate the trend using the extrapolated generative model.*

Description

Simulate the trend using the extrapolated generative model.

Usage

```
sample_predictive_trend(model, df, iteration)
```

Arguments

model	Prophet object.
df	Prediction dataframe.
iteration	Int sampling iteration to use parameters from.

Value

Vector of simulated trend over df$t.

setup_dataframe *Prepare dataframe for fitting or predicting.*

Description

Adds a time index and scales y. Creates auxillary columns 't', 't_ix', 'y_scaled', and 'cap_scaled'. These columns are used during both fitting and predicting.

Usage

```
setup_dataframe(m, df, initialize_scales = FALSE)
```

Arguments

m	Prophet object.
df	Data frame with columns ds, y, and cap if logistic growth.
initialize_scales	Boolean set scaling factors in m from df.

Value

list with items 'df' and 'm'.

set_auto_seasonalities *Set seasonalities that were left on auto.*

Description

Turns on yearly seasonality if there is >=2 years of history. Turns on weekly seasonality if there is >=2 weeks of history, and the spacing between dates in the history is <7 days.

Usage

```
set_auto_seasonalities(m)
```

Arguments

m	Prophet object.

Value

The prophet model with seasonalities set.

set_changepoints *Set changepoints*

Description

Sets m$changepoints to the dates of changepoints. Either: 1) The changepoints were passed in explicitly. A) They are empty. B) They are not empty, and need validation. 2) We are generating a grid of them. 3) The user prefers no changepoints be used.

Usage

```
set_changepoints(m)
```

Arguments

m	Prophet object.

Value

m with changepoints set.

validate_inputs *Validates the inputs to Prophet.*

Description

Validates the inputs to Prophet.

Usage

```
validate_inputs(m)
```

Arguments

m	Prophet object.

Index