MA Component: Add Bayesian Structure Time Series Model into rapbf

Justin Ji

May 15, 2018

Contents

Design

Overview

Add simple bayesian structure time series model into pick best forecast component.

Detailed behavior

The mathematical models behind the approach are:

- Observation Equation

$$TimeSeries = Trend + Seasonality + Regression + Noise$$
$$Y_t = U_t + S_t + \beta^T X_t + \epsilon_t$$

- State/Transition/Process Equation

$$U_t = U_{t-1} + \delta_{t-1} + \omega_t$$
$$\delta_t = \delta_{t-1} + v_t$$
$$S_t = \sum_{s=1}^{s-1} S_{t-s} + \gamma_t$$

Where ϵ_t, ω_t, v_t and γ_t are independent components of Gaussian random noise. U_t is the current level of the trend, the current slop of the trend is δ_t. The seasonal component S_t can be thought of as a set of s of seasons. X_t is the external regressors and β^T is the regression coefficients.

If the univariate time series data set is used for forecasting, the Bayesian structural model can be simplified as follow, since the series doesn't have any regressors.

- Observation Equation

$$TimeSeries = Trend + Seasonality + Noise$$
$$Y_t = U_t + S_t + \epsilon_t$$

- State/Transition/Process Equation

$$U_t = U_{t-1} + \delta_{t-1} + \omega_t$$
$$\delta_t = \delta_{t-1} + v_t$$
$$S_t = \sum_{s=1}^{s-1} S_{t-s} + \gamma_t$$

Considerations

- The methodology in the prototype is simple Bayesian time series model for the univariate time series forecasting.
  - The pros of the prototyped model is that Bayesian time series models are more transparent than ARIMA model. They also facilitate better handling of uncertainty, a key feature when planning for the future. Bayesian time series model fits perfectly with sequential learning and decision making and it directly leads to exact small sample results.
  - The cons of the prototyped model is that the partial seasonality will be threw into noise/error term. The accurary of the forecasting results might be slighly worse than ARIMA model for some cases.
- What are not in the prototype ?
  - The model with regressor (i.e. external variable) is not considered.
  - The input data is partial year data is not considered.
  - Since there is no project using Bayesian time series model at RA, the model is not covered in any Atlas Toolkit so far.

Function Overview

Function

```
rabsts(data, season.num, season.dur, pred.horizon)
```

Input

- ```
  data - Input data.
  ```
- ```
  season.num - The number of season to be modeled.
  i.e., for a time series with quarterly seasonality, season.num=4.
  ```
- ```
  season.dur - The number of time periods in each season.
  i.e., when season.num=4,
  if it is monthly data, then season.dur=3;
  if it is weekly data, then season.dur=13.
  ```
- ```
  pred.horizon - The number of periods you wish to predict.
  ```

  Note: seasonal component (season.num and season.dur) can be determined by external analysis.

Output

A data frame containing the following as columns:

- `result$mean` - The posterior mean of the prediction;
- `result$median` - The posterior median of the prediction.

Required R Package

- `bsts`

Prototype Code in R

Function Definition

```
## BEGIN: function ##
rabsts <- function(data, season.num, season.dur, pred.horizon) {
  ##get trend or/and seasonal state
  if(season.num==0){
    ss <- bsts::AddLocalLinearTrend(list(), data)}
  else{
    ss <- bsts::AddLocalLinearTrend(list(), data)
    ss <- bsts::AddSeasonal(ss, data, nseasons = season.num, season.duration = season.dur)}

  ##build Bayesial model
  bsts.model <- bsts::bsts(data, state.specification = ss, niter = 666, ping=0, seed=1000)

  ##predict
  result<-bsts::predict.bsts(bsts.model, horizon = pred.horizon, burn = SuggestBurn(0.1, bsts.model), q
  pred<-data.frame(result$mean,result$median)
  return(pred)
}
## END: function ##
```

Example of calling the function with sample data

First, we need to load the required R packages.

```
suppressMessages(library(bsts))
```

Now, we need some input data. The unit test data sets for rapbf package is hired for the prototype. It looks like:

Segment the data by seriesId:

```
#Input data
dat <- subset(inputData, seriesId=="1")
dat <- droplevels(dat)
#Holdout data
holdout.dat<-subset(holdoutData, seriesId=="1")
holdout.dat <- droplevels(holdout.dat)
```

Apply the Bayesial time series model for forecast:

- Seasonal series:

```
x<-rabsts(dat$value,season.num=12,season.dur=1,pred.horizon=3) #assume that there are 12 seasons and ea
data.frame(x,holdout.dat$value)
```

```
##   result.mean result.median holdout.dat.value
## 1   488120058     483865269         504393031
## 2   298829711     300141760         699517167
## 3   637861800     640252143         732156063
```

- Non-seasonal series:

```
y<-rabsts(dat$value,season.num=0,pred.horizon=3) #If no season, set season.num=0.
data.frame(y,holdout.dat$value)
```

```
##   result.mean result.median holdout.dat.value
## 1   472859177     470371758         504393031
## 2   462701885     465480191         699517167
## 3   459352088     454003189         732156063
```