Follow-up Forecasting Forum

Rob J Hyndman
18 April 2017
Outline

1. fpp2
2. forecast v8
3. forecastHybrid
4. opera
5. prophet
6. Forecasting Q&A
7. Wishlist for forecast v9.0
fpp2 package for data and functions on CRAN
Automatically loads forecast and ggplot2
Outline

1. fpp2
2. forecast v8
3. forecastHybrid
4. opera
5. prophet
6. Forecasting Q&A
7. Wishlist for forecast v9.0
head(lynx)

## Time Series:
## Start = 1821
## End = 1826
## Frequency = 1
## [1]  269  321  585  871 1475 2821
<table>
<thead>
<tr>
<th>Year Q</th>
<th>Consumption</th>
<th>Income</th>
<th>Production</th>
<th>Savings</th>
<th>Unemployment</th>
</tr>
</thead>
<tbody>
<tr>
<td>2015 Q2</td>
<td>0.708</td>
<td>0.955</td>
<td>-0.697</td>
<td>5.024</td>
<td>-0.1</td>
</tr>
<tr>
<td>2015 Q3</td>
<td>0.665</td>
<td>0.802</td>
<td>0.381</td>
<td>3.181</td>
<td>-0.3</td>
</tr>
<tr>
<td>2015 Q4</td>
<td>0.562</td>
<td>0.740</td>
<td>-0.846</td>
<td>3.483</td>
<td>0.0</td>
</tr>
<tr>
<td>2016 Q1</td>
<td>0.405</td>
<td>0.519</td>
<td>-0.418</td>
<td>2.237</td>
<td>0.0</td>
</tr>
<tr>
<td>2016 Q2</td>
<td>1.048</td>
<td>0.724</td>
<td>-0.203</td>
<td>-2.722</td>
<td>-0.1</td>
</tr>
<tr>
<td>2016 Q3</td>
<td>0.730</td>
<td>0.645</td>
<td>0.475</td>
<td>-0.573</td>
<td>0.0</td>
</tr>
</tbody>
</table>
gghistogram(lynx)
gghistogram(lynx, add.normal=TRUE)
gghistogram(lynx, add.kde=TRUE)
checkresiduals()

```r
fit <- auto.arima(WWWusage)
checkresiduals(fit)
```
checkresiduals()

## Ljung-Box test

## data: Residuals from ARIMA(1,1,1)
## Q* = 8, df = 8, p-value = 0.4

## Model df: 2. Total lags used: 10
Different types of residuals

```r
fit <- ets(woolyrnq)
res <- cbind(Residuals = residuals(fit),
             RespRes = residuals(fit, type='response'))
autoplot(res, facets=TRUE)
```
Different types of residuals

```r
fit <- Arima(lynx, order=c(4,0,0), lambda=0.5)
res <- cbind(Residuals = residuals(fit),
             RespRes = residuals(fit, type='response'))
autoplot(res, facets=TRUE)
```
Different types of residuals

```r
fit <- auto.arima(uschange[, "Consumption"],
    xreg = uschange[, "Income"])
res <- cbind(Residuals = residuals(fit),
    RegRes = residuals(fit, type = 'regression'))
autoplot(res, facets = TRUE)
```
Seasonal plot: USAccDeaths

- ggseasonplot()
ggseasonplot(polar=TRUE)

ggseasonplot(USAccDeaths, polar=TRUE)
WWWusage %>% ets %>% forecast(h=20) -> fc
autoplot(WWWusage, series="Data") +
autolayer(fc, series="Forecast") +
autolayer(fitted(fc), series="Fitted")
Time series cross-validation

Traditional evaluation

- Training data
- Test data

(time)
Time series cross-validation

Traditional evaluation

Time series cross-validation \((h = 1)\)

- Forecast accuracy averaged over test sets.
- Also known as “evaluation on a rolling forecasting origin”
Time series cross-validation

Traditional evaluation

Time series cross-validation \( (h = 2) \)

- Forecast accuracy averaged over test sets.
- Also known as “evaluation on a rolling forecasting origin”
Time series cross-validation

**Traditional evaluation**

- Training data
- Test data
- Time

**Time series cross-validation** \((h = 3)\)

- Forecast accuracy averaged over test sets.
- Also known as “evaluation on a rolling forecasting origin”
Time series cross-validation

Traditional evaluation

Time series cross-validation \((h = 4)\)

- Forecast accuracy averaged over test sets.
- Also known as “evaluation on a rolling forecasting origin”
Time series cross-validation

Traditional evaluation

Time series cross-validation \((h = 5)\)

- Forecast accuracy averaged over test sets.
- Also known as “evaluation on a rolling forecasting origin”
A good way to choose the best forecasting model is to find the model with the smallest RMSE computed using time series cross-validation.
Pipe function

**Ugly code:**

```r
e <- tsCV(dj, rwf, drift=TRUE, h=1)
sqrt(mean(e^2, na.rm=TRUE))
sqrt(mean(residuals(rwf(dj, drift=TRUE))^2, na.rm=TRUE))
```

**Better with a pipe:**

```r
dj %>% tsCV(forecastfunction=rwf, drift=TRUE, h=1) -> e
e^2 %>% mean(na.rm=TRUE) %>% sqrt
dj %>% rwf(drift=TRUE) %>% residuals -> res
res^2 %>% mean(na.rm=TRUE) %>% sqrt
```
Autoregressive cross-validation

5-fold cross-validation

5-fold non-dep. cross-validation

OOS evaluation

retraining and evaluation with new, unknown data
Autoregressive cross-validation

- OOS evaluation
- Non-dep. cross-validation

\[ y_t \quad y_{t+1} \quad y_{t-1} \quad y_{t-2} \quad y_{t-3} \]

Cross-Validation

- OOS
- Non-dep.
modelcv <- \texttt{CVar}(\texttt{lynx}, k=5, lambda=0.15)
\texttt{print(modelcv)}

## Series: lynx
## Call: \texttt{CVar}(y = lynx, k = 5, lambda = 0.15)
##
## 5-fold cross-validation
## Mean SD
## ME 57.384 437.579
## RMSE 975.938 238.318
## MAE 636.672 152.297
## MPE -17.982 19.380
## MAPE 56.278 15.635
## ACF1 0.144 0.264
## Theil's U 0.912 0.541
Bagged ETS

Algorithm: Generating bootstrapped series

```r
bootstrap ← function(ts, num.boot) {
    lambda ← BoxCox.lambda(ts, min=0, max=1)
    ts.bc ← BoxCox(ts, lambda)
    if(ts is seasonal) {
        [trend, seasonal, remainder] ← stl(ts.bc)
    } else {
        seasonal ← 0
        [trend, remainder] ← loess(ts.bc)
    }
    recon.series[1] ← ts
    for(i in 2:num.boot) {
        boot.sample[i] ← MBB(remainder)
        recon.series.bc[i] ← trend + seasonal + boot.sample[i]
        recon.series[i] ← InvBoxCox(recon.series.bc[i], lambda)
    }
    return(recon.series)
}
```
Bagged ETS
Bagged ETS

![Graph showing bagged ETS data, trend, seasonal, and remainder over time from 1982 to 1992. The graph includes a line for data, trend, seasonal, and remainder, with time on the x-axis and values on the y-axis.]
Bagged ETS

![Graph showing bagged ETS data, trend, seasonal, and remainder components over time from 1982 to 1992.]
Bagged ETS
Bagged ETS
baggedETS(Mcomp::M3[[1896]]$x) %>%
  forecast %>% autoplot +
xlab("Year") + ylab("M495")
Bagged ETS

- Intervals show range of point forecasts
- They are not prediction intervals
Different CO$_2$ forecasts

```r
train <- window(co2, end=c(1990,12))
test <- window(co2, start=c(1991,1))
h <- length(test)
ETS <- forecast(ets(train), h=h)
ARIMA <- forecast(auto.arima(train, lambda=0), h=h)
STL <- stlf(train, lambda=0, h=h)
```
Different CO$_2$ forecasts

```r
autoplot(co2) + xlab("Year") +
ylab(expression("Atmospheric concentration of CO"[2])) +
autolayer(ETS, PI=FALSE, series="ETS") +
autolayer(ARIMA, PI=FALSE, series="ARIMA") +
autolayer(STL, PI=FALSE, series="STL")
```
Fits ARIMA, ETS, Theta, NNETAR, STL-ETS and TBATS models

(Weighted) average of the point forecasts

No proper prediction intervals.
library(forecastHybrid)
fit1 <- hybridModel(train, weights="equal")

## Fitting the auto.arima model
## Fitting the ets model
## Fitting the thetam model
## Fitting the nnetar model
## Fitting the stlm model
## Fitting the tbats model

fit2 <- hybridModel(train, weights="insample")

## Fitting the auto.arima model
## Fitting the ets model
## Fitting the thetam model
autplot(fc1) + ggttitle("Hybrid 1") + xlab("Year") + ylab(expression("Atmospheric concentration of CO"[2])))
forecastHybrid

```r
autoplot(co2) + xlab("Year") +
  ylab(expression("Atmospheric concentration of CO"[2])) +
  autolayer(fc1, series="Hybrid1", PI=FALSE) +
  autolayer(fc2, series="Hybrid2", PI=FALSE)
```
mse <- c(Hybrid1=mean((test - fc1$mean)^2),
          Hybrid2=mean((test - fc2$mean)^2),
          ETS=mean((test - ETS$mean)^2),
          ARIMA=mean((test - ARIMA$mean)^2),
          STL=mean((test - STL$mean)^2))
round(mse,2)

## Hybrid1 Hybrid2 ETS ARIMA STL
## 0.93 0.73 5.92 2.35 2.04
Online Prediction by Expert Aggregation

- mixture function computes weights when combining forecasts based on how well it has done up to that point.
library(opera)
X <- cbind(ETS=ETS$mean, ARIMA=ARIMA$mean, STL=STL$mean)
MLpol0 <- mixture(model = "MLpol", loss.type = "square")
weights <- predict(MLpol0, X, test, type='weights')
head(weights)

##      X1  X2  X3
## [1,] 0.333 0.333 0.333
## [2,] 0.560 0.000 0.440
## [3,] 0.617 0.000 0.383
## [4,] 1.000 0.000 0.000
## [5,] 1.000 0.000 0.000
## [6,] 1.000 0.000 0.000
library(opera)
X <- cbind(ETS=ETS$mean, ARIMA=ARIMA$mean, STL=STL$mean)
MLpol0 <- mixture(model = "MLpol", loss.type = "square")
weights <- predict(MLpol0, X, test, type='weights')
tail(weights)

##     X1   X2   X3
## [79,] 0.360 0.294 0.346
## [80,] 0.280 0.334 0.386
## [81,] 0.277 0.336 0.387
## [82,] 0.358 0.298 0.344
## [83,] 0.214 0.369 0.417
## [84,] 0.233 0.359 0.408
z <- ts(predict(MLpol0, X, test, type='response'),
        start=c(1991,1), freq=12)
autoplot(co2, series="Data") + xlab("Year") +
ylab(expression("Atmospheric concentration of CO"[2])) +
autolayer(z, series="Mixture")
mse <- c(Opera=mean((test-z)^2),
    Hybrid1=mean((test - fc1$mean)^2),
    Hybrid2=mean((test - fc2$mean)^2))
round(mse,2)

## Opera Hybrid1 Hybrid2
## 0.25 0.93 0.73

- Opera weights are updated using past test data, so comparison not “fair”.
Outline

1. fpp2
2. forecast v8
3. forecastHybrid
4. opera
5. prophet
6. Forecasting Q&A
7. Wishlist for forecast v9.0
\[ y_t = g_t + s_t + h_t + \varepsilon_t \]

- \( y_t \) = daily time series.
- \( g_t \) = “growth function” (trend-cycle).
- \( s_t \) = Fourier seasonal terms: weekly and/or yearly
- \( h_t \) = holiday effect.
- \( \varepsilon_t \) = error (can be ARMA errors).
- Estimated as a Bayesian regression using Stan
Growth function

**Piecewise linear growth function**

\[ g_t = (k + a_t \delta)t + (b + a_t^T \gamma) \]

- Changepoints at times \( s_j, j = 1, \ldots, S \).
- \( a_{j,t} = \begin{cases} 
1 & \text{if } t \geq s_j \\
0 & \text{otherwise} 
\end{cases} \).
- Changepoints can be specified (e.g., product launches) or automatically selected.
- A piecewise logistic growth is also available.
Holidays and Events

Dummy holiday/event effects

\[ h_t = \sum_{i=1}^{L} \kappa_i 1(t \in D_i) \]

- \( L \) = number of different types of holidays.
- \( D_i \) = dates for holiday type \( i \).
prophet example
prophet example
library(prophet)
history <- data.frame(
  y = hyndssight,
  ds = seq(as.Date('2014-04-30'), as.Date('2015-04-29'), by = 'd')
)
m <- prophet(history)
future <- make_future_data_frame(m, periods = 365)
forecast <- predict(m, future)
prophet example

plot(m, forecast)
prophet pros and cons

Pros

- Completely automatic including changepoints
- Handles multiple seasonality and holiday effects

Cons

- Only for daily data
- Seems to overfit annual seasonality
- Number of Fourier terms is hard-coded

Compare

- Similar to dynamic harmonic regression with ARMA errors, but with changepoint selection automated.
Outline

1. fpp2
2. forecast v8
3. forecastHybrid
4. opera
5. prophet
6. **Forecasting Q&A**
7. Wishlist for forecast v9.0
Forecasting Q&A

- What forecasting have you been doing?
- Have you been using the forecast package?
- Have you run into any forecasting problems?
- Have you run into any R problems?
Outline

1. fpp2
2. forecast v8
3. forecastHybrid
4. opera
5. prophet
6. Forecasting Q&A
7. Wishlist for forecast v9.0
Wishlist for forecast v9.0

- What facilities would you like to see in the next version of the forecast package?
- What topics would you like to see covered in the fpp book?
My plans for forecast v9+

- New multiple-seasonality method which allows time-changing seasonality and covariates (cross between prophet and tbats).
- Methods for forecasting count time series.
- Improved method for selecting seasonal differencing in auto.arima().
- Somethink like forecastHybrid but with proper prediction intervals.
- Better forecast.ts() for a wider range of time series.
- PSO for ETS.
sugrrants

sugrrants package

- SUpporting GRaphs with R for ANalysing Time Series
- New package for time series data and visualization
- Works with tidyverse packages.
- Some parts of forecast to move?
- Calendar plots
- https://github.com/earowang/sugrrants