08 Timeseries Forecast mit prophet

Martin Hanewald

2019-02-19

Packages

```
library(tidyverse)
library(prophet)
library(lubridate)
library(dygraphs)
library(xts)
library(fpp2)
```

Überblick

Zeitreihenvorhersagen mit der `prophet` Bibliothek erzielen gute Ergebnisse mit minimalem Aufwand. Es ist möglich Sondereffekte mit zusätzlichen Regressoren oder Feiertage einzubringen. Prophet wählt automatisch das beste ARIMA Modell mit passender Saisonalität aus. Eine Besonderheit ist die inkludierte Funktion zur Time-Series-Cross-Validation, die ein robusteres Verfahren zur Modellevaluation darstellt als das übliche Splitting in Training- und Testset.

Zentrale Funktionen von `prophet` sind:

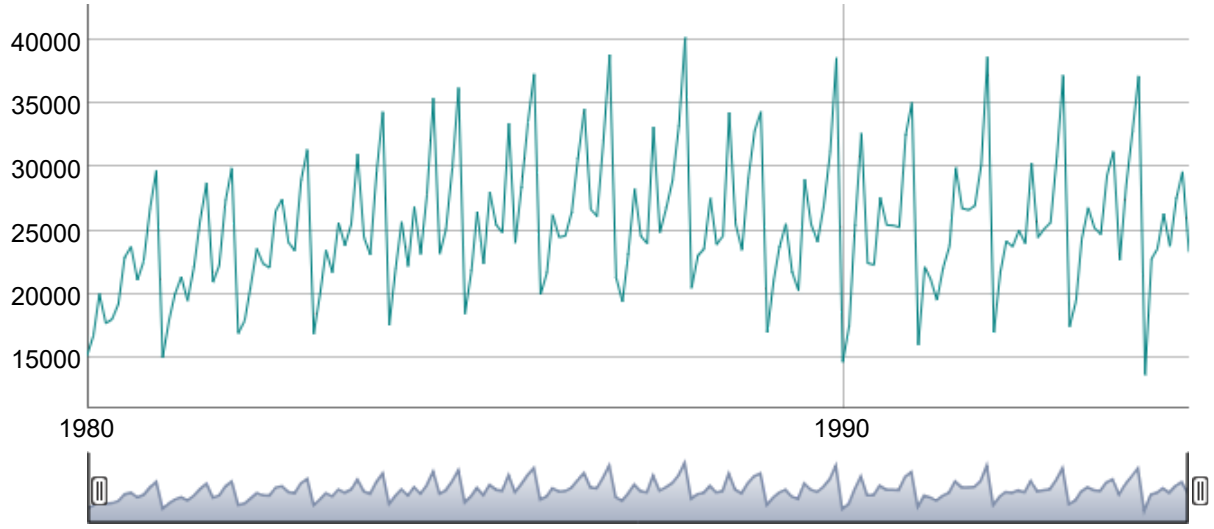
- `prophet()`: Automatisches ARIMA Fitting
- `make_future_dataframe()`: Erzeugung eines DF mit entsprechendem Zukunftshorizont für den Forecast
- `predict()`: Erzeugung des Forecasts
- `dyplot.prophet()`: Schöner interaktiver Plot des Forecasts
- `cross_validation()`: Berechnung der Time-Series Cross Validation
- `plot_cross_validation_metric()`: Plot zur CV

Dataset

Wir verwenden einen Datensatz aus dem `forecast`Paket: `wineind`. Es enthält eine Zeitreihe zum Weinverkauf australischer Winzer zwischen 1980 und 1994. Es eignet sich auf Grund seiner starken saisonalen Struktur sehr gut für einen Forecast.

```
data(wineind, package='forecast')
# prophet erwartet einen Dataframe im Format (ds = Datumsspalte, y = Wertespalte)
df <- as_tibble(wineind) %>%
    mutate(ds = time(wineind) %>% as_date()) %>%
    rename(y = x)
```

```
df %>% column_to_rownames('ds') %>%
    dygraph() %>% dygraphs::dyRangeSelector()
#> Warning: Setting row names on a tibble is deprecated.
```

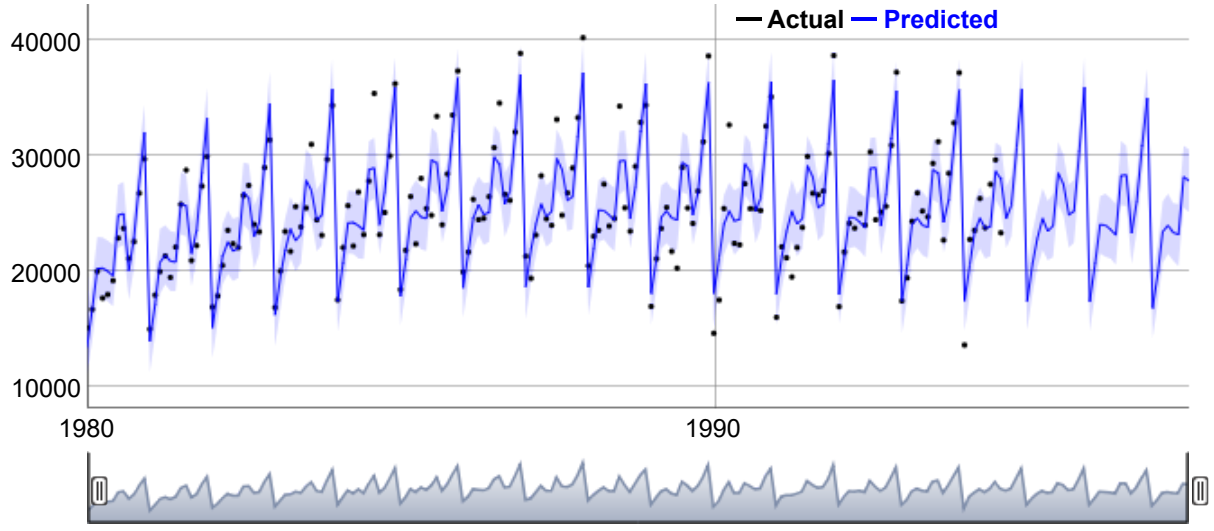


Training

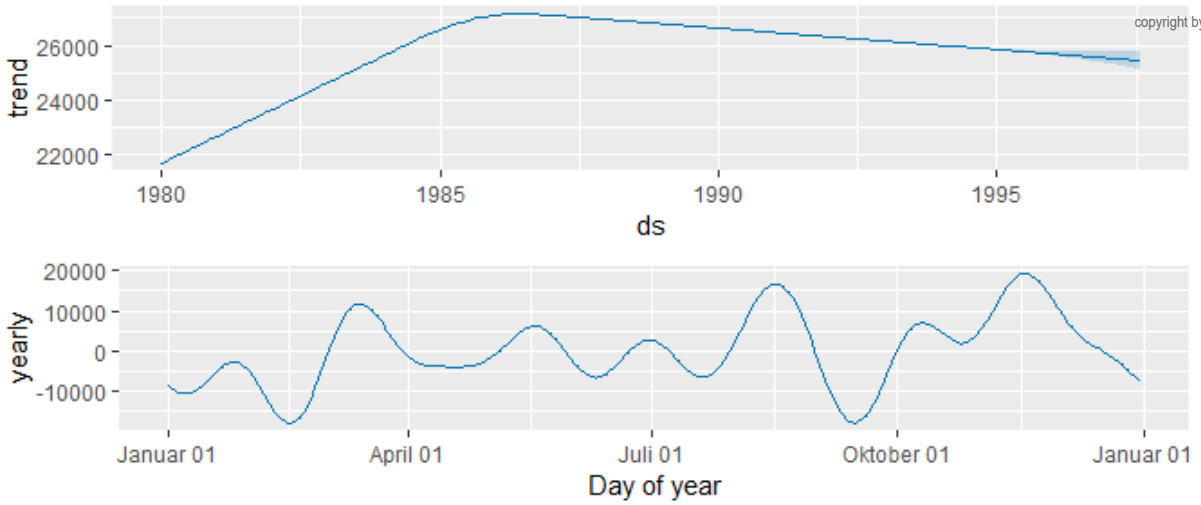
```
m <- prophet::prophet(df)
#> Disabling weekly seasonality. Run prophet with weekly.seasonality=TRUE to override this.
#> Disabling daily seasonality. Run prophet with daily.seasonality=TRUE to override this.
#> Initial log joint probability = -5.6225
#> Optimization terminated normally:
#>   Convergence detected: absolute parameter change was below tolerance

p <- prophet::make_future_dataframe(m, 36, freq = 'month') %>%
    predict(m, .)

prophet::dyplot.prophet(m, p)
```



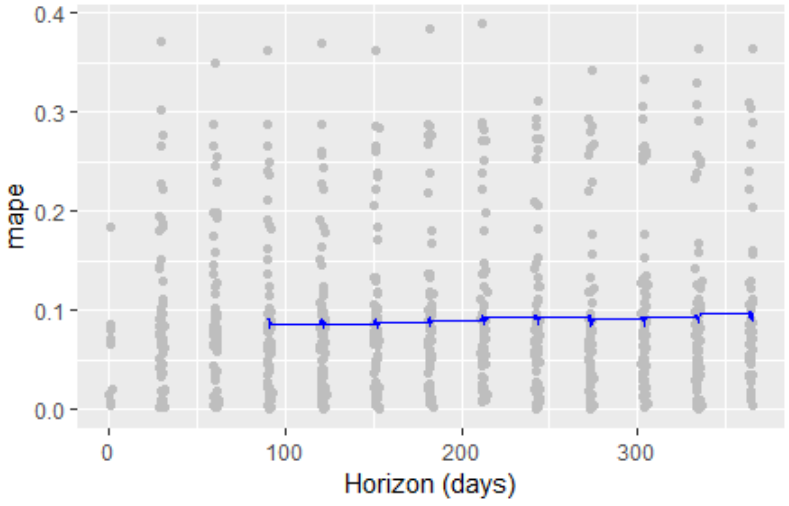

```
# Komponentenplot
prophet::prophet_plot_components(m, p)
```



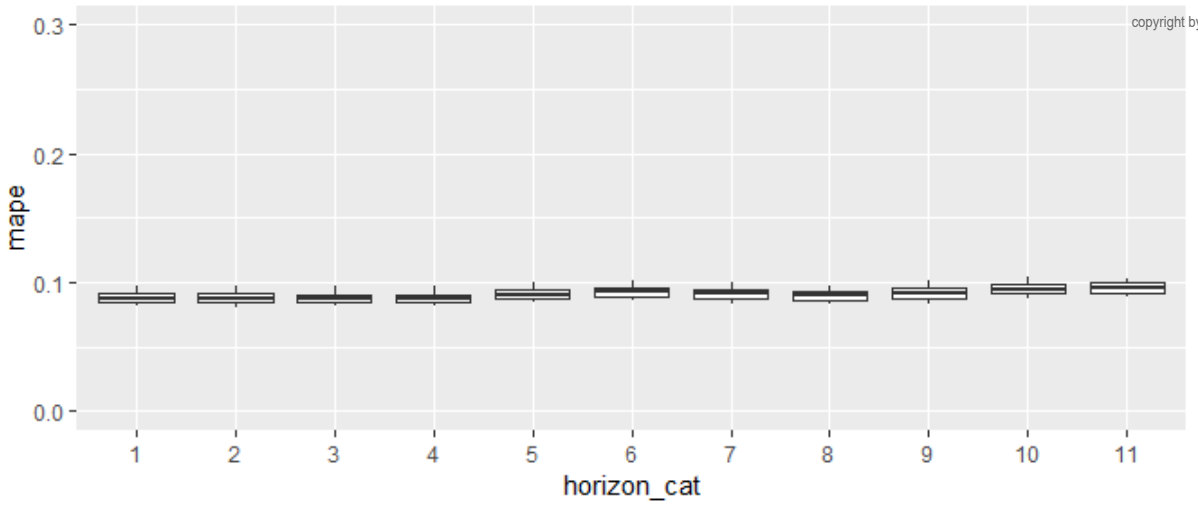
Cross-Validation

```
cv <- prophet::cross_validation(m, horizon = 365, initial=365*8, period = 365/12, units = 'days')
```

```
cv %>% prophet::plot_cross_validation_metric(metric='mape', rolling_window = .2)
```




```
cv %>% prophet::performance_metrics() %>% as_tibble() %>%
    mutate(horizon_cat = cut(as.numeric(horizon), 11, labels=1:11)) %>%
    ggplot(aes(x = horizon_cat, y= mape)) + geom_boxplot() + ylim(c(0,.3))
```



Die Cross-Validation zeigt, dass der Forecast Error MAPE im Schnitt leicht unter 10% liegt.

Aufgabe: Führe einen Timeseries Forecast mit zusätzlicher Regression durch

```
data(uschange, package='fpp2')

df <- as_tibble(uschange) %>%
    mutate(ds = time(uschange) %>% as_date) %>%
    rename(y = Consumption)

m <- prophet() %>%
    prophet::add_regressor('Income') %>%
    prophet::add_regressor('Production') %>%
    prophet::add_regressor('Savings') %>%
    prophet::add_regressor('Unemployment') %>%
    fit.prophet(df)
#> Disabling weekly seasonality. Run prophet with weekly.seasonality=TRUE to override this.
#> Disabling daily seasonality. Run prophet with daily.seasonality=TRUE to override this.
#> Initial log joint probability = -9.66493
#> Optimization terminated normally:
#>   Convergence detected: absolute parameter change was below tolerance

p <- df %>%
    predict(m, .)

dyplot.prophet(m, p)
```