

Rob J Hyndman
George Athanasopoulos

FORECASTING

PRINCIPLES AND PRACTICE

A comprehensive introduction to the latest forecasting methods using R. Learn to improve your forecast accuracy using dozens of real data examples.



3RD EDITION

 **OTexts**
OPEN TEXTS FOR PRACTICE

5. The forecaster's toolbox

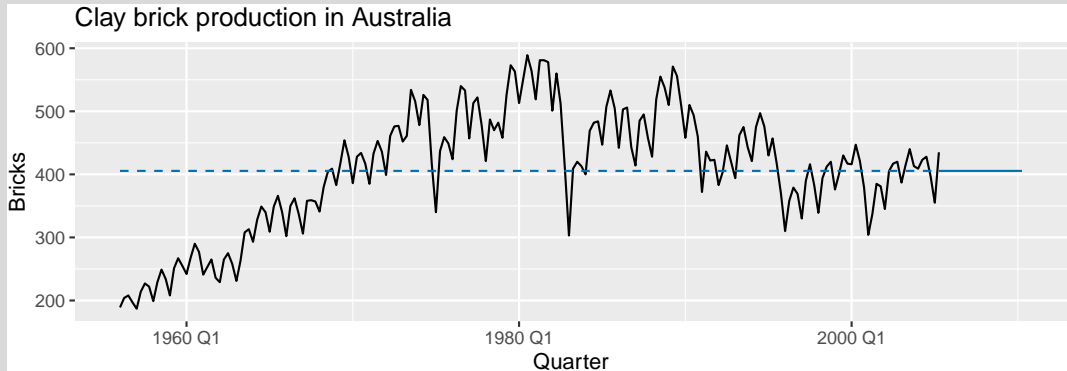
5.2 Some simple forecasting methods

OTexts.org/fpp3/

Some simple forecasting methods

MEAN(y): Average method

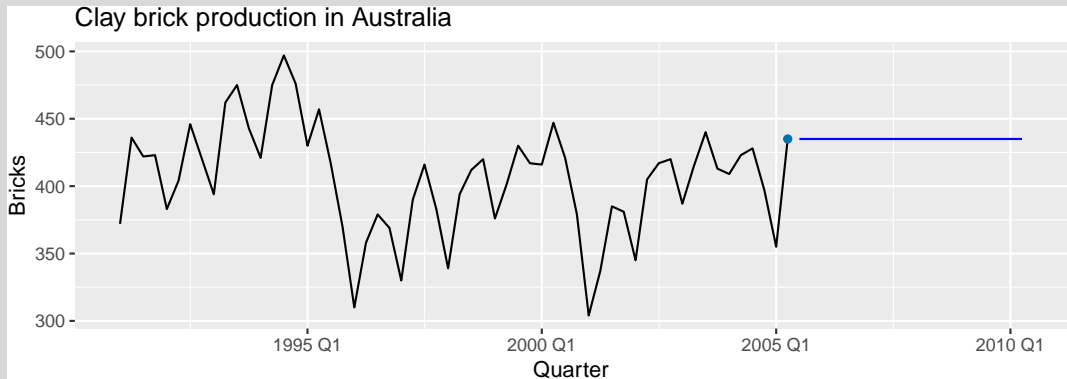
- Forecast of all future values is equal to mean of historical data $\{y_1, \dots, y_T\}$.
- Forecasts: $\hat{y}_{T+h|T} = \bar{y} = (y_1 + \dots + y_T)/T$



Some simple forecasting methods

NAIVE(y): Naïve method

- Forecasts equal to last observed value.
- Forecasts: $\hat{y}_{T+h|T} = y_T$.
- Consequence of efficient market hypothesis.

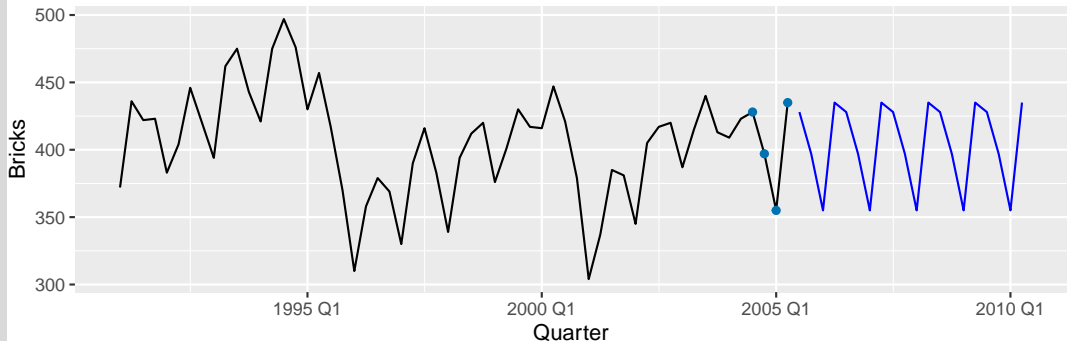


Some simple forecasting methods

SNAIVE($y \sim \text{lag}(m)$): Seasonal naïve method

- Forecasts equal to last value from same season.
- Forecasts: $\hat{y}_{T+h|T} = y_{T+h-m(k+1)}$, where m = seasonal period and k is the integer part of $(h-1)/m$.

Clay brick production in Australia



Some simple forecasting methods

`RW(y ~ drift())`: Drift method

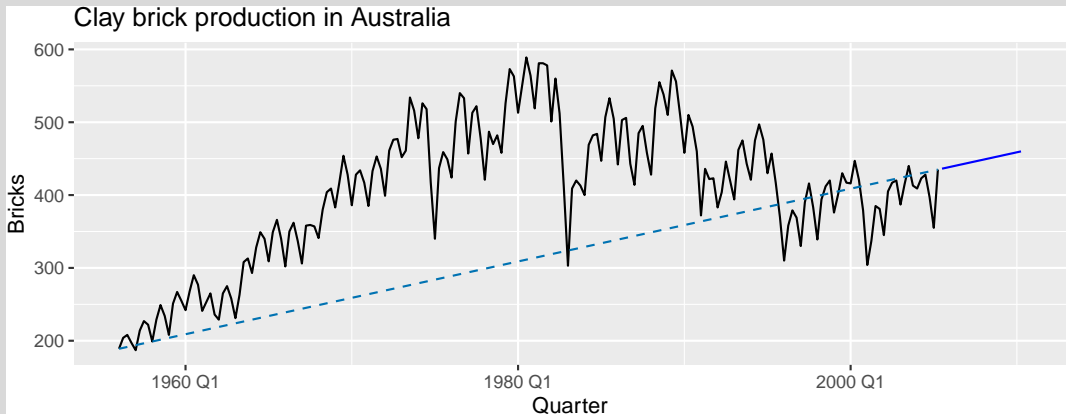
- Forecasts equal to last value plus average change.
- Forecasts:

$$\begin{aligned}\hat{y}_{T+h|T} &= y_T + \frac{h}{T-1} \sum_{t=2}^T (y_t - y_{t-1}) \\ &= y_T + \frac{h}{T-1} (y_T - y_1).\end{aligned}$$

- Equivalent to extrapolating a line drawn between first and last observations.

Some simple forecasting methods

Drift method



Model fitting

The `model()` function trains models to data.

```
brick_fit <- aus_production |>
  filter(!is.na(Bricks)) |>
  model(
    Seasonal_naive = SNAIVE(Bricks),
    Naive = NAIVE(Bricks),
    Drift = RW(Bricks ~ drift()),
    Mean = MEAN(Bricks)
  )
```

```
## # A mable: 1 x 4
##   Seasonal_naive   Naive         Drift    Mean
##           <model> <model>       <model> <model>
## 1           <SNAIVE> <NAIVE> <RW w/ drift> <MEAN>
```

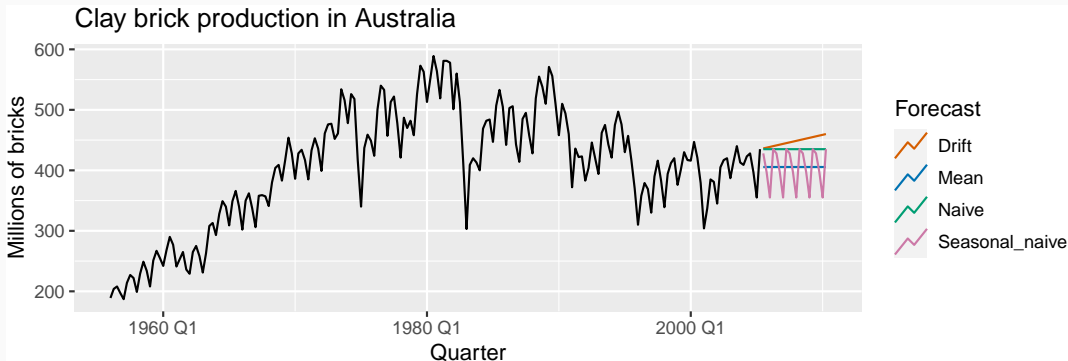
Producing forecasts

```
brick_fc <- brick_fit |>  
  forecast(h = "5 years")
```

```
## # A fable: 80 x 4 [1Q]  
## # Key:      .model [4]  
##   .model      Quarter      Bricks .mean  
##   <chr>        <qtr>        <dist> <dbl>  
## 1 Seasonal_naive 2005 Q3 N(428, 2336) 428  
## 2 Seasonal_naive 2005 Q4 N(397, 2336) 397  
## 3 Seasonal_naive 2006 Q1 N(355, 2336) 355  
## 4 Seasonal_naive 2006 Q2 N(435, 2336) 435  
## # ... with 76 more rows
```


Visualising forecasts

```
brick_fc |>  
  autoplot(aus_production, level = NULL) +  
  labs(title = "Clay brick production in Australia",  
        y = "Millions of bricks") +  
  guides(colour = guide_legend(title = "Forecast"))
```



Facebook closing stock price

```
# Extract training data
fb_stock <- gafa_stock |>
  filter(Symbol == "FB") |>
  mutate(trading_day = row_number()) |>
  update_tsibble(index = trading_day, regular = TRUE)

# Specify, estimate and forecast
fb_stock |>
  model(
    Mean = MEAN(Close),
    Naive = NAIVE(Close),
    Drift = RW(Close ~ drift())
  ) |>
  forecast(h = 42) |>
  autoplot(fb_stock, level = NULL) +
  labs(title = "Facebook closing stock price", y = "$US") +
  guides(colour = guide_legend(title = "Forecast"))
```

Facebook closing stock price

