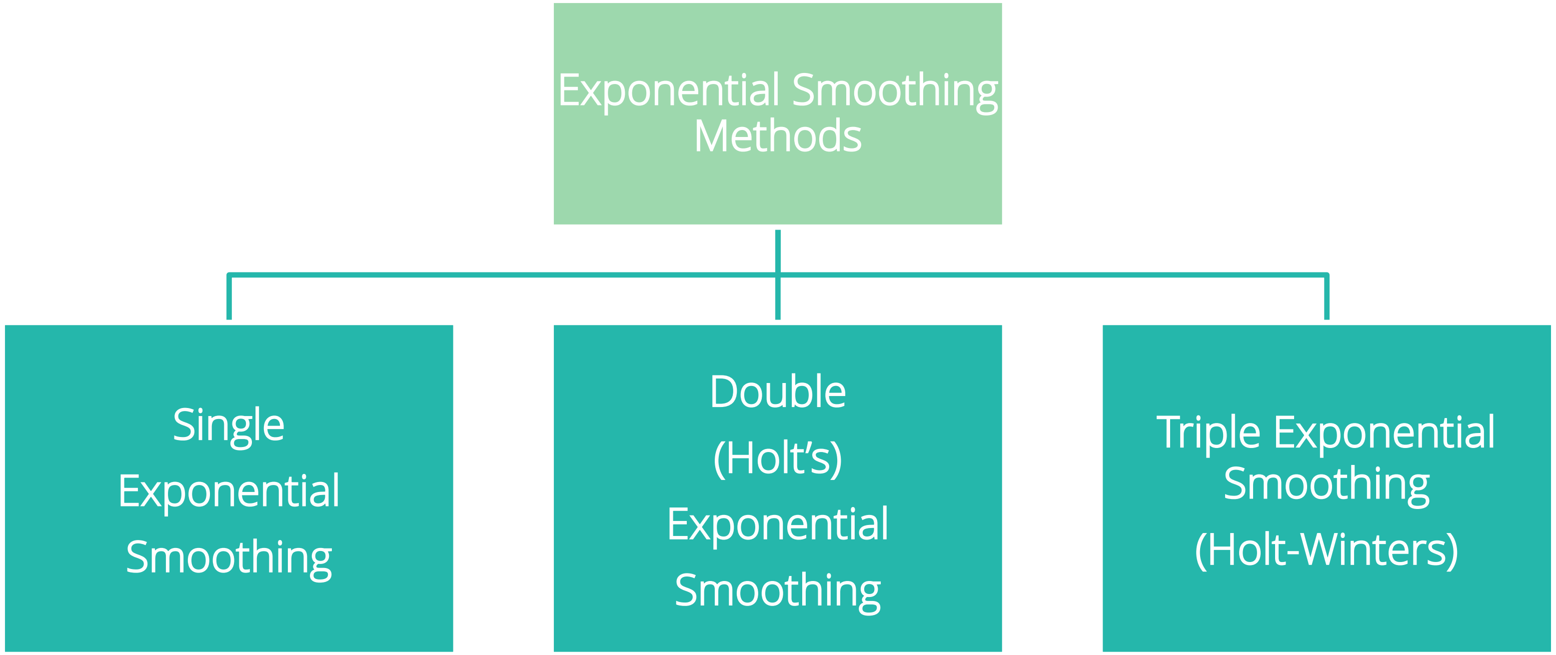
Time Series Analysis – Exponential Smoothing Methods for Forecasting

DATA SCIENCE INSTITUTE

Contents

Forecasting Using Smoothing Methods

- Random, unexplained variation in a time series can have an undesirable impact on forecasts
- Smoothing can cancel or reduce such impacts
- Smoothing can either be Simple (using Moving Averages) or Exponential



DATA SCIENCE INSTITUTE

Single Exponential Smoothing Model

Mathematical Model :

$$F_{t+1} = \alpha Y_t + (1 - \alpha) F_t$$

Where,

F_{t+1}	:	Forecast value for period t + 1
F_t	:	Forecast value for period t
Y_t	:	Actual value for period t
α	:	Alpha (Smoothing constant)

* Note that here we use the dataset "Sales Data for 3 Years"

DATA SCIENCE INSTITUTE

Single Exponential Smoothing Model

Assume α=0.8

t	yt	Ft	
1	23	-	
2	24	23	
3	26	23.80	=0.8*24+0.2*23
4	23.5	25.56	=0.8*26+0.2*23.8
5	27	23.91	
6	26.1	26.38	
7	28	26.16	
8	27	27.63	
9	29	27.13	
10	29.3	28.63	
11	28.2	29.17	
12	27	28.39	
		27.28	

* 1st future value will always be the previous value & then for rest future values exponential smoothing mathematical formula is applied.

DATA SCIENCE INSTITUTE

Single Exponential Smoothing Model - Smoothing Constant α

Values of α

close to one ✉ have less of a smoothing effect and give greater weight to recent changes in the data

closer to zero ✉ have a greater smoothing effect and are less responsive to recent changes

- There is no formally correct procedure for choosing α. Sometimes the statistician's judgment is used to choose an appropriate factor.
- Alternatively, α can be decided based on statistical measure such as Root Mean Squared Error.

Get an Edge!

Why the Name "Exponential"?

- This method gives weights to past observation in exponentially decreasing manner.

$$F_{t+1} = \alpha y_t + \alpha(1-\alpha) y_{t-1} + \alpha(1-\alpha)^2 y_{t-2} + \alpha(1-\alpha)^3 y_{t-3} \text{-----}$$

$$= \alpha y_t + (1-\alpha) [\alpha y_{t-1} + \alpha(1-\alpha) y_{t-2} + \alpha(1-\alpha)^2 y_{t-3} \text{----}]$$

$$= \alpha y_t + (1-\alpha) F_t$$

- Larger alpha gives more weight to recent values.

DATA SCIENCE INSTITUTE

Exponential Smoothing in R

```
#Importing the Data
salesseries<-ts(salesdata$Sales,start=c(2013,1), end=c(2015,12),
frequency=12)
```

```
#Single Exponential Smoothing
fit1<-HoltWinters(salesseries, beta=FALSE, gamma=FALSE)

predict(fit1,n.ahe
fit1
```

- **HoltWinters()** undertakes exponential smoothing.
- **beta = FALSE** and **gamma = FALSE** ensures single exponential smoothing is performed.

* Note that here we use the dataset “Sales Data for 3 Years”

Exponential Smoothing in R

Output

```
> predict(fit1,n.ahead=1)
          Jan
2016 314.4432
> fit1
Holt-Winters exponential smoothing without trend and without seasonal component.

Call:
HoltWinters(x = salesseries, beta = FALSE, gamma = FALSE)

Smoothing parameters:
 alpha: 0.754789
 beta : FALSE
 gamma: FALSE

Coefficients:
      [,1]
a 314.4432
```

Interpretation :

It returns predicted future value & value of alpha.

DATA SCIENCE INSTITUTE

Double (Holt) and Triple(Holt-Winters) Exponential Smoothing Methods

Double exponential smoothing has two equations

First equation is similar to single exponential smoothing method

Second equation updates trend using constant beta.

Double exponential smoothing method is used when there is a trend in the time series.

Triple exponential smoothing has three equations

First 2 equations are similar to double exponential smoothing method

Third equation updates seasonal component using constant gamma.

Triple exponential smoothing method is used when there is trend + seasonality in the time series.

Double Exponential Smoothing Model

Mathematical Model :

Where,

F_{t+1}	:	Forecast value for period t +1
F_t	:	Forecast value for period t
T_t	:	Trend component for period t
T_{t+1}	:	Trend component for period t +1
Y_t	:	Actual value for period t
α	:	Alpha (Smoothing constant)
β	:	Beta (Second smoothing constant)

DATA SCIENCE INSTITUTE

Double Exponential Smoothing in R

```
#Double Exponential Smoothing
fit2<-HoltWinters(salesseries, gamma=FALSE)
predict(fit2,n.ahead=1)
fit2
```

gamma = FALSE ensures double exponential smoothing

```
# Output
> predict(fit2,n.ahead=1)
          Jan
2016 306.1702
> fit2
Holt-Winters exponential smoothing with trend and without seasonal component.

Call:
HoltWinters(x = salesseries, gamma = FALSE)

Smoothing parameters:
 alpha: 0.3835632
 beta : 0.4889297
 gamma: FALSE

Coefficients:
       [,1]
a 294.81648
b  11.35368
```

Interpretation :

It returns predicted future value, value of alpha

DATA SCIENCE INSTITUTE

Triple Exponential Smoothing Model

Mathematical Model :

$$F_{t+1} = \alpha \frac{Y_t}{S_{t+1-k}} + (1-\alpha) F_t - T_t$$

where,

S_{t+1-k}	:	Seasonal smoothing value for period t +1
F_t	:	Forecast value for period t
F_{t+1}	:	Forecast value for period t +1
F_t	:	Forecast value for period t
T_t	:	Trend component for period t
T_{t+1}	:	Trend component for period t +1
Y_t	:	Actual value for period t

α : Alpha (Smoothing constant) β : Beta (Second smoothing constant)
: Gamma (Third smoothing constant)

DATA SCIENCE INSTITUTE

Triple Exponential Smoothing in R

```
#Triple Exponential Smoothing
fit3<-HoltWinters(salesseries)
predict(fit3,n.ahead=1)
fit3
```

```
# Output
> predict(fit3,n.ahead=1)
          Jan
2016 295.9492
> fit3
Holt-Winters exponential smoothing with trend and additive seasonal component.

Call:
HoltWinters(x = salesseries)

Smoothing parameters:
 alpha:  0.911556
 beta :  0
 gamma:  0.8681419

Coefficients:
             [,1]
a    296.725314337
b      4.522435897
s1    -5.298595983
s2    -1.290510241
s3    -3.658258513
s4    -0.005594377
s5    -3.580920942
s6    -2.015183167
s7     2.131638936
s8    -0.258654397
s9    -3.287793819
s10   -6.786550699
s11   -7.502574409
s12   31.125466834
```

Interpretation :

It returns predicted future value & value of

Get an Edge!

Always approach time series analysis in a systematic manner

Analyse
- Analyse trend and seasonality present in the data

Decompose
- Decompose time series into its components

Forecast
- Forecast future values of the time series

- For vector time series, investigate connections between two or more time series with the aim of using values of some of the processes to predict those of the others. (Eg. Pairs trading in stock market)

Exploring Built-In Time Series Data in R

```
#Fetching the Data
data("AirPassengers")

AirPassengers
```

AirPassengers() is Box-Jenkins data, with monthly totals of international airlines passengers (in thousands) from 1949 to 1960

```
# Output
```

```
> AirPassengers
     Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
1949 112 118 132 129 121 135 148 148 136 119 104 118
1950 115 126 141 135 125 149 170 170 158 133 114 140
1951 145 150 178 163 172 178 199 199 184 162 146 166
1952 171 180 193 181 183 218 230 242 209 191 172 194
1953 196 196 236 235 229 243 264 272 237 211 180 201
1954 204 188 235 227 234 264 302 293 259 229 203 229
1955 242 233 267 269 270 315 364 347 312 274 237 278
1956 284 277 317 313 318 374 413 405 355 306 271 306
1957 315 301 356 348 355 422 465 467 404 347 305 336
1958 340 318 362 348 363 435 491 505 404 359 310 337
1959 360 342 406 396 420 472 548 559 463 407 362 405
1960 417 391 419 461 472 535 622 606 508 461 390 432
```


DATA SCIENCE INSTITUTE

Exploring Built-In Time Series Data in R

```
# Checking Data Features
class(AirPassengers)
[1] "ts"

start(AirPassengers)
[1] 1949 1

end(AirPassengers)
[1] 1960 12

frequency(AirPassengers)
[1] 12
```

class() returns class of the data. AirPassengers data is a time series.

start() and **end()** return the first and last values in the data.

frequency() returns information about frequency of time points in the data, i.e. whether monthly, daily, etc.

Interpretation :

- The data is of class ts.
- The database starts at year 1949, month 1 and ends at year 1960 and month 12.
- Frequency = 12 suggests that the data is monthly.

Exploring Built-In Time Series Data in R

```
# Exploring Data
time(AirPassengers)
```

time() shows time at which the time series was sampled.

```
# Output
```

```
> time(AirPassengers)
          Jan      Feb      Mar      Apr      May      Jun      Jul      Aug      Sep      Oct      Nov      Dec
1949 1949.000 1949.083 1949.167 1949.250 1949.333 1949.417 1949.500 1949.583 1949.667 1949.750 1949.833 1949.917
1950 1950.000 1950.083 1950.167 1950.250 1950.333 1950.417 1950.500 1950.583 1950.667 1950.750 1950.833 1950.917
1951 1951.000 1951.083 1951.167 1951.250 1951.333 1951.417 1951.500 1951.583 1951.667 1951.750 1951.833 1951.917
1952 1952.000 1952.083 1952.167 1952.250 1952.333 1952.417 1952.500 1952.583 1952.667 1952.750 1952.833 1952.917
1953 1953.000 1953.083 1953.167 1953.250 1953.333 1953.417 1953.500 1953.583 1953.667 1953.750 1953.833 1953.917
1954 1954.000 1954.083 1954.167 1954.250 1954.333 1954.417 1954.500 1954.583 1954.667 1954.750 1954.833 1954.917
1955 1955.000 1955.083 1955.167 1955.250 1955.333 1955.417 1955.500 1955.583 1955.667 1955.750 1955.833 1955.917
1956 1956.000 1956.083 1956.167 1956.250 1956.333 1956.417 1956.500 1956.583 1956.667 1956.750 1956.833 1956.917
1957 1957.000 1957.083 1957.167 1957.250 1957.333 1957.417 1957.500 1957.583 1957.667 1957.750 1957.833 1957.917
1958 1958.000 1958.083 1958.167 1958.250 1958.333 1958.417 1958.500 1958.583 1958.667 1958.750 1958.833 1958.917
1959 1959.000 1959.083 1959.167 1959.250 1959.333 1959.417 1959.500 1959.583 1959.667 1959.750 1959.833 1959.917
1960 1960.000 1960.083 1960.167 1960.250 1960.333 1960.417 1960.500 1960.583 1960.667 1960.750 1960.833 1960.917
```

Interpretation :

- The output shows time for each observation as year followed by time stamp.
- Time stamp for Jan it’s 0/12, Feb it’s 1/12,……. So on

DATA SCIENCE INSTITUTE

Exploring Built-In Time Series Data in R

```
cycle(AirPassengers)
```

cycle() shows positions of each observation in the cycle.

```
# Output
```

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
1949	1	2	3	4	5	6	7	8	9	10	11	12
1950	1	2	3	4	5	6	7	8	9	10	11	12
1951	1	2	3	4	5	6	7	8	9	10	11	12
1952	1	2	3	4	5	6	7	8	9	10	11	12
1953	1	2	3	4	5	6	7	8	9	10	11	12
1954	1	2	3	4	5	6	7	8	9	10	11	12
1955	1	2	3	4	5	6	7	8	9	10	11	12
1956	1	2	3	4	5	6	7	8	9	10	11	12
1957	1	2	3	4	5	6	7	8	9	10	11	12
1958	1	2	3	4	5	6	7	8	9	10	11	12
1959	1	2	3	4	5	6	7	8	9	10	11	12
1960	1	2	3	4	5	6	7	8	9	10	11	12

Interpretation :

- The output shows frequency of data points, represented numerically

Plotting Data with Trend Line

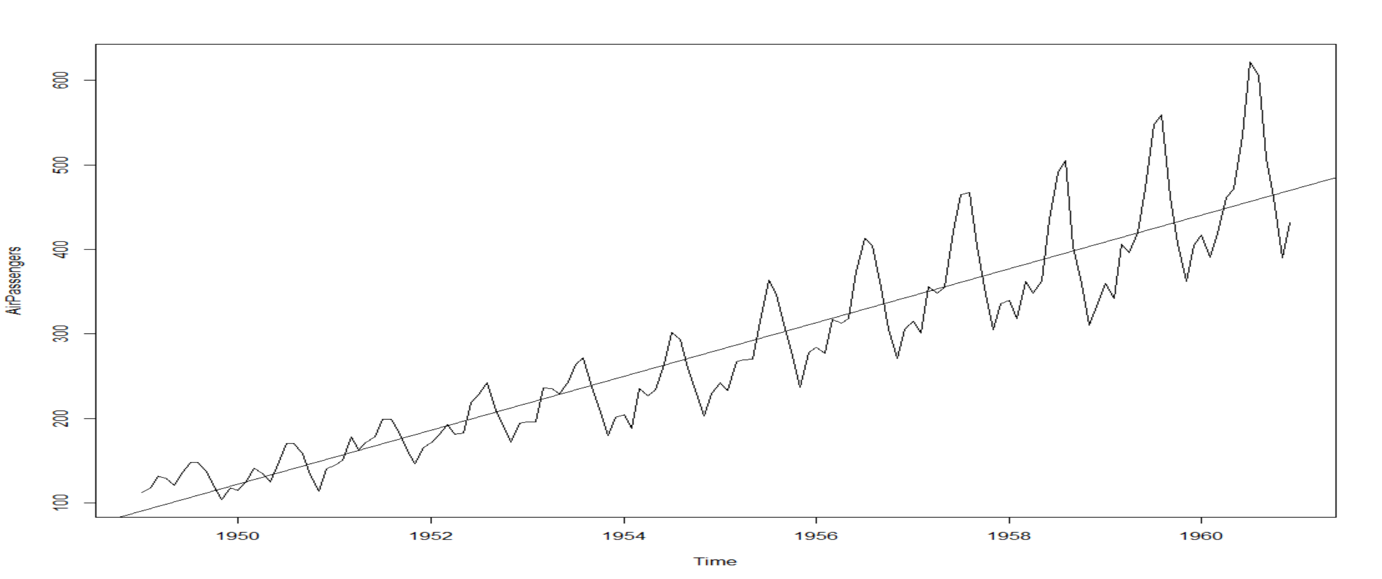
```
# Plotting Data with Trend Line
plot(AirPassengers)

model<-lm(AirPassengers~time(A

abline(model)
```

plot() returns a simple line plot of the entire data.

lm() fits a linear regression model on the data with observations as dependent variables and time stamps as independent variables.

abline() adds a straight line to the plot.

```
# Output
```



Interpretation :

- The plot shows a positive trend.

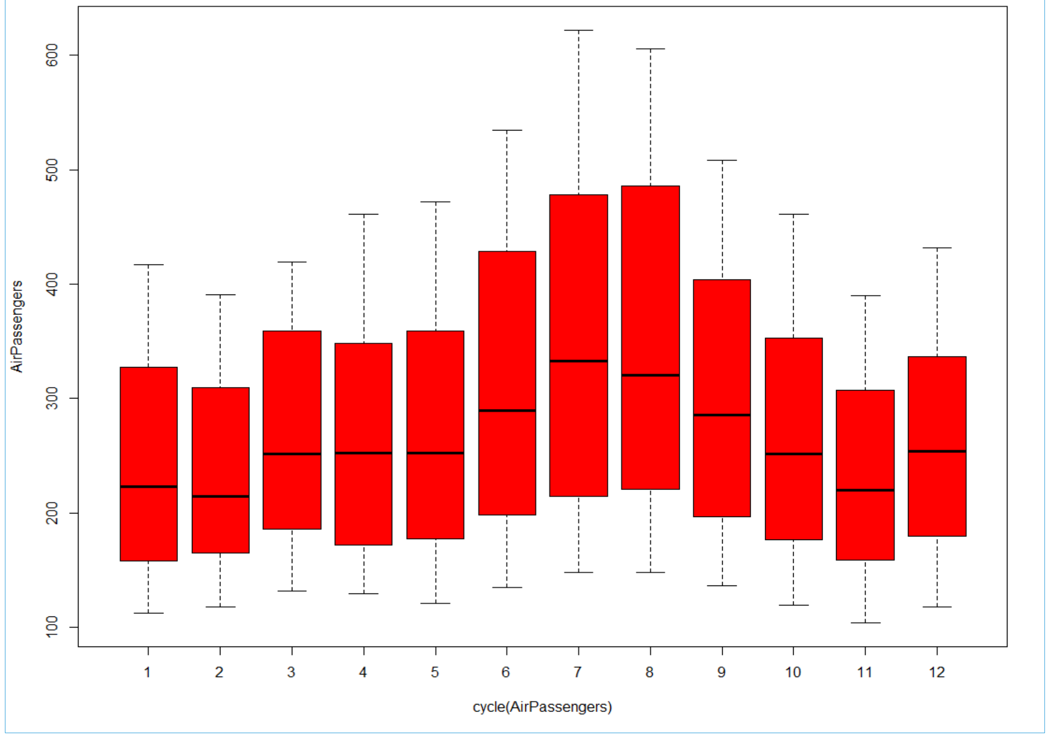
DATA SCIENCE INSTITUTE

Box Plot with Cycles

```
#Box Plot for Cycle
boxplot(AirPassengers~cycle(AirPassengers),col="red")
```

boxplot() generates a Box-plot for data cycles.

```
# Output
```



Interpretation :

- The plot gives a clear indication that number of passengers in the months of July and August were higher than the rest.

DATA SCIENCE INSTITUTE

Quick Recap

In this session, we learnt about **time series exponential smoothing**:

Smoothing	• Smoothing gives weights to past observations, in order to give more significance to seasonality and trend components of a time series.
Smoothing in R	• Use `HoltWinters()` to carry out exponential smoothing

DATA SCIENCE INSTITUTE