Lab experiment - 4

Name: Prithviraj Guntha

Reg. No.: 20BRS1188

Subject: Essentials of data analytics

Subject code: CSE3506

Professor: A Sheik Abdullah

Slot: L55+L56

1. **Calculate simple 3 year moving average forecast for the following sales data.**

year	1	2	3	4	5	6	7	8	9	10	11	12
Sales	5.2	4.9	5.5	4.9	5.2	5.7	5.4	5.8	5.9	6	5.2	4.8

```
sales <- c(5.2,4.9,5.5,4.9,5.2,5.7,5.4,5.8,5.9,6,5.2,4.8)
moving_average <- rollmean(sales, 3, align = "right")
print(moving_average)
forecasting_errors <- sales - moving_average
# Calculate the mean absolute error
mae <- mean(abs(forecasting_errors))
# Calculate the mean squared error
mse <- mean(forecasting_errors^2)
# Calculate the root mean squared error
rmse <- sqrt(mse)
# Calculate the mean absolute percentage error
mape <- mean(abs(forecasting_errors/sales))*100
print(mae)
print(mse)
print(rmse)
print(mape)
```

```r
#question 1:
sales <- c(5.2,4.9,5.5,4.9,5.2,5.7,5.4,5.8,5.9,6,5.2,4.8)
moving_average <- rollmean(sales, 3, align = "right")
print(moving_average)
forecasting_errors <- sales - moving_average
# Calculate the mean absolute error
mae <- mean(abs(forecasting_errors))
# Calculate the mean squared error
mse <- mean(forecasting_errors^2)
# Calculate the root mean squared error
rmse <- sqrt(mse)
# Calculate the mean absolute percentage error
mape <- mean(abs(forecasting_errors/sales))*100
print(mae)
print(mse)
print(rmse)
print(mape)
```

Output:

```
>
> #question 1:
> sales <- c(5.2,4.9,5.5,4.9,5.2,5.7,5.4,5.8,5.9,6,5.2,4.8)
> moving_average <- rollmean(sales, 3, align = "right")
> print(moving_average)
 [1] 5.200000 5.100000 5.200000 5.266667 5.433333 5.633333 5.700000 5.900000 5.700000
[10] 5.333333
> forecasting_errors <- sales - moving_average
Warning message:
In sales - moving_average :
  longer object length is not a multiple of shorter object length
> # Calculate the mean absolute error
> mae <- mean(abs(forecasting_errors))
> # Calculate the mean squared error
> mse <- mean(forecasting_errors^2)
> # Calculate the root mean squared error
> rmse <- sqrt(mse)
> # Calculate the mean absolute percentage error
> mape <- mean(abs(forecasting_errors/sales))*100
> print(mae)
[1] 0.2277778
> print(mse)
[1] 0.08314815
> print(rmse)
[1] 0.2883542
> print(mape)
[1] 4.225548
>
```

2. **Calculate the 4 year moving average for the following data,**

Year	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012
Sales	4	6	5	8	9	5	4	3	7	8

```
#question 2:
sales <- c(4,6,5,8,9,5,4,3,7,8)
moving_average <- rollmean(sales, 4, align = "right")
print(moving_average)
forecasting_errors <- sales - moving_average
# Calculate the mean absolute error
mae <- mean(abs(forecasting_errors))
```

```
# Calculate the mean squared error
mse <- mean(forecasting_errors^2)
# Calculate the root mean squared error
rmse <- sqrt(mse)
# Calculate the mean absolute percentage error
mape <- mean(abs(forecasting_errors/sales))*100
print(mae)
print(mse)
print(rmse)
print(mape)
```

```
#question 2:
sales <- c(4,6,5,8,9,5,4,3,7,8)
moving_average <- rollmean(sales, 4, align = "right")
print(moving_average)
forecasting_errors <- sales - moving_average
# Calculate the mean absolute error
mae <- mean(abs(forecasting_errors))
# Calculate the mean squared error
mse <- mean(forecasting_errors^2)
# Calculate the root mean squared error
rmse <- sqrt(mse)
# Calculate the mean absolute percentage error
mape <- mean(abs(forecasting_errors/sales))*100
print(mae)
print(mse)
print(rmse)
print(mape)
```

Output:

```
>
> #question 2:
> sales <- c(4,6,5,8,9,5,4,3,7,8)
> moving_average <- rollmean(sales, 4, align = "right")
> print(moving_average)
[1] 5.75 7.00 6.75 6.50 5.25 4.75 5.50
> forecasting_errors <- sales - moving_average
Warning message:
In sales - moving_average :
  longer object length is not a multiple of shorter object length
```

```
> # Calculate the mean absolute error
> mae <- mean(abs(forecasting_errors))
> # Calculate the mean squared error
> mse <- mean(forecasting_errors^2)
> # Calculate the root mean squared error
> rmse <- sqrt(mse)
> # Calculate the mean absolute percentage error
> mape <- mean(abs(forecasting_errors/sales))*100
> print(mae)
[1] 1.55
> print(mse)
[1] 3.4875
> print(rmse)
[1] 1.867485
> print(mape)
[1] 30.5625
```

3. **Explain what is autocorrelation, give the implementation steps with the help of an example. Bring out the inference from your results obtained.**

Autocorrelation is a statistical technique that measures the correlation of a time series data with its own lagged values. Any autocorrelation that may be present in time series data is determined using a correlogram, also known as an ACF plot. This is used to help you determine whether your series of numbers is exhibiting autocorrelation at all, at which point you can then begin to better understand the pattern that the values in the series may be predicting.

The coefficient of correlation between two values in a time series is called the autocorrelation function(ACF).
The formula for autocorrelation is given by,

$$r_k = \frac{\sum_{t=k+1}^{n}(Y_t - \overline{Y})(Y_{t-K} - \overline{Y})}{\sum_{t=1}^{n}(Y_t - \overline{Y})^2}$$

The resulting coefficients will be between -1 and 1. A positive coefficient means that the time series is positively correlated with its lags and a negative coefficient means that the time series is negatively correlated with its lags.

The steps that are involved in performing autocorrelation in Rstudio is as follows:

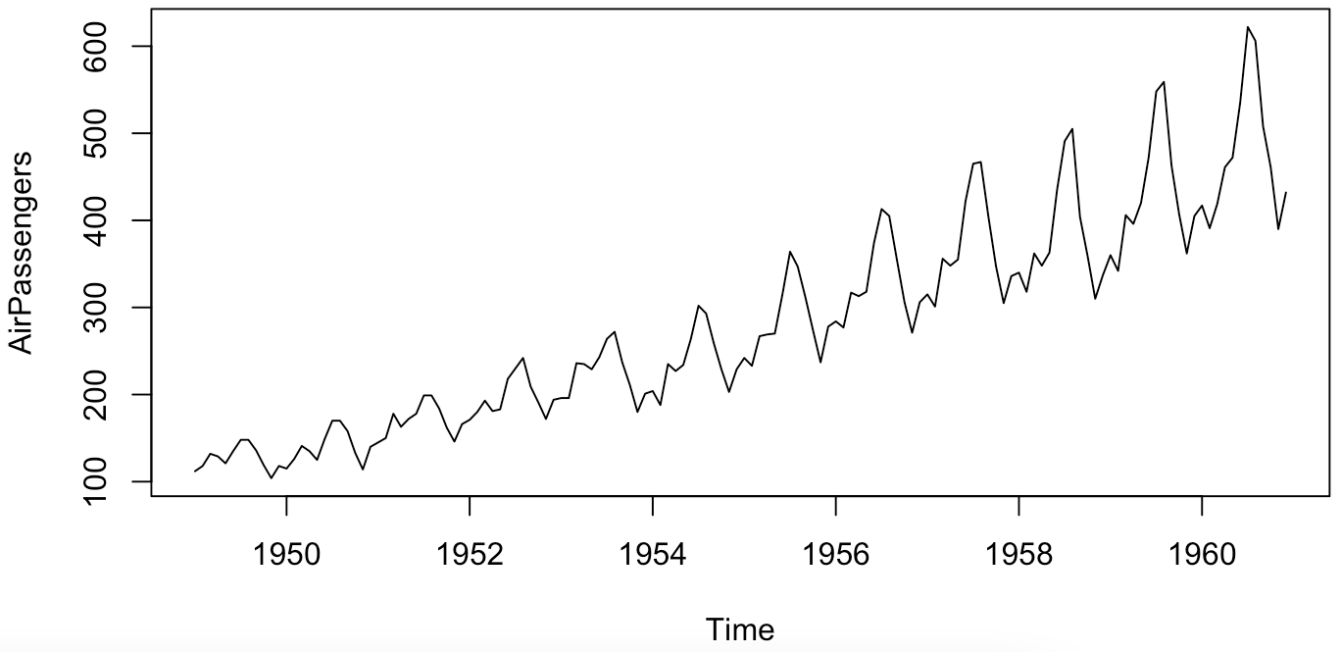
1. Import a time series dataset.
2. Calculate the autocorrelation using the 'acf()' function.
3. Plot the autocorrelation.
4. Find the autocorrelation coefficients.
5. Interpret the results and identify the patterns.

Here is an example of autocorrelation using the Air passengers dataset, that contains the number of passengers travelled for every month for different years.

```r
#question 3:
# Load the necessary library
library(forecast)

# Load the time series data
data(AirPassengers)
```

We now plot the data to see how it plays out,

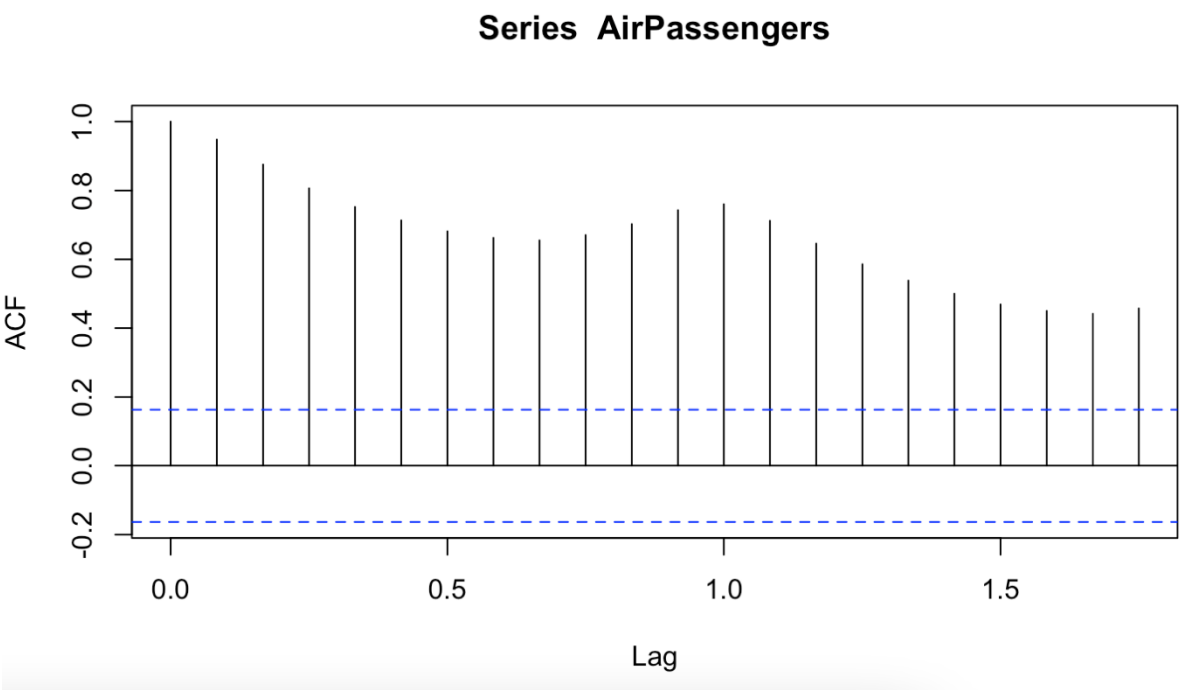


We now perform autocorrelation,

```
51
52  # Calculate the autocorrelation coefficient
53  autocorrelation <- acf(AirPassengers)
54
55  # Print the autocorrelation coefficient
56  print(autocorrelation$acf)
```

```
> # Print the autocorrelation coefficient
> print(autocorrelation$acf)
, , 1

           [,1]
 [1,] 1.0000000
 [2,] 0.9480473
 [3,] 0.8755748
 [4,] 0.8066812
 [5,] 0.7526254
 [6,] 0.7137700
 [7,] 0.6817336
 [8,] 0.6629044
 [9,] 0.6556105
[10,] 0.6709483
[11,] 0.7027199
[12,] 0.7432402
[13,] 0.7603950
[14,] 0.7126609
[15,] 0.6463423
[16,] 0.5859234
[17,] 0.5379552
[18,] 0.4997475
[19,] 0.4687340
[20,] 0.4498707
[21,] 0.4416288
[22,] 0.4572238
```



Inference:

The solid blue line in the graph represents the 95% confidence interval for the autocorrelation coefficient. If the autocorrelation coefficient falls outside of this interval, it is considered statistically significant. From the graph, it is clear that there is positive autocorrelation.