Joseph Flynn

STA 4155

1. Let us start by plotting the time series data as a scatterplot. To do this, we use the command:

plot(coffee$time,coffee$price,main="Coffee Price vs Time",ylab = 'Coffee Price',xlab = 'Time')

To produce the time series:

Chart, scatter chart

Description automatically generated

The time series above shows a downward trend with peaks that occur about every month. So, there maybe some seasonal element to the graph as these peaks happen within the same year. This element needs to be accounted for, in order to create an accurate prediction of the time series.

1. A method to account for the seasonal element is by using a simple moving average. We can create more than one simple moving average with different lengths because as the length of the moving average increases, it creates a smoother trend line. For this, we can create two simple moving averages with one having a length of 2 and is plotted as red, while the other has a length of 8 and is plotted as green. To so this, we use the commands:

library(TTR)

sma2 = SMA(coffee$price, n = 2)

sma8 = SMA(coffee$price, n = 8)

While to plot these two trends, we use the command:

plot(coffee$time,coffee$price,main="Coffee Price vs Time",ylab = 'Coffee Price',xlab = 'Time',)

lines(coffee$time,sma2, col = 'red')

lines(coffee$time,sma8, col = 'green')

legend("topright", legend=c("SMA with n = 2", "SMA with n = 8"),col=c("red", "green"),lty =1)

This produces the time series plot:

Chart, line chart

Description automatically generated

From the plot above, it is obvious that the length of 2 simple moving average follows the data points more closely and isn’t as smooth. Plus, the length of 2 starts at the second data point, while the length of 8 starts at the eight data point. In contrast, the length of 8 has a smoother line, but still follows the overall trend.

1. However, you may want to try another method like single exponential smoothing. This method involves an alpha, with the larger the alpha means the less smooth the trend. We can create two with different alphas, one with an alpha of 0.8 in orange and another with an alpha of 0.2 in purple. To do this, we use the commands (an n of 1 for single exponential smoothing):

ema0.8 = EMA(coffee$price, ratio = 0.8, n = 1)

ema0.2 = EMA(coffee$price, ratio = 0.2, n = 1)

To plot these:

plot(coffee$time,coffee$price,main="Coffee Price vs Time",ylab = 'Coffee Price',xlab = 'Time')

lines(coffee$time,ema0.8, col = 'orange')

lines(coffee$time,ema0.2, col = 'purple')

legend("topright", legend=c("SES with a = 0.8", "SES with a = 0.2"),col=c("orange", "purple"),lty =1)

This creates the graph below.

Chart, histogram

Description automatically generated

The graph above shows the 0.8 alpha having the most recent value being weighted heaviest and it follows more closely to the data. While the lower alpha of 0.2 doesn’t weigh most recent points as heavily and creates a smoother curve.

1. Another method is an autoregressive model. To do this, we first need to setup lags for the data. To do this, we use the command:

acf(coffee$price,lag.max = 5, plot = FALSE)

This produces the graph:

Text

Description automatically generated

From here, we eliminate the lags from 4 to 5 because they are less than 0.8. So we will use lags 1 to 3. To create the model, we use the commands:

ar1 = ar(coffee$price, aic = FALSE, order.max = 3, demean = FALSE, intercept = TRUE, method = 'ols')

This produces the model:

Graphical user interface, text, application, email

Description automatically generated

ˆyt = 7.047 + 0.8456(yt−1) + 0.1032(yt−2) + 0.0004(yt−3)

The model created weighs heavily the first lag, due to the coefficient of that lag having the largest coefficient in comparison to the others. This maybe to the model weighing the most recent data points heavier, in order to better follow the trend. From here, we can add the model to the plot by using the command:

fitted.ar1 = coffee$price-ar1$resid

plot(coffee$time,coffee$price,main="Coffee Price vs Time",ylab = 'Coffee Price',xlab = 'Time')

lines(coffee$time,fitted.ar1,col='blue')

This produces the plot:

Chart

Description automatically generated

1. But, to check how well the models preformed is by calculating the ABS for each model. The lower the ABS means the closer to the actual answer the model. In this case, the actual value for the predicted value is 138.90. To calculate the ABS, we use the commands:

#ABS

#SMA

abs.sma2 = abs(yhat.actual - yhat.sma2)/abs(yhat.actual)\*100

abs.sma8 = abs(yhat.actual - yhat.sma8)/abs(yhat.actual)\*100

#EMA

abs.ema0.2 = abs(yhat.actual - yhat.ema0.2)/abs(yhat.actual)\*100

abs.ema0.8 = abs(yhat.actual - yhat.ema0.8)/abs(yhat.actual)\*100

#Auto-reg

abs.ar1 = abs(yhat.actual - yhat.ar1)/abs(yhat.actual)\*100

From these, we get the ABS for each being:

* ABS for SMA with length of 2 being 2.105838%
* ABS for SMA with length of 8 being 2.28582%
* ABS for SES with alpha of 0.2 being 1.997467%
* ABS for SES with alpha of 0.8 being 2.085957%
* ABS for autoregressive model being 1.937486%

From this, we can see that the autoregressive model is the best for predicting the next value because it had the lowest ABS at 1.937486%. However, all are fairly close in absolute percent error.