ISYE 6501 Week 4 Homework

2023-09-12

Question 7.1

Describe a situation or problem from your job, everyday life, current events, etc., for which exponential smoothing would be appropriate. What data would you need? Would you expect the value of alpha (the first smoothing parameter) to be closer to 0 or 1, and why?

An exponential smoothing model would be an appropriate model to analyze investment earnings over time. For this model, I would need access to monthly bank statements for data on account balances over a fixed time period. The value of alpha would depend largely on the type of investment. For low risk investments like bonds or index funds, I would expect the value of alpha to be closer to 1. This is because there is not as much randomness, therefore, recent market trends can be a good indicator of future performance. Conversely, I would expect the value of alpha to be closer to 0 for high risk investments because there is more randomness and volatility.

Question 7.2

Using the 20 years of daily high temperature data for Atlanta (July through October) from Question 6.2 (file temps.txt), build and use an exponential smoothing model to help make a judgment of whether the unofficial end of summer has gotten later over the 20 years. (Part of the point of this assignment is for you to think about how you might use exponential smoothing to answer this question. Feel free to combine it with other models if you'd like to. There's certainly more than one reasonable approach.

```r
#set up work space and load libraries
rm(list = ls()) #clear out environment
library(ggplot2)
library(tidyr)
library(plotly)
```

```
##
## Attaching package: 'plotly'

## The following object is masked from 'package:ggplot2':
##
##     last_plot

## The following object is masked from 'package:stats':
##
##     filter

## The following object is masked from 'package:graphics':
##
##     layout
```

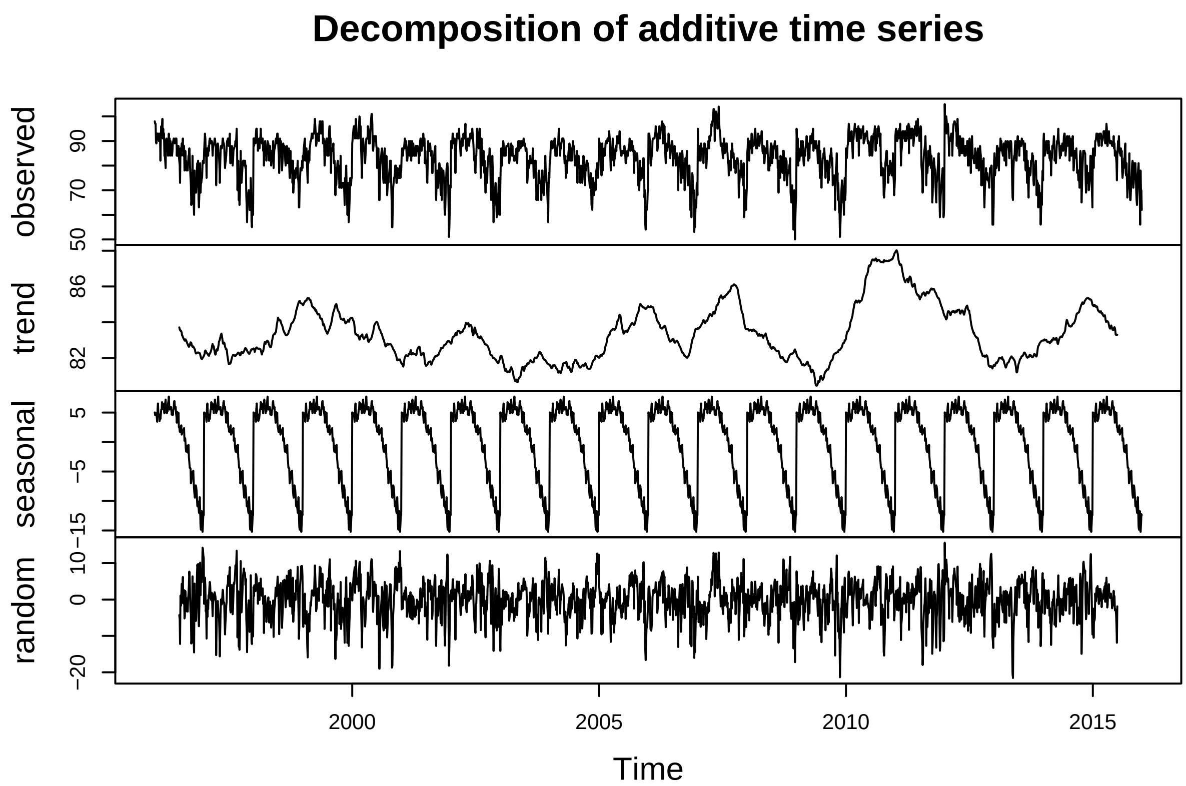
```r
#load the data
temp_data <- read.table("temps.txt", header=TRUE)
```

As a first step to answer this question, I reformatted my data as a time series object so I could use exponential smoothing. To do this, I restructured the temp_data data frame within two columns: DATE and TEMP. Then, I passed this data frame into the ts() function to generate a time series object.

```r
#pivot the table
temp_data <- pivot_longer(temp_data,
                          cols = -"DAY",
                          names_to = "YEAR",
                          values_to = "TEMP"
                          )
temp_data <- temp_data[,c("YEAR", "DAY", "TEMP")]

#sort table by "YEAR" in ascending order
temp_data <- temp_data[order(temp_data$YEAR),]

#remove X from the "YEAR" (e.g., X1996 becomes 1996)
temp_data$YEAR <- substr(temp_data$YEAR, 2, nchar(temp_data$YEAR))

#combine "YEAR" and "DAY" to make a date column
date_string <- paste(temp_data$YEAR, temp_data$DAY, sep="-")
temp_data$DATE <- as.Date(date_string, format="%Y-%d-%b")

#update table to have only "DATE" and "TEMP"
temp_data <- subset(temp_data, select=-c(1,2))
temp_data <- temp_data[, c("DATE", "TEMP")]

#convert data to a time series object
ts_temp_data <- ts(temp_data$TEMP,
                   frequency=123,
                   start=c(1996, 1)
                   )
```

Before fitting an exponential smoothing model, I used the decompose() function to observe the trend, seasonal, and random components of my time series data. As shown in the graph below, the time series data exhibits seasonality, but no consistent trend over time. Additionally, the function automatically selected additive decomposition over multiplicative decomposition since the seasonal patterns appear to have consistent magnitude over time.

```r
#decompose time series into seasonal, trend, and irregular components
components <- decompose(ts_temp_data, type=c("additive", "multiplicative"))
plot(components)
```



Next, I fitted an exponential smoothing model using the base R HoltWinters() function. In this model, I left the alpha, beta, and gamma parameters as NULL and let R figure out the optimal parameters on its own. As shown below, this model determined alpha = 0.66, beta = 0, and gamma = 0.62 as optimal parameters. The chosen alpha parameter indicates that the model relies slightly more on recent data than historical data; the beta parameter indicates there is no clear trend in the time series data; and the gamma parameter indicates that the model is incorporating a moderate level of smoothing to the seasonal component.

Also shown below is the model's sum of squared error (SSE). This can be used to measure how well the model fits the historical time series data by quantifying the difference between observed values and predicted values. A lower SSE is generally the goal, as it indicates a better fit of the model to the historical data.

```r
#fit Holt Winters, let R figure out tuning parameters on its own
hw_model1 <- HoltWinters(ts_temp_data, seasonal="additive")

#save and print hw_model1 alpha, beta, and gamma parameters
HW1_alpha <- hw_model1[["alpha"]]
HW1_beta <- hw_model1[["beta"]]
HW1_gamma <- hw_model1[["gamma"]]

print(paste0("alpha: ", round(HW1_alpha, 2)))
```

```
## [1] "alpha: 0.66"
```

```r
print(paste0("beta: ", round(HW1_beta, 2)))
```

```
## [1] "beta: 0"
```

```r
print(paste0("gamma: ", round(HW1_gamma, 2)))
```

```
## [1] "gamma: 0.62"
```

```r
#print SSE
print(paste0("SSE: ", round(hw_model1[["SSE"]], 2)))
```

```
## [1] "SSE: 66244.25"
```

To be comprehensive, I re-fitted the model using multiplicative seasonality and compared the parameters and SSE to my previous model. As you can see below, the parameters are similar, but the SSE is higher than in hw_model1, indicating hw_mdoel2 does not fit the data as well.

```r
#fit Holt Winters, let R figure out tuning parameters on its own
hw_model2 <- HoltWinters(ts_temp_data, seasonal="multiplicative")

#save and print hw_model1 alpha, beta, and gamma parameters
HW2_alpha <- hw_model2[["alpha"]]
HW2_beta <- hw_model2[["beta"]]
HW2_gamma <- hw_model2[["gamma"]]

print(paste0("alpha: ", round(HW2_alpha, 2)))
```

```
## [1] "alpha: 0.62"
```

```r
print(paste0("beta: ", round(HW2_beta, 2)))
```

```
## [1] "beta: 0"
```

```r
print(paste0("gamma: ", round(HW2_gamma, 2)))
```

```
## [1] "gamma: 0.55"
```

```r
#print SSE
print(paste0("SSE: ", round(hw_model2[["SSE"]], 2)))
```

```
## [1] "SSE: 68904.57"
```

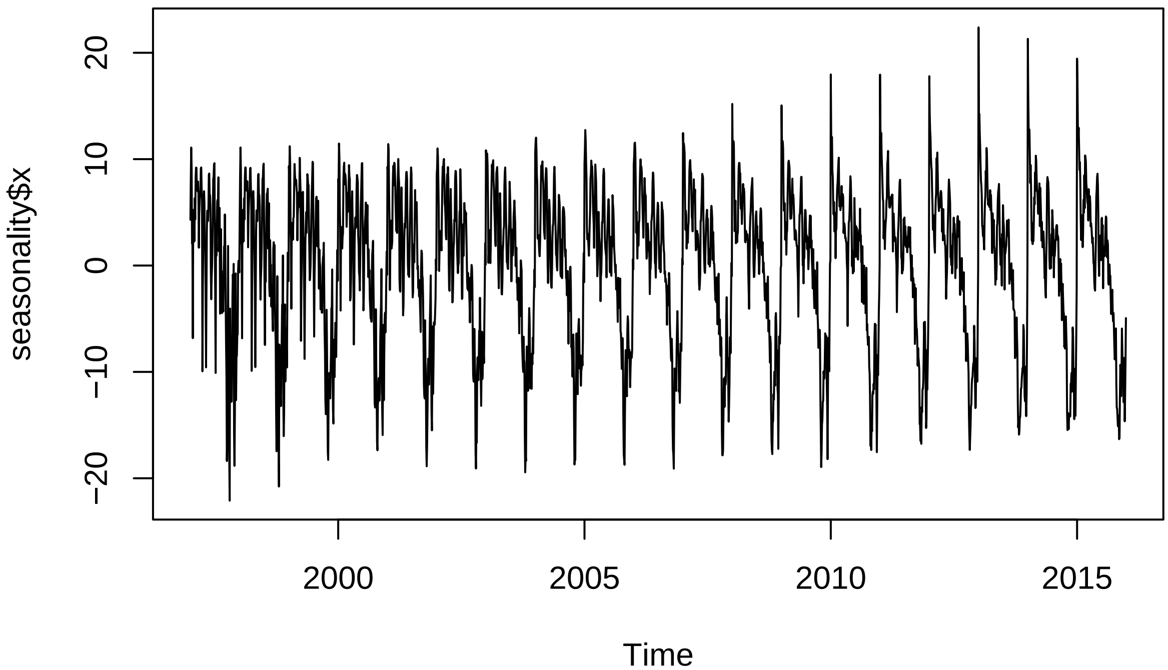
To further explore, I used the CUSUM approach on the seasonal component of the fitted exponential smoothing model. A detected change in the seasonal component would imply that the magnitude of summer temperatures has increased over time. Consequently, the end of summer would also be later.

To start, I took the seasonal component of the fitted exponential smoothing model and stored it in its own data frame. I also added an empty column to store a value of "St."

```r
seasonality <- as.data.frame(hw_model1$fitted[,4])
seasonality$St <- NA
```

To approximate the target mean, I plotted the seasonality component to see if there are any clear increases or decreases in the seasonality component over time. Because there is no clear trend, I used the entire data frame to approximate the target mean. I also used the data frame's standard deviation to calculate a starting threshold and C-value.

```r
#visualize seasonality component
plot(seasonality$x)
```




```r
#calculate target mean for CUSUM approach
mean <- mean(seasonality$x)
mean
```

```
## [1] -0.01576207
```

```r
#determine threshold for CUSUM approach
std_dev <- sd(seasonality$x)
threshold <- 5*std_dev
threshold
```

```
## [1] 35.58207
```

```r
#determine c-value for CUSUM approach
C <- 2*std_dev
C
```

```
## [1] 14.23283
```

Then, I applied the CUSUM approach to determine if there were any values of St that exceeded the pre-determined threshold.

```r
#calculate first value of St
first_St <- max(0, mean - seasonality[1, "x"])
seasonality[1, "St"] <- first_St

#calculate remaining values of St
for(i in 2:nrow(seasonality)){

  #calculate x and St-1
  x <- seasonality[i, "x"]
  previous_St <- seasonality[i-1, "St"]

  #calculate St
  St <- max(0, previous_St +(x - mean - C))

  #update table with value of St
  seasonality[i, "St"] <- St
}

#does St exceed threshold?
for(i in 1:nrow(seasonality)){
  x <- NULL
  if(seasonality[i, "St"] >= threshold){
    x <- i
  }
}

if(is.numeric(x)){
  print("St exceeds threshold")
} else{
  print("St does NOT exceed threshold")
}
```

```
## [1] "St does NOT exceed threshold"
```

NOTE: I tested out the CUSUM approach with different values of T and C. Even with small values of T and C (meaning a more sensitive model) St did not exceed threshold.

All things considered, there is not enough information to conclude that the unofficial end of summer has gotten later over the 20 years.