Homework 4 by Haritha Pulletikurti

2020-09-16

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 α (the first smoothing parameter) to be closer to 0 or 1, and why?

Answer:

The Exponential Smoothing can be used in analyzing the traffic delays in airports. We can analyze the delays in flights over the period of time and smooth out the data to find a better cusum for detecting increase or decrease in the delay for a period of time.

Let is consider certain flights coming from places where the weather is usually bad.

Trend – Usually the flight is delayed due to weather. Due to global warming there may be an increas ing trend in delays. Seasonality and randomness also exist.

Hence alpha is closer to 0.3 – 0.6 Beta is closer to 0.5 and gamma also closer to 0.5 .

Randomness is more when the value of alpha is closer to 0.

Seasonality and Trend both exist. Hence I choose the above values.

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.)

Note: in R, you can use either HoltWinters (simpler to use) or the smooth package's es function (harder to use, but more general). If you use es, the Holt-Winters model uses model="AAM" in the function call (the first and second constants are used "A"dditively, and the third (seasonality) is used "M"ultiplicatively; the documentation doesn't make that clear).

Solution:

Results from Assignment -3 to show peaks in the Cusum results as the data is not smoothed.

We can notice peaks in the below table of decreased Cusum which finds the end of summer for each year.

	Year	End of Summer	temp	Cusum S(t)	C	Threshold	
A	X1996	30-Sep	80	25.1463414634146	5	24	
B	X1997	27-Sep	79	30.0243902439024	5	24	
C	X1998	10-Oct	82	27.3008130081301	5	24	
D	X1999	1-Oct	86	26.2926829268293	5	24	peaks
E	X2000	7-Sep	91	26.0650406504065	5	24	peaks
F	X2001	27-Sep	71	24.2113821138211	5	24	
G	X2002	29-Sep	71	27.5121951219512	5	24	
H	X2003	2-Oct	84	32.3983739837399	5	24	
I	X2004	13-Oct	78	35.1138211382114	5	24	
J	X2005	9-Oct	71	26.4308943089431	5	24	
K	X2006	13-Oct	77	32.390243902439	5	24	
L	X2007	13-Oct	62	28.1951219512195	5	24	
M	X2008	19-Oct	76	31.5365853658537	5	24	
N	X2009	6-Oct	82	27.9349593495934	5	24	
O	X2010	1-Oct	76	26.2682926829269	5	24	
P	X2011	8-Sep	93	26.3821138211382	5	24	peaks
Q	X2012	3-Oct	75	25.6016260162602	5	24	
R	X2013	17-Aug	87	24	5	24	peak
S	X2014	5-Oct	84	28.260162601626	5	24	
T	X2015	27-Sep	78	31.1056910569105	5	24	

Exponential Smoothing is used for smoothing out any jumps (peaks and valleys) in the time series data. It exponentially weights all the past observations and the most recent observations are given higher weights and assigns exponentially decreasing order of weights for the past observations. Exponential Smoothing considers Trends, Cyclic Pattens and Seasonality in the data to determine the baseline estimate and to forecast the baseline for a future time t+1.

S(t) = A(x(t)) + (1-A)(S(t-1) + T(t-1))   Here A is alpha , T is trend parameter.

T(t) = B (S(t) -S(t-1)) +(1-B)T(t-1)   Here B is beta , S is the expected baseline at time t.

For Trend and Seasonality: St = Ax(t)/C(t-l) + (1-A)(S(t-1) +T(t-1)

If Cyclic factor is also added to the baseline formula: Ct = Y(x(t)/S(t) + (1-Y) C(t-1) Here Y is gamma

Step 1: Install the necessary libraries and get the data

```
options(warn=-1)
rm(list = ls())
library(IRkernel)
library(forecast)

library(ggplot2)
library(reshape)

data <- read.table("temps.txt",stringsAsFactors = FALSE, header=TRUE)
head(data[1:4,])

##     DAY X1996 X1997 X1998 X1999 X2000 X2001 X2002 X2003 X2004 X2005 X2006 X2007
## 1 1-Jul    98    86    91    84    89    84    90    73    82    91    93    95
## 2 2-Jul    97    90    88    82    91    87    90    81    81    89    93    85
## 3 3-Jul    97    93    91    87    93    87    87    87    86    86    93    82
## 4 4-Jul    90    91    91    88    95    84    89    86    88    86    91    86
##   X2008 X2009 X2010 X2011 X2012 X2013 X2014 X2015
## 1    85    95    87    92   105    82    90    85
## 2    87    90    84    94    93    85    93    87
## 3    91    89    83    95    99    76    87    79
## 4    90    91    85    92    98    77    84    85
```

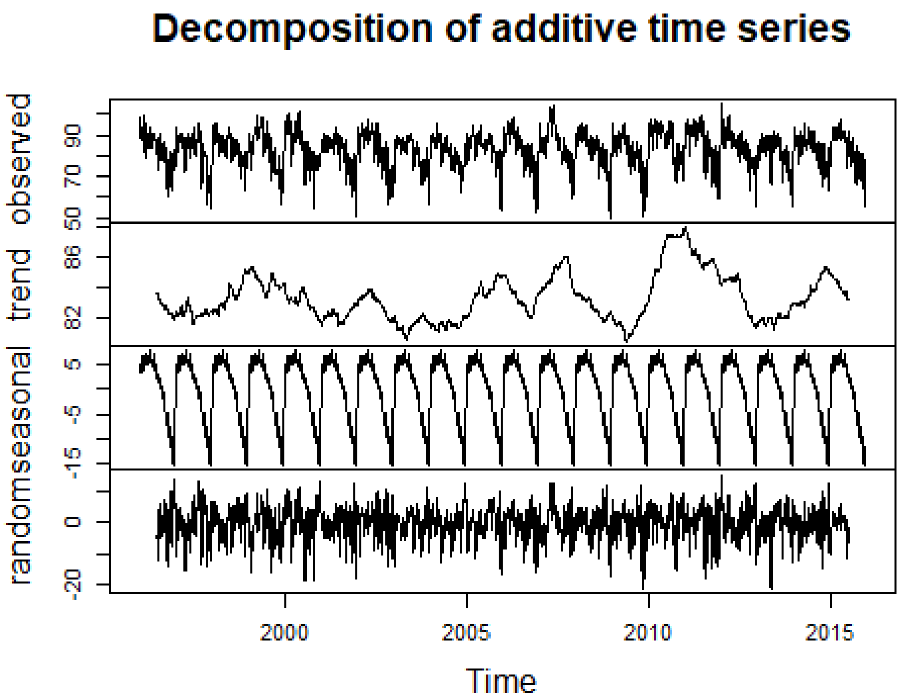
Step 2: Convert the data into time series

```
data_vec <- as.vector(unlist(data[,2:21]))
data_ts <- ts(data_vec, start = 1996, frequency = 123)
```

Step 3: Decompose the timeseries data to find the presence of trend and seasonality.

```
plot(decompose(data_ts))
```

Based on the decompose graph, we notice there is trend, seasonality, and randomness in the timeseries data. Hence HoltsWinters Algorithm suits best to smooth this timeseries data.



Step 4: Comparing Additive and Multiplcative methods of Holtswinter Algorithm

There are two ways to perform HoltsWinters Algorithm

One is Additive method and the other is Multiplicative method.
Let is try both and pick the best among them.

- **Additive:** The Additive model is more useful when the magnitude of the seasonal variations around the trend-cycle do not vary with the level of time series.
- **Multiplicative:** This Model is more useful when the seasonal pattern variation around the trend-cycle is proportional to the level of the timeseries.

Let us run both the models and analyze the results

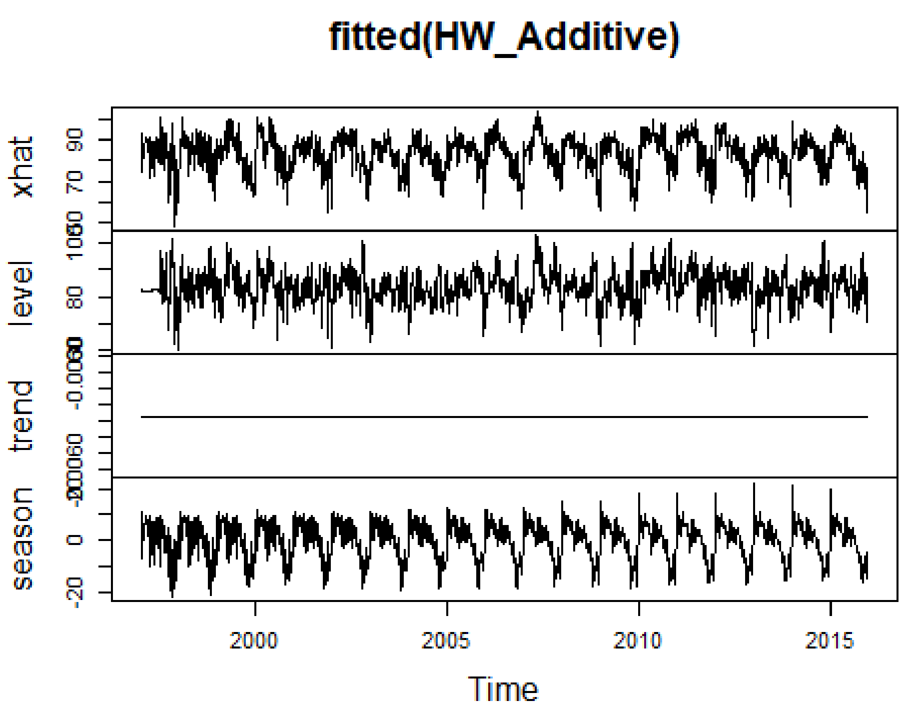
```
set.seed(1)
#Holtwinter: using additive model
HW_Additive<- HoltWinters(data_ts,seasonal="additive")
summary(HW_Additive)
```

```
##              Length Class  Mode
## fitted       9348   mts    numeric
## x            2460   ts     numeric
## alpha           1   -none- numeric
## beta            1   -none- numeric
## gamma           1   -none- numeric
## coefficients  125   -none- numeric
## seasonal        1   -none- character
## SSE             1   -none- numeric
## call            3   -none- call
```

```
cat("HoltsWinter Additive method Results:\n\tBaseline factor alpha:", HW_Additive$alpha,"\n\tTrend factor beta:",HW_Additive$beta,"\n\tSeasonal factor gamma:",HW_Additive$gamma,"\n\tSum of Squared Errors:", HW_Additive$SSE,"\n")
```

```
## HoltsWinter Additive method Results:
## Baseline factor alpha: 0.6610618
## Trend factor beta: 0
## Seasonal factor gamma: 0.6248076
## Sum of Squared Errors: 66244.25
```

```
par(mfrow=c(1,2))
plot(fitted(HW_Additive))
```




```
HW_Multiplicative <- HoltWinters(data_ts,alpha=NULL,beta=NULL,gamma=NULL,seasonal = "multiplicative")
cat("HoltsWinter Multiplicative method Results:\n\tBaseline factor alpha:", HW_Multiplicative$alpha,"\n\tTrend factor beta:",HW_Multiplicative$beta,"\n\tSeasonal factor gamma:",HW_Multiplicative$gamma,"\n\tSum of Squared Errors:", HW_Multiplicative$SSE,"\n")
```

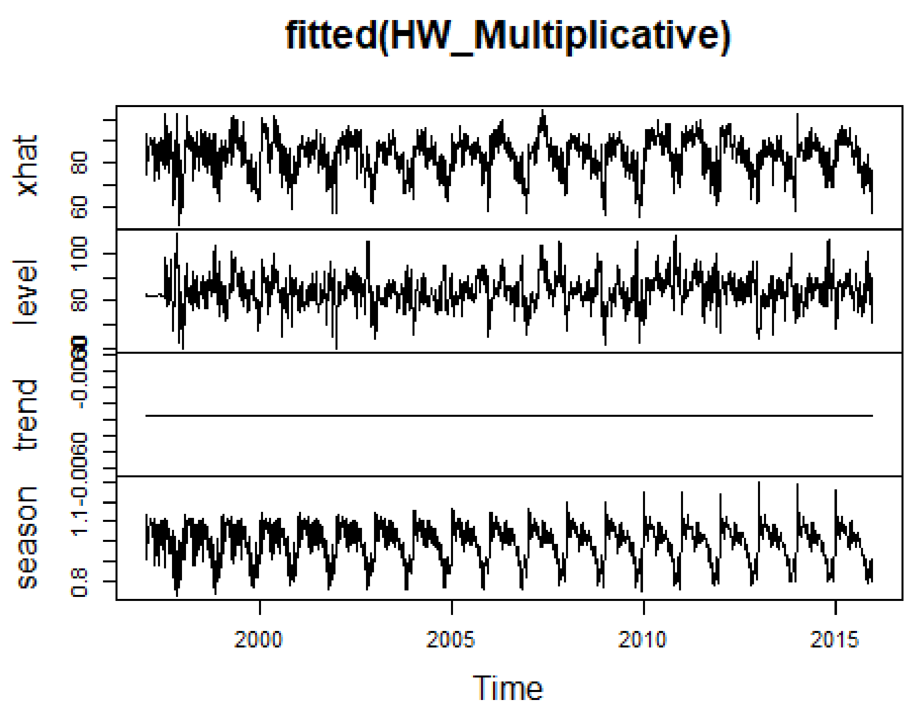
```
## HoltsWinter Multiplicative method Results:
##  Baseline factor alpha: 0.615003
##  Trend factor beta: 0
```

```
##  Seasonal factor gamma: 0.5495256
##  Sum of Squared Errors: 68904.57

summary(HW_Multiplicative)

##              Length Class  Mode
## fitted       9348   mts    numeric
## x            2460   ts     numeric
## alpha           1   -none- numeric
## beta            1   -none- numeric
## gamma           1   -none- numeric
## coefficients  125   -none- numeric
## seasonal        1   -none- character
## SSE             1   -none- numeric
## call            6   -none- call

par(mfrow=c(1,2))
plot(fitted(HW_Multiplicative))
```



Based on the Summary reports of both the models, the mean squared error of the additive method (66244.25) is less than the multiplicative method (68905.7).

So Additive method will give the least error.

Choosing the Method:

But as multiplicative method is taught in depth in the class I will stick with that method and continue the process.

```
HW_Multiplicative$fitted

## Time Series:
## Start = c(1997, 1)
## End = c(2015, 123)
## Frequency = 123
##                xhat     level        trend    season
## 1997.000  87.23653  82.87739 -0.004362918 1.0526529
## 1997.008  90.42182  82.15059 -0.004362918 1.1007422
## 1997.016  92.99734  81.91055 -0.004362918 1.1354128
## 1997.024  90.94030  81.90763 -0.004362918 1.1103378
## 1997.033  83.99917  81.93634 -0.004362918 1.0252306
## 1997.041  84.04496  81.93247 -0.004362918 1.0258379
## 1997.049  75.06333  81.90115 -0.004362918 0.9165601
## 1997.057  87.04945  81.85429 -0.004362918 1.0635250
## 1997.065  84.02220  81.82134 -0.004362918 1.0269532
…………………….(skipped this data)….
## 2015.927  75.12802  86.85003 -0.004362918 0.8650751
## 2015.935  77.61628  91.02020 -0.004362918 0.8527777
## 2015.943  73.71182  89.85022 -0.004362918 0.8204254
## 2015.951  76.54551  87.81303 -0.004362918 0.8717307
## 2015.959  69.70436  81.07435 -0.004362918 0.8598048
## 2015.967  57.02909  71.26750 -0.004362918 0.8002607
## 2015.976  72.14646  87.37935 -0.004362918 0.8257107
## 2015.984  73.89293  85.77627 -0.004362918 0.8615051
## 2015.992  75.83100  82.99285 -0.004362918 0.9137532
```

When we analyze the above fitted values for the Holtswinter Multiplicative method, the trend component is almost negligible which means we cannot say that the temperatures steadily rise of decrease over the years.

Seasonal component (gamma) and baseline component alpha exists > 0.5. So there is increase and decrease in temperatures but as the data is now smoothed out, the peaks and valleys are smoothed out resulting in better visualization of the cusum result.

The below are the seasonal factors:

```
HW_M_seasonalfactor <-matrix(HW_Multiplicative$fitted[,4],nrow=123)
head(HW_M_seasonalfactor)

##          [,1]     [,2]     [,3]     [,4]     [,5]     [,6]     [,7]     [,8]
## [1,] 1.052653 1.049468 1.120607 1.103336 1.118390 1.108172 1.140906 1.140574
## [2,] 1.100742 1.099653 1.108025 1.098323 1.110184 1.116213 1.126827 1.154074
## [3,] 1.135413 1.135420 1.139096 1.142831 1.143201 1.138495 1.129678 1.156092
## [4,] 1.110338 1.110492 1.117079 1.125774 1.134539 1.126117 1.130758 1.137722
## [5,] 1.025231 1.025233 1.044684 1.067291 1.084725 1.097239 1.115055 1.103877
## [6,] 1.025838 1.025722 1.028169 1.042340 1.053954 1.067494 1.080203 1.094312
##          [,9]    [,10]    [,11]    [,12]    [,13]    [,14]    [,15]    [,16]
## [1,] 1.125438 1.122063 1.161415 1.198102 1.198910 1.243012 1.243781 1.238435
## [2,] 1.142187 1.131889 1.144549 1.134661 1.153433 1.165431 1.172935 1.190735
## [3,] 1.165657 1.147982 1.149459 1.135756 1.153310 1.155197 1.157286 1.169773
## [4,] 1.150639 1.146992 1.142497 1.150162 1.151169 1.157751 1.163844 1.159343
## [5,] 1.120818 1.133733 1.132167 1.142714 1.139244 1.112909 1.132435 1.132045
## [6,] 1.102680 1.092178 1.075766 1.088547 1.082185 1.103092 1.115071 1.118575
##         [,17]    [,18]    [,19]
## [1,] 1.300204 1.290647 1.254521
```

```
## [2,] 1.191956 1.219190 1.228826
## [3,] 1.189915 1.172309 1.169045
## [4,] 1.166605 1.167993 1.158956
## [5,] 1.145230 1.168161 1.170449
## [6,] 1.121598 1.134962 1.145475
```

Below are the smoothed xhat values

```
HW_xhat <- matrix(HW_Multiplicative$fitted[,1],nrow=123)
rownames(HW_xhat) <-data[,1]
columns = colnames(data[,-2])
colnames(HW_xhat) <-columns[-1]

head(HW_xhat)

##          X1997    X1998    X1999    X2000    X2001    X2002    X2003    X2004
## 1-Jul 87.23653 65.04516 90.29613 83.39938 87.68863 78.07509 73.10059 87.27074
## 2-Jul 90.42182 84.87634 85.44878 86.44444 84.78855 86.02384 72.13247 85.01878
## 3-Jul 92.99734 89.61560 85.65942 92.85774 88.70570 90.23022 77.77739 82.68648

##          X2005    X2006    X2007    X2008    X2009    X2010    X2011    X2012
## 1-Jul 92.29714 78.50826 81.58696 84.72917 79.51855 86.74604 93.88371 82.30605
## 2-Jul 92.85614 88.18138 88.52648 80.39548 85.65722 81.47324 87.43846 92.55001
## 3-Jul 92.33884 92.43570 86.72311 84.53380 88.31357 82.29310 90.24836 91.18746

##          X2013     X2014    X2015
## 1-Jul 84.88750 102.54643 90.07756
## 2-Jul 76.18707  89.57468 85.16854
## 3-Jul 81.46207  88.15080 82.09161
```

Now let us perform CUSUM on the smoothed out Xhat values from the

multiplicative Holtswinter result set.

```
Initialize the variables

S= c() # St of cusum
std=c() # standard deviation
DetectedDecreaseIndex = c()# detected decreased index from the data
DetectedDecrease = c() # the Temperature that is detected as decreased from cusum
S[0] = 0 // St of each year
total = 0 //total SD
```

We need to find the Threshold T value. I am taking 3 times the standard deviation of the smoothed out data set.

```
for(j in 1:ncol(HW_xhat))
{
  std[j]= sd(HW_xhat[,j])
  total = total + std[j]
}
t= (total/ncol(HW_xhat))*3
t

## [1] 24.4838
```

I will use T= 24 and C=5 for finding when the temperature decrease happens for each year 1997-2015

```
t=24 // Theshold T
C=5  // C

for(j in 2:ncol(HW_xhat))
{

  for(i in 1:nrow(HW_xhat))
  {
    S[i] = max(0,S[i-1]+(mean(HW_xhat[,j])-HW_xhat[i,j] - C))
    if(S[i]>t)
    {
      DetectedDecreaseIndex[j-1] = i
      DetectedDecrease[j-1]=S[i]
      break
    }
  }
}

rows=rownames(HW_xhat)
cusum_year = colnames(HW_xhat)
cusum_decrease_date = c()
cusum_c = c()
cusum_t =c()
cusum_st = c()
cusum_dt=c()
temp=c()
for(k in 1:length(DetectedDecreaseIndex))
{

  cusum_decrease_date[k] = HW_xhat[DetectedDecreaseIndex[k],1]
  cusum_st[k]=DetectedDecrease[k]
  cusum_c[k] = C
  cusum_t[k] = t
  cusum_dt[k]=rows[DetectedDecreaseIndex[k]]
  temp[k]=HW_xhat[DetectedDecreaseIndex[k],k]
}


matrix.c = cbind(cusum_year,cusum_dt,temp,cusum_st,cusum_c,cusum_t)
colnames(matrix.c) = c("Year","Date","Ending Temperature", "Cusum S(t)", "C","Threshold")
matrix.c = as.table(matrix.c)
matrix.c
```

Below is the result from the CUSUM approach.

```
##   Year  Date   Ending Temperature Cusum S(t)       C Threshold  Diff
## A X1997 11-Oct 78.9190894105648   24.8455839791495 5 24
## B X1998 30-Sep 69.291047578795    26.6867353045359 5 24      11 days early
## C X1999 10-Sep 79.1866844600578   25.5316914278139 5 24      20 days early
## D X2000 29-Sep 69.5893423257103   24.4603950874326 5 24      19 days late
## E X2001 30-Sep 67.1563783984011   32.3680141545398 5 24      1 day late
## F X2002 2-Oct  79.3597049644767   30.1621407333443 5 24      2 days late
## G X2003 14-Oct 75.0176156665381   27.0386820586508 5 24      12 days late
## H X2004 9-Oct  72.4274638086104   27.2652810752102 5 24      5 day early
## I X2005 15-Oct 75.9827836778864   32.9201637926231 5 24      1 day early
```

```
## J X2006 14-Oct 66.6294848951239   27.7403908386628 5 24     7 day late
## K X2007 21-Oct 74.8352760309822   32.397072249358  5 24     7 day late
## L X2008 7-Oct  76.4756047418944   33.8227128042536 5 24     14 day early
## M X2009 2-Oct  74.6014312642295   24.5363976996178 5 24     5 day early
## N X2010 3-Oct  77.8689297870613   26.1190004599661 5 24     1 day late
## O X2011 8-Oct  74.1899206832538   32.5988594083823 5 24     5 days late
## P X2012 21-Oct 76.5073212181581   24.7777685778158 5 24     13 days late
## Q X2013 30-Sep 72.1794608892659   26.7466238546176 5 24     11 days early
## R X2014 28-Sep 75.3604036983112   28.4196926845551 5 24     2 days early
## S X2015 11-Oct 78.9190894105648   24.8455839791495 5 24     13 days late
```

Let us now analyze the above result.

Conclusion:

I started with presenting the Cusum result of unsmoothed data to indicate peaks and valleys.

For Smoothing the data, HoltsWinter – Additive and Multiplicative methods are analyzed. Based on the analysis Additive method has less error than the multiplicative method.

As Multiplicative method is being focusedin the class, I have chosen to use the Holtswinter Multipli cative method.

Then using this method,I smoothed the data which eliminated the peaks shown in the unsmoothed data.

Then on the smoothed xhat data, I performed the CUSUM technique to detect when the summer end s for each year. (The smoothed data was from 1997-2015)

Based on the above data, the last column "diff" indicates the difference from the previous year, whe ther the summer ended early or later than its previous year. (only diff column is computed manual ly the rest is the output from the R), We cannot find any trend in the result as some years the summer ends early while some years the summer is ending late.

The Lowest temperature that reached for this Cusum result is 66.62 on 14th October on 2006. Ther e is seasonality and randomness in the obtained result. So we cannot say that the unofficial end of s ummer has reached later over the 20 years. Usually the summer may end in between the last week of summer and 2nd week of October based on the results.