Homework Week 3

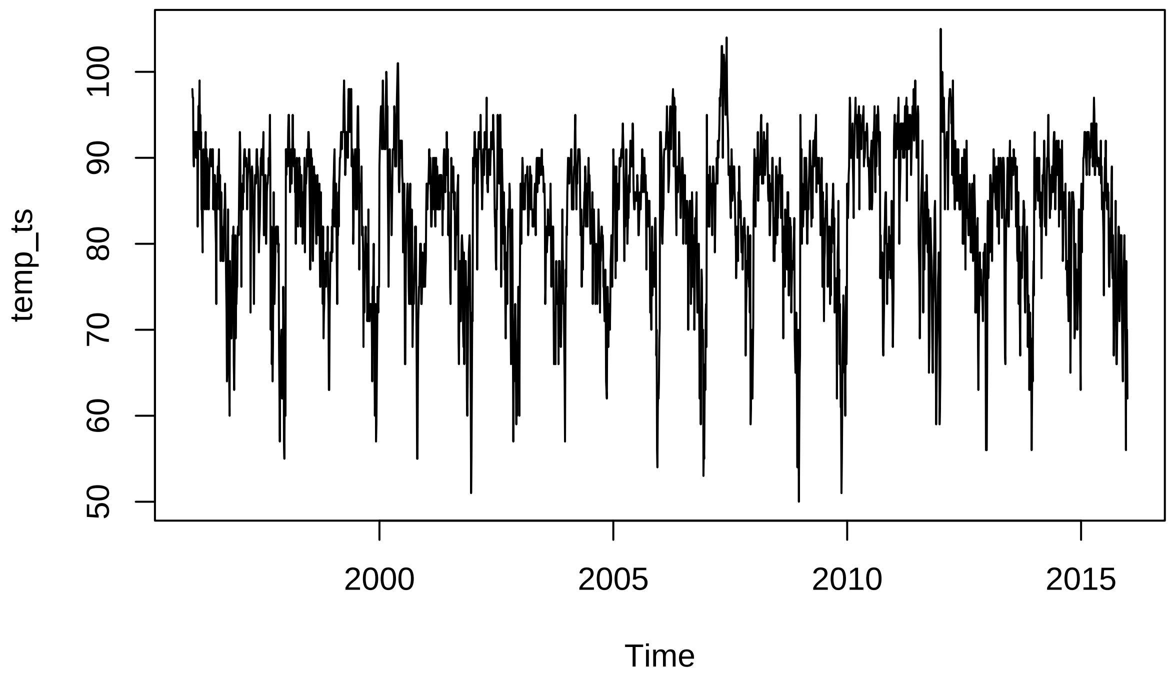
Question 7.1
Answer:

The exponential smoothing model could be used to predict stock prices in near future. To run this model, we need the data of historic daily closing prices for the stocks we want to predict.
Since there's a lot of randomness in stock price fluctuation (some scholars even believe that stock prices follow a "Random Walk" pattern), we would expect the alpha value closer to 0.

Question 7.2
Answer:
First, read the data and convert the temperature data (column 2 to 21) into a vector. Then, we will use ts() function to convert the vector into a time series object which will be used as our time series data. The time series data will be displayed in a plot.

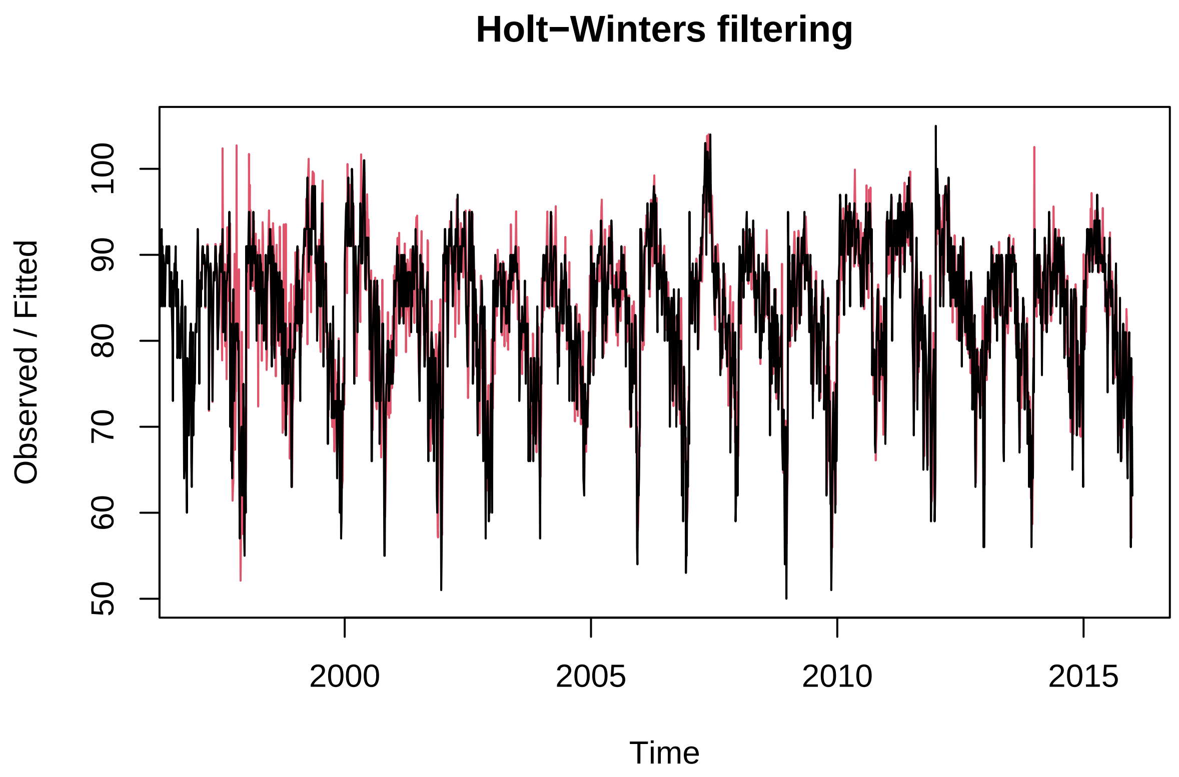
```r
#read file
rm(list = ls())
set.seed(19)
data <- read.table("C:\\Data\\week 3 data-summer\\data 7.2\\temps.txt",
                   stringsAsFactors = FALSE, header = TRUE)
#Convert the temperature data into a time series object
temp_vec <- as.vector(unlist(data[2:21]))
temp_ts <- ts(temp_vec, start = 1996, frequency = 123)

#Display the teime series
plot.ts(temp_ts)
```



Then we are going to run ***HoltWinters()*** function on the time series we just created. We will define alpha, beta and gamma to be null so that the model will estimate the best parameters for us. We will also set "seasonal" to be multiplicative.
The comparison of our models estimations and the real temperature data will be displayed in a plot. The black line represents the actual temperature data while the red line represents the estimation of our model.

```
temp_es <- HoltWinters(temp_ts, alpha = NULL, beta = NULL, gamma = NULL,
                       seasonal = "multiplicative")
plot(temp_es)
```



```
summary(temp_es)
```

```
##              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
```

```
head(temp_es$fitted)
```

```
##          xhat    level        trend   season
## [1,] 87.23653 82.87739 -0.004362918 1.052653
## [2,] 90.42182 82.15059 -0.004362918 1.100742
## [3,] 92.99734 81.91055 -0.004362918 1.135413
## [4,] 90.94030 81.90763 -0.004362918 1.110338
## [5,] 83.99917 81.93634 -0.004362918 1.025231
## [6,] 84.04496 81.93247 -0.004362918 1.025838
```

To determine whether the unofficial end of summer has gotten later over the 20 years, we are going to take the following approach:

First, we are going to load the seasonal factors returned by ***HoltWinters()*** function into a data frame with similar structure as the raw data (dates in rows, and years in columns). Let's display the trends of each year's seasonal factor in line chart:

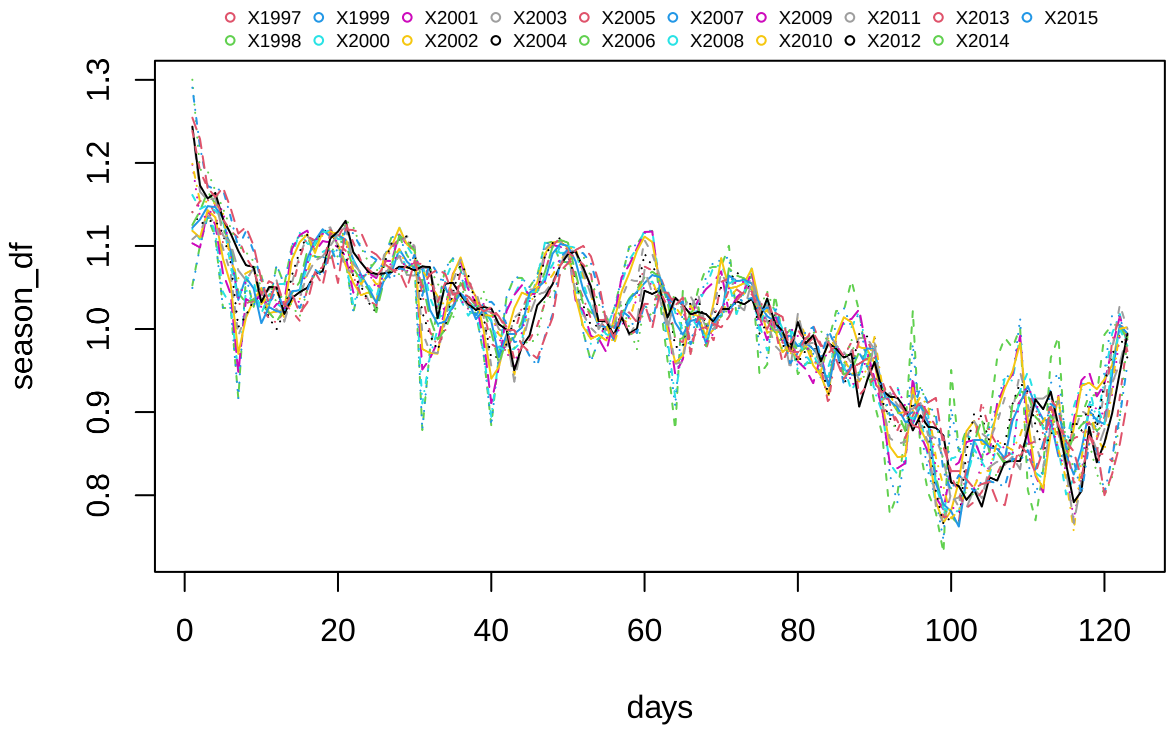
```r
#Convert seasonal factors into matrix
season_df <- as.data.frame(matrix(temp_es$fitted[,4], nrow = 123))
season_df <- cbind(data[,1],season_df)
colnames(season_df)[1] <- "Dates"
colnames(season_df)[2:ncol(season_df)] = colnames(data[3:21])

matplot(season_df, type = "l",pch=1, col = 2:20, xlab = "days")
```

```
## Warning in xy.coords(x, y, xlabel, ylabel, log = log, recycle = TRUE): NAs
## introduced by coercion
```

```
## Warning in xy.coords(x, y, xlabel, ylabel, log): NAs introduced by coercion
```

```r
legend("top", legend = colnames(season_df[,-1]), col=2:20, xpd = TRUE, pch=1,
       inset=c(0,-0.13), cex =0.6, bty = "n",  ncol=10)
```



Then we are going to run a CUSUM model to find the date when the seasonal factor significantly decreases for each year. The dates we find will be the unofficial end of summer. Then we are going to observe if the date is getting later across the time periods.
Since we are going to detect significant decrease, consider following equation for CUSUM model:

$$S_t = \max\{0, S_{t-1} + (\mu - x_t - C)\}$$

$$\text{Is } S_t \geq T\text{ ?}$$

To make sure that **μ** represents the average seasonal factors in summer. We will define **μ** as each year's average seasonal factor in July only.
Since seasonal factor is a coefficient returned by exponential smoothing model instead of a real temperature. It's not easy to define the value of **C** and **T** using common sense. Therefore we are going to use the standard deviation of each years' seasonal factors (July only) as a benchmark.
For each year, we define **C** equals the 2 times that year's standard deviation seasonal factor (July only), and **T** equals 6 times that year's standard deviation of seasonal factor (July only).

Note:
I tried several difference set of numbers to determine how many times of standard deviation C and T should be. If the numbers of times are too big or too small, the unofficial summer end dates returned by CUSUM will be too early or too late. I choose 2 and 6 in the end since this combination returns my some "reasonable" dates of summer end.

```r
#initialize cusum_df with the values of the seasonal factor data set
cusum_df <- season_df
year_vec <- c()
summer_end_vec <- c()

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

  #Define mu, C and T
  mu <- mean(unlist(season_df[1:31,j]))
  C <- 2*sd(unlist(season_df[1:31,j]))
  T <- 6 * sd(unlist(season_df[1:31,j]))

  #initialize the first row
  cusum_df[1,j] <- max(0, mu - cusum_df[1,j] - C)

  #Calculate St for each date
  for (i in 2:nrow(season_df)){
    cusum_df[i,j] <- max(0, cusum_df[i-1,j] + mu - cusum_df[i,j] - C)
  }
}

# find summer end date for each year
for (j in 2:ncol(season_df)){
  year <- colnames(season_df)[j]
  for (i in 2:nrow(season_df)){
    if (cusum_df[i,j] >= T){
      year_vec <- append(year_vec, year)
      summer_end_vec <- append(summer_end_vec, season_df[i,1])
      break
    }
  }
}

summer_end_df <- data.frame("Year" = year_vec,
                            "Summer End Date" = summer_end_vec)
#show the unofficial summer end date for each year
```

```r
print(summer_end_df)
```

```
##      Year Summer.End.Date
## 1  X1997           1-Oct
## 2  X1998           1-Oct
## 3  X1999           1-Oct
## 4  X2000           1-Oct
## 5  X2001           1-Oct
## 6  X2002          30-Sep
## 7  X2003          30-Sep
## 8  X2004          29-Sep
## 9  X2005          29-Sep
## 10 X2006          26-Sep
## 11 X2007          26-Sep
## 12 X2008          27-Sep
## 13 X2009          28-Sep
## 14 X2010          29-Sep
## 15 X2011          29-Sep
## 16 X2012          30-Sep
## 17 X2013           1-Oct
## 18 X2014           2-Oct
## 19 X2015           1-Oct
```

From the table above, we can see that the unofficial summer end date didn't significant change from 1997 to 2015. Therefore, the unofficial end of summer hasn't gotten later over the timer period.

Question 8.1
Answer:

I am a basketball fan and played campus basketball when I was at school. An important factor that middle school basketball coaches would consider when they select their players is how tall the young players would be in the near future. Let's assume we are selecting some young basketball players who are 12 years old, and we want to know how tall they would be when they are 15. In this scenario, we can use linear regression model to make the prediction.
The predictors may include: the student's height at the age of 12, the heights of the student's parents, student's gender, number of hours the student sleeps everyday (assuming more sleep helps teenagers grow taller), etc.

Question 8.2
Answer:

First, read the data and display the first few rows.

```r
#read file
rm(list = ls())
set.seed(19)
data <- read.table("C:\\Data\\week 3 data-summer\\data 8.2\\uscrime.txt",
                   stringsAsFactors = FALSE, header = TRUE)

head(data)
```

```
##       M So   Ed  Po1  Po2    LF   M.F Pop   NW    U1   U2 Wealth Ineq     Prob
## 1 15.1  1  9.1  5.8  5.6 0.510  95.0  33 30.1 0.108 4.1   3940 26.1 0.084602
## 2 14.3  0 11.3 10.3  9.5 0.583 101.2  13 10.2 0.096 3.6   5570 19.4 0.029599
## 3 14.2  1  8.9  4.5  4.4 0.533  96.9  18 21.9 0.094 3.3   3180 25.0 0.083401
## 4 13.6  0 12.1 14.9 14.1 0.577  99.4 157  8.0 0.102 3.9   6730 16.7 0.015801
## 5 14.1  0 12.1 10.9 10.1 0.591  98.5  18  3.0 0.091 2.0   5780 17.4 0.041399
## 6 12.1  0 11.0 11.8 11.5 0.547  96.4  25  4.4 0.084 2.9   6890 12.6 0.034201
##      Time Crime
## 1 26.2011   791
## 2 25.2999  1635
## 3 24.3006   578
## 4 29.9012  1969
## 5 21.2998  1234
## 6 20.9995   682
```

Next, we are going to run a linear regression model using “Crime” as results and all the other 15 variables as predictors.

```
model <- lm(Crime~M+So+Ed+Po1+Po2+LF+M.F+Pop+NW+U1+U2+Wealth+Ineq+Prob+Time,
            data = data)
summary(model)
```

```
##
## Call:
## lm(formula = Crime ~ M + So + Ed + Po1 + Po2 + LF + M.F + Pop +
##     NW + U1 + U2 + Wealth + Ineq + Prob + Time, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -395.74  -98.09   -6.69  112.99  512.67
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -5.984e+03  1.628e+03  -3.675 0.000893 ***
## M            8.783e+01  4.171e+01   2.106 0.043443 *
## So          -3.803e+00  1.488e+02  -0.026 0.979765
## Ed           1.883e+02  6.209e+01   3.033 0.004861 **
## Po1          1.928e+02  1.061e+02   1.817 0.078892 .
## Po2         -1.094e+02  1.175e+02  -0.931 0.358830
## LF          -6.638e+02  1.470e+03  -0.452 0.654654
## M.F          1.741e+01  2.035e+01   0.855 0.398995
## Pop         -7.330e-01  1.290e+00  -0.568 0.573845
## NW           4.204e+00  6.481e+00   0.649 0.521279
## U1          -5.827e+03  4.210e+03  -1.384 0.176238
## U2           1.678e+02  8.234e+01   2.038 0.050161 .
## Wealth       9.617e-02  1.037e-01   0.928 0.360754
## Ineq         7.067e+01  2.272e+01   3.111 0.003983 **
## Prob        -4.855e+03  2.272e+03  -2.137 0.040627 *
## Time        -3.479e+00  7.165e+00  -0.486 0.630708
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 209.1 on 31 degrees of freedom
## Multiple R-squared:  0.8031, Adjusted R-squared:  0.7078
## F-statistic: 8.429 on 15 and 31 DF,  p-value: 3.539e-07
```

As we can see from the results above, there are a few coefficient that have p values larger the threshold of 0.1. This means that these coefficient are not significant and may cause overfitting issue. We may need to exclude them from the model. Here's the model after these coefficients are excluded:

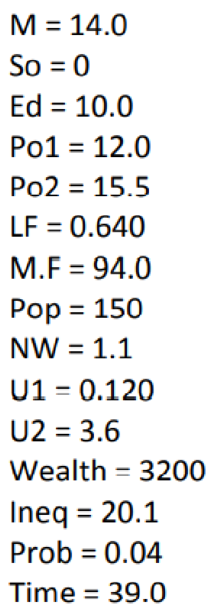
```
model2 <- lm(Crime~M+Ed+Po1+U2+Ineq+Prob,
             data = data)
summary(model2)
```

```
##
## Call:
## lm(formula = Crime ~ M + Ed + Po1 + U2 + Ineq + Prob, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -470.68  -78.41  -19.68  133.12  556.23
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -5040.50     899.84  -5.602 1.72e-06 ***
## M             105.02      33.30   3.154  0.00305 **
## Ed            196.47      44.75   4.390 8.07e-05 ***
## Po1           115.02      13.75   8.363 2.56e-10 ***
## U2             89.37      40.91   2.185  0.03483 *
## Ineq           67.65      13.94   4.855 1.88e-05 ***
## Prob        -3801.84    1528.10  -2.488  0.01711 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 200.7 on 40 degrees of freedom
## Multiple R-squared:  0.7659, Adjusted R-squared:  0.7307
## F-statistic: 21.81 on 6 and 40 DF,  p-value: 3.418e-11
```

In the second model, we excluded all the insignificant coefficient and kept only six predictors. We can find from their p values that all these predictors have significant coefficients. Therefore we are going to use this model to make predictions.

We are going to use the adjusted R squared value of this model as an estimate of the quality of fit, which equals 0.7307.

If we plug in the predictors provided in the questions:



M = 14.0
So = 0
Ed = 10.0
Po1 = 12.0
Po2 = 15.5
LF = 0.640
M.F = 94.0
Pop = 150
NW = 1.1
U1 = 0.120
U2 = 3.6
Wealth = 3200
Ineq = 20.1
Prob = 0.04
Time = 39.0

We can make following prediction:
Crime = -5040.5 + 105.02 * 14 + 196.47 * 10 + 115.02 * 12 + 89.37 * 3.6 + 67.65 * 20.1 - 3801.84 * 0.04
= 1304.245

```
input <- data.frame("M" = 14.0,
                    "Ed" = 10.0,
                    "Po1" = 12.0,
                    "U2" = 3.6,
                    "Ineq" = 20.1,
                    "Prob" = 0.04)
prediction <- predict(model2, newdata = input)
prediction
```

```
##        1
## 1304.245
```