ISYE6501 Homework week 3

Dylan Peters

Question 1

A situation from real life where an exponential smoothing model would be appropriate:

The best example of this I can think of is tracking an individual's weight. To track my own weight, I would log the daily weight and apply an ES model to it over several months. It would help smooth out any upward or downward blips and determine whether a higher weight means a dangerous trend or is just part of daily random variation. There could likely be a seasonal component over holidays as weight spikes and then settles back down again.

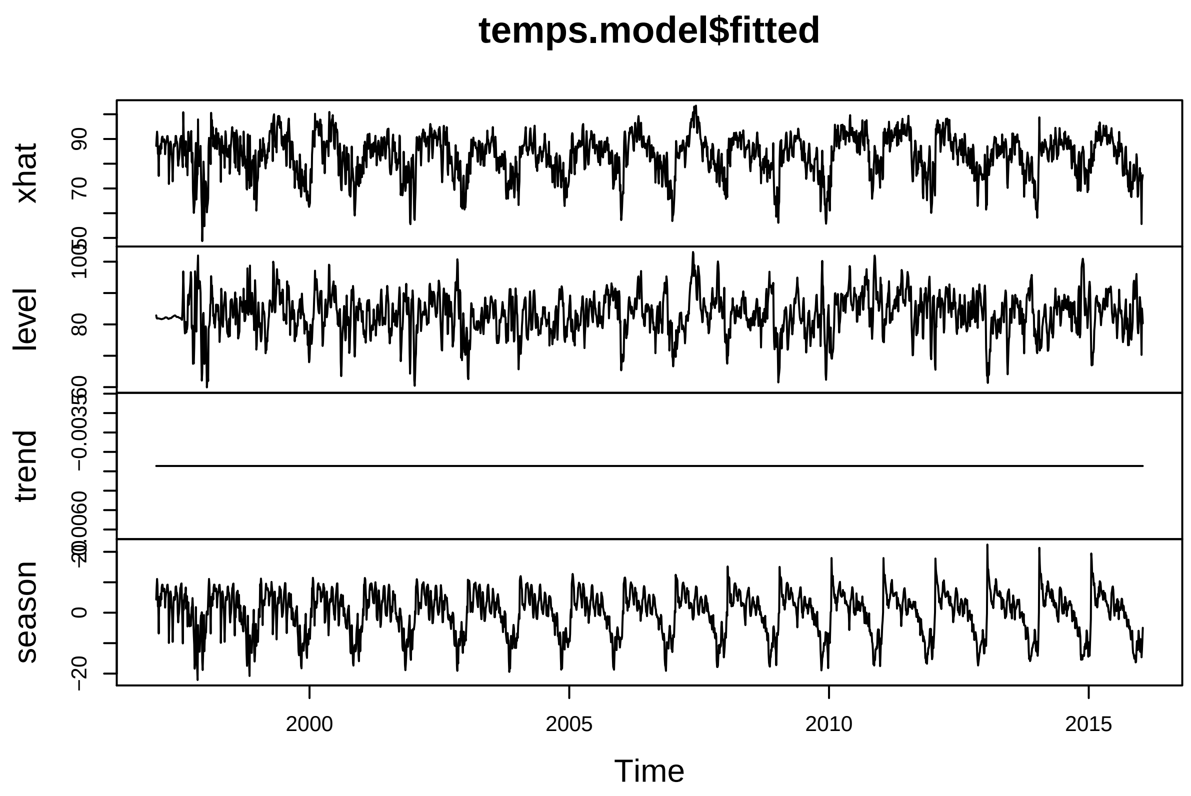
Question 2

1. Apply an exponentail smoothing model to Atlanta temperature data to determine whether the unofficial end of summer has gotten later over the 20 years:

```r
# Load the data and convert the dataframe to a single vector
temps <- read.delim("temps.tsv")
temps.vec <- as.vector(as.matrix(temps[-1]))
```

We have to convert the vector to a time series object and then run a model on it.

```r
temps.ts <- ts(temps.vec, frequency = 123, start=c(1996,7))
temps.model <- HoltWinters(temps.ts)
#plot(temps.model)
plot(temps.model$fitted)
```



The plotted trend line is flat, and the beta coefficient is flat, indicating no trend up or down. However this covers the data as a whole and doesn't look at whether the seasonal variation changes in later years. My plan is to create a model based on the first 15 years, run a forecast for the next 5, and compare the forecast to the actual temperature data.

```
library(forecast)
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
## Loading required package: timeDate
```

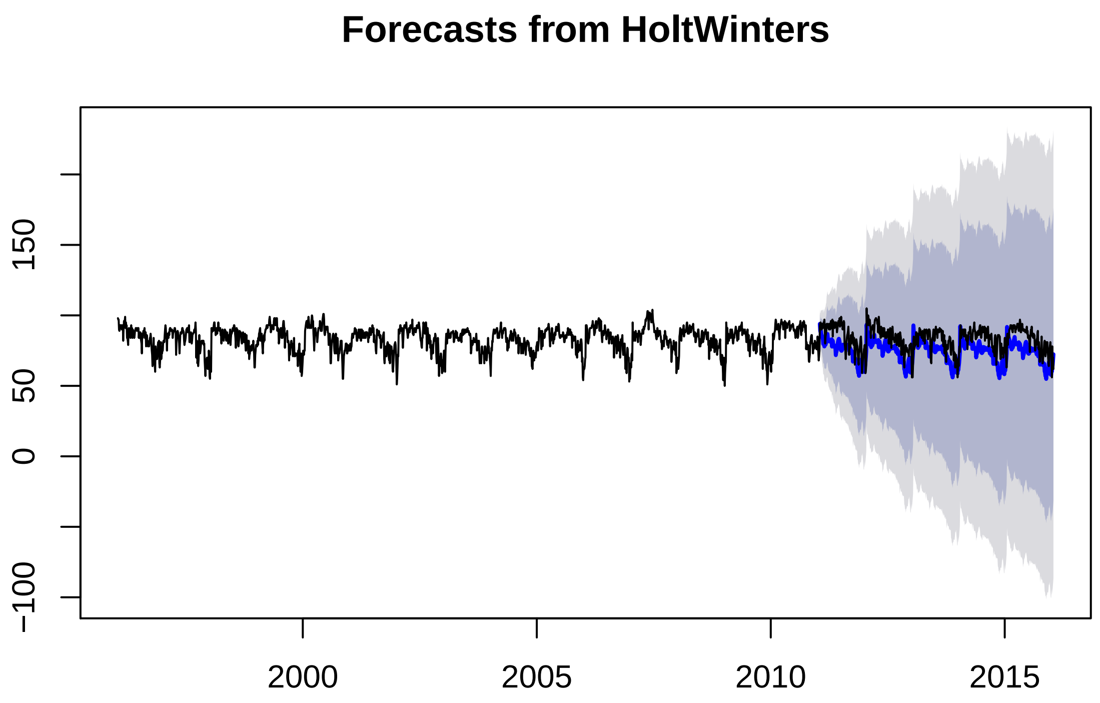
```
## Warning: package 'timeDate' was built under R version 3.3.3
```

```
## This is forecast 7.3
```

```
temps.vec.train <- temps.vec[1:(123*15)] # 123 days/year times 15 years
temps.ts.train <- ts(temps.vec.train, frequency = 123, start=c(1996,7))
temps.model.2 <- HoltWinters(temps.ts.train)
Predicted <- forecast.HoltWinters(temps.model.2, h=(123*5))

PredictedValues <- Predicted$mean
ActualValues <- temps.ts[(123*15+1):(123*20)]
```

```r
# plot the predicted values, then the actual
plot(Predicted, type="n")
lines(temps.ts)
```



Looking at the predicted (blue) versus the actual temps (black), it appears that the summers are warmer and longer than expected. A better approach could be to look at only the seasonality and see if it changes over time. One caveat is that warmer temperatures in the summer might give the false impression of an early end when they return to normal.

```r
require(qcc)
```

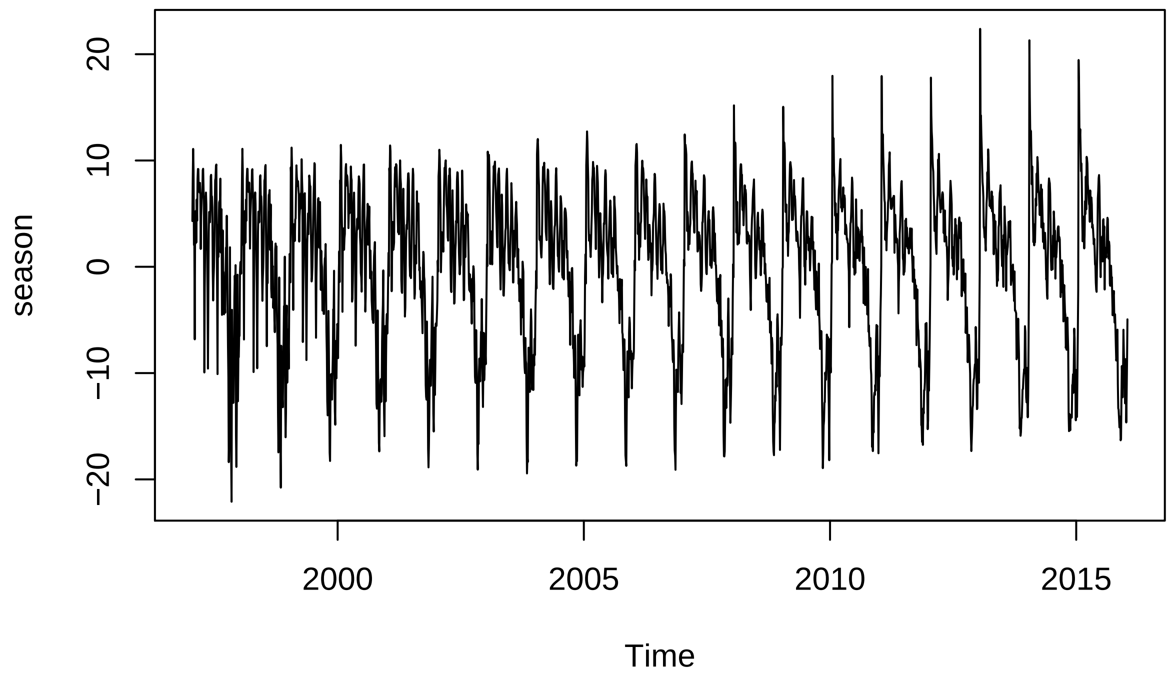
```
## Loading required package: qcc

## Warning: package 'qcc' was built under R version 3.3.3

## Package 'qcc', version 2.6

## Type 'citation("qcc")' for citing this R package in publications.
```

```r
season <- temps.model$fitted[,4]
plot(season)
```

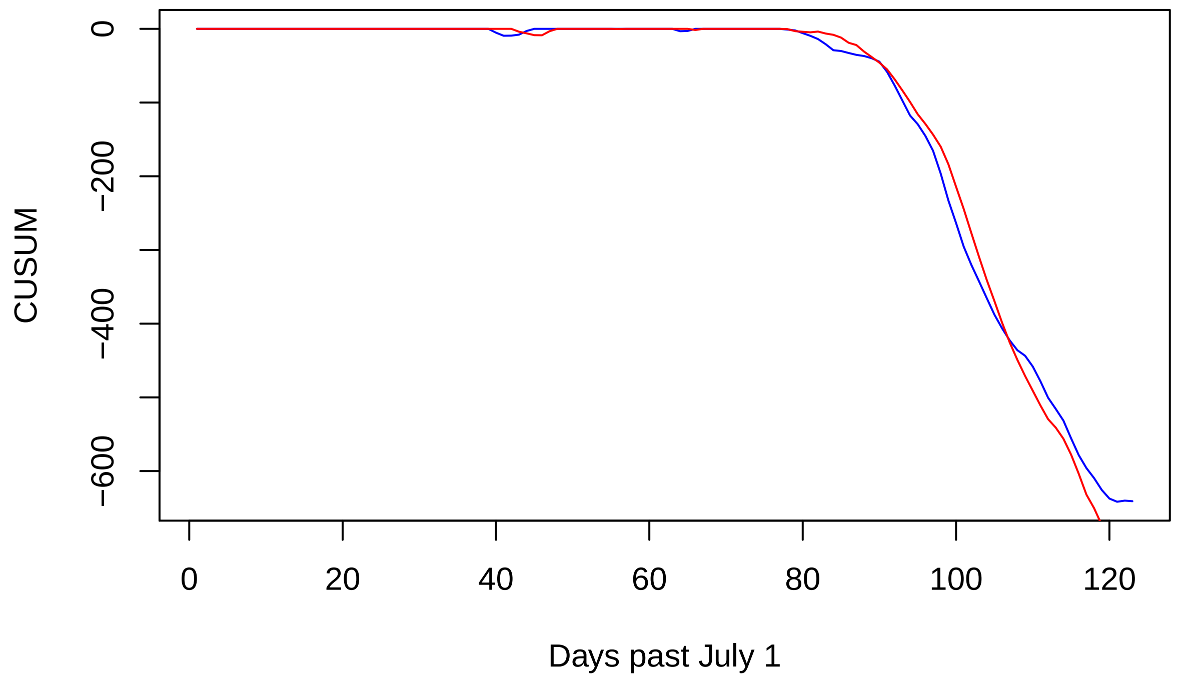


```
# convert season back to matrix form:
season.m <- matrix(season, c(123,20))

cusum.first <- cusum(season.m[,1:14], plot=FALSE)
cusum.last <- cusum(season.m[,15:19], plot=FALSE)

season.pairs <- cbind(dimnames(cusum.first$data)$Group, cusum.first$neg, cusum.last$neg)

# Plot first set of seasons
plot(season.pairs[,1], season.pairs[,2], col="blue", type="l", xlab="Days past July 1", ylab="CUSUM")
# Plot last set of seasons
lines(season.pairs[,1], season.pairs[,3], col="red")
```



The blue line is the base line based on the first 15 years. The red line is the line based on the last five years. The chart shows that the temperatures cool down later from 2011-2015.

Question 3

Describe a problem where a Regression model would be appropriate:

I have an interest in economic modeling of a worker's income based on demographic factors, education, and other factors such as geography. Significant factors would be: level of education, education quality (compared to state or national standards), race, gender, national origin, poverty rate of parents (and grandparents), mobility, disability, etc. I would like to answer questions like: How does being raised in a single parent household affect long term income? Do workers who are more mobile make more than workers who stay in the same town? What are the economic prospects of students who fail to graduate high school, or who graduate without marketable skills?

Question 4

Make a regression model for Crime Data:

Load the data:

```r
uscrime <- read.delim("http://www.statsci.org/data/general/uscrime.txt")
dim(uscrime)
```

```
## [1] 47 16
```

The following step looks at the correlations in the data and removes any variables that are highly correlated (abs > .9).

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.3.3
```

```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```

```
## Warning: package 'ggplot2' was built under R version 3.3.3
```

```
print(findCorrelation(cor(uscrime), names=TRUE))
```

```
## [1] "Po1"
```

```
uscrime <- subset(uscrime, select = -c(Po1))
```

The findCorrelation function shows that the "Po1" variable is likely redundant, so we will remove it.

Since there are only 47 observations in the dataset, I will use Leave-One-Out cross validation, and tune several different versions. Make sure to scale the data first.

```
Train_X <- subset(uscrime, select = -c(Crime))
Train_Y <- uscrime[,"Crime"]
preProc <- preProcess(Train_X, method = c("center","scale"))

Train_X.scaled <- predict(preProc, Train_X)

## Set for leave-one-out CV
traincontrol1 <- trainControl(method="LOOCV", search="grid")
lm.model <- train(Train_X.scaled, Train_Y, method="lm", trControl = traincontrol1)
```

We have one model with all the parameters. To find if there's a better subset, now use the RFE (Recursive Feature Elimination) function in the caret package to filter down to the best features.

```
rfecontrol <- rfeControl(functions=lmFuncs, method="LOOCV")

rfeResult <- rfe(Train_X.scaled, Train_Y, sizes=2:ncol(Train_X.scaled), rfeControl = rfecontrol)
```

Comparing the two models, we see that they have similar performance:

```
print(paste("AIC of first model",AIC(lm.model$finalModel)))
```

```
## [1] "AIC of first model 652.785685177313"
```

```
print(paste("AIC of second model", AIC(rfeResult$fit)))
```

```
## [1] "AIC of second model 650.79162413827"
```

```
print(paste("RMSE of first model", summary(lm.model$finalModel)$sigma))
```

```
## [1] "RMSE of first model 216.452332513202"
```

```
print(paste("RMSE of second model", summary(rfeResult$fit)$sigma))
```

```
## [1] "RMSE of second model 213.160989858815"
```

```
print(paste("Adjusted R Squared of first model", summary(lm.model)$adj.r.squared))
```

```
## [1] "Adjusted R Squared of first model 0.686790172758196"
```

```
print(paste("Adjusted R Squared of second model", summary(rfeResult$fit)$adj.r.squared))
```

```
## [1] "Adjusted R Squared of second model 0.696242999069925"
```

The results are very similar. The Adjusted R Squared tells us that the model explains about 70% of the variation in the data. Not too shabby. We will use the second model for the rest of the answer. The full summary including coeffcients are below:

```
summary(rfeResult$fit)
```

```
##
## Call:
## lm(formula = y ~ ., data = tmp)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -476.26 -134.89    0.13  136.61  428.76
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  905.085     31.093  29.109  < 2e-16 ***
## Ineq         294.668     92.130   3.198 0.003046 **
## Po2          274.151     72.828   3.764 0.000653 ***
## Ed           189.809     69.606   2.727 0.010158 *
## U2           152.888     69.991   2.184 0.036143 *
## M            114.007     52.592   2.168 0.037491 *
## U1          -103.922     76.649  -1.356 0.184362
## Wealth       100.396    101.031   0.994 0.327594
## Prob         -90.834     43.359  -2.095 0.043934 *
## M.F           63.059     60.091   1.049 0.301624
## Pop          -22.231     48.140  -0.462 0.647250
## NW            21.708     64.541   0.336 0.738736
## So             1.103     72.009   0.015 0.987871
## LF            -3.726     59.144  -0.063 0.950154
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 213.2 on 33 degrees of freedom
## Multiple R-squared:  0.7821, Adjusted R-squared:  0.6962
## F-statistic: 9.111 on 13 and 33 DF,  p-value: 1.557e-07
```

We can see that the Time variable has been removed.

Now to calculate based on the provided test set of one observation.

```
feature_names <- names(Train_X.scaled)

test <- c(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)

test.scaled.df <- data.frame(predict(preProc, t(test)))
```

Predict the test data based on the final model:

```
Crime.predict <- predict(rfeResult$fit, test.scaled.df)
Crime.predict
```

```
##        1
## 1131.577
```

The predicted crime rate is 1132.