Homework_3

Richie Phan

5/31/2020

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

At my job the production of our product requires a lot of materials. We make diagnostic tests for food pathogens, so the amount of product we make on a monthly basis can be different based on the holiday season, time of farmer's harvest, or on random food outbreaks.If we could see the potential growth along with the seasonal need of each product, we can make sure we can secure the right amount of materials before we hit peaks in demand.We could use exponential smoothing with a alpha value closer to 0 because we tend to get a lot of randomness in the system due to random outbreaks or other factors like farmers needing to harvest quicker due to weather.

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.)

To start I clear the environment, call the forecast package, then bring in the data.

```
rm(list = ls())
library(forecast)
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
library(stats)
temps_data <- read.table("C:/Users/phan_/Documents/R/temps.txt", stringsAsFactors = F, header = T)
head(temps_data)
```

```
##     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
## 5 5-Jul    89    84    91    90    96    86    93    80    90    89    90    88
## 6 6-Jul    93    84    89    91    96    87    93    84    90    82    81    87
##   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
```

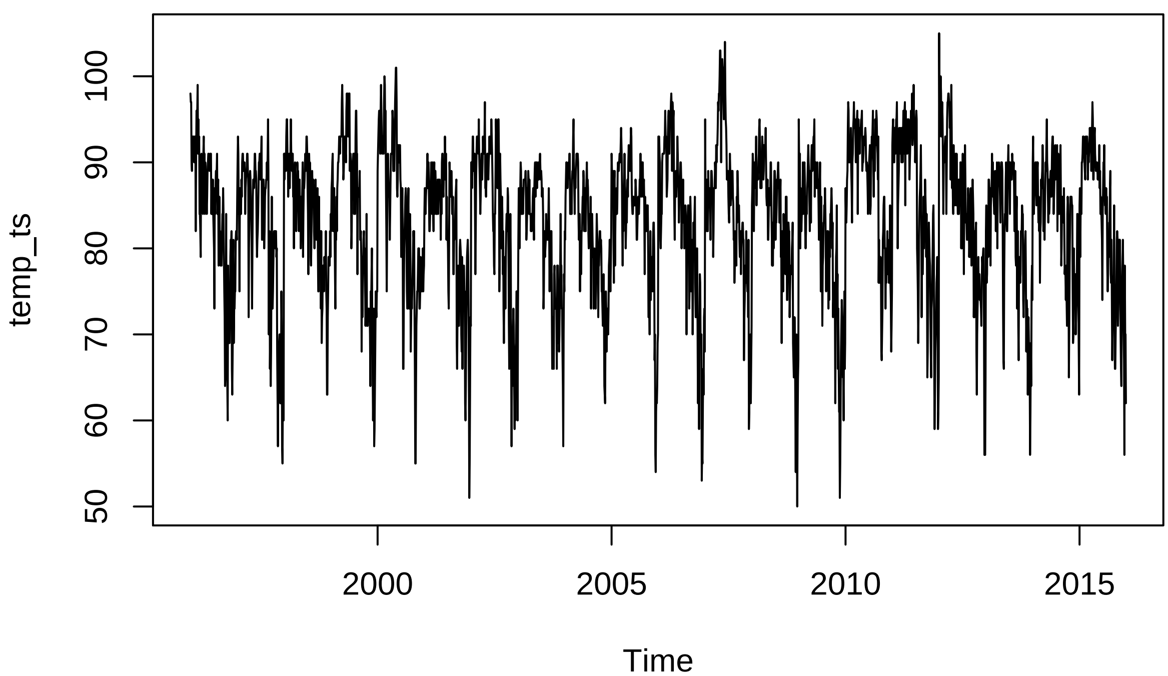
```
## 4     90     91     85     92     98     77     84     85
## 5     88     80     88     90    100     83     86     84
## 6     82     87     89     90     98     83     87     84
```

```
DAY X1996 X1997 X1998 X1999 X2000 X2001 X2002 X2003 X2004 X2005 X2006 X2007 X2008 X2009 X2010 X2011 X20
```

1 1-Jul 98 86 91 84 89 84 90 73 82 91 93 95 85 95 87 92 105 2 2-Jul 97 90 88 82 91 87 90 81 81 89 93 85 87 90 84 94 93 3 3-Jul 97 93 91 87 93 87 87 87 86 86 93 82 91 89 83 95 99 4 4-Jul 90 91 91 88 95 84 89 86 88 86 91 86 90 91 85 92 98 5 5-Jul 89 84 91 90 96 86 93 80 90 89 90 88 88 80 88 90 100 6 6-Jul 93 84 89 91 96 87 93 84 90 82 81 87 82 87 89 90 98 X2013 X2014 X2015 1 82 90 85 2 85 93 87 3 76 87 79 4 77 84 85 5 83 86 84 6 83 87 84

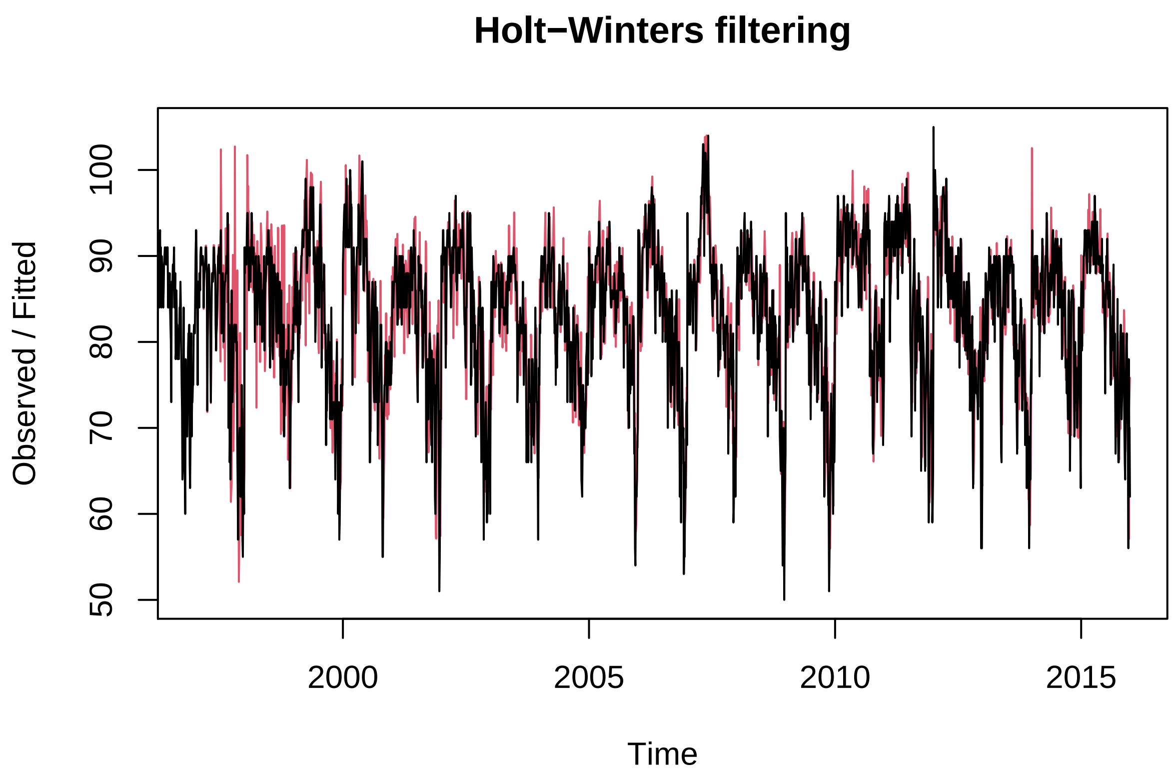
To create a time series I first only want the years and temperature of each day, I created a vector and unlisted the data to get all the atomic components, and also removed the first column as well. I created a time series object using ts, making the frequency 123 (days in the data set) and set the start point at 1996.

```r
temp_vector <- as.vector(unlist(temps_data[,2:21]))
temp_ts <- ts(data = temp_vector, frequency = 123, start = 1996)
ts.plot(temp_ts)
```



Now we will use the Holtswinter model onto the timeseries data to help smooth it out. From how I understood from reading the details in the HoltWinters details page, the function will filter a timeseries and find the optimal values for our parameters, which is the default of the function. I just had to indicate that it is seasonality type, inserting the “multiplicitive” in the function. I was not sure if the seasonality was additive or multiplicitive, so I tried multiplicative first since the amplitude does not look too constant. You can see in the plot the red line represents the fitted values (xhat) of the model.

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



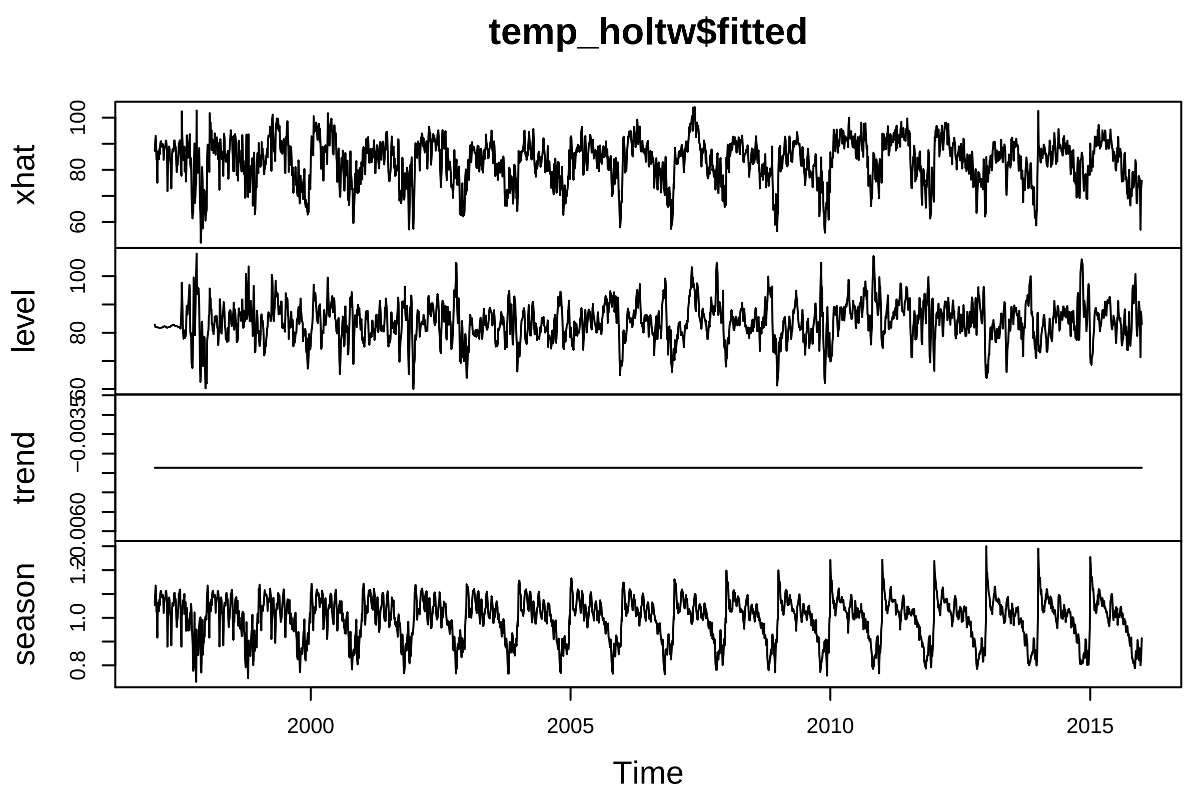

```r
summary(temp_holtw)
```

```
##              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
```

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

when I looked at the fitted version of the filtered data, it confirmed the seasonality as multiplicative for me. You can see that for the season parameter, the amplitude is increasing as time passes, specifically around 2010. I noticed that trend seems to be a horizontal straight line, but that makes sense, the global temperature should be oscillating over time as the season changes, having minimal changes in yearly temperature change in terms of degrees, so there is not much change in trend and dampening needed there.

```
plot(temp_holtw$fitted)
```



I then converted the smoothed data back into a matrix and gave the rows and columns back their names. I then exported the data as a csv so I could use excel to do cusum.

```
temps_holtw_smoothed <- matrix(temp_holtw$fitted[,1], nrow = 123)
head(temps_holtw_smoothed)
```

```
##          [,1]     [,2]     [,3]     [,4]     [,5]     [,6]     [,7]     [,8]
## [1,] 87.23653 65.04516 90.29613 83.39938 87.68863 78.07509 73.10059 87.27074
## [2,] 90.42182 84.87634 85.44878 86.44444 84.78855 86.02384 72.13247 85.01878
## [3,] 92.99734 89.61560 85.65942 92.85774 88.70570 90.23022 77.77739 82.68648
## [4,] 90.94030 88.47600 84.80741 91.55309 86.98750 87.27931 83.52416 83.37312
## [5,] 83.99917 83.11178 81.14293 88.80208 81.40681 86.06745 83.86090 83.64904
## [6,] 84.04496 88.00054 85.21673 91.04477 81.83758 87.87757 78.93483 86.79140
##          [,9]    [,10]    [,11]    [,12]    [,13]    [,14]    [,15]    [,16]
## [1,] 92.29714 78.50826 81.58696 84.72917 79.51855 86.74604 93.88371 82.30605
## [2,] 92.85614 88.18138 88.52648 80.39548 85.65722 81.47324 87.43846 92.55001
## [3,] 92.33884 92.43570 86.72311 84.53380 88.31357 82.29310 90.24836 91.18746
```

```
## [4,] 87.29596 92.69774 83.30574 89.62822 88.56597 82.90566 93.69353 95.13130
## [5,] 84.25223 90.58916 84.18954 89.27001 89.12501 80.92784 90.14667 94.60910
## [6,] 85.75665 86.91496 82.21750 84.28967 79.32562 84.52016 88.67069 96.75445
##         [,17]     [,18]    [,19]
## [1,] 84.88750 102.54643 90.07756
## [2,] 76.18707  89.57468 85.16854
## [3,] 81.46207  88.15080 82.09161
## [4,] 76.56780  87.11605 79.49314
## [5,] 75.42085  85.20682 83.69666
## [6,] 78.42462  83.25423 82.08838
```

```
colnames(temps_holtw_smoothed) <- colnames(temps_data[,3:21])
rownames(temps_holtw_smoothed) <-temps_data[,1]
write.csv(temps_holtw_smoothed, file = "C:/Users/phan_/Documents/smoothed_temp_data.csv")
```

See the attached excel to see the cusum results. Based on the results the end of summer seems to not be really getting later according to the data. The end of summer just seems to be going up and down from year to year, not really having an upward trend or downward trend so far.

Question 8.1

Describe a situation or problem from your job, everyday life, current events, etc., for which a linear regression model would be appropriate. List some (up to 5) predictors that you might use.

A way to use regression is to find out how much rent varies in Seattle and I can compare it against the square footage of the place, how many rooms it has, distance from downtown Seattle, what part of Seattle, or age of the place.

Question 8.2

Using crime data from http://www.statsci.org/data/general/uscrime.txt (file uscrime.txt, description at http://www.statsci.org/data/general/uscrime.html), use regression (a useful R function is lm or glm) to predict the observed crime rate in a city with the following data.

First I will clear the environment. Call the data and any packages I may be using.

```
rm(list = ls())
set.seed(1)
library(ggplot2)
crime_data <- read.table("C:/Users/phan_/Documents/R/uscrime.txt", stringsAsFactors = F, header = T)
View(crime_data)
head(crime_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
```

```
test<-data.frame(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)
```

```
 M So   Ed  Po1  Po2    LF   M.F Pop   NW    U1  U2 Wealth Ineq     Prob    Time Crime
```

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 26.2011 791 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 25.2999 1635 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 24.3006 578 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 29.9012 1969 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 21.2998 1234 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 20.9995 682

I tried building a model using all the features and it gave a high adjust R-squared and low overall p-value, but since there are also so many features and insignificant features, the p=value and r-squared may not be accurate. I tested it on the test data set and got 155, lower than the min of the data. The best way to figure this out is to pick the best features from the 15 we are given. Of note I did not split the data into a training and validation. I thought our data set was already small, and splitting it may have been more detrimental.

```
lm_model_1 <- lm(Crime~.,data = crime_data)
summary(lm_model_1)
```

```
##
## Call:
## lm(formula = Crime ~ ., data = crime_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
```

Call: lm(formula = Crime ~ ., data = crime_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

```r
predict(lm_model_1, test)
```

```
##        1
## 155.4349
```

```r
#155.4349
min(crime_data$Crime)
```

```
## [1] 342
```

```r
#[1] 342
```

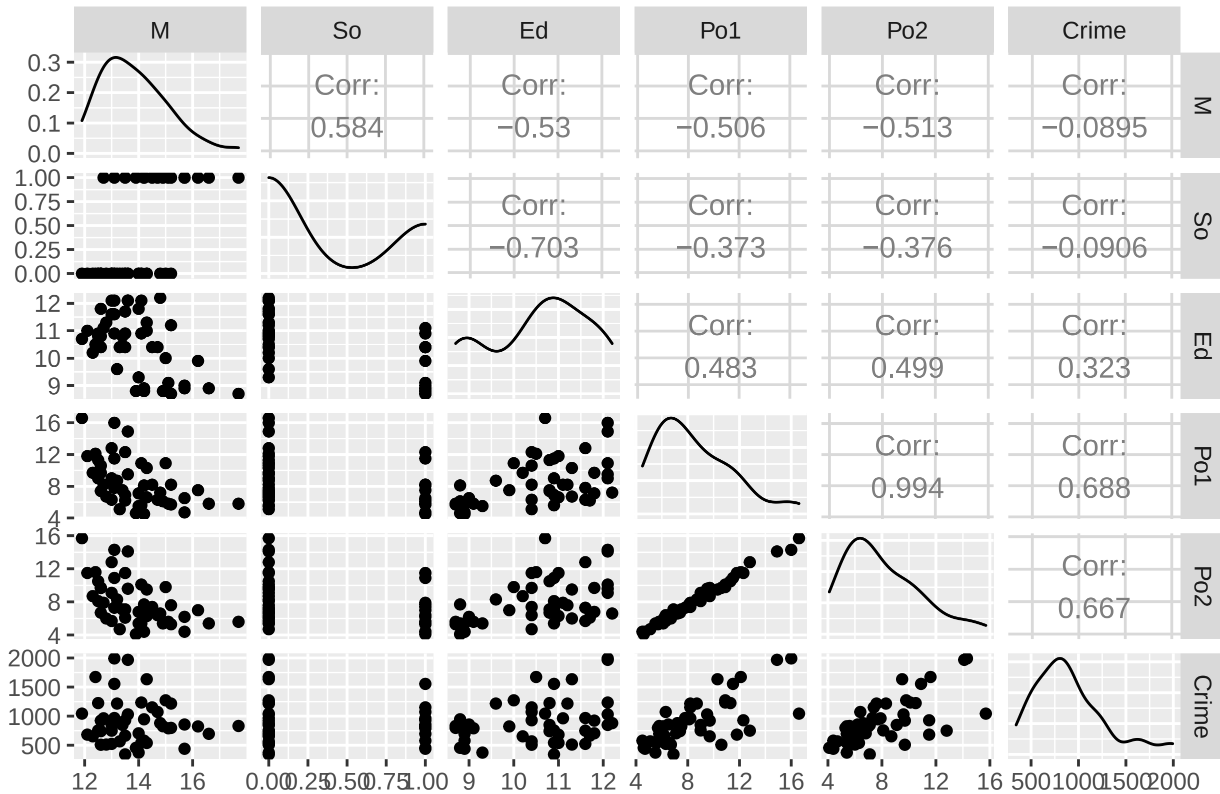
I decided to pick the best features by using the ggpairs function in the GGally package. This allows me to plot variables against each other and give me a correlation. I made vectors that split the data into 3 groups so we can visually see the scatter plots better. From this method I narrowed down the features with the strongest correlation to be Po1 and Po2.

```r
library(GGally)
```

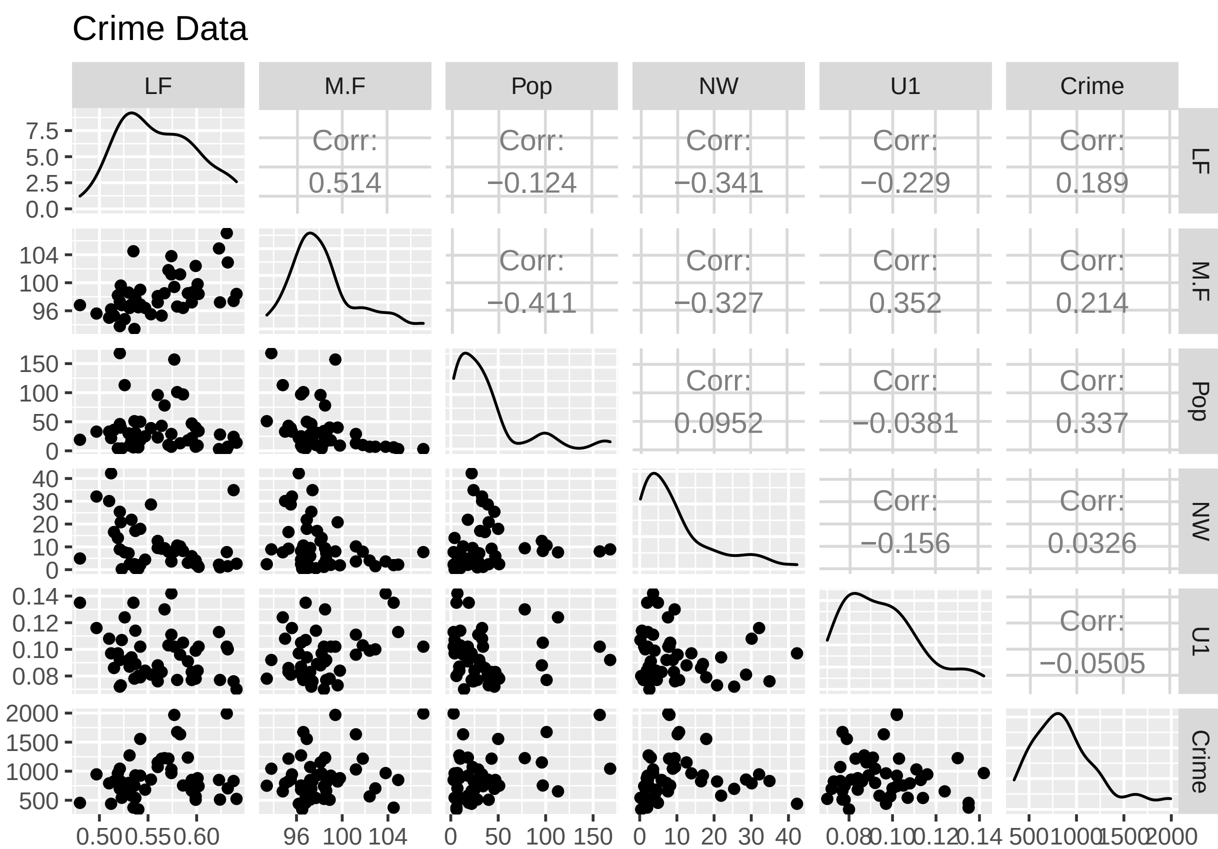
```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```r
crime_plot_1 <- ggpairs(data=crime_data, columns = c(1:5, 16), title = "Crime Data")
crime_plot_2 <- ggpairs(data=crime_data, columns = c(6:10, 16), title = "Crime Data")
crime_plot_3 <- ggpairs(data=crime_data, columns = c(10:15, 16), title = "Crime Data")
crime_plot_1
```

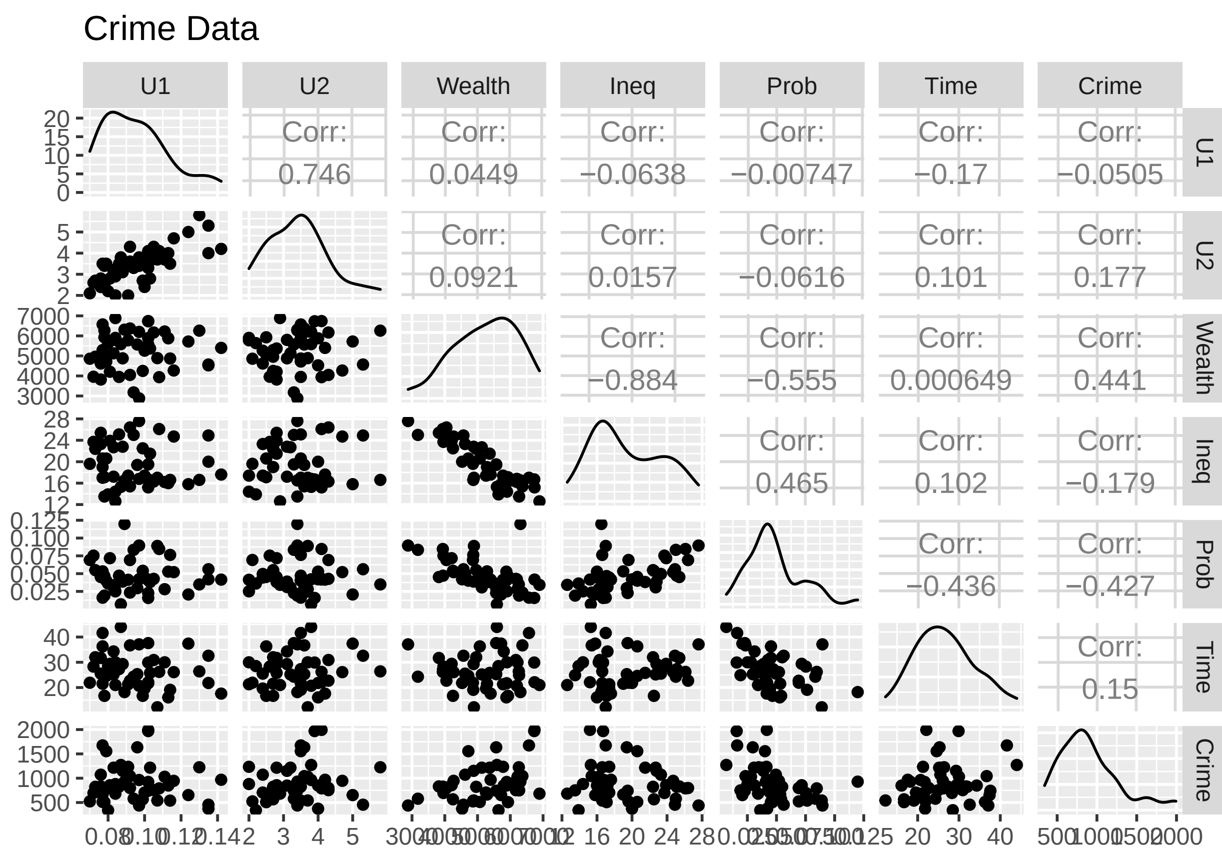
Crime Data




```
crime_plot_2
```


Crime Data
LF
M.F
Pop
NW
U1
Crime
Corr: 0.514
Corr: −0.124
Corr: −0.341
Corr: −0.229
Corr: 0.189
Corr: −0.411
Corr: −0.327
Corr: 0.352
Corr: 0.214
Corr: 0.0952
Corr: −0.0381
Corr: 0.337
Corr: −0.156
Corr: 0.0326
Corr: −0.0505

```
crime_plot_3
```



I made a new model using those 2 features and got a better result when I ran it on the test.

```r
set.seed(1)
lm_model_2 <- lm(Crime ~ Po1 + Po2,data = crime_data)
summary(lm_model_2)
```

```
##
## Call:
## lm(formula = Crime ~ Po1 + Po2, data = crime_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -636.09 -168.62   35.44  141.80  532.10
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)    158.3      125.9   1.257   0.2155
## Po1            256.2      123.4   2.076   0.0438 *
## Po2           -178.3      131.2  -1.359   0.1810
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 281.3 on 44 degrees of freedom
## Multiple R-squared:  0.494,  Adjusted R-squared:  0.471
## F-statistic: 21.48 on 2 and 44 DF,  p-value: 3.094e-07
```

Call: lm(formula = Crime ~ Po1 + Po2, data = crime_data)

Residuals: Min 1Q Median 3Q Max -636.09 -168.62 35.44 141.80 532.10

Coefficients: Estimate Std. Error t value Pr(>|t|)
(Intercept) 158.3 125.9 1.257 0.2155
Po1 256.2 123.4 2.076 0.0438 * Po2 -178.3 131.2 -1.359 0.1810
— Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 281.3 on 44 degrees of freedom Multiple R-squared: 0.494, Adjusted R-squared: 0.471 F-statistic: 21.48 on 2 and 44 DF, p-value: 3.094e-07

```
predict(lm_model_2, test)
```

```
##       1
## 468.632
```

468.632

I decided I wanted to try out a package that automatically evaluates each feature.This method is from the caret package. It uses a random forest algorithm on each iteration to evaluate the model. It explores all possible subsets of attributes and gives the top attributes.

```
library(randomForest)
```

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
library(caret)
```

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

```
set.seed(1)
control <- rfeControl(functions = rfFuncs, method = "cv", number = 10, repeats = 25)
results <- rfe(crime_data[,1:15], crime_data[,16], sizes = c(1:15), rfeControl = control)
print(results)
```

```
##
## Recursive feature selection
##
## Outer resampling method: Cross-Validated (10 fold)
##
## Resampling performance over subset size:
##
```

```
##  Variables  RMSE Rsquared   MAE RMSESD RsquaredSD  MAESD Selected
##          1 341.3   0.4705 269.7 133.98     0.2978 117.90
##          2 371.5   0.4407 289.8 124.33     0.3473 125.09
##          3 320.6   0.5945 248.4  93.51     0.2962  88.42
##          4 275.6   0.6830 209.1  84.61     0.2776  76.81        *
##          5 290.1   0.6734 217.6  91.39     0.2783  81.47
##          6 294.9   0.6413 220.4  89.95     0.2834  87.64
##          7 291.7   0.6617 214.9  90.56     0.2986  89.87
##          8 280.9   0.7162 208.8  87.60     0.3093  86.10
##          9 290.0   0.6652 215.5  93.05     0.3127  86.20
##         10 280.3   0.6850 207.0  89.16     0.2772  76.45
##         11 284.1   0.6898 214.5  88.28     0.3133  78.63
##         12 276.5   0.7094 211.4  91.58     0.3063  80.59
##         13 277.4   0.7068 211.6  77.14     0.2960  70.15
##         14 279.9   0.7146 211.5  80.69     0.3149  72.44
##         15 284.8   0.6894 216.2  76.83     0.3209  68.48
##
## The top 4 variables (out of 4):
##    Po1, Po2, NW, Prob
```

Recursive feature selection

Outer resampling method: Cross-Validated (10 fold)

Resampling performance over subset size:

Variables RMSE Rsquared MAE RMSESD RsquaredSD MAESD Selected 1 347.4 0.4851 273.1 161.0 0.3692 124.1
2 308.5 0.5293 256.5 143.6 0.3471 122.3
3 272.6 0.6149 224.5 140.3 0.3199 118.8
4 236.7 0.7032 190.4 133.2 0.2688 109.7
5 228.8 0.6988 184.6 131.6 0.2672 107.6
6 228.2 0.7232 180.2 125.1 0.2397 103.6 * 7 228.9 0.7123 181.9 133.5 0.2439 111.3
8 237.2 0.6981 188.7 129.2 0.2642 108.5
9 235.3 0.6912 189.3 124.8 0.2356 108.2
10 234.6 0.6884 188.5 128.2 0.2344 107.0
11 231.6 0.6897 187.5 128.7 0.2313 108.3
12 233.4 0.6832 186.8 129.1 0.2424 112.1
13 230.5 0.7058 183.9 123.9 0.1997 106.9
14 234.2 0.7113 186.3 126.0 0.2132 107.0
15 232.4 0.7033 186.3 126.2 0.2185 109.0

The top 5 variables (out of 6): Po1, Po2, NW, Prob, Wealth

```r
lm_model_3 <- lm(Crime ~ Po1 + Po2 + NW + Prob + Wealth + Ed,data = crime_data)
predict(lm_model_3, test)
```

```
##       1
## 458.752
```

458.752

In the end when I ran the model on the test, I got a similar result to the second model.