week04_homework

Brian Ritz

May 12, 2015

```r
require(gdata)
```

```
## Loading required package: gdata
```

```
## Warning: package 'gdata' was built under R version 3.1.3
```

```
## gdata: read.xls support for 'XLS' (Excel 97-2004) files ENABLED.
##
## gdata: read.xls support for 'XLSX' (Excel 2007+) files ENABLED.
##
## Attaching package: 'gdata'
##
## The following object is masked from 'package:stats':
##
##     nobs
##
## The following object is masked from 'package:utils':
##
##     object.size
```

```r
library(lubridate)
```

```r
library(forecast)
```

```
## Warning: package 'forecast' was built under R version 3.1.3
```

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

```
## Warning: package 'zoo' was built under R version 3.1.3
```

```
##
## Attaching package: 'zoo'
##
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
##
## Loading required package: timeDate
## This is forecast 6.0
```

```r
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
##
## The following objects are masked from 'package:lubridate':
##
##     intersect, setdiff, union
##
```

```
## The following objects are masked from 'package:gdata':
##
##     combine, first, last
##
## The following object is masked from 'package:stats':
##
##     filter
##
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
# import all the data

date.char <- "July-1"



import.day.data <- function(date.char){

  a <- read.xls(paste0("../input/I-57-2013-", date.char, ".xls"), header=T, skip=1, stringsAsFa
ctors=F)

  #cnames<-names(a)
  #first.row <- as.character(as.vector(a[1,]))

  actual.data <- a[3:26,c(3,5)]
  date.vector <- rep(date.char, nrow(actual.data))
  actual.data$date <- date.vector
  return (actual.data)
}
```

```r
all.dates<- c("June-16", "June-17", "June-18", "June-19", "June-20", "June-21", "June-22", "June-23", "June-24", "June-25", "June-26", "June-27", "June-28", "June-29", "June-30", "July-1")

all.data<-sapply(all.dates, FUN=import.day.data, simplify=F)

all.data<-do.call(rbind, all.data)

all.data$date <- sapply(all.data$date, FUN=function(x) paste0(x, "-2013"))
all.data$Time <- sapply(all.data$Time, FUN=function(x) paste0(x, ":00"))
all.data$DateTime <- mdy_hms(mapply(FUN=function(x, y) (paste(x, y, sep=" ")), all.data$date, all.data$Time, USE.NAMES=F), tz="America/Chicago")
all.data$I80E <- as.numeric(all.data$I80E)
saveRDS(all.data, file="I80_traffic.Rds")
```

```r
# make the time series and fit using auto-arima

I80.traffic<-ts(as.numeric(all.data$I80E), frequency = 24)

june.traffic <- window(I80.traffic, start=c(1,1), end=c(15, 24))
july.traffic <- window(I80.traffic, start=c(16,1), end=c(16, 24))

fit <- auto.arima(june.traffic)
```

DO AICc and BIC select the same model as the best model?

```r
summary(fit <- auto.arima(june.traffic, ic="aicc", seasonal = F))
```

```
## Series: june.traffic
## ARIMA(2,0,3) with non-zero mean
```

```
##
## Coefficients:
##          ar1      ar2      ma1      ma2      ma3  intercept
##       1.8088  -0.8853  -0.5348  -0.2671  -0.1157   746.3181
## s.e.  0.0288   0.0287   0.0600   0.0596   0.0654     6.8586
##
## sigma^2 estimated as 13219:  log likelihood=-2220.78
## AIC=4455.56   AICc=4455.88   BIC=4482.77
##
## Training set error measures:
##                     ME     RMSE    MAE  MPE MAPE    MASE         ACF1
## Training set -1.390098 114.9732 79.019 -Inf  Inf 0.80082 -0.003018285
```

```r
summary(fit <- auto.arima(june.traffic, ic="bic", seasonal=F))
```

```
## Series: june.traffic
## ARIMA(2,0,2) with non-zero mean
##
## Coefficients:
##          ar1      ar2      ma1      ma2  intercept
##       1.8308  -0.9072  -0.5916  -0.3254   746.3649
## s.e.  0.0229   0.0228   0.0488   0.0471     6.9120
##
## sigma^2 estimated as 13327:  log likelihood=-2222.26
## AIC=4456.52   AICc=4456.76   BIC=4479.83
##
## Training set error measures:
##                     ME     RMSE      MAE  MPE MAPE      MASE       ACF1
## Training set -1.376334 115.4405 79.43987 -Inf  Inf 0.8050852 0.02457699
```

Evaluating using AICc or BIC does make a difference. Using AICc, auto.arima produced an arima of (2,0,3), while BIC produced a model of order (2,0,2).

Instead of changing p and q manually and checking the AICc and BIC of the resulting fit, I used auto.arima to iterate through many p and q parameters and pick the best one based on AICc (in the first case) and BIC (in the second case).

PART 2 Use the day of week seasonal ARIMA(p,d,q)x(P,D,Q)s to forecast July 1, which is Monday.

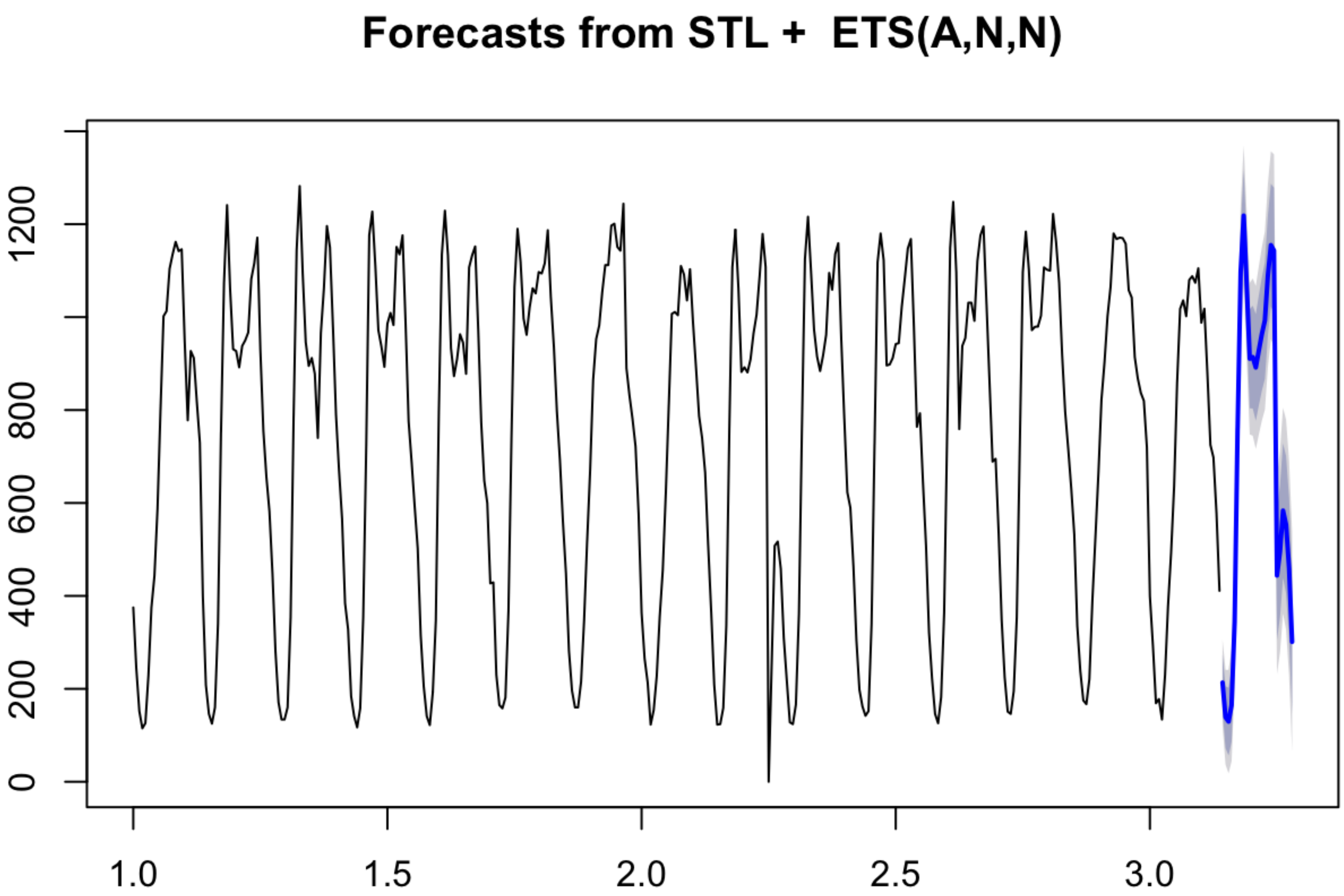
```
# make a weekly time series
weekly.time.series <- ts(june.traffic, frequency=24*7)

# fit the time series, the seasonality component defaults to the frequency of the time series
summary(weekly.fit<- auto.arima(weekly.time.series, seasonal=T))
```

```
## Series: weekly.time.series
## ARIMA(0,1,2)(0,1,0)[168]
##
## Coefficients:
##           ma1      ma2
##       -0.4741  -0.4853
## s.e.   0.0593   0.0586
##
## sigma^2 estimated as 7007:  log likelihood=-1121.66
## AIC=2249.31   AICc=2249.44   BIC=2259.07
##
## Training set error measures:
##                    ME     RMSE      MAE  MPE MAPE      MASE      ACF1
## Training set 2.143998 60.97278 24.90656 -Inf  Inf 0.5210919 0.0387665
```

Now forecast for July 1:

```
plot(july.1.forecast <- forecast(weekly.time.series, h = 24))
```




```
summary(july.1.forecast)
```

```
##
## Forecast method: STL +  ETS(A,N,N)
##
## Model Information:
## ETS(A,N,N)
##
## Call:
##  ets(y = x, model = etsmodel, allow.multiplicative.trend = allow.multiplicative.trend)
##
##   Smoothing parameters:
##     alpha = 0.4928
##
##   Initial states:
##     l = 739.9329
##
##   sigma:  46.5995
##
##      AIC     AICc      BIC
## 4888.941 4888.975 4896.714
##
## Error measures:
##                      ME     RMSE     MAE  MPE MAPE      MASE      ACF1
## Training set 0.07314343 46.59945 25.2849 -Inf  Inf 0.5290075 0.1828475
##
## Forecasts:
##          Point Forecast      Lo 80     Hi 80      Lo 95     Hi 95
## 3.142857       213.4702  153.75057  273.1898  122.13693  304.8034
## 3.148810       138.8634   72.28531  205.4414   37.04102  240.6857
## 3.154762       129.5572   56.76404  202.3503   18.22967  240.8847
## 3.160714       164.6290   86.11115  243.1468   44.54632  284.7116
```

```
## 3.166667        342.5877  258.73508  426.4402  214.34621  470.8291
## 3.172619        760.4705  671.60281  849.3381  624.55911  896.3818
## 3.178571       1097.8057 1004.19122 1191.4201  954.63472 1240.9767
## 3.184524       1218.4032 1120.27130 1316.5351 1068.32340 1368.4830
## 3.190476       1067.4875  965.03717 1169.9379  910.80322 1224.1719
## 3.196429        910.5747  803.98069 1017.1687  747.55322 1073.5962
## 3.202381        913.9949  803.41243 1024.5775  744.87358 1083.1163
## 3.208333        891.6979  777.26584 1006.1300  716.68917 1066.7067
## 3.214286        928.2133  810.05706 1046.3696  747.50891 1108.9178
## 3.220238        963.4818  841.71521 1085.2485  777.25585 1149.7078
## 3.226190        992.6566  867.38359 1117.9295  801.06809 1184.2450
## 3.232143       1090.2479  961.56407 1218.9317  893.44298 1287.0528
## 3.238095       1154.9122 1022.90568 1286.9188  953.02564 1356.7988
## 3.244048       1143.3740 1008.12629 1278.6217  936.53050 1350.2175
## 3.250000        443.9122  305.49924  582.3251  232.22786  655.5965
## 3.255952        495.5100  354.00261  637.0174  279.09312  711.9269
## 3.261905        583.5438  439.00814  728.0794  362.49559  804.5920
## 3.267857        553.4150  405.91329  700.9167  327.83059  778.9994
## 3.273810        460.0600  309.65072  610.4694  230.02883  690.0913
## 3.279762        301.1217  147.85989  454.3834   66.72801  535.5153
```

** PART 3 ** Hour of the day seasonality included:

```
summary(day.seasonality.fit <- auto.arima(june.traffic), seasonal=T)
```

```
## Series: june.traffic
## ARIMA(2,0,1)(2,0,0)[24] with non-zero mean
##
## Coefficients:
##          ar1      ar2      ma1    sar1    sar2  intercept
```

```
##       1.7922  -0.8685  -0.9146  0.4866  0.1010   743.7286
## s.e.  0.0299   0.0291   0.0257  0.0555  0.0557    13.6793
##
## sigma^2 estimated as 10558:  log likelihood=-2184.12
## AIC=4382.23   AICc=4382.55   BIC=4409.43
##
## Training set error measures:
##                     ME     RMSE      MAE  MPE MAPE      MASE       ACF1
## Training set -1.090589 102.7539 70.18552 -Inf  Inf 0.7112968 0.09755528
```

The ARIMA(2,0,1)(2,0,0)[24] tells us that the frequency of the seasonality is 24 hours, or a day. This is daily seasonality.

July Forecast for the daily seasonality:

```
july.1.forecast.daily <- forecast(day.seasonality.fit, h=24)

(july.1.forecast.daily.means <- july.1.forecast.daily$mean)
```

```
## Time Series:
## Start = c(16, 1)
## End = c(16, 24)
## Frequency = 24
##  [1]  288.2947  292.1714  295.9657  369.2010  416.8783  533.8109  661.2150
##  [8]  756.5516  854.0998  970.2372 1044.1427 1037.2500 1004.1374 1009.4760
## [15]  981.8795  945.2603  933.5026  846.3402  845.4994  757.6490  677.0171
## [22]  665.1854  614.4364  534.6310
```

The forecast for July 1 at 8:00 is `july.1.forecast.daily.means[8]`, the forecast for 9:00 is `july.1.forecast.daily.means[9]`, the forecast for 17:00 is `july.1.forecast.daily.means[17]`, the forecast for 18:00 is `july.1.forecast.daily.means[18]`.

** PART 4 ** For the July 1 8:00, 9:00, 17:00 and 18:00 forecasts, which model is better (part 2 or part 3) ?

```
actuals <- july.traffic
weekly.forecast <- july.1.forecast$mean
daily.forecast <- july.1.forecast.daily.means

weekly.error <- (as.vector(weekly.forecast) - as.vector(actuals))
daily.error <-  (as.vector(daily.forecast) - as.vector(actuals))
error.comparison <- weekly.error / daily.error

(error.comparison[8])
```

```
## [1] 0.03063663
```

```
(error.comparison[9])
```

```
## [1] 0.1661291
```

```
(error.comparison[17])
```

```
## [1] -0.06192993
```

```
(error.comparison[18])
```

```
## [1] -0.05085257
```

For all of 8:00, 9:00, 17:00, and 18:00, the weekly forecast performed better than the daily forecast by a wide margin–mostly greater than 10 to 1.