Assignment_8

Weijie Gao

11/28/2017

```
suppressWarnings(library(forecast))
suppressWarnings(library(xts))
```

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

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

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

```
suppressWarnings(library(fGarch))
```

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

```
## Loading required package: timeSeries
```

```
##
## Attaching package: 'timeSeries'
```

```
## The following object is masked from 'package:zoo':
##
##     time<-
```

```
## Loading required package: fBasics
```

```
##
```

```
## Rmetrics Package fBasics
```

```
## Analysing Markets and calculating Basic Statistics
```

```
## Copyright (C) 2005-2014 Rmetrics Association Zurich
```

```
## Educational Software for Financial Engineering and Computational Science
```

```
## Rmetrics is free software and comes with ABSOLUTELY NO WARRANTY.
```

```
## https://www.rmetrics.org --- Mail to: info@rmetrics.org
```

```
dataPath <- "~/Google Drive/2017 Fall/Time Series/Week8"
seat_price <- read.csv(paste(dataPath,"seat_price.csv",sep='/'), header=TRUE)
classification_data <- read.csv(paste(dataPath,"Contracts_Classification.csv"
,sep='/'), header=TRUE)
```

```
volume_data <- read.csv(paste(dataPath,"Contracts_Volume.csv",sep='/'), heade
r=TRUE)

head(seat_price)

##     Date    CME      IMM    IOM
## 1 2001/1 188000 183125.0 130000
## 2 2001/2 250000 225000.0 170000
## 3 2001/3 250000 292500.0 242500
## 4 2001/4 287500 298750.0 291500
## 5 2001/5 325000 305000.0 275750
## 6 2001/6 375000 355333.3 260000

head(classification_data)

##   Commodity.Code Division
## 1             48      CME
## 2             56      CME
## 3             62      CME
## 4             BZ      CME
## 5             CB      CME
## 6            CPO      CME

head(volume_data)

##         Date Commodity.Indicator Product.Short.Desc Future.Option
## 1 01/01/2000                  ED       EURO DLR FUT             F
## 2 01/01/2000                  SP        S&P 500 FUT             F
## 3 01/01/2000                  ES      EMINI S&P FUT             F
## 4 01/01/2000                  ED      EURO DLR CALL             O
## 5 01/01/2000                  ED       EURO DLR PUT             O
## 6 01/01/2000                  J1      JAPAN YEN FUT             F
##   Electronic.Volume Total.Volume
## 1           232,796      8379642
## 2            88,426      1915082
## 3         1,302,447      1305618
## 4             6,600       972970
## 5             6,303       942420
## 6            39,288       436505
```

1. CME

Task A

```
seat <- "CME"
commodity <- c(as.character(classification_data$Commodity.Code[classification
_data$Division=="CME"]))
commodity <- unique(commodity)
commodity

##  [1] "48"   "56"   "62"   "BZ"   "CB"   "CPO"  "CSC"  "DA"   "DB"   "DK"
## [11] "DP"   "DY"   "EG"   "EQ"   "ET"   "FB"   "FC"   "GD"   "GN"   "GNP"
## [21] "H1"   "H2"   "H3"   "H4"   "ISM"  "LB"   "LC"   "LH"   "LN"   "MX"
```

```
## [31] "NF"   "PB"   "PC"   "UF"   "UL"   "Z-BB" "Z-BD" "Z-E"  "Z-ET" "Z-LB"
## [41] "Z-LU" "Z-PO" "Z-PY" "Z-ST" "Z-TR" "Z-UC"

tradable_commodity <- is.element(volume_data$Commodity.Indicator,commodity)
head(tradable_commodity,3)

## [1] FALSE FALSE FALSE

volume <- volume_data[tradable_commodity,]
head(volume,3)

##          Date Commodity.Indicator Product.Short.Desc Future.Option
## 9  01/01/2000                  48      LV CATTLE FUT             F
## 17 01/01/2000                  LN      LEAN HOGS FUT             F
## 23 01/01/2000                  62     FDR CATTLE FUT             F
##    Electronic.Volume Total.Volume
## 9                  0       347290
## 17                 0       175399
## 23                 0        73232

volume$Electronic.Volume <- as.numeric(gsub(",","",volume$Electronic.Volume,f
ixed=TRUE))
volume$Floor.Volume <- volume$Total.Volume-volume$Electronic.Volume
head(volume,3)

##          Date Commodity.Indicator Product.Short.Desc Future.Option
## 9  01/01/2000                  48      LV CATTLE FUT             F
## 17 01/01/2000                  LN      LEAN HOGS FUT             F
## 23 01/01/2000                  62     FDR CATTLE FUT             F
##    Electronic.Volume Total.Volume Floor.Volume
## 9                  0       347290       347290
## 17                 0       175399       175399
## 23                 0        73232        73232

# aggregate upon derivative types.
cme_volume_0 <- aggregate(cbind(Electronic.Volume, Floor.Volume,
Total.Volume)~Date+Commodity.Indicator, data=volume, sum)
head(cme_volume_0,3)

##         Date Commodity.Indicator Electronic.Volume Floor.Volume
## 1 01/01/2000                  48                 0       397437
## 2 01/01/2001                  48                 0       595662
## 3 01/01/2002                  48                 0       371730
##   Total.Volume
## 1       397437
## 2       595662
## 3       371730

dim(cme_volume_0)

## [1] 2038    5
```

```r
# drop any data before date 1/1/2001
cme_volume <- cme_volume_0[!as.Date(cme_volume_0$Date, "%m/%d/%Y") < as.Date(
"2001-01-01"),]
head(cme_volume,3)
```

```
##         Date Commodity.Indicator Electronic.Volume Floor.Volume
## 2 01/01/2001                  48                0       595662
## 3 01/01/2002                  48                0       371730
## 4 01/01/2003                  48                0       411141
##   Total.Volume
## 2       595662
## 3       371730
## 4       411141
```

```r
dim(cme_volume)
```

```
## [1] 1942    5
```

```r
names(cme_volume)
```

```
## [1] "Date"                "Commodity.Indicator" "Electronic.Volume"
## [4] "Floor.Volume"        "Total.Volume"
```

```r
# aggregate CME based on Date
cme_volume<-aggregate(cbind(Electronic.Volume, Floor.Volume, Total.Volume)~Da
te, data=cme_volume,sum)
head(cme_volume,3)
```

```
##         Date Electronic.Volume Floor.Volume Total.Volume
## 1 01/01/2001                0       939008       939008
## 2 01/01/2002                7       633643       633650
## 3 01/01/2003              929       729796       730725
```

```r
# order by date
cme_volume <- cme_volume[order(as.Date(cme_volume$Date, "%m/%d/%Y")),]
head(cme_volume,5)
```

```
##          Date Electronic.Volume Floor.Volume Total.Volume
## 1  01/01/2001                0       939008       939008
## 14 02/01/2001                0       692906       692906
## 27 03/01/2001                0       803275       803275
## 40 04/01/2001                0       612603       612603
## 53 05/01/2001                0       714467       714467
```

```r
tail(cme_volume,5)
```

```
##           Date Electronic.Volume Floor.Volume Total.Volume
## 104 08/01/2013           1851029       309396      2160425
## 117 09/01/2013           2309003       389281      2698284
## 130 10/01/2013           2208718       381331      2590049
## 143 11/01/2013           2013552       370374      2383926
## 156 12/01/2013           1679846       317249      1997095
```

```r
# combine trading volume with seat price.
cme_seat_price <- seat_price[[toupper(seat)]]
head(cme_seat_price)
```

```
## [1] 188000 250000 250000 287500 325000 375000
```

```r
cme_volume$Seat.Price <- cme_seat_price
head(cme_volume,3)
```

```
##          Date Electronic.Volume Floor.Volume Total.Volume Seat.Price
## 1  01/01/2001                 0       939008       939008     188000
## 14 02/01/2001                 0       692906       692906     250000
## 27 03/01/2001                 0       803275       803275     250000
```

Split data into train and test

```r
# Partion the training and testing data based on the date of trading.
# training data: data with date before year 2013
# testing data: data with date in year 2013
cme_train <- cme_volume[as.Date(cme_volume$Date,"%m/%d/%Y") < as.Date("2013-01-01") ,]
cme_test <- cme_volume[!as.Date(cme_volume$Date,"%m/%d/%Y") < as.Date("2013-01-01") ,]
# create an empty list to store forecast values
seat_price_forecast <- list()
```

1.1 Linear regression

```r
cme_lm <- lm(Seat.Price~Electronic.Volume+Floor.Volume, data=cme_train)
summary(cme_lm)
```

```
##
## Call:
## lm(formula = Seat.Price ~ Electronic.Volume + Floor.Volume, data = cme_train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -498555 -141758  -31015   66022  936120
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)       9.297e+04  6.514e+04   1.427    0.156
## Electronic.Volume 1.658e-01  2.721e-02   6.094 9.95e-09 ***
## Floor.Volume      4.308e-01  5.954e-02   7.235 2.73e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 235200 on 141 degrees of freedom
## Multiple R-squared:  0.3251, Adjusted R-squared:  0.3155
## F-statistic: 33.96 on 2 and 141 DF,  p-value: 9.144e-13
```

```r
seat_price_forecast$lm <- predict(cme_lm, cme_test)
```

1.2 Linear regression with ARMA errors (use arima with xreg)

```
cme_lm_arma_errors <- auto.arima(cme_train$Seat.Price, xreg=cme_train[,c(2,3)
], allowdrift = FALSE)
summary(cme_lm_arma_errors)
```

```
## Series: cme_train$Seat.Price
## Regression with ARIMA(1,1,2) errors
##
## Coefficients:
##           ar1     ma1     ma2  Electronic.Volume  Floor.Volume
##       -0.5312  0.5857  0.3338             0.0414        0.0090
## s.e.   0.1497  0.1480  0.0928             0.0367        0.0249
##
## sigma^2 estimated as 3.787e+09:  log likelihood=-1777.43
## AIC=3566.85   AICc=3567.47   BIC=3584.63
##
## Training set error measures:
##                    ME     RMSE      MAE       MPE     MAPE    MASE
## Training set 806.638 60241.22 40170.73 0.1031762 6.501965 1.00873
##                      ACF1
## Training set -0.01528629
```

```
seat_price_forecast$lm_arma_errors <- predict(cme_lm_arma_errors, n.ahead=12,
 newxreg = cme_test[,c(2,3)])$mean
```

1.3 ARIMA

```
cme_arima <- auto.arima(cme_train$Seat.Price, allowdrift = FALSE)
summary(cme_arima)
```

```
## Series: cme_train$Seat.Price
## ARIMA(1,1,2)
##
## Coefficients:
##           ar1     ma1     ma2
##       -0.5097  0.5635  0.3139
## s.e.   0.1527  0.1538  0.0915
##
## sigma^2 estimated as 3.792e+09:  log likelihood=-1778.53
## AIC=3565.05   AICc=3565.34   BIC=3576.9
##
## Training set error measures:
##                     ME     RMSE      MAE       MPE     MAPE     MASE
## Training set 1180.256 60715.16 40093.21 0.1353883 6.484681 1.006783
##                      ACF1
## Training set -0.01507773
```

```
seat_price_forecast$arima <- forecast(cme_arima,h=12)$mean
```

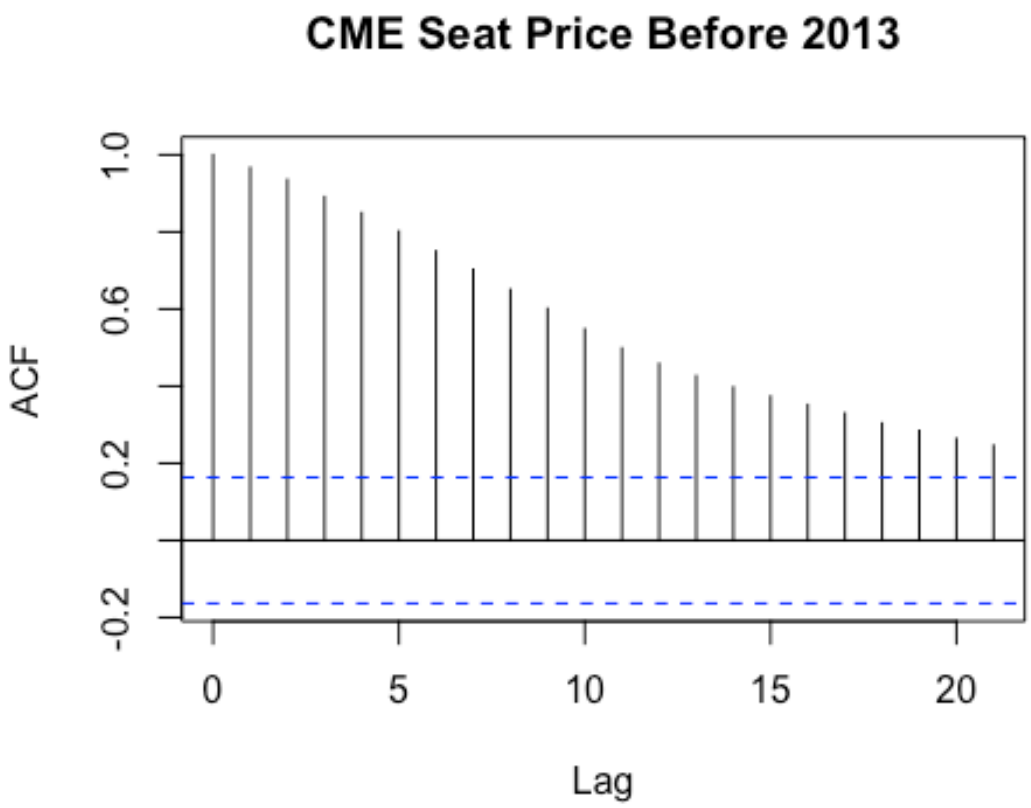
1.4 Seasonal ARIMA (SARIMA)

```
cme_sarima <- auto.arima(ts(cme_train$Seat.Price,frequency = 12), allowdrift
= FALSE)
summary(cme_sarima)
```

```
## Series: ts(cme_train$Seat.Price, frequency = 12)
## ARIMA(1,1,2)
##
## Coefficients:
##           ar1     ma1     ma2
##       -0.5097  0.5635  0.3139
## s.e.   0.1527  0.1538  0.0915
##
## sigma^2 estimated as 3.792e+09:  log likelihood=-1778.53
## AIC=3565.05   AICc=3565.34   BIC=3576.9
##
## Training set error measures:
##                     ME     RMSE      MAE       MPE     MAPE      MASE
## Training set 1180.256 60715.16 40093.21 0.1353883 6.484681 0.2048874
##                      ACF1
## Training set -0.01507773
```

1.5 Fractional ARIMA (ARFIMA)

```
acf(cme_train$Seat.Price, main = "CME Seat Price Before 2013")
```

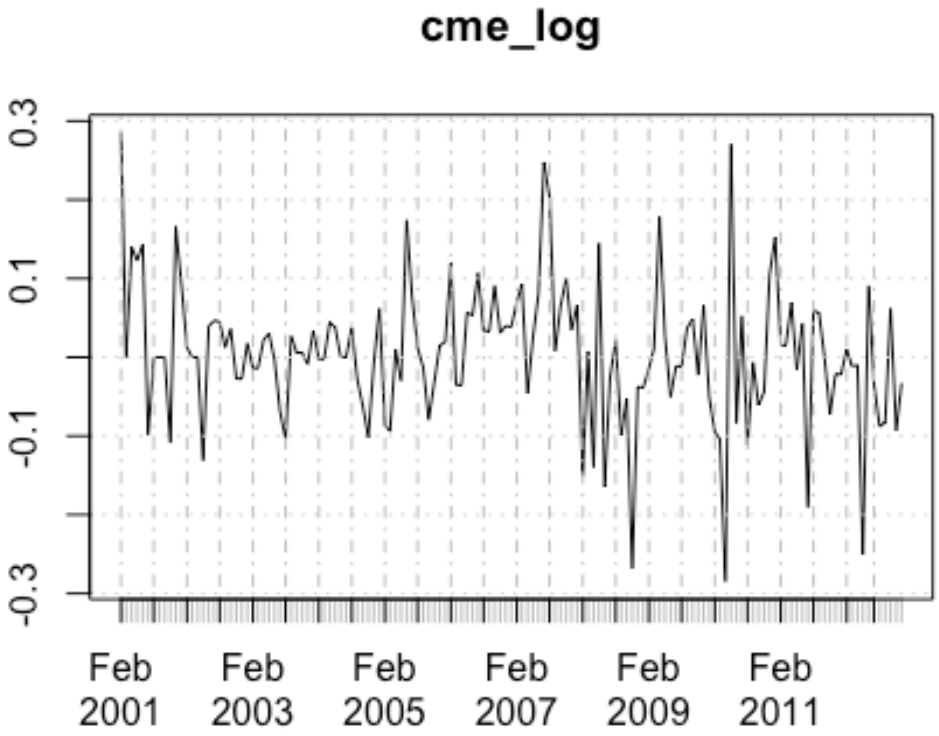



```
cme_arfima <- arfima(cme_train$Seat.Price)
summary(cme_arfima)

##
## Call:
##   arfima(y = cme_train$Seat.Price)
##
## Coefficients:
##          Estimate Std. Error    z value Pr(>|z|)
## d       2.744e-01  7.653e-02  3.585e+00 0.000337 ***
## ar.ar1  2.708e-01  3.446e-11  7.860e+09  < 2e-16 ***
## ar.ar2  6.374e-01  2.677e-10  2.381e+09  < 2e-16 ***
## ma.ma1 -4.392e-01  4.964e-02 -8.848e+00  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## sigma[eps] = 61381.44
## [d.tol = 0.0001221, M = 100, h = 1.889e-05]
## Log likelihood: -1792 ==> AIC = 3594.38 [5 deg.freedom]

seat_price_forecast$arfima <- forecast(cme_arfima , h=12)$mean
```

1.6 ARMA and GARCH combination

```
cme_seat_price_xts <- xts(cme_train$Seat.Price, order.by = as.Date(cme_train$
Date, "%m/%d/%Y"))
```

```r
cme_log <- log(cme_seat_price_xts)
# in order to make data stationary, need to take first difference of cme_log
cme_log <- diff(cme_log)[-1]
plot(cme_log)
```




```r
# find p,q
cme_arma <- auto.arima(cme_log)
summary(cme_arma)
```

```
## Series: cme_log
## ARIMA(0,0,0) with zero mean
##
## sigma^2 estimated as 0.007943:  log likelihood=142.83
## AIC=-283.66   AICc=-283.63   BIC=-280.7
##
## Training set error measures:
##                       ME       RMSE        MAE MPE MAPE      MASE
## Training set 0.005604398 0.08912215 0.06304191 100  100 0.7582197
##                     ACF1
## Training set 0.04236152
```

```r
# from summary results, we will choose p=2, q=2
cme_garch <- garchFit(~arma(2,2) + garch(1,1), data=cme_log, cond.dist = "std
```

```
", trace=F)
summary(cme_garch)

##
## Title:
##  GARCH Modelling
##
## Call:
##  garchFit(formula = ~arma(2, 2) + garch(1, 1), data = cme_log,
##     cond.dist = "std", trace = F)
##
## Mean and Variance Equation:
##  data ~ arma(2, 2) + garch(1, 1)
## <environment: 0x7f8b47118c78>
##  [data = cme_log]
##
## Conditional Distribution:
##  std
##
## Coefficient(s):
##          mu          ar1          ar2          ma1          ma2
##  0.00077357   0.28505793   0.48505406  -0.17507216  -0.43417078
##       omega       alpha1        beta1        shape
##  0.00724948   0.49659210   0.00000001   2.79138169
##
## Std. Errors:
##  based on Hessian
##
## Error Analysis:
##          Estimate  Std. Error  t value Pr(>|t|)
## mu      7.736e-04   2.221e-03    0.348  0.72756
## ar1     2.851e-01   2.765e-01    1.031  0.30265
## ar2     4.851e-01   1.569e-01    3.091  0.00200 **
## ma1    -1.751e-01   2.720e-01   -0.644  0.51974
## ma2    -4.342e-01   1.378e-01   -3.150  0.00163 **
## omega   7.249e-03   5.321e-03    1.362  0.17304
## alpha1  4.966e-01   6.139e-01    0.809  0.41855
## beta1   1.000e-08   6.784e-01    0.000  1.00000
## shape   2.791e+00   9.903e-01    2.819  0.00482 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Log Likelihood:
##  164.562    normalized:  1.150784
##
## Description:
##  Fri Dec  1 21:42:35 2017 by user:
##
##
## Standardised Residuals Tests:
```

```
##                                 Statistic p-Value
##  Jarque-Bera Test   R    Chi^2  31.49075  1.451682e-07
##  Shapiro-Wilk Test  R    W      0.949012  4.179596e-05
##  Ljung-Box Test     R    Q(10)  3.674726  0.9608301
##  Ljung-Box Test     R    Q(15)  10.31597  0.7994079
##  Ljung-Box Test     R    Q(20)  12.52497  0.8968233
##  Ljung-Box Test     R^2  Q(10)  7.148593  0.7113464
##  Ljung-Box Test     R^2  Q(15)  8.476763  0.903251
##  Ljung-Box Test     R^2  Q(20)  15.39135  0.7535939
##  LM Arch Test       R    TR^2   8.116551  0.7759572
##
## Information Criterion Statistics:
##       AIC       BIC       SIC      HQIC
## -2.175693 -1.989220 -2.183007 -2.099919
```

```
# forecast value
forecast_cme_log_garch <- predict(cme_garch, n.ahead=12)$meanForecast

# compute the price from forecast_cme_log_garch
seat_price_forecast$garch <- as.numeric(tail(cme_seat_price_xts,1)*exp(cumsum
(forecast_cme_log_garch)))
```

Task B (sMAPE)

```
smape <- function(fitted, actual) {
          return(2*mean(abs(fitted - actual) / (abs(fitted) + abs(actual))))
          }
cme_smape <- mapply(smape, seat_price_forecast, list(cme_test$Seat.Price))
cme_smape
```

```
##        lm     arima    arfima     garch
## 0.3697493 0.1173591 0.0920966 0.1194568
```

```
cme_smape[which.min(cme_smape)]
```

```
##    arfima
## 0.0920966
```

From sMAPE results, the ARFIMA model is the best one to forecast monthly prices for CME seat classes since ARFIMA model has the smallest sMAPE thus the best.

2. IMM

Task A

```
# IMM commodity subset
seat <- "IMM"
commodity_imm <- c(as.character(classification_data$Commodity.Code[classifica
tion_data$Division=="IMM"]))
commodity_imm <- unique(commodity_imm)
commodity_imm
```

```
##   [1] "16E"  "36E"  "3F"   "AC"   "AD"   "AJ"   "AN"   "BF"   "BP"   "BY"

##  [11] "C1"   "CA"   "CC"   "CD"   "CN"   "CNH"  "CU"   "CY"   "CZ"   "DJ"

##  [21] "DNV"  "E1"   "E5B"  "E7"   "EB"   "EC"   "ED"   "EED"  "EK"   "EL"

##  [31] "EM"   "EY"   "FR"   "FT"   "FXD"  "HC"   "IS"   "IY"   "J1"   "J7"

##  [41] "JB"   "JY"   "K"    "KE"   "KEV"  "KFV"  "KJ"   "KRW"  "KTV"  "LBA"

##  [51] "M6A"  "M6B"  "M6C"  "M6E"  "M6J"  "M6S"  "MB"   "MCD"  "MIR"  "MJY"

##  [61] "MNH"  "MSF"  "NB"   "NE"   "OSP"  "PZ"   "R"    "RF"   "RMB"  "RME"

##  [71] "RMY"  "RP"   "RU"   "RY"   "S0"   "S2"   "S5"   "SE"   "SEV"  "SF"

##  [81] "SIR"  "SJ"   "SKV"  "T1"   "TB"   "TRE"  "TRY"  "TZ"   "UN"   "YR"

##  [91] "Z"    "ZAR"  "Z-DC" "Z-DD" "Z-DM" "Z-DP" "Z-DY" "Z-FE" "Z-FM" "Z-FR
"
## [101] "Z-IP" "Z-JM" "Z-MS" "Z-RD" "Z-RP" "Z-RY"
```

```
# filter out tradable commodities
tradable_commodity_imm <- is.element(volume_data$Commodity.Indicator,commodit
y_imm)
head(tradable_commodity_imm,3)
```

```
## [1]  TRUE FALSE FALSE
```

```
volume_imm <- volume_data[tradable_commodity_imm,]
head(volume_imm,3)
```

```
##         Date Commodity.Indicator Product.Short.Desc Future.Option
## 1 01/01/2000                  ED       EURO DLR FUT             F
## 4 01/01/2000                  ED      EURO DLR CALL             O
## 5 01/01/2000                  ED       EURO DLR PUT             O
##   Electronic.Volume Total.Volume
## 1           232,796      8379642
## 4             6,600       972970
## 5             6,303       942420
```

```
# floor volume
volume_imm$Electronic.Volume <- as.numeric(gsub(",","",volume_imm$Electronic.
Volume,fixed=TRUE))
volume_imm$Floor.Volume <- volume_imm$Total.Volume-volume_imm$Electronic.Volu
me
head(volume_imm,3)
```

```
##         Date Commodity.Indicator Product.Short.Desc Future.Option
## 1 01/01/2000                  ED       EURO DLR FUT             F
```

```
## 4 01/01/2000                  ED      EURO DLR CALL             O
## 5 01/01/2000                  ED       EURO DLR PUT             O
##   Electronic.Volume Total.Volume Floor.Volume
## 1            232796      8379642      8146846
## 4              6600       972970       966370
## 5              6303       942420       936117

# aggregate upon derivative types.
imm_volume_0 <- aggregate(cbind(Electronic.Volume, Floor.Volume,
Total.Volume)~Date+Commodity.Indicator, data=volume_imm, sum)
head(imm_volume_0,3)

##         Date Commodity.Indicator Electronic.Volume Floor.Volume
## 1 02/01/2011                 16E                11            0
## 2 03/01/2011                 16E                24            0
## 3 04/01/2011                 16E                85            0
##   Total.Volume
## 1           11
## 2           24
## 3           85

dim(imm_volume_0)

## [1] 5723    5

# drop any data before date 1/1/2001
imm_volume <- imm_volume_0[!as.Date(imm_volume_0$Date, "%m/%d/%Y") < as.Date(
"2001-01-01"),]
head(imm_volume,3)

##         Date Commodity.Indicator Electronic.Volume Floor.Volume
## 1 02/01/2011                 16E                11            0
## 2 03/01/2011                 16E                24            0
## 3 04/01/2011                 16E                85            0
##   Total.Volume
## 1           11
## 2           24
## 3           85

dim(imm_volume)

## [1] 5539    5

# aggregate IMM based on Date
imm_volume<-aggregate(cbind(Electronic.Volume, Floor.Volume, Total.Volume)~Da
te, data=imm_volume,sum)
head(imm_volume,3)

##         Date Electronic.Volume Floor.Volume Total.Volume
## 1 01/01/2001            619988     22520720     23140708
## 2 01/01/2002           1069075     29434061     30503136
## 3 01/01/2003           1383632     18505884     19889516
```

```r
# order by date
imm_volume <- imm_volume[order(as.Date(imm_volume$Date, "%m/%d/%Y")),]
head(imm_volume,3)
```

```
##          Date Electronic.Volume Floor.Volume Total.Volume
## 1  01/01/2001            619988     22520720     23140708
## 14 02/01/2001            554640     17537808     18092448
## 27 03/01/2001            730558     21537410     22267968
```

```r
# combine trading volume with seat price.
imm_seat_price <- seat_price[[toupper(seat)]]
head(imm_seat_price)
```

```
## [1] 183125.0 225000.0 292500.0 298750.0 305000.0 355333.3
```

```r
imm_volume$Seat.Price <- imm_seat_price
head(imm_volume,3)
```

```
##          Date Electronic.Volume Floor.Volume Total.Volume Seat.Price
## 1  01/01/2001            619988     22520720     23140708     183125
## 14 02/01/2001            554640     17537808     18092448     225000
## 27 03/01/2001            730558     21537410     22267968     292500
```

Split data into train and test

```r
imm_train <- imm_volume[as.Date(imm_volume$Date,"%m/%d/%Y") < as.Date("2013-01-01") ,]
imm_test <- imm_volume[!as.Date(imm_volume$Date,"%m/%d/%Y") < as.Date("2013-01-01") ,]
# create an empty list to store forecast values
seat_price_forecast_imm <- list()
```

2.1 Linear regression

```r
imm_lm <- lm(Seat.Price~Electronic.Volume+Floor.Volume, data=imm_train)
summary(imm_lm)
```

```
##
## Call:
## lm(formula = Seat.Price ~ Electronic.Volume + Floor.Volume, data = imm_train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -239141  -73553   -7386   47466  453221
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)       1.235e+05  3.584e+04   3.445 0.000752 ***
## Electronic.Volume 3.761e-03  4.276e-04   8.794 4.54e-15 ***
## Floor.Volume      8.460e-03  1.407e-03   6.014 1.48e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 119600 on 141 degrees of freedom
## Multiple R-squared:  0.3719, Adjusted R-squared:  0.363
## F-statistic: 41.75 on 2 and 141 DF,  p-value: 5.736e-15
```

2.2 Linear regression with ARMA errors (use arima with xreg)

```
imm_lm_arma_errors <- auto.arima(imm_train$Seat.Price, xreg=imm_train[,c(2,3)
], allowdrift = FALSE)
summary(imm_lm_arma_errors)
```

```
## Series: imm_train$Seat.Price
## Regression with ARIMA(0,0,0) errors
##
## Coefficients:
##       Electronic.Volume  Floor.Volume
##                  0.0048        0.0127
## s.e.             0.0003        0.0007
##
## sigma^2 estimated as 1.54e+10:  log likelihood=-1892.26
## AIC=3790.51   AICc=3790.68   BIC=3799.42
##
## Training set error measures:
##                     ME     RMSE      MAE       MPE     MAPE     MASE
## Training set 9546.991 123219.7 91730.28 -2.75446 22.19796 3.748694
##                    ACF1
## Training set 0.6110219
```

```
seat_price_forecast_imm$lm_arma_errors <- predict(imm_lm_arma_errors, n.ahead
=12, newxreg = imm_test[,c(2,3)])$mean
```

2.3 ARIMA

```
imm_arima <- auto.arima(imm_train$Seat.Price, allowdrift = FALSE)
summary(imm_arima)
```

```
## Series: imm_train$Seat.Price
## ARIMA(0,1,1)
##
## Coefficients:
##          ma1
##       0.4417
## s.e.  0.0768
##
## sigma^2 estimated as 1.233e+09:  log likelihood=-1699.18
## AIC=3402.36   AICc=3402.45   BIC=3408.29
##
## Training set error measures:
##                      ME     RMSE      MAE        MPE     MAPE      MASE
## Training set -83.58633 34863.45 22949.52 -0.2361661 5.862202 0.9378663
##                     ACF1
## Training set 0.01505295
```

```r
seat_price_forecast_imm$arima <- forecast(imm_arima,h=12)$mean
```

2.4 Seasonal ARIMA (SARIMA)

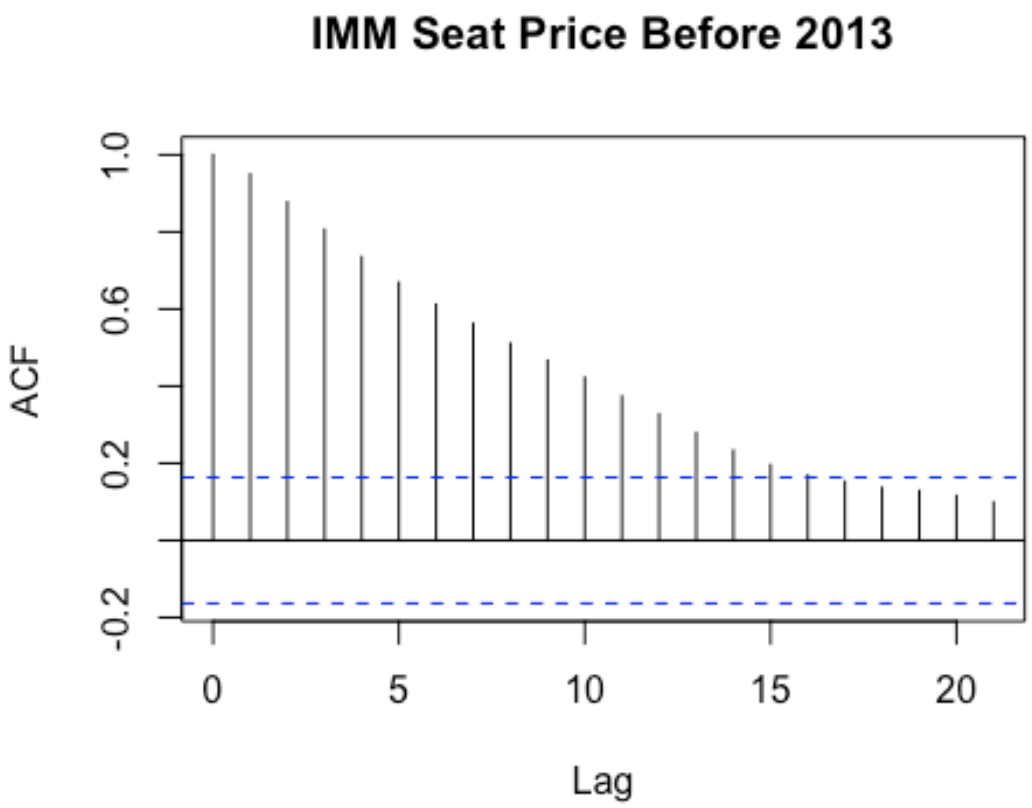
```r
imm_sarima <- auto.arima(ts(imm_train$Seat.Price,frequency = 12), allowdrift
= FALSE)
summary(imm_sarima)
```

```
## Series: ts(imm_train$Seat.Price, frequency = 12)
## ARIMA(0,1,1)
##
## Coefficients:
##          ma1
##       0.4417
## s.e.  0.0768
##
## sigma^2 estimated as 1.233e+09:  log likelihood=-1699.18
## AIC=3402.36   AICc=3402.45   BIC=3408.29
##
## Training set error measures:
##                     ME     RMSE      MAE        MPE     MAPE      MASE
## Training set -83.58633 34863.45 22949.52 -0.2361661 5.862202 0.1951832
##                     ACF1
## Training set 0.01505295
```

```r
seat_price_forecast_imm$sarima <- forecast(imm_sarima , h=12)$mean
```

2.5 Fractional ARIMA (ARFIMA)

```r
acf(imm_train$Seat.Price, main = "IMM Seat Price Before 2013")
```

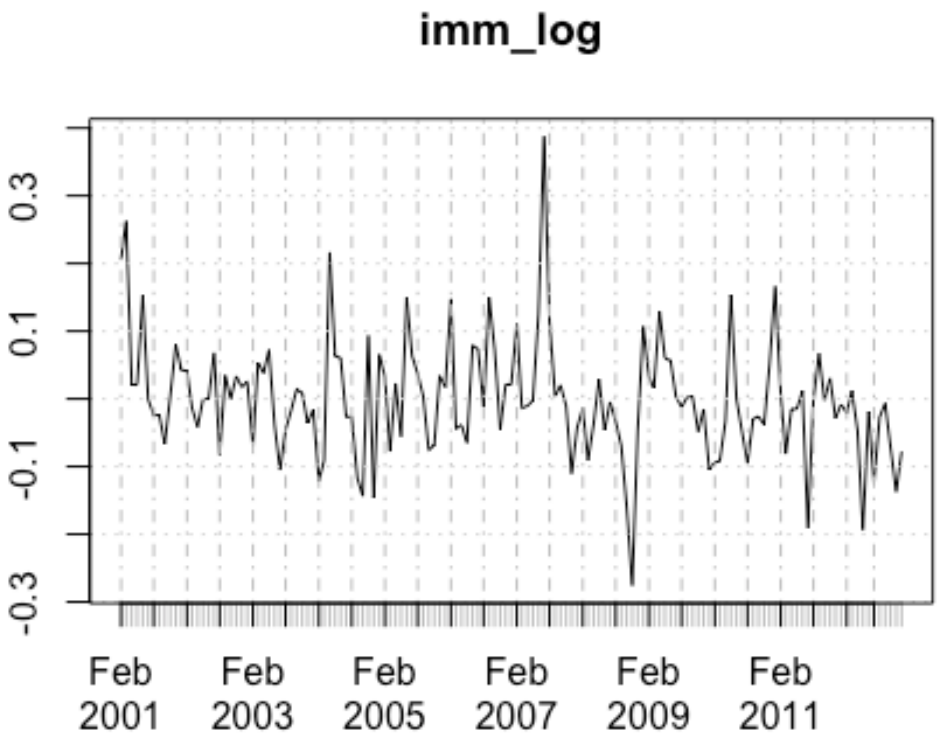



```
imm_arfima <- arfima(imm_train$Seat.Price)
summary(imm_arfima)

##
## Call:
##   arfima(y = imm_train$Seat.Price)
##
## Coefficients:
##        Estimate Std. Error z value Pr(>|z|)
## d       0.03585    0.06041   0.593    0.553
## ar.ar1  0.94468    0.15868   5.953 2.63e-09 ***
## ma.ma1 -0.43898    0.03266 -13.440  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## sigma[eps] = 34603.46
## [d.tol = 0.0001221, M = 100, h = 1.802e-05]
## Log likelihood: -1709 ==> AIC = 3426.743 [4 deg.freedom]

seat_price_forecast_imm$arfima <- forecast(imm_arfima, h=12)$mean
```

2.6 ARMA and GARCH combination

```
imm_seat_price_xts <- xts(imm_train$Seat.Price, order.by = as.Date(imm_train$
Date, "%m/%d/%Y"))
imm_log <- log(imm_seat_price_xts)
```

```r
# in order to make data stationary, need to take first difference of imm_log
imm_log <- diff(imm_log)[-1]
plot(imm_log)
```




```r
# find p,q
imm_arma <- auto.arima(imm_log)
summary(imm_arma)
```

```
## Series: imm_log
## ARIMA(1,1,1)
##
## Coefficients:
##          ar1      ma1
##       0.3045  -0.9694
## s.e.  0.0858   0.0282
##
## sigma^2 estimated as 0.006914:  log likelihood=151.58
## AIC=-297.16   AICc=-296.98   BIC=-288.29
##
## Training set error measures:
##                       ME       RMSE        MAE MPE MAPE      MASE
## Training set -0.01429273 0.08227589 0.05985959 NaN  Inf 0.7920005
```

```
##                      ACF1
## Training set -0.01195338

# from summary results, we will choose p=0, q=2
imm_garch <- garchFit(~arma(0,2) + garch(1,1), data=imm_log, cond.dist = "std
", trace=F)
summary(imm_garch)

##
## Title:
##  GARCH Modelling
##
## Call:
##  garchFit(formula = ~arma(0, 2) + garch(1, 1), data = imm_log,
##     cond.dist = "std", trace = F)
##
## Mean and Variance Equation:
##  data ~ arma(0, 2) + garch(1, 1)
## <environment: 0x7f8b495e8a68>
##  [data = imm_log]
##
## Conditional Distribution:
##  std
##
## Coefficient(s):
##          mu          ma1          ma2        omega       alpha1
## -0.00520074   0.24340748   0.02667907   0.75169186   0.00000001
##       beta1        shape
##  0.93755904   2.00038512
##
## Std. Errors:
##  based on Hessian
##
## Error Analysis:
##          Estimate  Std. Error    t value Pr(>|t|)
## mu     -5.201e-03   6.399e-03 -8.130e-01   0.4164
## ma1     2.434e-01   6.217e-02  3.915e+00 9.03e-05 ***
## ma2     2.668e-02   7.117e-02  3.750e-01   0.7078
## omega   7.517e-01   4.153e-01  1.810e+00   0.0703 .
## alpha1  1.000e-08   4.564e+01  0.000e+00   1.0000
## beta1   9.376e-01   1.914e-02  4.898e+01  < 2e-16 ***
## shape   2.000e+00   2.470e-09  8.100e+08  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Log Likelihood:
##  173.5959    normalized:  1.213958
##
## Description:
##  Fri Dec  1 21:42:38 2017 by user:
```

```
##
##
## Standardised Residuals Tests:
##                                 Statistic p-Value
##  Jarque-Bera Test   R    Chi^2  68.38173  1.44329e-15
##  Shapiro-Wilk Test  R    W      0.9453694 2.157733e-05
##  Ljung-Box Test     R    Q(10)  8.713318  0.5595097
##  Ljung-Box Test     R    Q(15)  15.49319  0.4165094
##  Ljung-Box Test     R    Q(20)  20.26966  0.4411787
##  Ljung-Box Test     R^2  Q(10)  4.586855  0.9170157
##  Ljung-Box Test     R^2  Q(15)  6.671226  0.9661393
##  Ljung-Box Test     R^2  Q(20)  14.23188  0.8185472
##  LM Arch Test       R    TR^2   6.608638  0.8823567
##
## Information Criterion Statistics:
##       AIC       BIC       SIC      HQIC
## -2.330013 -2.184979 -2.334514 -2.271078
```

```
# forecast value
forecast_imm_log_garch <- predict(imm_garch, n.ahead=12)$meanForecast
# compute the price from forecast_cme_log_garch

seat_price_forecast_imm$garch <- as.numeric(tail(imm_seat_price_xts,1)*exp(cu
msum(forecast_imm_log_garch)))
```

Task B

```
smape <- function(fitted, actual) {
          return(2*mean(abs(fitted - actual) / (abs(fitted) + abs(actual))))
          }
imm_smape <- mapply(smape, seat_price_forecast_imm, list(imm_test$Seat.Price)
)
imm_smape
```

```
##     arima    sarima    arfima     garch
## 0.1829424 0.1829424 0.1347388 0.2192750
```

```
imm_smape[which.min(imm_smape)]
```

```
##    arfima
## 0.1347388
```

From sMAPE results, the Fractional ARIMA model is the best one to forecast monthly prices for IMM seat classes since ARFIMA model has the smallest sMAPE thus the best.

3. IOM

Task A

```
# IOM commodity subset
seat <- "IOM"
commodity_iom <- c(as.character(classification_data$Commodity.Code[classifica
tion_data$Division=="IOM"]))
```

```r
commodity_iom <- unique(commodity_iom)
head(commodity_iom,30)
```

```
##  [1] "12" "13" "48" "56" "62" "1#" "1A" "1B" "1C" "1F" "1J" "1K" "1P" "1S"
## [15] "1T" "1U" "1X" "1Y" "1Z" "2#" "2A" "2B" "2C" "2D" "2J" "2K" "2P" "2S"
## [29] "2T" "2U"
```

```r
# filter out tradable commodities
tradable_commodity_iom <- is.element(volume_data$Commodity.Indicator,commodit
y_iom)
head(tradable_commodity_iom,3)
```

```
## [1] TRUE TRUE TRUE
```

```r
volume_iom <- volume_data[tradable_commodity_iom,]
head(volume_iom,3)
```

```
##         Date Commodity.Indicator Product.Short.Desc Future.Option
## 1 01/01/2000                  ED       EURO DLR FUT             F
## 2 01/01/2000                  SP        S&P 500 FUT             F
## 3 01/01/2000                  ES      EMINI S&P FUT             F
##   Electronic.Volume Total.Volume
## 1           232,796      8379642
## 2            88,426      1915082
## 3         1,302,447      1305618
```

```r
# floor volume
volume_iom$Electronic.Volume <- as.numeric(gsub(",","",volume_iom$Electronic.
Volume,fixed=TRUE))
volume_iom$Floor.Volume <- volume_iom$Total.Volume-volume_iom$Electronic.Volu
me
head(volume_iom,3)
```

```
##         Date Commodity.Indicator Product.Short.Desc Future.Option
## 1 01/01/2000                  ED       EURO DLR FUT             F
## 2 01/01/2000                  SP        S&P 500 FUT             F
## 3 01/01/2000                  ES      EMINI S&P FUT             F
##   Electronic.Volume Total.Volume Floor.Volume
## 1            232796      8379642      8146846
## 2             88426      1915082      1826656
## 3           1302447      1305618         3171
```

```r
# aggregate upon derivative types.
iom_volume_0 <- aggregate(cbind(Electronic.Volume, Floor.Volume,
Total.Volume)~Date+Commodity.Indicator, data=volume_iom, sum)
head(iom_volume_0,3)
```

```
##         Date Commodity.Indicator Electronic.Volume Floor.Volume
## 1 01/01/2008                  1A                0           25
## 2 01/01/2013                  1A             1095           20
## 3 02/01/2012                  1A              388            0
##   Total.Volume
```

```
## 1           25
## 2         1115
## 3          388
```

```
# drop any data before date 1/1/2001
iom_volume <- iom_volume_0[!as.Date(iom_volume_0$Date, "%m/%d/%Y") < as.Date(
"2001-01-01"),]
head(iom_volume,3)
```

```
##         Date Commodity.Indicator Electronic.Volume Floor.Volume
## 1 01/01/2008                 1A                 0           25
## 2 01/01/2013                 1A              1095           20
## 3 02/01/2012                 1A               388            0
##   Total.Volume
## 1           25
## 2         1115
## 3          388
```

```
# aggregate IOM based on Date
iom_volume<-aggregate(cbind(Electronic.Volume, Floor.Volume, Total.Volume)~Da
te, data=iom_volume,sum)
head(iom_volume,3)
```

```
##         Date Electronic.Volume Floor.Volume Total.Volume
## 1 01/01/2001           4765408     26755761     31521169
## 2 01/01/2002           9805903     34264048     44069951
## 3 01/01/2003          19835643     23001043     42836686
```

```
# order by date
iom_volume <- iom_volume[order(as.Date(iom_volume$Date, "%m/%d/%Y")),]
head(iom_volume,3)
```

```
##          Date Electronic.Volume Floor.Volume Total.Volume
## 1  01/01/2001           4765408     26755761     31521169
## 14 02/01/2001           4787364     21688980     26476344
## 27 03/01/2001           6623579     27416016     34039595
```

```
# combine trading volume with seat price.
iom_seat_price <- seat_price[[toupper(seat)]]
head(iom_seat_price)
```

```
## [1] 130000 170000 242500 291500 275750 260000
```

```
iom_volume$Seat.Price <- iom_seat_price
head(iom_volume,3)
```

```
##          Date Electronic.Volume Floor.Volume Total.Volume Seat.Price
## 1  01/01/2001           4765408     26755761     31521169     130000
## 14 02/01/2001           4787364     21688980     26476344     170000
## 27 03/01/2001           6623579     27416016     34039595     242500
```

Split data into train and test

```
# Partion the training and testing data based on the date of trading.
# training data: data with date before year 2013
# testing data: data with date in year 2013
iom_train <- iom_volume[as.Date(iom_volume$Date,"%m/%d/%Y") < as.Date("2013-01-01") ,]
iom_test <- iom_volume[!as.Date(iom_volume$Date,"%m/%d/%Y") < as.Date("2013-01-01") ,]
# create an empty list to store forecast values
seat_price_forecast_iom <- list()
```

3.1 Linear regression

```
iom_lm <- lm(Seat.Price~Electronic.Volume+Floor.Volume, data=iom_train)
summary(iom_lm)
```

```
##
## Call:
## lm(formula = Seat.Price ~ Electronic.Volume + Floor.Volume, data = iom_train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -167073  -73531  -26641   53026  441262
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)       6.018e+04  3.463e+04   1.738   0.0844 .
## Electronic.Volume 3.927e-04  1.861e-04   2.110   0.0366 *
## Floor.Volume      5.491e-03  1.083e-03   5.071 1.23e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 100200 on 141 degrees of freedom
## Multiple R-squared:  0.1578, Adjusted R-squared:  0.1458
## F-statistic: 13.21 on 2 and 141 DF,  p-value: 5.537e-06
```

```
seat_price_forecast_iom$lm <- predict(iom_lm, iom_test)
```

3.2 Linear regression with ARMA errors (use arima with xreg)

```
iom_lm_arma_errors <- auto.arima(iom_train$Seat.Price, xreg=iom_train[,c(2,3)], allowdrift = FALSE)
summary(iom_lm_arma_errors)
```

```
## Series: iom_train$Seat.Price
## Regression with ARIMA(0,0,0) errors
##
## Coefficients:
##       Electronic.Volume  Floor.Volume
##                    6e-04        0.0072
## s.e.               2e-04        0.0005
##
```

```
## sigma^2 estimated as 1.017e+10:  log likelihood=-1862.42
## AIC=3730.85   AICc=3731.02   BIC=3739.76
##
## Training set error measures:
##                    ME   RMSE     MAE       MPE     MAPE     MASE      ACF1
## Training set 3496.334 100164 77969.5 -16.07025 39.36036 4.640622 0.8024459
```

3.3 ARIMA

```
iom_arima <- auto.arima(iom_train$Seat.Price, allowdrift = FALSE)
summary(iom_arima)
```

```
## Series: iom_train$Seat.Price
## ARIMA(0,1,1)
##
## Coefficients:
##          ma1
##       0.3028
## s.e.  0.0825
##
## sigma^2 estimated as 697499902:  log likelihood=-1658.41
## AIC=3320.82   AICc=3320.91   BIC=3326.75
##
## Training set error measures:
##                      ME     RMSE      MAE        MPE     MAPE      MASE
## Training set -360.7213 26226.18 15843.39 -0.7418089 7.064388 0.9429735
##                      ACF1
## Training set -0.01440655
```

```
seat_price_forecast_iom$arima <- forecast(iom_arima,h=12)$mean
```

3.4 Seasonal ARIMA (SARIMA)

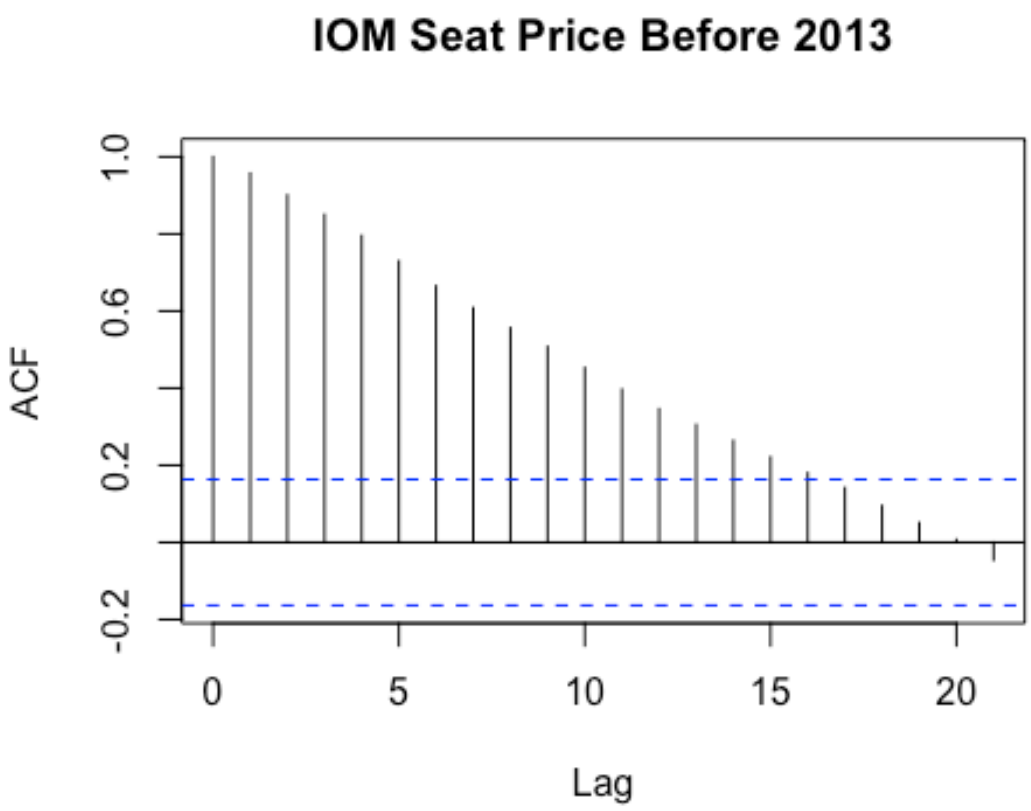
```
iom_sarima <- auto.arima(ts(iom_train$Seat.Price,frequency = 12), allowdrift
= FALSE)
summary(iom_sarima)
```

```
## Series: ts(iom_train$Seat.Price, frequency = 12)
## ARIMA(0,1,1)(0,0,1)[12]
##
## Coefficients:
##          ma1     sma1
##       0.2997  -0.1322
## s.e.  0.0833   0.0852
##
## sigma^2 estimated as 689493117:  log likelihood=-1657.18
## AIC=3320.37   AICc=3320.54   BIC=3329.26
##
## Training set error measures:
##                      ME     RMSE      MAE        MPE     MAPE      MASE
## Training set -377.2664 25983.24 15709.39 -0.866572 7.043824 0.1808763
##                      ACF1
## Training set -0.01406828
```

```
seat_price_forecast_iom$sarima <- forecast(iom_sarima , h=12)$mean
```

3.5 Fractional ARIMA (ARFIMA)

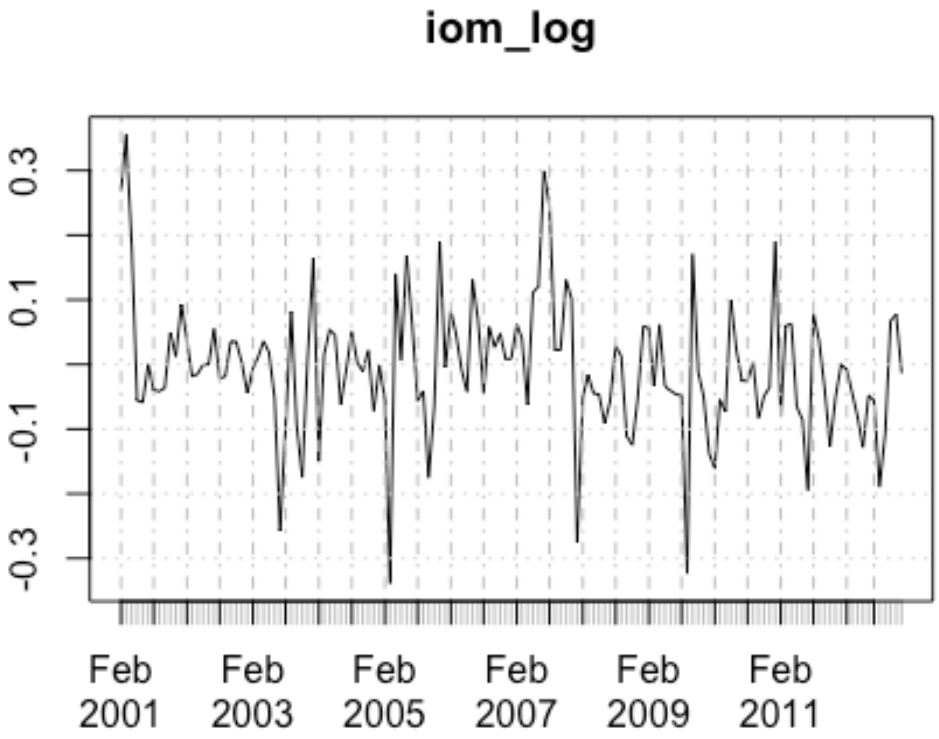
```
acf(iom_train$Seat.Price, main = "IOM Seat Price Before 2013")
```




```
iom_arfima <- arfima(iom_train$Seat.Price)
summary(iom_arfima)

##
## Call:
##   arfima(y = iom_train$Seat.Price)
##
## Coefficients:
##        Estimate Std. Error z value Pr(>|z|)
## d       0.03855    0.03120   1.235  0.21665
## ar.ar1  1.19938    0.08587  13.967  < 2e-16 ***
## ar.ar2 -0.24362    0.08444  -2.885  0.00391 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## sigma[eps] = 26205.18
## [d.tol = 0.0001221, M = 100, h = 1.76e-05]
## Log likelihood: -1669 ==> AIC = 3346.682 [4 deg.freedom]

seat_price_forecast_iom$arfima <- forecast(iom_arfima, h=12)$mean
```

3.6 ARMA and GARCH combination

```
iom_seat_price_xts <- xts(iom_train$Seat.Price, order.by = as.Date(iom_train$
Date, "%m/%d/%Y"))
iom_log <- log(iom_seat_price_xts)
# in order to make data stationary, need to take first difference of iom_log
iom_log <- diff(iom_log)[-1]
plot(iom_log)
```




```
# find p,q
iom_arma <- auto.arima(iom_log)
summary(iom_arma)
```

```
## Series: iom_log
## ARIMA(0,0,1) with zero mean
##
## Coefficients:
##          ma1
##       0.2545
## s.e.  0.0798
##
## sigma^2 estimated as 0.0101:  log likelihood=126.14
## AIC=-248.28   AICc=-248.19   BIC=-242.35
##
```

```
## Training set error measures:
##                       ME      RMSE        MAE MPE MAPE      MASE
## Training set -0.004016066 0.1001315 0.07078396 Inf  Inf 0.7964714
##                      ACF1
## Training set -0.004951168
```

```
# from summary results, we will choose p=2, q=0
iom_garch <- garchFit(~arma(2,0) + garch(1,1), data=iom_log, cond.dist = "std
", trace=F)
summary(iom_garch)
```

```
##
## Title:
##  GARCH Modelling
##
## Call:
##  garchFit(formula = ~arma(2, 0) + garch(1, 1), data = iom_log,
##     cond.dist = "std", trace = F)
##
## Mean and Variance Equation:
##  data ~ arma(2, 0) + garch(1, 1)
## <environment: 0x7f8b45d89540>
##  [data = iom_log]
##
## Conditional Distribution:
##  std
##
## Coefficient(s):
##         mu         ar1         ar2       omega      alpha1       beta1
## -0.0076529   0.2771151  -0.1130667   0.0028510   0.3862317   0.5838514
##      shape
##  2.6862100
##
## Std. Errors:
##  based on Hessian
##
## Error Analysis:
##         Estimate  Std. Error  t value Pr(>|t|)
## mu     -0.007653    0.005926   -1.291  0.19654
## ar1     0.277115    0.085677    3.234  0.00122 **
## ar2    -0.113067    0.070787   -1.597  0.11020
## omega   0.002851    0.002790    1.022  0.30689
## alpha1  0.386232    0.360624    1.071  0.28417
## beta1   0.583851    0.195253    2.990  0.00279 **
## shape   2.686210    0.837810    3.206  0.00134 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Log Likelihood:
##  148.5329    normalized:  1.038691
```

```
##
## Description:
##  Fri Dec  1 21:42:42 2017 by user:
##
##
## Standardised Residuals Tests:
##                                 Statistic p-Value
##  Jarque-Bera Test   R    Chi^2  94.65229  0
##  Shapiro-Wilk Test  R    W      0.9285878 1.328659e-06
##  Ljung-Box Test     R    Q(10)  4.307938  0.932388
##  Ljung-Box Test     R    Q(15)  10.46502  0.7895149
##  Ljung-Box Test     R    Q(20)  25.34871  0.1884145
##  Ljung-Box Test     R^2  Q(10)  2.639922  0.9886761
##  Ljung-Box Test     R^2  Q(15)  4.690286  0.9944397
##  Ljung-Box Test     R^2  Q(20)  16.08586  0.7112834
##  LM Arch Test       R    TR^2   4.710597  0.9669535
##
## Information Criterion Statistics:
##       AIC       BIC       SIC      HQIC
## -1.979481 -1.834446 -1.983982 -1.920546
```

Task B (sMAPE)

```
smape <- function(fitted, actual) {
          return(2*mean(abs(fitted - actual) / (abs(fitted) + abs(actual))))
        }
iom_smape <- mapply(smape, seat_price_forecast_iom, list(iom_test$Seat.Price)
)
iom_smape

##        lm     arima    sarima    arfima
## 0.9694257 0.1539462 0.1815043 0.5300529

iom_smape[which.min(iom_smape)]

##     arima
## 0.1539462
```

From sMAPE results, the ARIMA model is the best one to forecast monthly prices for IOM seat classes since the ARIMA combination model has the smallest sMAPE.