ARMA-X Analysis

Contents

Data

```r
# 1. Load Political Social Media

#contains posts from Twitter & TruthSocial
social <- read.csv(here("data/mothership", "social.csv"))

# 2. Load Financial

#S&P500
SPY <- read.csv(here("data/mothership", "SPY.csv"))

#STOXX50
VGK <- read.csv(here("data/mothership", "VGK.csv"))

#CSI 300 (China)
ASHR <- read.csv(here("data/mothership", "ASHR.CSV"))


#temporary while we figure out mothership
names(SPY) = gsub(pattern = "SPY*",
                         replacement = "", x = names(SPY))
```

```r
#make posixct
SPY$timestamp = as.POSIXct(SPY$timestamp,format = "%Y-%m-%d %H:%M:%S")
VGK$timestamp = as.POSIXct(VGK$timestamp,format = "%Y-%m-%d %H:%M:%S")
ASHR$timestamp = as.POSIXct(ASHR$timestamp,format = "%Y-%m-%d %H:%M:%S")
social$timestamp = as.POSIXct(social$timestamp,format = "%Y-%m-%d %H:%M:%S")
```

Volatility Calculation

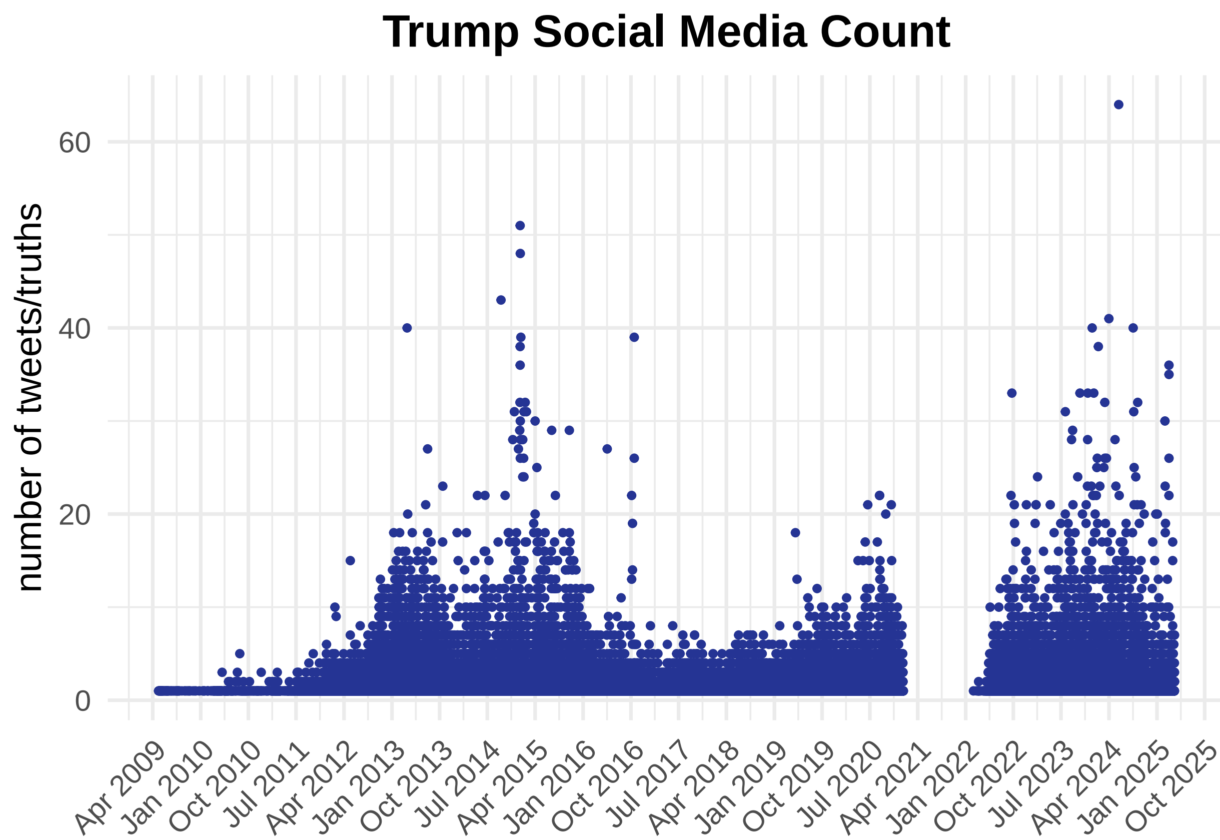
```r
#find hourly volatility
#NOTE: this ignores tweets made outside trading hours!!
SPY_volatility_alltime = r.vol_hourly(SPY,merge=F)

#select time period
SPY_volatility = filter(SPY_volatility_alltime,
                  between(timestamp,
                          as.Date('2019-01-01'),
                          as.Date('2025-04-10')))
colnames(SPY_volatility)[1] <- "timestamp_hour"
```

Social Media Variables

Count Number of Posts

```r
#convert to datatable
social = as.data.table(social)

#count by hour
tweet_count = social[, .N, by=.(year(timestamp), month(timestamp),
                                day(timestamp), hour(timestamp))]

#fix timestamp
tweet_count$timestamp = as.POSIXct(sprintf("%04d-%02d-%02d %02d:00:00",
                         tweet_count$year, tweet_count$month, tweet_count$day,
                         tweet_count$hour), format = "%Y-%m-%d %H:00:00")

#remove useless columns and reorder by oldest first
tweet_count = dplyr::select(tweet_count, timestamp, N)
tweet_count = tweet_count[ order(tweet_count$timestamp , decreasing = F ),]

#plot
ggplot(tweet_count, aes(x = timestamp, y = N)) +
    geom_point(color = "#253494", size = 1) +
    scale_x_datetime(date_labels = "%b %Y", date_breaks = "9 month") +
    labs(title = "Trump Social Media Count",
         x = NULL,
         y = "number of tweets/truths") +
    theme_minimal(base_size = 14) +
    theme(axis.text.x = element_text(angle = 45, hjust = 1),
          plot.title = element_text(face = "bold", hjust = 0.5))
```

```
## Warning: Removed 1 row containing missing values or values outside the scale range
## (`geom_point()`).
```



Dummy for Social Media Post

```
tweet_count = tweet_count %>% mutate(dummy = if_else(N > 0, 1, 0))
```

ARMA-X Models

Tweet Count on Volatility by hour

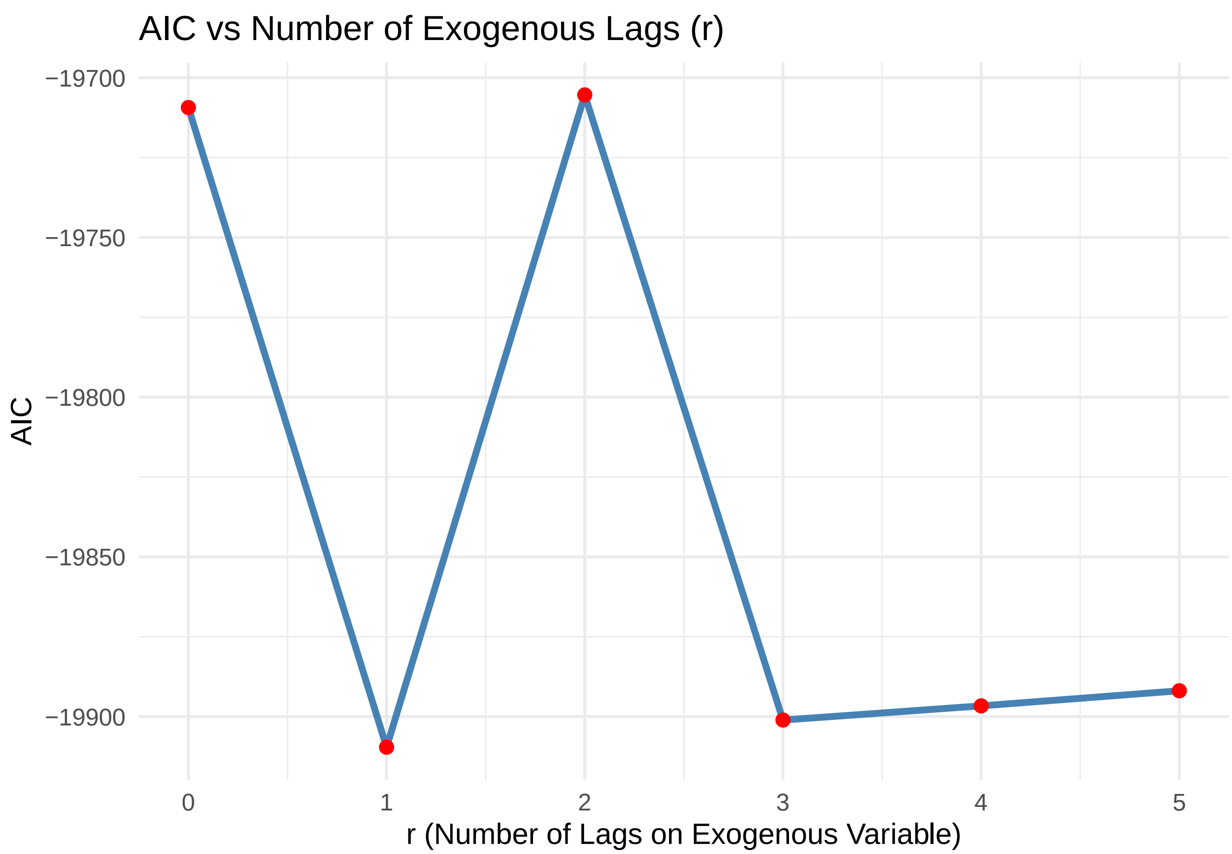
```
#take all relevant data for armax
countvol = merge(SPY_volatility, tweet_count, by.x = "timestamp_hour",
                 by.y = "timestamp", all.x = T)

#NA tweets means no tweets
countvol$N[is.na(countvol$N)] = 0



#find best armax model and fit
armax_fit <- select_armax(countvol$r_vol_h, countvol$N,
                          max_p = 5, max_q = 5, max_r = 5, criterion = "AIC")
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```r
summary(armax_fit$model)
```

```
## Series: y_trimmed
## Regression with ARIMA(4,0,5) errors
##
## Coefficients:
##          ar1     ar2     ar3      ar4     ma1      ma2      ma3     ma4     ma5
##       0.0817  1.7072  0.0182  -0.8094  0.2524  -1.6998  -0.5889  0.8488  0.2425
## s.e.  0.0186  0.0138  0.0176   0.0139  0.0216   0.0116   0.0299  0.0111  0.0152
##       intercept  X1_Lag_0  X1_Lag_1
##          0.0343    -5e-04    -4e-04
## s.e.     0.0244     4e-04     4e-04
##
## sigma^2 = 0.009626:  log likelihood = 9967.8
## AIC=-19909.6   AICc=-19909.57   BIC=-19814.58
##
## Training set error measures:
##                        ME       RMSE        MAE       MPE     MAPE     MASE
## Training set 0.0008183625 0.09805822 0.02191816 -73.69928 116.6322 1.096845
##                      ACF1
## Training set -0.004616607
```

```r
armax_fit$ICplot
```




```
armax_fit$params
```

```
## $p
## [1] 4
##
## $q
## [1] 5
##
## $r
## [1] 1
```

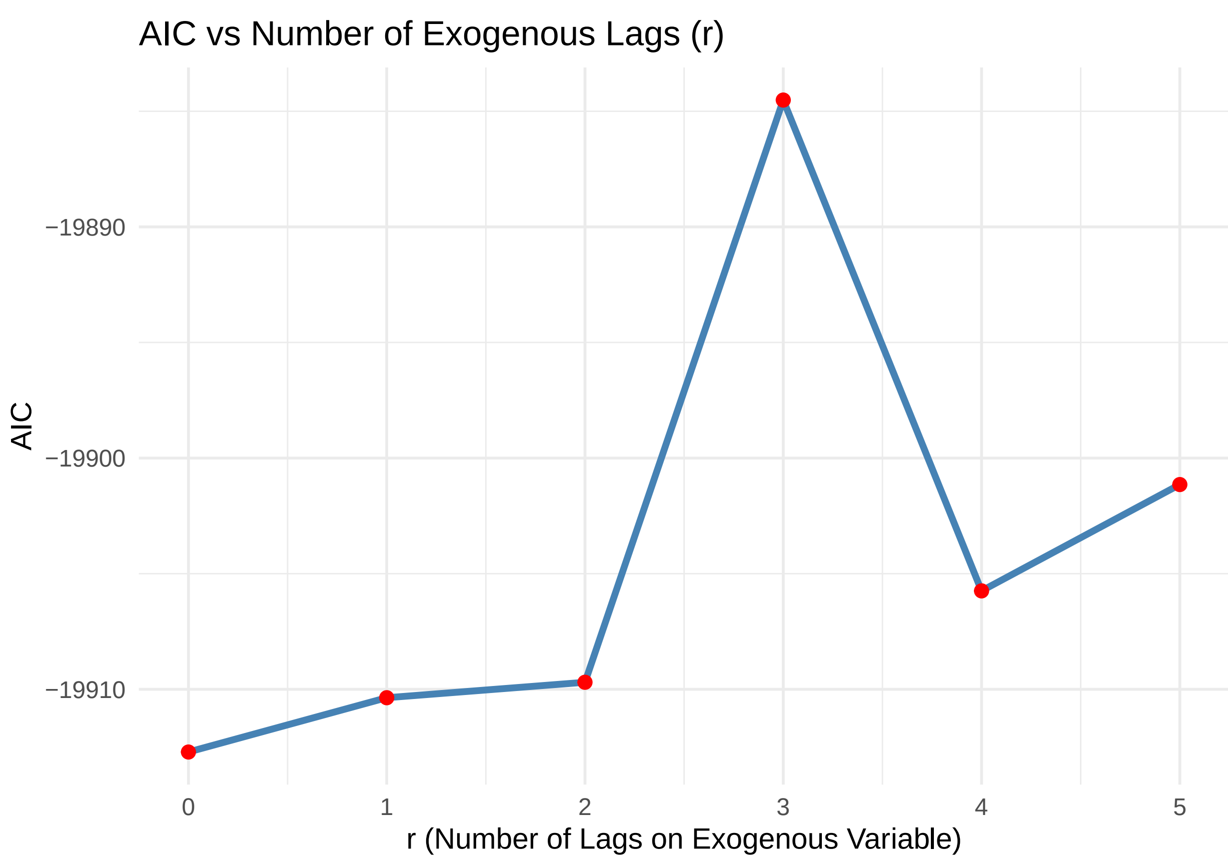
Tweet Dummy on Volatility by hour

```
#NA tweets means no tweets
countvol$dummy[is.na(countvol$dummy)] = 0

#find best armax model and fit
armax_fit <- select_armax(countvol$r_vol_h, countvol$dummy,
                          max_p = 5, max_q = 5, max_r = 5, criterion = "AIC")

summary(armax_fit$model)
```

```
## Series: y_trimmed
```

```
## Regression with ARIMA(4,0,5) errors
##
## Coefficients:
##          ar1     ar2     ar3      ar4     ma1      ma2      ma3     ma4     ma5
##       0.0834  1.7079  0.0162  -0.8098  0.2507  -1.7008  -0.5870  0.8498  0.2421
## s.e.  0.0184  0.0137  0.0175   0.0138  0.0214   0.0114   0.0298  0.0109  0.0152
##       intercept  X1_Lag_0
##          0.0332   -0.0002
## s.e.     0.0243    0.0018
##
## sigma^2 = 0.009626:  log likelihood = 9968.36
## AIC=-19912.71   AICc=-19912.68   BIC=-19825
##
## Training set error measures:
##                        ME       RMSE        MAE       MPE     MAPE    MASE
## Training set 0.0008330655 0.09806133 0.02188351 -74.27044 116.0456 1.09518
##                     ACF1
## Training set -0.00458639
```

```
armax_fit$ICplot
```




```
armax_fit$params
```

```
## $p
## [1] 4
```

```
##
## $q
## [1] 5
##
## $r
## [1] 0
```

```r
nb.lags <- 3 #r
count_lags <- embed(countvol_day$N, nb.lags + 1)
dummy_lags <- embed(countvol_day$dummy, nb.lags + 1)
colnames(count_lags) <- paste0("Lag_", 0:nb.lags)

#align volatility to match count rows (for lag)
vol_aligned <- tail(countvol_day$r_vol_d, nrow(count_lags))

#choosing how many lags
# fit an ARMA(0,0,0) model with lm (with r set above)
eq <- lm(vol_aligned ~ count_lags)
eq2 <- lm(vol_aligned ~ dummy_lags)

#compute Newey-West HAC standard errors
var.cov.mat <- NeweyWest(eq, lag = 7, prewhite = FALSE)
robust_se <- sqrt(diag(var.cov.mat))
#for both
var.cov.matD <- NeweyWest(eq2, lag = 7, prewhite = FALSE)
robust_seD <- sqrt(diag(var.cov.matD))

#output table; significant lags are how many we choose
stargazer(eq, eq, type = "text",
          column.labels = c("(no HAC)", "(HAC)"), keep.stat = "n",
          se = list(NULL, robust_se), no.space = TRUE)

#output table; significant lags are how many we choose
stargazer(eq2, eq2, type = "text",
          column.labels = c("(no HAC)", "(HAC)"), keep.stat = "n",
          se = list(NULL, robust_se), no.space = TRUE)
```