ARMA-X Analysis

Contents

Data

Raw Data

```
# 1. Political

#truthsocial
raw_truths <- read.csv(here("data/political_data", "truths_new.csv"))

#twitter
raw_tweets <- read.csv(here("data/political_data", "tweets.csv"))


# 2. Financial

#S&P500
data_loader(symbol="SPY")

#STOXX50
data_loader(symbol="VGK")

#CSI 300 (China)
data_loader(symbol="ASHR")
```

Tweet Cleanup & Count

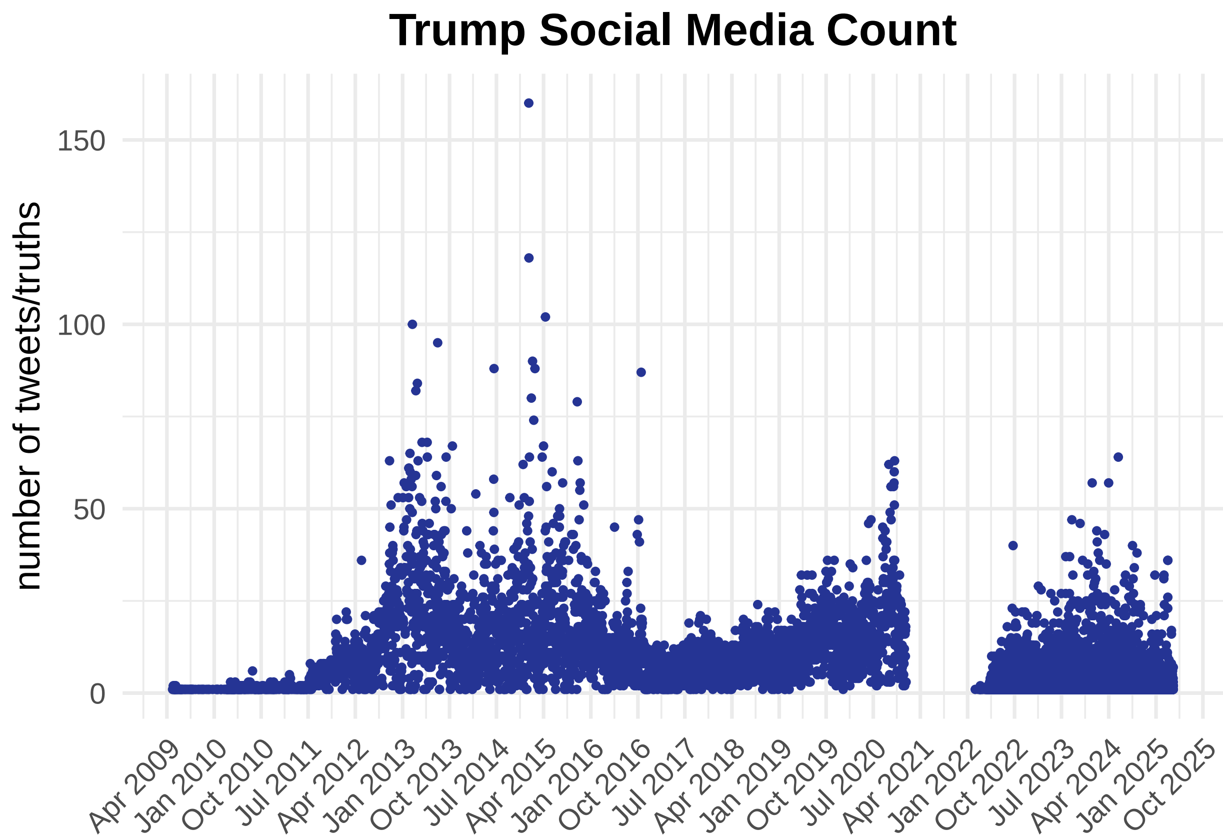
```
tweets = raw_tweets

#only keep original Tweets
tweets <- tweets %>% filter(isRetweet != "t")
tokens <- tokens(tweets$text)
dfm <- dfm(tokens)

#cleanup
tweets = as.data.table(tweets)
names(tweets)[names(tweets) == 'date'] <- 'timestamp'
tweets <- tweets[order(tweets$timestamp, decreasing=T), ]

#count by hour
tweet_count = tweets[, .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 = select(tweet_count, timestamp, N)
tweet_count = tweet_count[ order(tweet_count$timestamp , decreasing = F ),]
```

Truths Cleanup & Count

```r
truthsbackup <- truths_processer(raw_truths)

#cleanup
truths = as.data.table(truthsbackup)
names(truths)[names(truths) == 'date_time_parsed'] <- 'timestamp'
truths <- truths[order(truths$timestamp, decreasing=T), ]

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

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

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

Tweets & Truths Merge

```r
tt_count = rbind(tweet_count,truth_count) #tweets & truths

ggplot(tt_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))
```



Volatility - Daily

```r
#find daily volatility
SPY_dvolatility_alltime = r.vol_daily(raw_SPY,merge=F)

#select time period
SPY_dvolatility = filter(SPY_dvolatility_alltime,
                  between(timestamp, as.Date('2024-09-20'), as.Date('2025-04-10')))
colnames(SPY_dvolatility)[1] <- "timestamp_day"
tt_count = filter(tt_count,
                  between(timestamp, as.Date('2024-09-20'), as.Date('2025-04-10')))
```

Volatility - Hourly

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

#select time period
SPY_hvolatility = filter(SPY_hvolatility_alltime,
                  between(timestamp, as.Date('2024-09-20'), as.Date('2025-04-10')))
colnames(SPY_hvolatility)[1] <- "timestamp_hour"
```

```
tt_count = filter(tt_count,
                  between(timestamp, as.Date('2024-09-20'), as.Date('2025-04-10')))
```

ARMA-X Models

Tweet Count on Daily Volatility

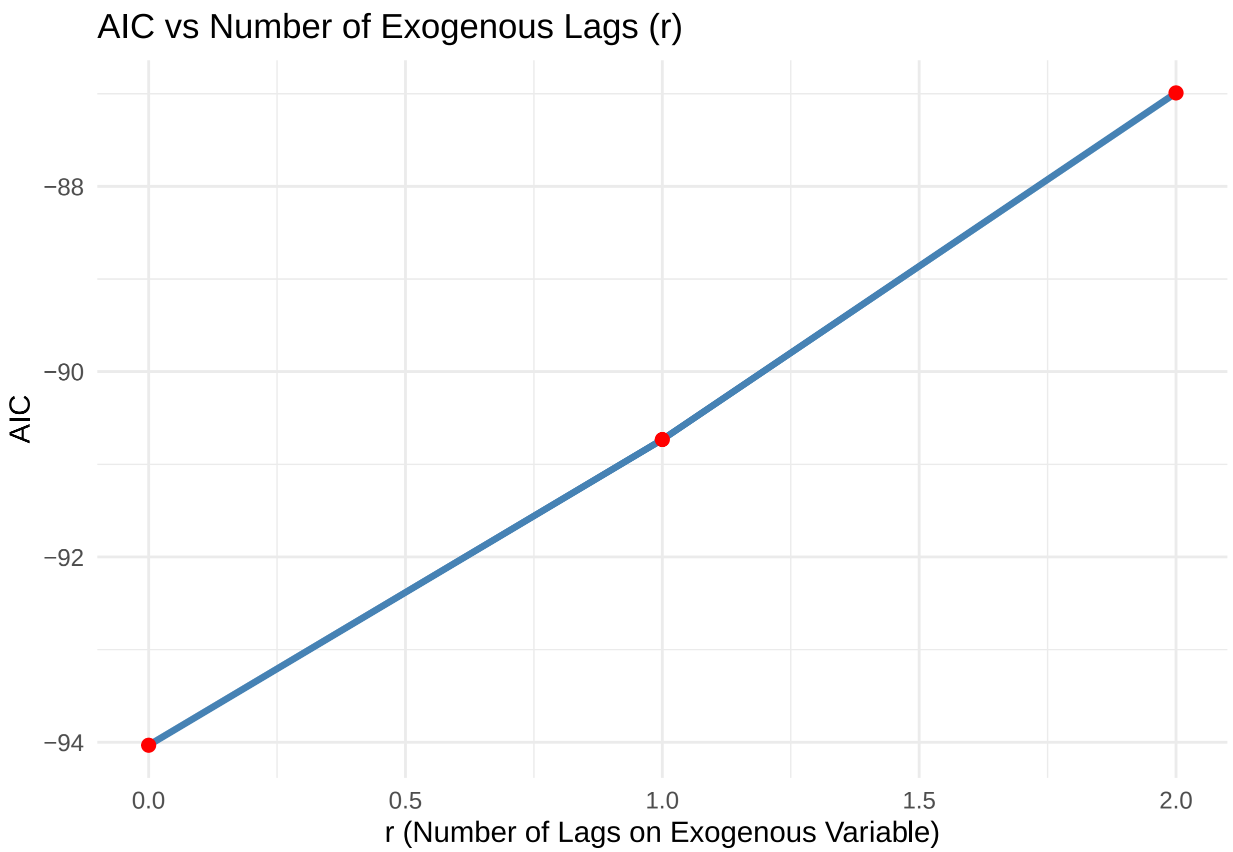
```
#take all relevant data for armax
countvol_day = merge(SPY_dvolatility, tt_count, by.x = "timestamp_day",
                   by.y = "timestamp", all.x = T)

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

#find best armax model and fit
armax_dayfit <- select_armax(countvol_day$r_vol_d, countvol_day$N,
                       max_p = 6, max_q = 6, max_r = 2, 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.
```

```
summary(armax_dayfit$model)
```

```
## Series: y_trimmed
## Regression with ARIMA(2,0,0) errors
##
## Coefficients:
##          ar1     ar2  intercept    Lag_0
##       0.2192  0.7084     0.2039  -0.0008
## s.e.  0.0649  0.0744     0.1886   0.0048
##
## sigma^2 = 0.02807:  log likelihood = 52.02
## AIC=-94.03   AICc=-93.58   BIC=-79.36
##
## Training set error measures:
##                       ME      RMSE        MAE       MPE     MAPE      MASE
## Training set 0.003610082 0.1651074 0.04166751 -82.72722 98.37072 0.7203116
##                    ACF1
## Training set 0.00497567
```

```
armax_dayfit$ICplot
```



```
armax_dayfit$params
```

```
## $p
## [1] 2
##
## $q
## [1] 0
##
## $r
## [1] 0
```

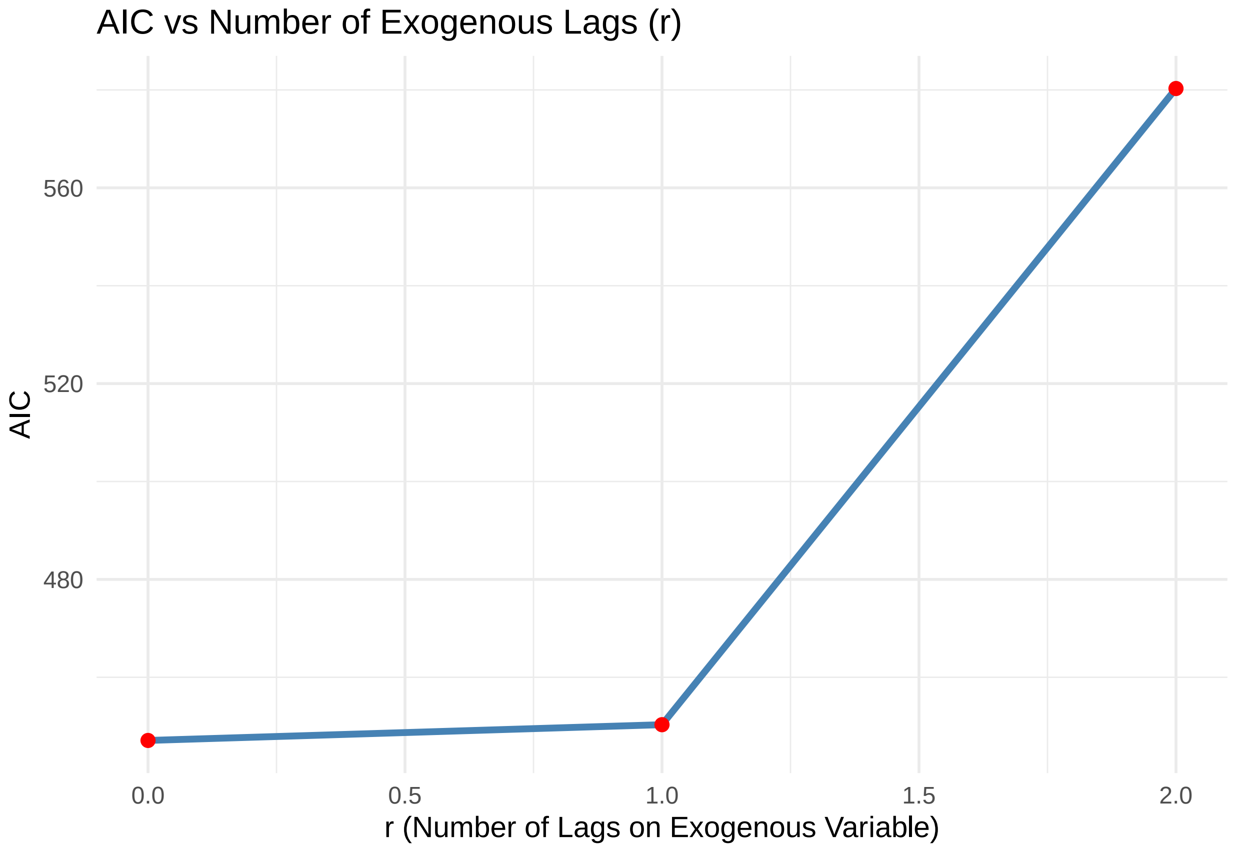
Tweet Count on Hourly Volatility

```
#take all relevant data for armax
countvol_hour = merge(SPY_hvolatility, tt_count, by.x = "timestamp_hour",
                      by.y = "timestamp", all.x = T)

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

#find best armax model and fit
armax_hourfit <- select_armax(countvol_hour$r_vol_h, countvol_hour$N,
                       max_p = 6, max_q = 6, max_r = 2, criterion = "AIC")
```

```
summary(armax_hourfit$model)
```

```
## Series: y_trimmed
## Regression with ARIMA(6,0,3) errors
##
## Coefficients:
##           ar1     ar2     ar3      ar4     ar5     ar6     ma1      ma2
##       -0.5939  0.9354  0.6385  -0.0501  0.0244  0.0390  0.9599  -0.7234
## s.e.   0.0365  0.0454  0.0556   0.0580  0.0484  0.0467  0.0178   0.0317
##           ma3  intercept   Lag_0
##       -0.9021     0.1093  0.0057
## s.e.   0.0183     0.6357  0.0030
##
## sigma^2 = 0.09111:  log likelihood = -211.55
## AIC=447.09   AICc=447.42   BIC=505.57
##
## Training set error measures:
##                       ME      RMSE        MAE       MPE     MAPE     MASE
## Training set 0.01078858 0.3001205 0.05908487 -63.29265 149.1663 1.063584
##                       ACF1
## Training set -0.0008528121
```

```
armax_hourfit$ICplot
```



```
armax_hourfit$params
```

```
## $p
## [1] 6
##
## $q
## [1] 3
##
## $r
## [1] 0
```