# Bitcoin Report

Modified Stock-to-Flow Model

## Contents

## Load Packages

```
library(data.table)
library(ggplot2)
library(magrittr)
library(patchwork)
library(rvest)
library(glmnet)
```

```
## Loading required package: Matrix
```

```
## Loaded glmnet 4.1
```

```
library(survival)
library(splines)
library(tidyquant)
```

```
## Loading required package: lubridate
```

```
##
## Attaching package: 'lubridate'
```

```
## The following objects are masked from 'package:data.table':
##
##     hour, isoweek, mday, minute, month, quarter, second, wday, week,
##     yday, year
```

```
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
## Loading required package: PerformanceAnalytics
```

```
## Loading required package: xts
```

```
## Loading required package: zoo

##
## Attaching package: 'zoo'

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

##
## Attaching package: 'xts'

## The following objects are masked from 'package:data.table':
##
##     first, last

##
## Attaching package: 'PerformanceAnalytics'

## The following object is masked from 'package:graphics':
##
##     legend

## Loading required package: quantmod

## Loading required package: TTR

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

## == Need to Learn tidyquant? ============================================
## Business Science offers a 1-hour course - Learning Lab #9: Performance Analysis & Portfolio Optimization with tidyquant!
## </> Learn more at: https://university.business-science.io/p/learning-labs-pro </>
```

```
library(ggrepel)
```

## Prepare Price data

```
newPrices <-
  read_html("https://www.cryptodatadownload.com/cdd/Bitstamp_BTCUSD_d.csv") %>%
  html_text() %>%
  gsub(pattern = "https://www.CryptoDataDownload.com\n", replacement = "", fixed = TRUE) %>%
  strsplit("\n") %>%
  unlist() %>%
  strsplit(",") %>%
  unlist() %>%
  matrix(ncol = 9, byrow = TRUE) %>%
  as.data.table() %>%
  setnames(x = ., new = unlist(.[1])) %>%
  .[-1]

newPrices %>%
  setnames(., "close", "Price") %>%
  setnames(., "Volume BTC", "VolBTC") %>%
  .[,time := as.POSIXct(newPrices$date)] %>%
  .[,MergeTime := paste(year(time), month(time), mday(time))] %>%
  .[,Price := as.numeric(Price)]

oldPriceDat <- readRDS(file =  "/Users/Shared/S2F/myBitcoinPriceDat.RDS")
```

## Prepare Block Data

```
currentBlock <- "https://mempool.space/api/blocks/tip/height" %>%
                  read_html() %>%
                  html_text() %>%
                  as.integer()
```

```
currentDate <- Sys.time()
blocksToModel <- 1000000

newPrices <- newPrices[time > max(oldPriceDat$time)]

myDat <- rbind(newPrices[,.(MergeTime, time, Price)],
               oldPriceDat[,.(MergeTime, time, Price)])

myDat[,Price := as.numeric(Price)]

saveRDS(object = myDat,file =  "/Users/Shared/S2F/myBitcoinPriceDat.RDS")

date = Sys.Date() %m+% years(-10)
SP500 = tq_get("^GSPC", from = date) %>%
        as.data.table() %>%
        .[,MergeTime := paste(year(date), month(date), mday(date))]

myDat <- merge(x = myDat,
               y = SP500[,.(MergeTime, SP500 = close)],
               by = "MergeTime")
```

## Make function to compute the Stock to flow

```
S2F <- function(dat, days=365) {
  setorder(x = dat, "Block")
  dat[,S2F := Stock / frollsum(x = BlockReward, n = round(days*6*24)) * days / 365]
  maxVal <- dat[,max(S2F,na.rm = TRUE)]
  dat[is.na(S2F), S2F := maxVal]
  dat
}

dat <- data.table(Block = 1:blocksToModel) %>%
       .[,Time := (currentDate + (Block - currentBlock)*60*10)] %>%
       .[,BlockReward := round(50 * (0.5) ^ floor((Block-1)/210000),8)] %>%
       .[,DaysToHalving := (210000 -(Block %% 210000))/(6*24)] %>%
```

```
      .[,Stock := cumsum(BlockReward)] %>%
      .[,MergeTime := paste(year(Time), month(Time), mday(Time))]
```

## Add the P-Spline Terms on DaysToNextHalving

```
tmp <- dat[,pspline(DaysToHalving, nterm = 4)] %>% as.matrix()
class(tmp) <- "matrix"
tmp <- as.data.table(tmp)
names(tmp) <- paste0("DaysToHalving", 1:ncol(tmp))

dat <- cbind(dat, tmp)

dat <- merge(x = dat,
             y = myDat,
             by = "MergeTime",
             all.x = TRUE)

currentDaysToHalving <- dat[Block == currentBlock]$DaysToHalving
```

## Set up function to fit the model

```
trainAndPlotS2FModel <- function(maxTrainingYear = 2021,
                                 daysInS2F = 463,
                                 yearsToPlot = 2011:2023,
                                 daysToPlot = c(1,15),
                                 initialParam = c(-1.5, 3, 0, 0, 0, 0, 0, 0)) {

  cat("\n\nFitting Stock to Flow\n\n")

  S2F(dat, days = daysInS2F)

  trainDat <- dat[year(Time) <= maxTrainingYear & !is.na(Price)]
```

```
priceModel <- function(x, S2F, DaysToHalving1, DaysToHalving2, DaysToHalving3, DaysToHalving4, DaysToHalving5, DaysToHalving6) {

  result <- (exp(x[1] + x[3]*DaysToHalving1 + x[4]*DaysToHalving2 + x[5]*DaysToHalving3 + x[6]*DaysToHalving4 + x[7]*DaysToHalving5 + x[

  return(result)
}

tukey_loss <- function(r, c) {
  ifelse(abs(r) <= c,
         c^2 / 6 * (1 - (1 - (r / c)^2)^3),
         c^2 / 6)
}

squareError <- function(x) {
  #trainDat[,sum(tukey_loss(1 - priceModel(x, S2F, DaysToHalving1, DaysToHalving2, DaysToHalving3)/Price, 4.685), na.rm = TRUE)]
  trainDat[,sum((1 - priceModel(x, S2F, DaysToHalving1, DaysToHalving2, DaysToHalving3, DaysToHalving4, DaysToHalving5, DaysToHalving6)/
}
tmp <- optim(initialParam, squareError)
par <- tmp$par

print(par)

dat[,S2F_predicted_price := priceModel(x = par, S2F, DaysToHalving1, DaysToHalving2, DaysToHalving3, DaysToHalving4, DaysToHalving5, Day
dat[,AtoE_S2F_Price := Price/S2F_predicted_price]

# dat[,SmoothedAtoE_S2F_Price := frollmean(AtoE_S2F_Price, 3)]
dat[,SmoothedAtoE_S2F_Price := AtoE_S2F_Price]

cat("\n\nPreparing Plots and Data\n\n")

S2FpredPriceCurrent <- dat[Block == currentBlock]$S2F_predicted_price %>% signif(.,3)
S2FpredPrice90Day <- dat[Block == currentBlock + 90*24*6]$S2F_predicted_price %>% signif(.,3)
S2FpredPrice2021YE <- dat[MergeTime == "2021 12 31"]$S2F_predicted_price %>% signif(.,3) %>% max()
date90Day <- dat[Block == currentBlock + 90*24*6]$MergeTime

dat <<- dat

dat <- dat[Time >= dat[!is.na(Price)][,min(Time)]]
```

```
dat[,KeepMe := max(.SD$S2F) == .SD$S2F,MergeTime]
dat <- dat[KeepMe == TRUE][,KeepMe:=NULL]

dat[,Yearr := year(Time)]

result <- list()
result$plotts <- list()

result$plotts$S2FModelPlot <-
  ggplot(dat[year(Time) %in% yearsToPlot][mday(Time) %in% daysToPlot][]) +
    geom_point(aes(x = Time, y = S2F_predicted_price), color = 'black') +
    geom_point(aes(x = Time, y = Price, color = DaysToHalving)) +
    scale_color_gradient(low = "green", high = "red") +
    scale_y_log10() +
    labs(title = paste(daysInS2F, "day Stock-to-Flow model fit to & incl.", maxTrainingYear),
         subtitle = paste0("Today S2F : $", S2FpredPriceCurrent,"\n",
                           "'21 YE S2F: $", S2FpredPrice2021YE,
                           "\nModel adjusted to include a non-linear expression in days to next halving",
                           " in order to capture sentiment over the halving cycle.\nFormula based on PlanB's S2F model."),
         xlab = NULL)

result$plotts$S2FAtoEPlot <-
    ggplot(dat[year(Time) %in% yearsToPlot][mday(Time) %in% daysToPlot]) +
      geom_hline(yintercept = 1, color = "black") +
      geom_point(aes(x = Time, y = AtoE_S2F_Price, color = DaysToHalving)) +
      scale_color_gradient(low = "green", high = "red") +
      ylim(c(0, 3))

result$plotts$StockPlot <-
    ggplot(dat[year(Time) %in% yearsToPlot][mday(Time) %in% daysToPlot]) +
      geom_point(aes(x = Time, y = S2F, color = DaysToHalving)) +
      scale_color_gradient(low = "green", high = "red")

result$plotts$AtoEvsDaysToHalving1 <-
    ggplot(dat[!is.na(Price)][!is.na(SmoothedAtoE_S2F_Price)][,.(AtoE_S2F_Price = pmin(2,AtoE_S2F_Price), DaysToHalving, Time, SmoothedA
      geom_hline(yintercept = 1, color = 'red') +
      geom_point(aes(Time, SmoothedAtoE_S2F_Price, color = DaysToHalving)) +
      scale_color_gradient(low = "green", high = "red") +
```

```
      ylim(c(0, 3))

dat[,pointSize := fifelse(year(Time) == max(year(Time)), 1, 0.25)]


result$plotts$AtoEvsDaysToHalving2 <-
    ggplot(dat[!is.na(Price)][!is.na(SmoothedAtoE_S2F_Price)][,.(AtoE_S2F_Price = pmin(2,AtoE_S2F_Price), DaysToHalving, Time, SmoothedA
      geom_hline(yintercept = 1, color = 'red') +
      geom_point(aes(DaysToHalving, SmoothedAtoE_S2F_Price, color = Time, alpha = pointSize), size = 3) +
      scale_color_viridis_c() +
      scale_x_reverse(breaks = round(seq(min(dat$DaysToHalving), max(dat$DaysToHalving), by = 100))) +
      ylim(c(0, 3)) +
      labs(title = "Actual divided Predicted Price vs Days-to-next-halving",
           subtitle = "Yellow dots = most recent halving cycle",
           ylab = "Actual / Predicted Price")

daysLB <- currentDaysToHalving-200
daysUB <- currentDaysToHalving+50

dat[,DaysToHalving := ceiling(DaysToHalving)]

tmp <- copy(dat) %>%
       .[,DaysToHalving := round(DaysToHalving/10)*10] %>%
       .[, CycleSmoothAtoE := as.numeric(NA)] %>%
       .[, CycleSmoothAtoE := mean(SmoothedAtoE_S2F_Price, na.rm = TRUE), DaysToHalving]

ggplot(tmp) + geom_point(aes(DaysToHalving, CycleSmoothAtoE))

tmp[, PredCyclePrice := CycleSmoothAtoE * S2F_predicted_price]
tmp[,showMe := as.numeric(NA)]
tmp[between(DaysToHalving, daysLB,daysUB) &
    DaysToHalving != shift(DaysToHalving) &
    substr(MergeTime,1,4) == "2021",
    showMe := PredCyclePrice]

tmp[!is.na(showMe)][rev(order(DaysToHalving))]

tmp[,textLabelYpos := pmin(CycleSmoothAtoE, 3)]
```

```
  result$plotts$AtoEvsDaysToHalving3 <-
      ggplot(dat[!is.na(Price)][!is.na(SmoothedAtoE_S2F_Price)][between(DaysToHalving,daysLB,daysUB)
              ][,.(DaysToHalving, Time, SmoothedAtoE_S2F_Price = pmin(SmoothedAtoE_S2F_Price, 5),
                   pointSize)]) +
        geom_hline(yintercept = 1, color = 'red') +
        geom_point(aes(DaysToHalving, SmoothedAtoE_S2F_Price, color = Time, alpha = pointSize), size = 3) +
        geom_text_repel(aes(DaysToHalving, textLabelYpos, label = paste0("$",signif(showMe/1000,2),"k")),
                        size = 4,
                        data = tmp[between(DaysToHalving,daysLB,daysUB)][!is.na(showMe)]) +
        scale_color_viridis_c() +
        scale_x_reverse(breaks = round(seq(daysLB, daysUB, by = 30))) +
        ylim(c(0, 3)) +
        labs(title = "Actual divided Predicted Price vs Days-to-next-halving",
             subtitle = paste0("Yellow dots = most recent halving cycle\nCurrent days to halving :",round(currentDaysToHalving)),
             ylab = "Actual / Predicted Price",
             caption = "Text Labels = Price relative to this S2F model if Historic AtoEs repeat")


  result$plotts$AtoEDistribution <-
      ggplot(dat[!is.na(Price)][,.(AtoE_S2F_Price = pmin(3,AtoE_S2F_Price), DaysToHalving, Time)]) +
        geom_histogram(aes(AtoE_S2F_Price), bins = 50) +
        geom_vline(xintercept = dat[!is.na(AtoE_S2F_Price)]$AtoE_S2F_Price %>% tail(1), color = 'red', size = 3) +
        labs(title = "Current AtoE vs Entire AtoE Distribution",
             subtitle = paste0("Current AtoE = ", dat[!is.na(AtoE_S2F_Price)]$AtoE_S2F_Price %>% tail(1) %>% signif(2),
                               "\nCurrent Price = ", dat[!is.na(AtoE_S2F_Price)]$Price %>% tail(1) %>% signif(2),
                               "\nModel Price = ", S2FpredPriceCurrent))

  return(result)
}
```

## Fit the model

```
result <-
trainAndPlotS2FModel(maxTrainingYear = 2021,
                     daysInS2F = 463,
```

```
                      yearsToPlot = 2010:2025,
                      daysToPlot = 1:31)
```

```
##
##
## Fitting Stock to Flow
##
## [1] -3.12740693  3.62526921  0.29723983 -0.60118875  0.61959307 -0.48600523
## [7]  0.06010253 -0.74567180
##
##
## Preparing Plots and Data
```

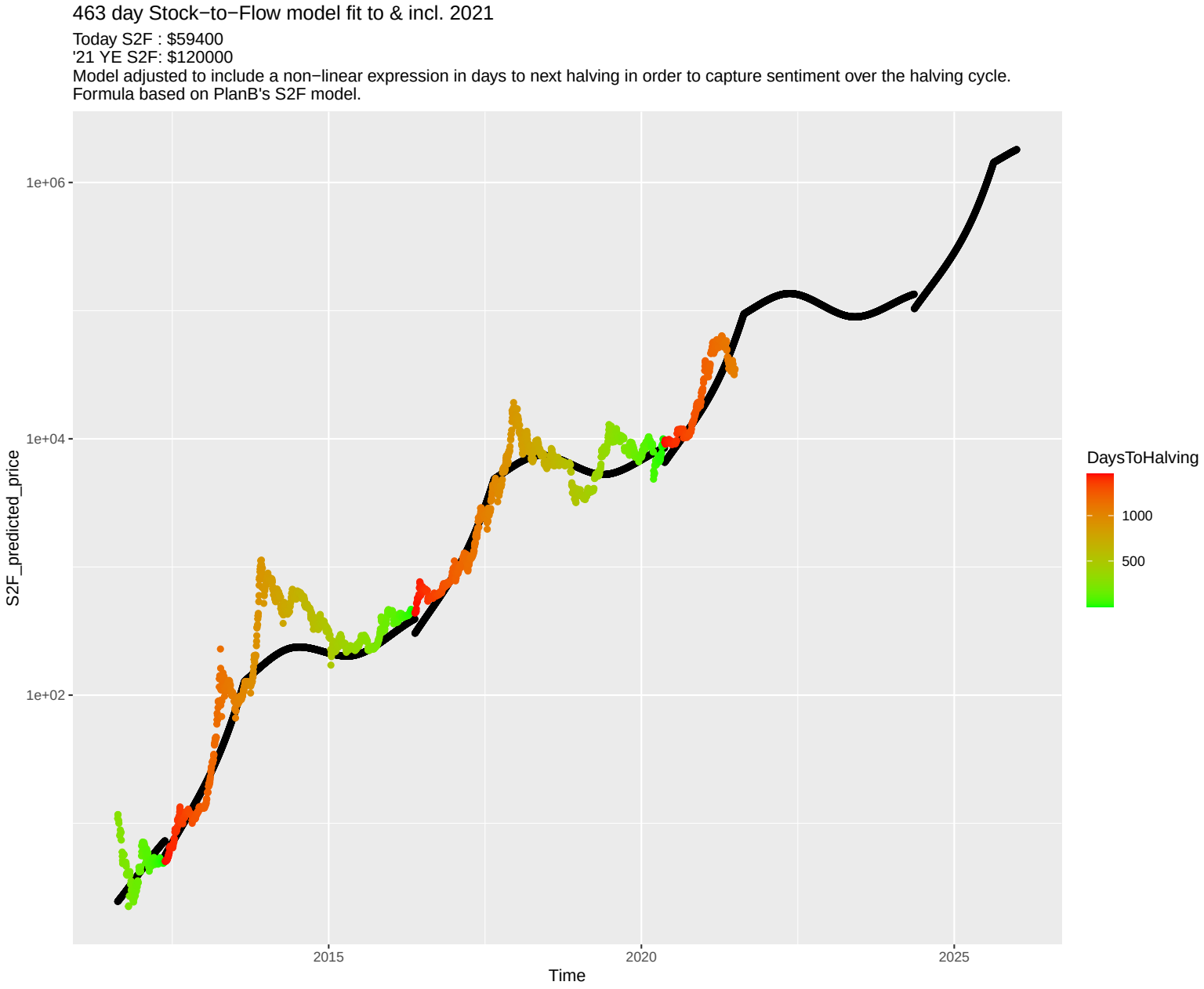
## Show the plots

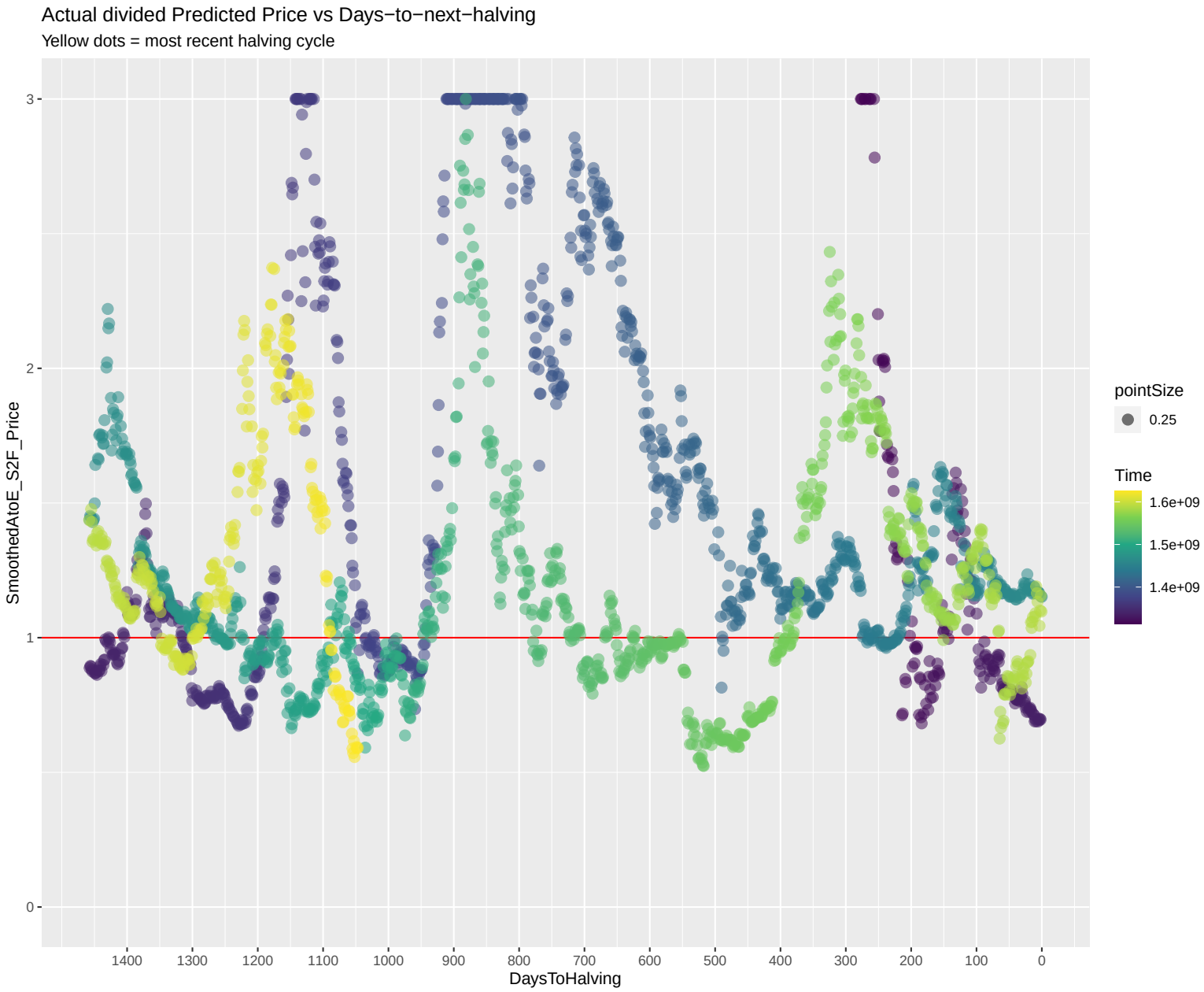
The current date is 2021-07-01 and the current block number is 689300.

Our price data goes back to 2021 6 30.

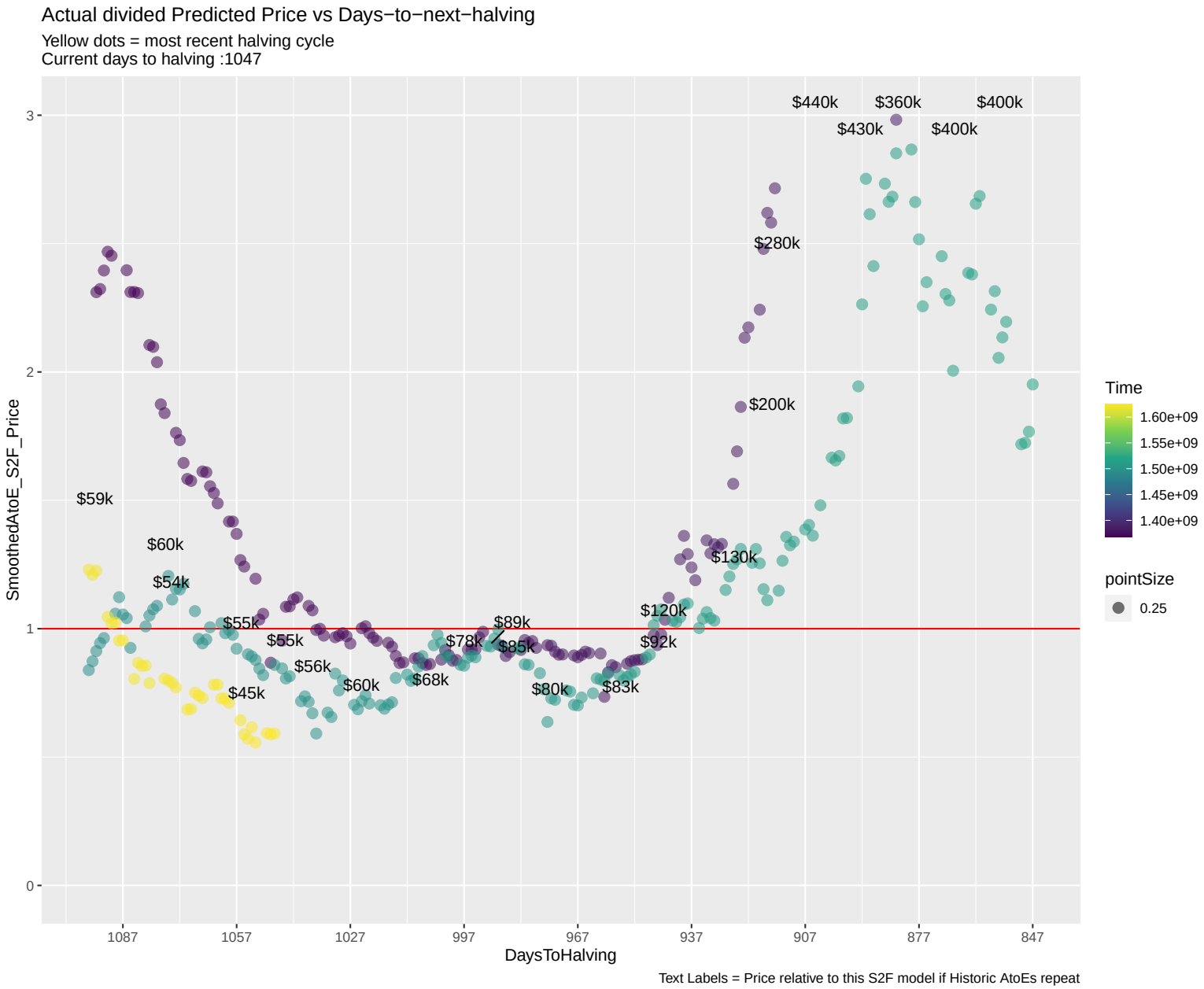
The latest closing price in the data is $3.5 \times 10^4$

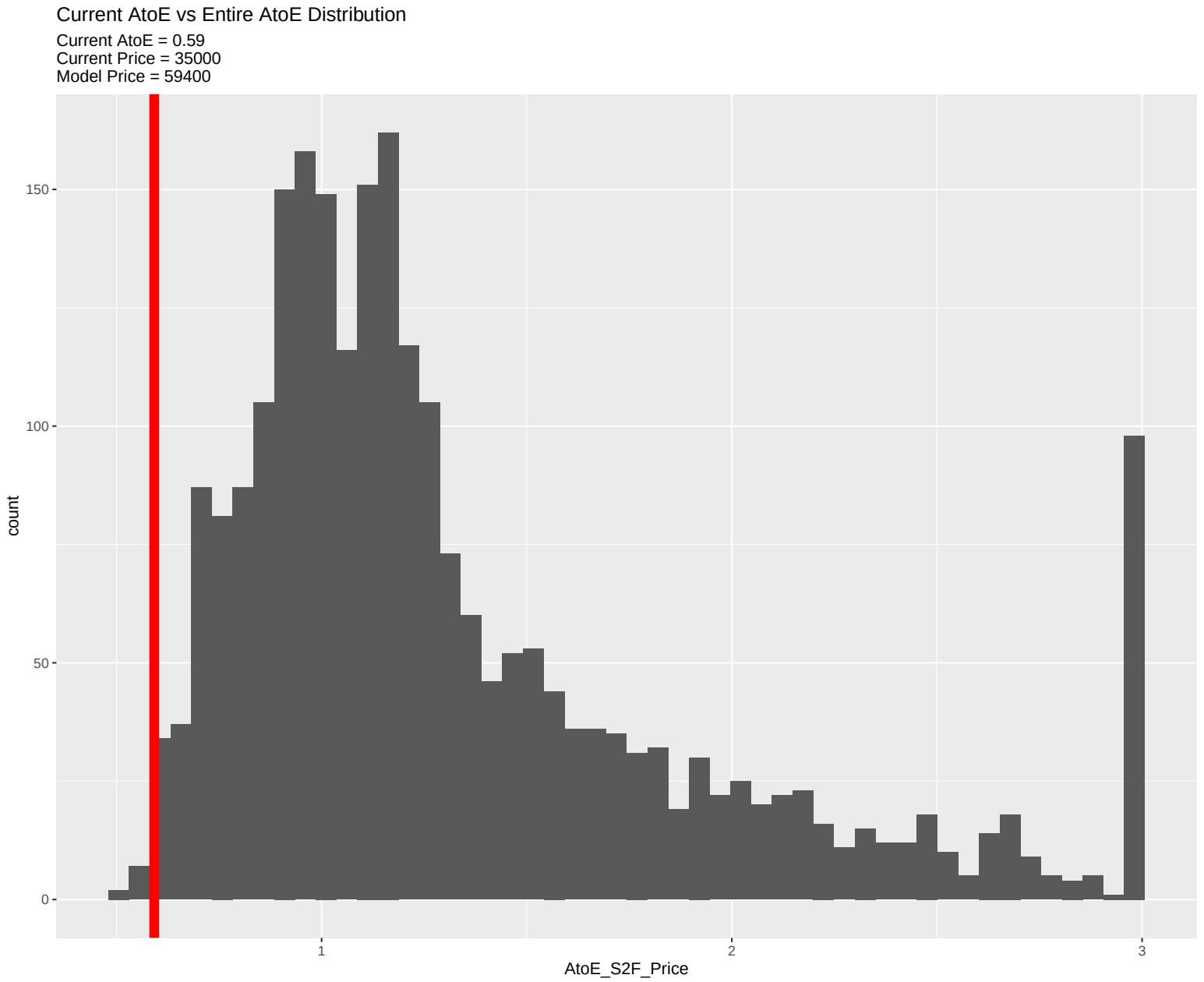
```
## Warning: Removed 2791 rows containing missing values (geom_point).
```


463 day Stock−to−Flow model fit to & incl. 2021
Today S2F : $59400
'21 YE S2F: $120000
Model adjusted to include a non−linear expression in days to next halving in order to capture sentiment over the halving cycle.
Formula based on PlanB's S2F model.
S2F_predicted_price
1e+06
1e+04
1e+02
2015
2020
2025
Time
DaysToHalving
1000
500


Actual divided Predicted Price vs Days−to−next−halving
Yellow dots = most recent halving cycle
SmoothedAtoE_S2F_Price
3
2
1
0
1400
1300
1200
1100
1000
900
800
700
600
500
400
300
200
100
0
DaysToHalving
pointSize
0.25
Time
1.6e+09
1.5e+09
1.4e+09

```
## Warning: Removed 44 rows containing missing values (geom_point).
```


Actual divided Predicted Price vs Days−to−next−halving
Yellow dots = most recent halving cycle
Current days to halving :1047
SmoothedAtoE_S2F_Price
DaysToHalving
Text Labels = Price relative to this S2F model if Historic AtoEs repeat
Time
1.60e+09
1.55e+09
1.50e+09
1.45e+09
1.40e+09
pointSize
0.25
$440k
$360k
$400k
$430k
$400k
$280k
$200k
$130k
$120k
$92k
$89k
$85k
$78k
$83k
$80k
$68k
$60k
$56k
$55k
$55k
$45k
$59k
$60k
$54k


Current AtoE vs Entire AtoE Distribution
Current AtoE = 0.59
Current Price = 35000
Model Price = 59400
150
100
50
0
count
1
2
3
AtoE_S2F_Price