In today’s post, we will add a benchmark so that we can compare our IPO portfolios to something besides themselves. Next time, we will delve into return attribution to visualize how individual equities have contributed to portfolios over time.

I won’t review the code from the previous posts, but briefly we imported prices for every ticker that IPO’d between 2004 and 2014, found the monthly returns of those tickers, then constructed portfolios on a year-by-year basis, so that we had a portfolio formed each year consisting of equal weights in every IPO for that year.

# object holding time series of monthly closing prices, monthly returns, tickers, IOP year and sector

ipo\_riingo\_prices\_pins

# function to calculate returns of portfolios constructed by investing equally in each IPO in each year

ipo\_by\_year\_portfolios <- function(year, show\_growth = F){

ipo\_riingo\_prices\_pins %>%

select(ticker, date, monthly\_returns, ipo.year) %>%

filter(ipo.year == year) %>%

tq\_portfolio(assets\_col = ticker,

returns\_col = monthly\_returns,

col\_rename = paste(year, "\_port\_returns", sep = ""),

wealth.index = show\_growth,

rebalance\_on = "months")

}

# vector of years

years\_numeric <- seq(2004, 2014, by = 1)

# pass the years and the indiviual returns object to the function

returns\_each\_year\_ipo\_portfolios <-

map(years\_numeric, ipo\_by\_year\_portfolios) %>%

reduce(left\_join)

And here is the resulting object of portfolio returns:

returns\_each\_year\_ipo\_portfolios %>%

tail()

# A tibble: 6 x 12

date `2004\_port\_retu… `2005\_port\_retu… `2006\_port\_retu…

1 2019-05-31 00:00:00 -0.101 -0.0575 0.261

2 2019-06-28 00:00:00 0.382 0.0635 0.0533

3 2019-07-31 00:00:00 0.00659 0.0185 0.0346

4 2019-08-30 00:00:00 -0.0229 -0.0317 -0.00789

5 2019-09-30 00:00:00 0.0256 0.00545 -0.00539

6 2019-10-31 00:00:00 0.0347 0.0180 0.0233

# … with 8 more variables: `2007\_port\_returns` ,

# `2008\_port\_returns` , `2009\_port\_returns` ,

# `2010\_port\_returns` , `2011\_port\_returns` ,

# `2012\_port\_returns` , `2013\_port\_returns` ,

# `2014\_port\_returns`

All that was done last time, and it gave us the returns of our 11 IPO portfolios from formation to today.

Now, let’s calculate the returns of a benchmark portfolio so we can compare those IPO portfolios to something besides themselves. We will use SPY as the benchmark and start by importing monthly prices since 2004. I’ll also go ahead and calculate monthly returns in the same piped flow.

spy\_benchmark <-

"SPY" %>%

riingo\_prices(start\_date = "2004-01-01", end\_date = "2019-10-31", resample\_frequency = "monthly") %>%

select(ticker, date, close) %>%

mutate(spy\_monthly\_returns = close/lag(close) - 1) %>%

na.omit()

spy\_benchmark %>%

head()

# A tibble: 6 x 4

ticker date close spy\_monthly\_returns

1 SPY 2004-02-27 00:00:00 115. 0.0136

2 SPY 2004-03-31 00:00:00 113. -0.0167

3 SPY 2004-04-30 00:00:00 111. -0.0189

4 SPY 2004-05-31 00:00:00 113. 0.0171

5 SPY 2004-06-30 00:00:00 115. 0.0148

6 SPY 2004-07-30 00:00:00 111. -0.0322

From here, it’s straightforward to compare these benchmark returns to those of the 2004 IPO portfolio. First, we line up the two columns of returns.

returns\_each\_year\_ipo\_portfolios %>%

select(date, `2004\_port\_returns`) %>%

add\_column(benchmark = spy\_benchmark$spy\_monthly\_returns) %>%

tail()

# A tibble: 6 x 3

date `2004\_port\_returns` benchmark

1 2019-05-31 00:00:00 -0.101 -0.0638

2 2019-06-28 00:00:00 0.382 0.0644

3 2019-07-31 00:00:00 0.00659 0.0151

4 2019-08-30 00:00:00 -0.0229 -0.0167

5 2019-09-30 00:00:00 0.0256 0.0148

6 2019-10-31 00:00:00 0.0347 0.0221

Then we pivot\_longer() and apply the SharpeRatio() function, same as we did last time.

returns\_each\_year\_ipo\_portfolios %>%

select(date, `2004\_port\_returns`) %>%

add\_column(benchmark = spy\_benchmark$spy\_monthly\_returns) %>%

pivot\_longer(-date, names\_to = "portfolio", values\_to = "monthly\_return") %>%

group\_by(portfolio) %>%

arrange(portfolio, date) %>%

filter(![is.na](http://is.na)(monthly\_return)) %>%

tq\_performance(Ra = monthly\_return,

performance\_fun = SharpeRatio,

Rf = 0,

FUN= "StdDev")

# A tibble: 2 x 2

# Groups: portfolio [2]

portfolio `StdDevSharpe(Rf=0%,p=95%)`

1 2004\_port\_returns 0.234

2 benchmark 0.153

Here’s the result piped straight to ggplot().

returns\_each\_year\_ipo\_portfolios %>%

select(date, `2004\_port\_returns`) %>%

add\_column(benchmark = spy\_benchmark$spy\_monthly\_returns) %>%

pivot\_longer(-date, names\_to = "portfolio", values\_to = "monthly\_return") %>%

group\_by(portfolio) %>%

arrange(portfolio, date) %>%

filter(![is.na](http://is.na)(monthly\_return)) %>%

tq\_performance(Ra = monthly\_return,

performance\_fun = SharpeRatio,

Rf = 0,

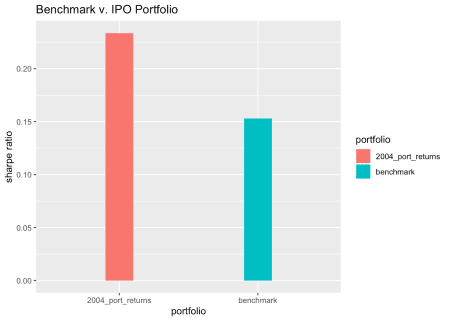
FUN= "StdDev") %>%

`colnames<-`(c("portfolio", "port\_sharpe")) %>%

ggplot(aes(x = portfolio, y = port\_sharpe, fill = portfolio)) +

geom\_col(width = .2) +

labs(y = "sharpe ratio", title = "Benchmark v. IPO Portfolio")



Our IPO portfolio has a higher Sharpe Ratio, but remember that we built this without regard to survivorship bias, we didn’t invest in any companies that haven’t survived to 2019.

That’s a nice comparison of one portfolio to the benchmark, but we want to run this same analysis on all of our portfolios.

First, let’s calculate the Sharpes for all of our IPO portfolios, same as we did last time.

years\_numeric <- seq(2004, 2014, by = 1)

port\_sharpes <-

returns\_each\_year\_ipo\_portfolios %>%

pivot\_longer(-date, names\_to = "portfolio\_by\_year", values\_to = "monthly\_return") %>%

group\_by(portfolio\_by\_year) %>%

arrange(portfolio\_by\_year, date) %>%

filter(![is.na](http://is.na)(monthly\_return)) %>%

tq\_performance(Ra = monthly\_return,

performance\_fun = SharpeRatio,

Rf = 0,

FUN= "StdDev") %>%

`colnames<-`(c("portfolio\_by\_year", "port\_sharpe"))%>%

add\_column(year = years\_numeric)

port\_sharpes

# A tibble: 11 x 3

# Groups: portfolio\_by\_year [11]

portfolio\_by\_year port\_sharpe year

1 2004\_port\_returns 0.234 2004

2 2005\_port\_returns 0.192 2005

3 2006\_port\_returns 0.249 2006

4 2007\_port\_returns 0.190 2007

5 2008\_port\_returns 0.142 2008

6 2009\_port\_returns 0.220 2009

7 2010\_port\_returns 0.279 2010

8 2011\_port\_returns 0.152 2011

9 2012\_port\_returns 0.309 2012

10 2013\_port\_returns 0.182 2013

11 2014\_port\_returns 0.218 2014

And now, let’s calculate the Sharpe Ratio for the benchmark for each year. That means we will build or organize 11 different return streams for SPY, each starting in a year from 2004 to 2014, and then calculate the Sharpes for each of those 11 return streams.

Here’s how we do it for just 2004.

start\_year <- "2004"

start\_date <- ymd(parse\_date(start\_year, format = "%Y"))

spy\_benchmark %>%

filter(date >= start\_date) %>%

tq\_performance(Ra = spy\_monthly\_returns,

performance\_fun = SharpeRatio,

Rf = 0,

FUN= "StdDev")

# A tibble: 1 x 1

`StdDevSharpe(Rf=0%,p=95%)`

1 0.153

This looks like a good candidate for a function that accepts one argument, the start\_year, that we can pass a vector of years.

spy\_sharpe\_function <- function(start\_year){

start\_date <- ymd(parse\_date(start\_year, format = "%Y"))

spy\_benchmark %>%

filter(date >= start\_date) %>%

tq\_performance(Ra = spy\_monthly\_returns,

performance\_fun = SharpeRatio,

Rf = 0,

FUN = "StdDev") %>%

`colnames<-`("spy\_sharpe") %>%

mutate(year = as.numeric(start\_year))

}

Let’s pass in one year and peek at the result.

spy\_sharpe\_function("2005")

# A tibble: 1 x 2

spy\_sharpe year

1 0.150 2005

Now, let’s map across different years.

years\_character <- as.character(years\_numeric)

spy\_sharpes <-

map\_dfr(years\_character, spy\_sharpe\_function)

spy\_sharpes

# A tibble: 11 x 2

spy\_sharpe year

1 0.153 2004

2 0.150 2005

3 0.152 2006

4 0.139 2007

5 0.141 2008

6 0.258 2009

7 0.253 2010

8 0.263 2011

9 0.310 2012

10 0.300 2013

11 0.229 2014

That worked! Let’s join our benchmark results with the IPO portfolio results for ease of comparison and pipe straight to ggplot().

port\_sharpes %>%

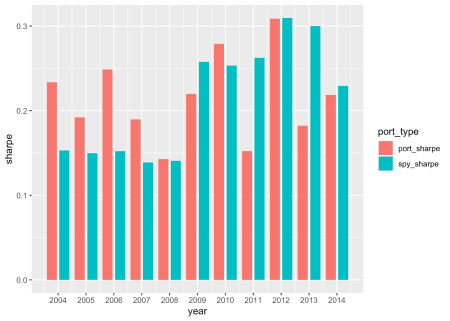
left\_join(spy\_sharpes, by = "year") %>%

pivot\_longer(c(-year, -portfolio\_by\_year), names\_to = "port\_type", values\_to = "sharpe") %>%

ggplot(aes(x = year, y = sharpe, fill = port\_type)) +

geom\_col(position = position\_dodge2(padding = .2)) +

scale\_x\_continuous(breaks = scales::pretty\_breaks(n = 10))



It looks like our IPO portfolios outperformed in the years 2004-2007. That might be due to our survivorship bias since we’re only investing in companies that we know, with hindsight, have survived to 2019.

Let’s also remember that Sharpe Ratios aren’t everything. Our IPO portfolios might be so volatile that we wouldn’t have a the gumption to stick with them through the hard times. To get a better sense of what we’d have faced, let’s visualize the drawdowns for the 2004 IPO portfolio versus the benchmark.

returns\_each\_year\_ipo\_portfolios %>%

select(date, `2004\_port\_returns`) %>%

left\_join(spy\_benchmark %>% select(date, spy\_monthly\_returns), by = "date") %>%

pivot\_longer(-date, names\_to = "fund", values\_to = "drawdown") %>%

mutate(drawdown = case\_when(drawdown > 0 ~ 0,

TRUE ~ drawdown),

drawdown = drawdown \* 100) %>%

plot\_ly(type = 'scatter', x = ~date, y = ~drawdown, color = ~fund,

mode = 'lines', fill = 'tonexty') %>%

layout(yaxis = list(ticksuffix = "%"))

Click on the legend to isolate the chart of either the IPO portfolio or the benchmark and notice the much rougher history of the IPO portfolio.  
Our IPO portfolio had a large drawdown of around 26% in 2008 – would we have stuck with it?

Let’s do that for the drawdown chart so that all we have to do is supply a year. This would be helpful in a Shiny application where we wanted to explore different years interactively.

drawdown\_vis\_fun <- function(start\_year){

start\_date <- ymd(parse\_date(start\_year, format = "%Y"))

spy\_benchmark <-

spy\_benchmark %>%

filter(date >= start\_date)

ipo\_port <- paste(start\_year, "\_port\_returns", sep = "")

returns\_each\_year\_ipo\_portfolios %>%

select(date, `ipo\_port`) %>%

left\_join(spy\_benchmark %>% select(date, spy\_monthly\_returns), by = "date") %>%

pivot\_longer(-date, names\_to = "fund", values\_to = "drawdown") %>%

mutate(drawdown = case\_when(drawdown > 0 ~ 0,

TRUE ~ drawdown),

drawdown = drawdown \* 100) %>%

plot\_ly(type = 'scatter', x = ~date, y = ~drawdown, color = ~fund,

mode = 'lines', fill = 'tonexty') %>%

layout(yaxis = list(ticksuffix = "%"))

}

Now, let’s pass the function a single year and examine the drawdown history.

drawdown\_vis\_fun("2007")

That’s all for today’s addendum.