Brinson Attribution Analysis

With a 'Real World' Example

Investment Management Division

Arizona State Retirement System

April 6, 2017

Outline

Outline

Concept

- Performance Attribution is a set of techniques used to explain why a portfolio's performance differed from the benchmark
- Classic attribution approach is the Brinson, Hood and Beebower methodology where active return (difference between portfolio and benchmark) is broken down into three components
  - Allocation effect is simply the contribution to active return due tot he over- and underweighting of asset class
  - Selection effect is the contribution to active return due to the over- and underweighting of portfolios within the asset class
  - Interaction effect is the contribution to active return due to the over- and underweighting of out- or underperforming asset classes

Brinson Model

- Formulas:

Figure: Brinson, Hood, and Beebower Attribution Analysis Formulas

$$Allocation\ Effect = \sum (W_{portfolio} - W_{benchmark}) \times R_{benchmark} \tag{1}$$

$$Selection\ Effect = \sum W_{benchmark} \times (R_{portfolio} - R_{benchmark}) \tag{2}$$

$$Interaction\ Effect = \sum (W_{portfolio} - W_{benchmark}) / times (R_{portfolio} - R_{benchmark}) \tag{3}$$

- Inputs
  - Target weights of asset classes
  - Benchmark returns
  - Portfolio returns
- These formulas are effective for single period analysis but a residual is introduced once returns are compounded through time
  - For example, if you outperform in period 1 by 2% and underperform in period 2 by 1%; the classic brinson model may not fully account for the total 1% outperformance due to the compounding between the periods

Frongello Method

Reason for the Frongello Method

- Method indroduced in 2004 to solve the challenge of linking single period attribution results without an unexplained residual
  - Problem: Returns can't be summed or compounded
  - Solution: Adjust attributes so they can be summed

Frongello Method

Reason for the Frongello Method

- Method indroduced in 2004 to solve the challenge of linking single period attribution results without an unexplained residual
  - Problem: Returns can't be summed or compounded
  - Solution: Adjust attributes so they can be summed
- At a high level, monthly returns are transformed into dollars utilizing logarithmics
  - From the example above, at period 0, you had \$100; at period 1, you had 102 dollars; and at period 3, you have \$100.98

Why?

Frongello Adjustments

- Adjustment 1:
    - Scale the current attribute by the total portfolio return through the prior period
- Adjustment 2:
    - Multiply the prior attributes by the current benchmark return

Frongello Adjusted Attributes

Figure: Frongello Algorithm

$$F_{tb} = G_{tb}\prod_{j=1}^{t-1}(1+R_j) + \bar{R}_t\sum_{j=1}^{t-1}F_{jb} \qquad (4)$$

Intuitive interpretation:

- Each original attribute is scaled by the portfolio total return through the prior period and the current period return of the benchmark compounds with the total return due to that attribute through the prior period
- Curr Attribute * [(1 + Port Ret Thru n-1) + (1 + Bench Ret Thru n-1)] /2 + [Curr Port Ret + Curr Bench Ret] /2 * Sum Prior Adj Attributes

Outline

Introduction to Time Series in R

- There are multiple packages available for time series but 2 are commonly used within IMD: xts & zoo
  - There are subtle but significant differences between the two
    - XTS package is actually an extension of the zoo package that allows more attributes of the data to be saved
    - An xts object is also a zoo class object but not vice versa
- Time series are special cases of data frames where the dates are printed like rownames but are actually an index
  - Helpful functions for dealing with time series objects include **coredata** (results in a data frame of the data without the date) & **time** (vector of the dates in the time series)

Creating a Time Series

- To create a zoo object time series:
  - function to change an object to date class is: as.Date(date column, format="%m/%d/%Y")
  - Another useful date object is type yearmon: as.yearmon(date column that is Date class)
  - data = zoo(data, dates as date class or yearmon class)

Brinson Attribution

- To calculate the monthly allocation, selection, and interaction effects, it's simple math
  - Selection Effect = Target weight * (Return of composite - Return of benchmark)
  - Allocation Effect = Benchmark Return * (Weight of composite - Target weight)
  - Interaction Effect = (Return of composite - Return of benchmark) * (Weight of composite - Target weight)

Table: Last 6 Rows of Monthly Brinson Attribution in %

	IRS.sel	HY.sel	EMD.sel	IRS.allo	HY.allo	EMD.allo	IRS.inter
Aug 2016	-0.08	-0.00	0.00	-0.01	-0.14	0.00	-0.01
Sep 2016	-0.02	-0.03	0.00	-0.00	-0.04	0.00	-0.00
Oct 2016	0.02	-0.14	0.00	-0.05	-0.03	0.00	0.00
Nov 2016	0.04	-0.01	0.00	-0.10	0.02	0.00	0.00
Dec 2016	0.02	-0.05	0.00	0.01	-0.08	0.00	0.00
Jan 2017	-0.01	-0.05	0.00	0.01	-0.06	0.00	-0.00

After Frongello Algorithm

Table: Frongello Output to Graph in %

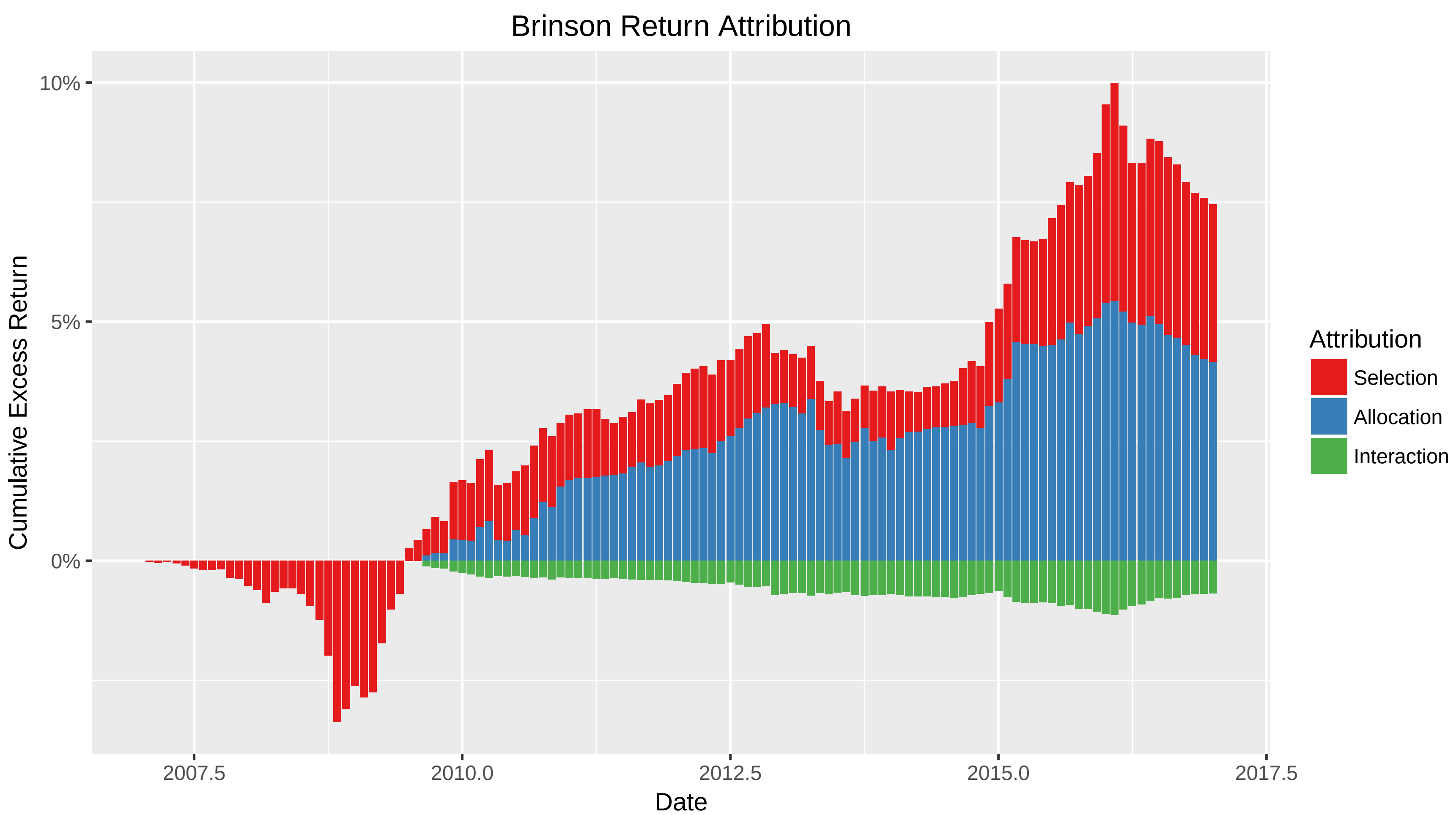
	Selection	Allocation	Interaction
Aug 2016	3.71	4.73	-0.79
Sep 2016	3.63	4.66	-0.78
Oct 2016	3.41	4.51	-0.72
Nov 2016	3.39	4.30	-0.70
Dec 2016	3.38	4.21	-0.69
Jan 2017	3.30	4.16	-0.68

Bar chart Used in Code

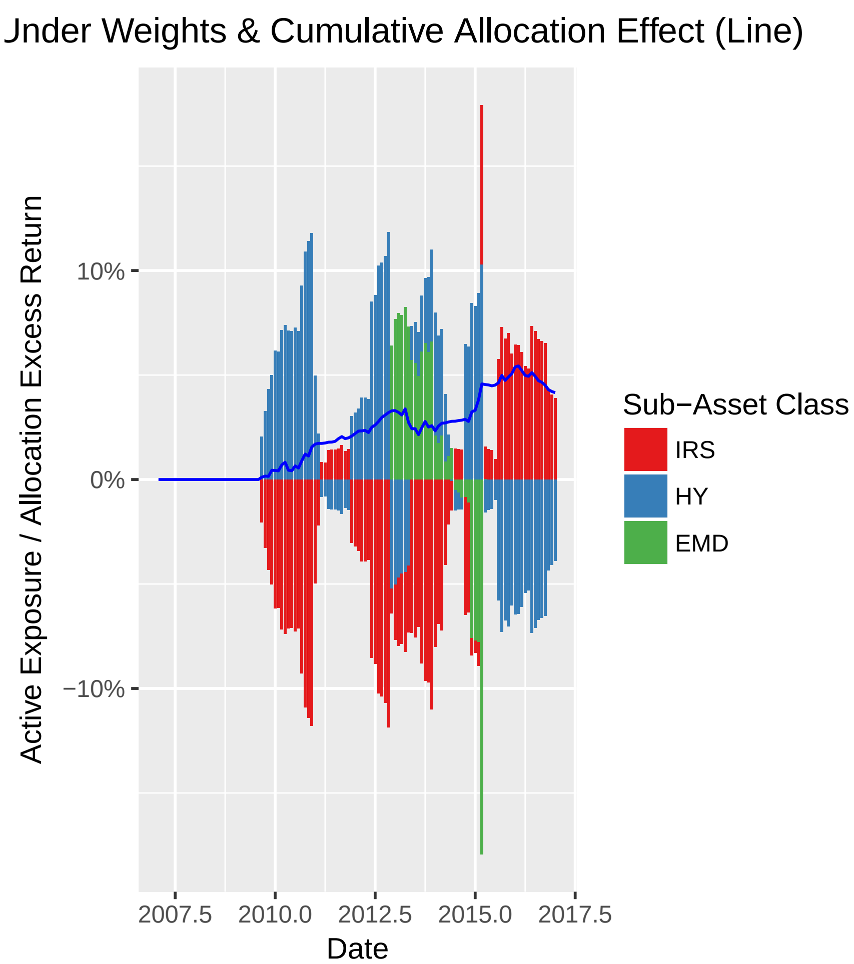
- Explanatoin of the code used to create the bar chart
  - The bar charts are stacked bar charts so to have both negative and positive values graphed, 2 bar charts are required
  - The data is split into 2 data frames using the subset function

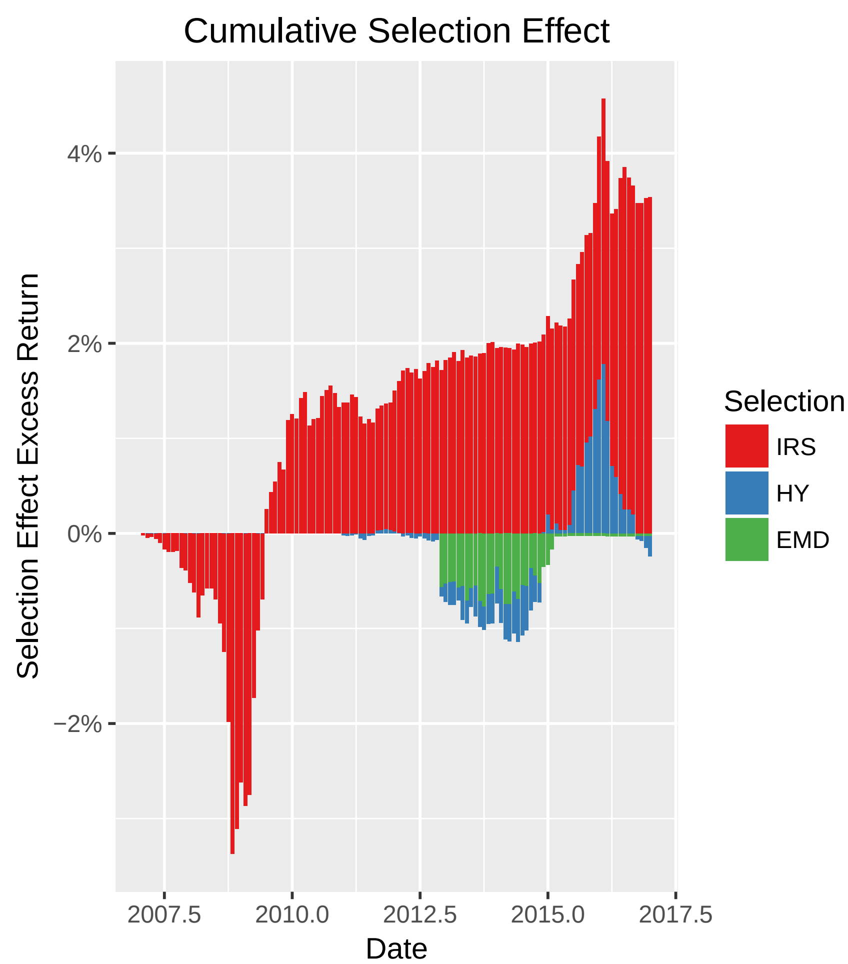
```
ggplot() + geom_bar(data=attrpos, aes(x=Date, y=Growth_of_Dollar,
fill=Attribution), stat="identity")+
geom_bar(data=attrneg, aes(x=Date, y=Growth_of_Dollar,
fill=Attribution), stat="identity")+
scale_fill_manual(values = colorpalette) +
scale_y_continuous(labels=percent)+ggtitle("title")+
theme(plot.title = element_text(hjust=0.5))
```

Summary Chart

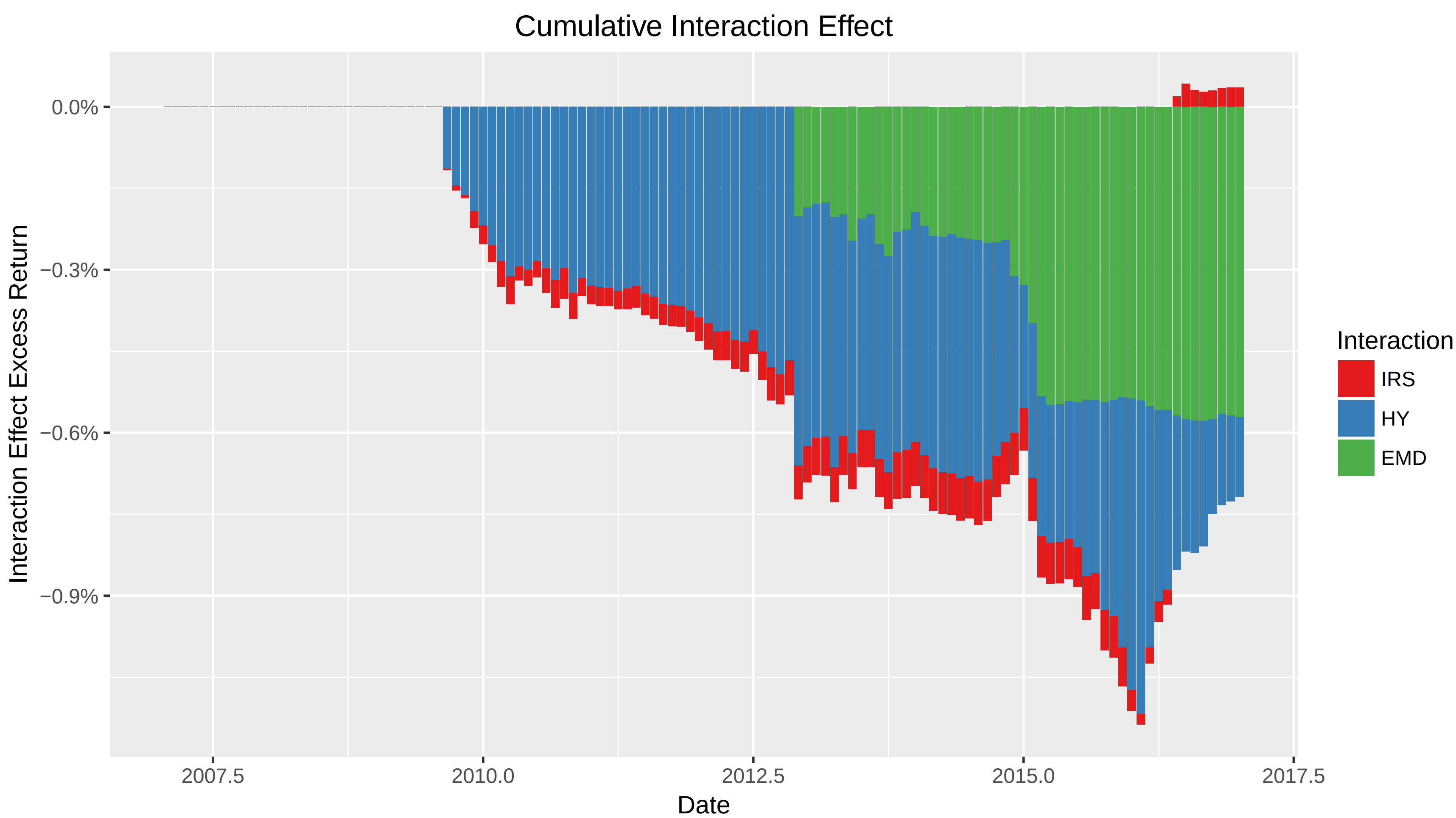


Allocation and Selection Effects Only





Interaction Effect



Outline

Attribution for a 60/40 Portfolio

- In finance, the classic portfolio is the 60/40 portfolio which is composed of 60% stocks and 40% bonds
- Utilizing simulated data shown below, complete an attribution analysis with targets of 60% stocks and 40% bonds
- Use the simulated data or create your own.
  - Change the weights or add other asset classes with their own return stream

Simulated Data

- The code to simulate data is fairly straigtforward
  - First thing is to run the function set.seed which ensures that results will be reproducible (pseudorandom number generator)
    - example: **set.seed(1234)**
  - The function **rnorm** simulates data with 3 inputs: number of observations, mean, and standard deviation
    - This creates a data set that has a specified number of observations, a specified mean, and standard deviation that is normally distributed

```
set.seed(1234)
stocks = rnorm(n=120,mean=.09, sd=.15)
bonds = rnorm(n=120, mean=.04, sd=.06)
data = zoo(cbind(stocks,bonds),timeline)
weight.s = rnorm(n=120, mean=.6, sd=.05)
weight.b = 1 - weight.s
weights = zoo(cbind(weight.s,weight.b),timeline)
target.weights = zoo(data.frame("Stock Target"=rep(.6,120),
                                "Bond Target"= rep(.4,120)),timeline)
sp500 = rnorm(n=120, mean=.085, sd=.16)
lba = rnorm(n=120, mean=.039, sd=.057)
bms = zoo(cbind(sp500,lba), timeline)
composite.actual = zoo(rowSums(data * weights), timeline)
```