

A Time Series Platform For The Tidyverse

Matt Dancho
Founder & CEO, Business Science
business-science.io

June 1, 2018



About Business Science



- We are **applications** people that build tools to **solve tough problems**
- We **serve** the data science **community**
- We **empower** data scientists via **education**



- Data Science For Business
 - BSPF
 - Tidy Eval
 - H2O
 - LIME
- Real World **CHURN** Problem!
- Enrollment
 - Student Satisfaction Rating: **9.1 / 10**
 - **DS4B_15** (15% OFF in June)
 - university.business-science.io

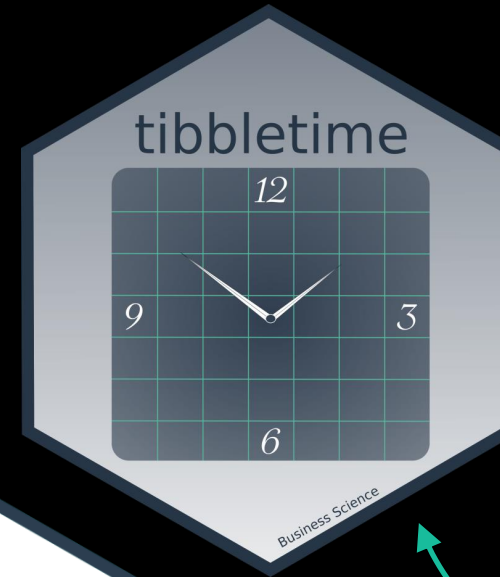
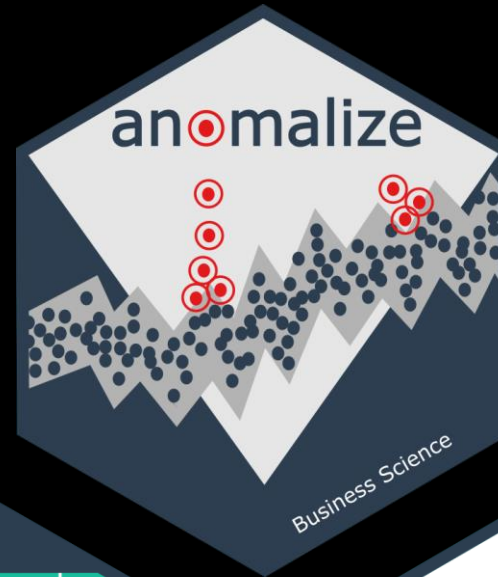
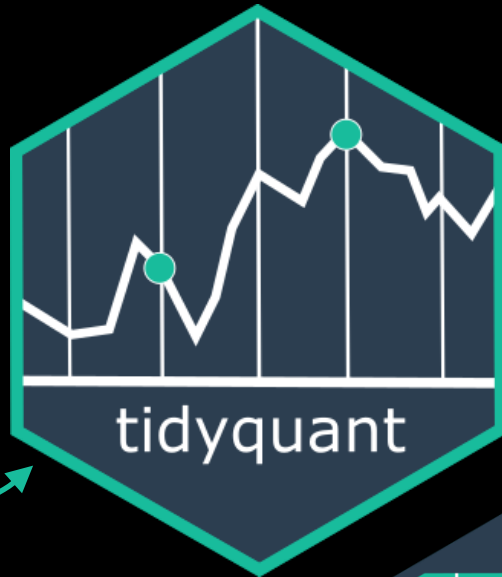


Open Source Packages

NEW!!
Anomaly
Detection



Most Popular
Financial Data
Stock/Portfolio
Analysis



Helps Do
Time-Based
Forecasting
w/ forecast



Time-Aware
tibbles

Time Series
ML

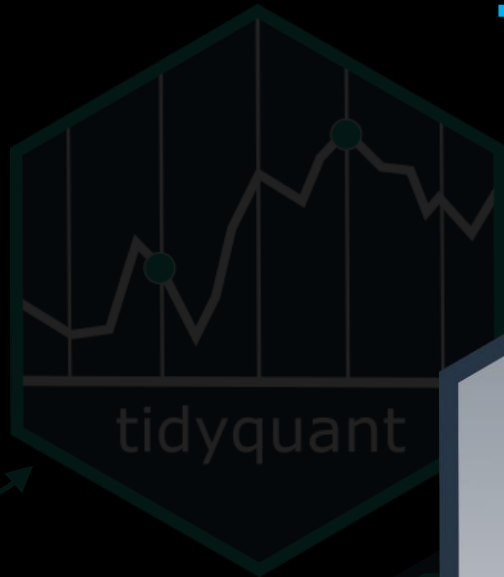
Open Source Time Series Packages

Time Series Platform

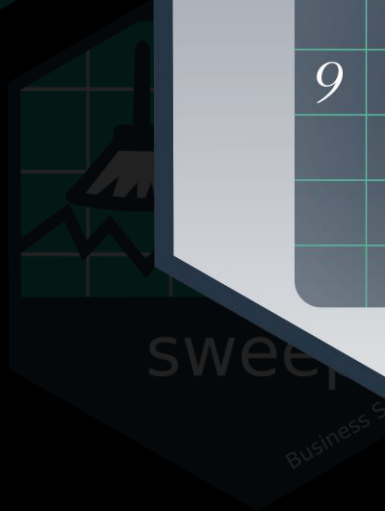


tidyverse

Most Popular
Financial Data
Stock/Portfolio
Analysis



Helps Do
Time-Based
Forecasting
w/ forecast



Time-Aware
tibbles
(dplyr)

Time Series
ML

Business Science

Business Science

Business Science

What Is **tibbletime**?



```
library(tibbletime)
library(dplyr)

# library(tidyquant)
# FANG <- tidyquant::tq_get(c("FB", "AMZN", "NFLX", "GOOG"))
data[FANG]
```

```
FANG_time <- FANG %>%
  tbl_time(date) %>%
  group_by(symbol)
```

FANG_time

Set up a time-aware
tibbletime object

```
> FANG_time
# A time tibble: 4,032 x 8
# Index:   date
# Groups:   symbol [4]
  symbol date      open  high  low close  volume adjusted
  <chr>  <date>    <dbl> <dbl> <dbl> <dbl>    <dbl>    <dbl>
1 FB    2013-01-02  27.4  28.2  27.4  28      69846400    28
2 FB    2013-01-03  27.9  28.5  27.6  27.8    63140600    27.8
3 FB    2013-01-04  28.0  28.9  27.8  28.8    72715400    28.8
4 FB    2013-01-07  28.7  29.8  28.6  29.4    83781800    29.4
5 FB    2013-01-08  29.5  29.6  28.9  29.1    45871300    29.1
6 FB    2013-01-09  29.7  30.6  29.5  30.6   104787700    30.6
7 FB    2013-01-10  30.6  31.5  30.3  31.3    95316400    31.3
8 FB    2013-01-11  31.3  32.0  31.1  31.7    89598000    31.7
9 FB    2013-01-14  32.1  32.2  30.6  31.0    98892800    31.0
10 FB   2013-01-15  30.6  31.7  29.9  30.1   173242600    30.1
# ... with 4,022 more rows
```

What Is **tibbletime**?



```
FANG_time %>%
```

```
collapse_by(period = "year") %>%
```

```
group_by(symbol, date) %>%
```

```
summarise(mean_adj = mean(adjusted))
```

Work with time index
using **tidyverse**
infrastructure

```
# A time tibble: 16 x 3
```

```
# Index: date
```

```
# Groups: symbol [?]
```

	symbol <chr>	date <date>	mean_adj <dbl>
1	AMZN	2013-12-31	298.
2	AMZN	2014-12-31	333.
3	AMZN	2015-12-31	478.
4	AMZN	2016-12-30	700.
5	FB	2013-12-31	35.5
6	FB	2014-12-31	68.8
7	FB	2015-12-31	88.8
8	FB	2016-12-30	117.
9	GOOG	2013-12-31	442.
10	GOOG	2014-12-31	561.
11	GOOG	2015-12-31	602.
12	GOOG	2016-12-30	743.
13	NFLX	2013-12-31	35.3
14	NFLX	2014-12-31	57.5
15	NFLX	2015-12-31	91.9
16	NFLX	2016-12-30	102.

What Is **tibbletime**?



- Scalable grouped analysis
at your fingertips
- Popular functions:
 - **collapse_by()**
 - **rollify()**
 - **filter_time()**
 - **as_period()**

Documentation:

<https://business-science.github.io/tibbletime/>

```
# A time tibble: 16 x 3
# Index:   date
# Groups:  symbol [?]
  symbol date      mean_adj
  <chr>  <date>      <dbl>
1 AMZN   2013-12-31    298.
2 AMZN   2014-12-31    333.
3 AMZN   2015-12-31    478.
4 AMZN   2016-12-30    700.
5 FB     2013-12-31     35.5
6 FB     2014-12-31     68.8
7 FB     2015-12-31     88.8
8 FB     2016-12-30    117.
9 GOOG   2013-12-31    442.
10 GOOG   2014-12-31    561.
11 GOOG   2015-12-31    602.
12 GOOG   2016-12-30    743.
13 NFLX   2013-12-31     35.3
14 NFLX   2014-12-31     57.5
15 NFLX   2015-12-31     91.9
16 NFLX   2016-12-30    102.
```


NEW: Business Science Labs

PROJECT 001:

Algorithmic Trading
Backtest Optimization
with
Quantopian's Zipline
in R

tibbletime

+

furrr

+

flyingfox



ALGORITHMIC TRADING: USING QUANTOPIAN'S ZIPLINE PYTHON LIBRARY IN R FOR BACKTESTING BY GRID SEARCH OPTIMIZATION

Written by Davis Vaughan and Matt Dancho on May 31, 2018

Categories: Business-Science-Labs

Tags: R-Project, R, Financial Analysis, Backtesting, Quantopian, Zipline, flyingfox, furrr, tibbletime

<http://www.business-science.io/business-science-labs/2018/05/31/backtesting-quantopian-zipline-tibbletime-furrr-flyingfox.html>



What Is **furrr**?

- **furrr** = future + purrr
- **future**
 - Parallel Processing
- **purrr**
 - Iteration
- **furrr**
 - Parallel Iteration

```
plan("multiprocess")
```

← New

```
results_tbl <- ma_grid tbl %>%  
  mutate(results = future_map(  
    .x = f,  
    .f = ~ fly_run_algorithm(  
      initialize = fly_initialize,  
      handle_data = .x,  
      start      = as.Date("2013-01-01"),  
      end        = as.Date("2016-01-01")  
    )  
  ))
```

← Replaces
map()

What Is **flyingfox**?



- Quantopian's **Zipline**
 - Backtesting Library for Algorithmic Trading Strategies
 - Python Only
- **reticulate**
 - Connects to Python
- **flyingfox**
 - Connects to **Zipline** using **reticulate**
 - Can Backtest ANY R Trading Strategy



Quantopian



Zipline

Usage Case: Backtested Order Optimizations

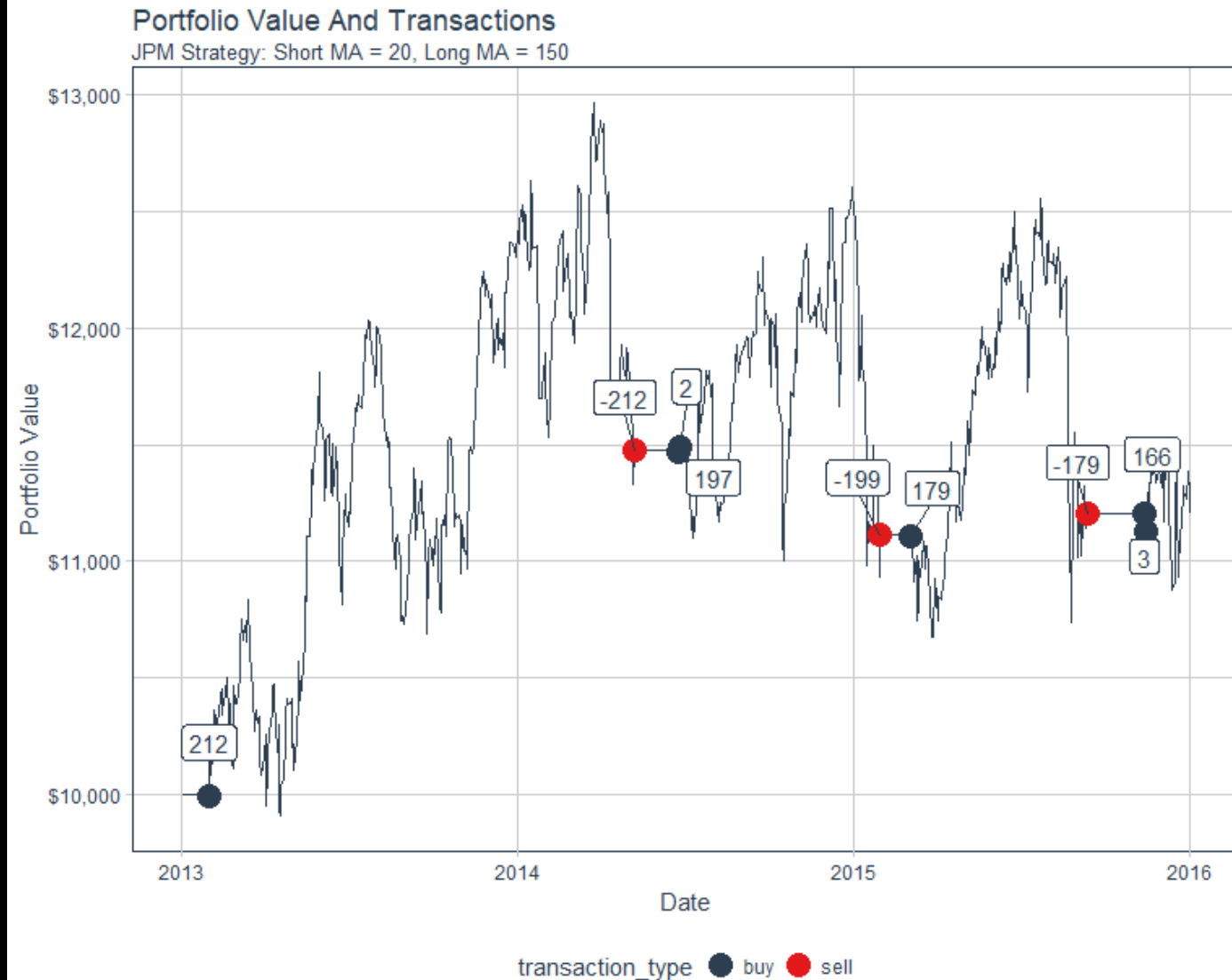


Zipline in R

JPM

Simple Moving
Average Strategy

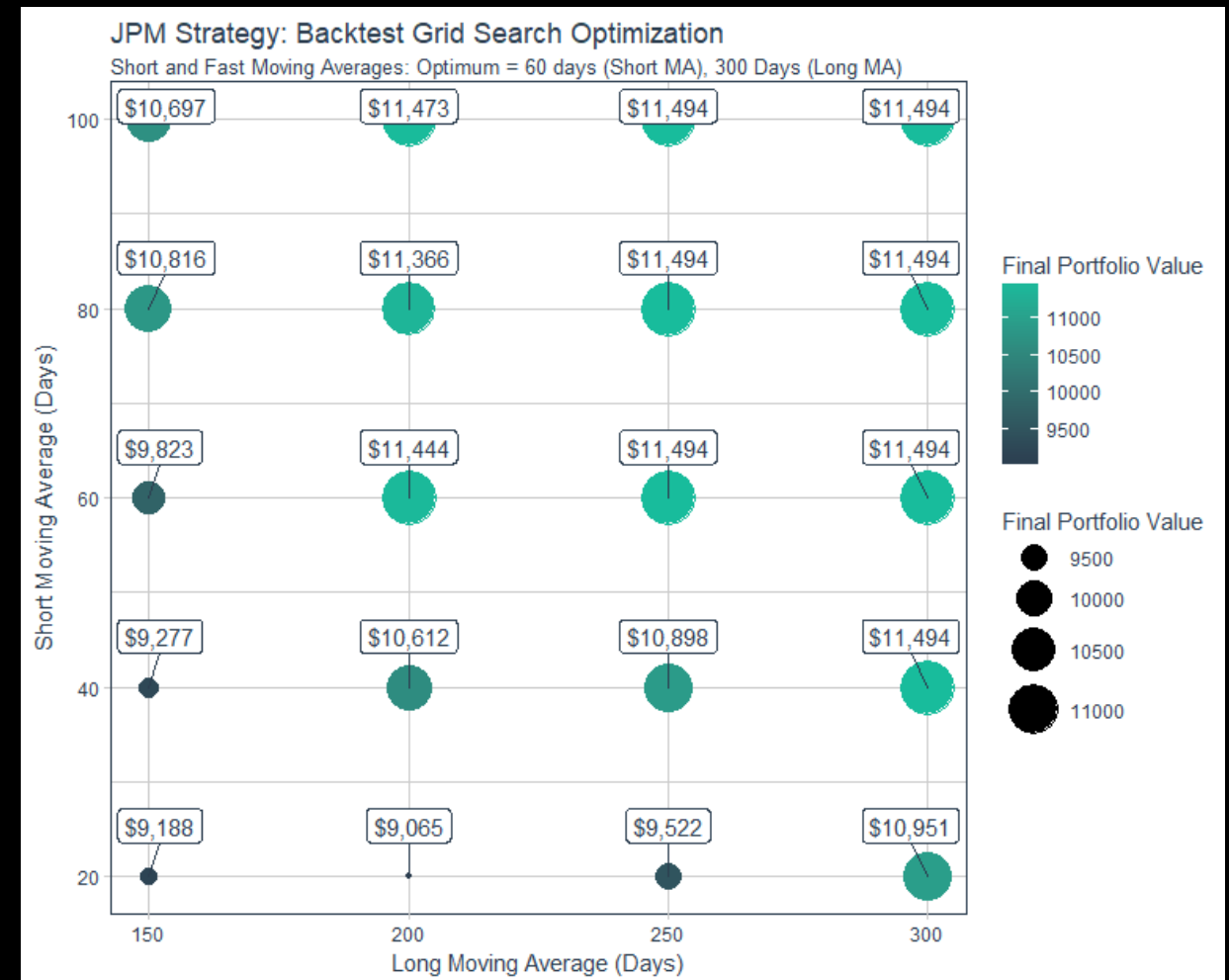
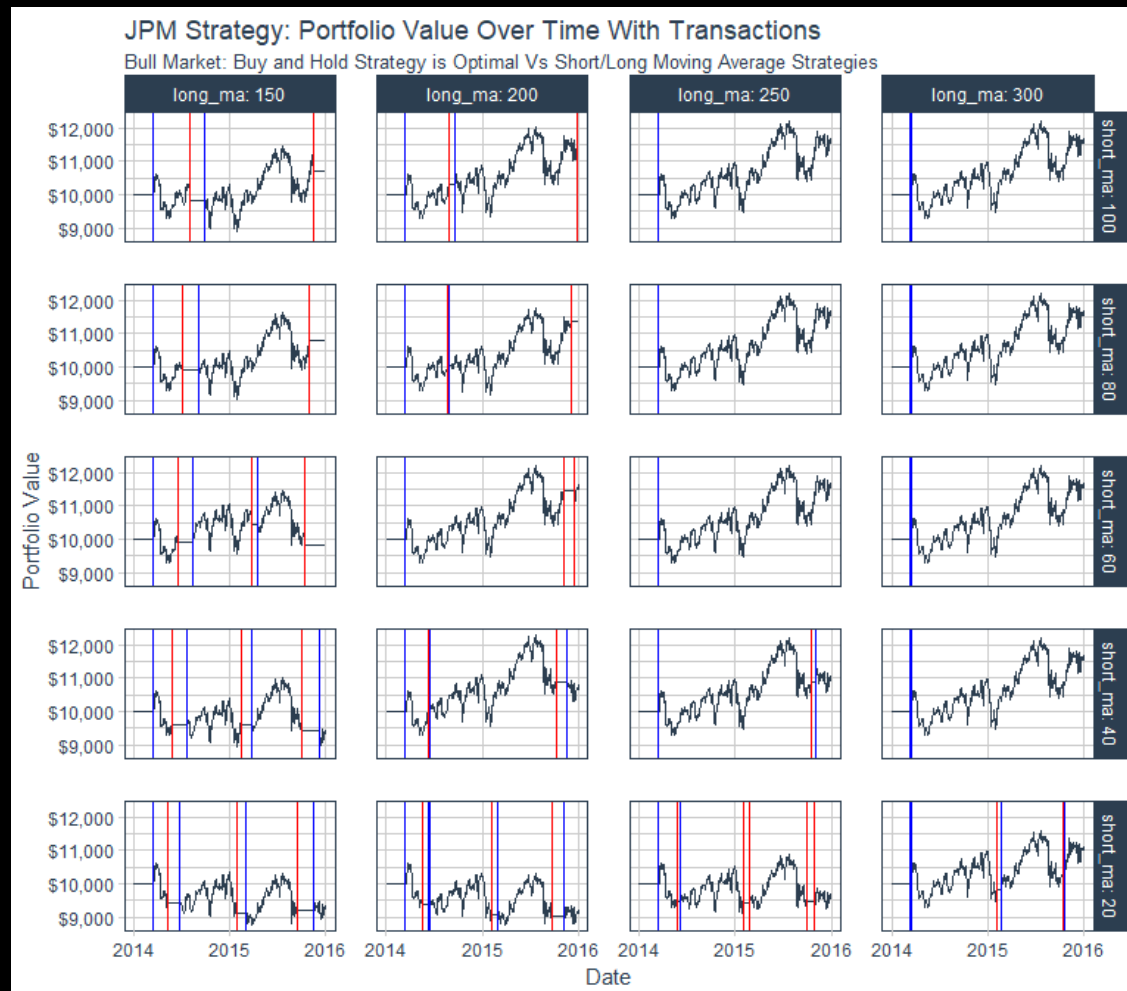
- Short MA = 20 Days
- Long MA = 150 Days



Usage Case: Backtested Order Optimizations



Optimized Backtests Using Grid Search – Parallelized With furrr



Try Backtesting With **flyingfox**



Experiment with **flyingfox** on
RStudio Cloud

<https://rstudio.cloud/project/38291>

The screenshot shows the RStudio Cloud interface for a project named 'flyingfox-demo-lab'. The code editor displays R code for backtesting a trading strategy using the 'flyingfox' package. The console shows the execution of the code, and the environment pane lists the loaded packages and data objects. A plot titled 'Portfolio value over time' is visible in the bottom right corner.

```
# Or graph some metrics
library(ggplot2)
library(dplyr)
library(tidyverse)
library(flyingfox)

orders <- performance %>%
  filter(map_lgl(orders, ~ length(.x) > 0)) %>%
  unnest()

buys <- filter(orders, filled > 0) %>% pull(date)
sells <- filter(orders, filled < 0) %>% pull(date)

performance %>%
  ggplot(aes(x = date, y = portfolio_value)) +
  geom_line() +
  geom_vline(xintercept = buys, color = "blue") +
  geom_vline(xintercept = sells, color = "red") +
  scale_y_continuous(labels = scales::dollar) +
  ggtitle("Portfolio value over time") +
  labs(x = "", y = "Portfolio Value") +
  theme_minimal()
```

The environment pane shows the following data objects:

- orders: 18 obs. of 14 variables
- performance: 73 obs. of 41 variables
- fly_handle_data: function (context, data)
- fly_initialize: function (context)

The plot titled 'Portfolio value over time' shows the portfolio value over time, with vertical lines indicating buy and sell points. The y-axis is labeled 'Portfolio Value' and ranges from \$11,000 to \$13,000. The x-axis shows dates from 2013 to 2016.

Business + Data Science



business-science.io

university.business-science.io

DS4B_15 for 15%-OFF through June