Financial Forecasting of Dow Jones Index Stocks

Partha Sarathi Mukherjee

5/24/2020

Contents

1 Overview

This project is related to "Choose your own" project for HarvardX: PH125.9x Data Science: Capstone course. The objective of the course is to build a complex machine learning algorithm to solve a real-life problem.

1.1 Introduction

The Dow Jones Industrial Average (DJIA), Dow Jones, or simply the Dow, is a stock market index that measures the stock performance of 30 large companies listed on stock exchanges in the United States. More details of the DOW can be found https://en.wikipedia.org/wiki/Dow_Jones_Industrial_Average.

University of Maryland project (https://archive.ics.uci.edu/ml/datasets/Dow+Jones+Index#) used 2 quarter (Jan to June 2011) of Dow Jones Industrial Average(DJIA) stocks to build a stock forecasting model. The goal of the project was to find the stock which will provide the greatest rate of return if you buy the stock next week open and close the position before the weekend.

Looking into only 2 quarters of data is very limiting. For this project, we will use 20 years(1999-2019) of weekly data. More precisely from the first Monday of 1999 to the last Friday in 2019. This big-time period will provide a robust model that will see the euphoria(2000 and 2007) and despair (2002 and 2008).

1.2 Challenges

There are many challenges in forecasting Stock prices.

1.2.1 Individual Stock Issues

Forecasting stock movement in the future is very dicey as the future is unknown. Also, an individual stock is affected by many unforeseen events like a news report, natural calamity, earning surprises, etc. Also stock has asymmetric moves from planned events and explosive move on surprise events like earning surprise.

In this project, we will just accept these risks and assume that over time the deltas will balance out.

1.2.2 DOW issues

The DOW stock composition changed over the years and the events are documented here https://en.wikipedia.org/wiki/Historical_components_of_the_Dow_Jones_Industrial_Average. This will pose a challenge to the data needed to reflect these changes. We know that a stock tends to get bumped when added to DJIA and dumped when dropped off the index.

In this project, current DJIA listed stocks will be used but limit the data to the date the stock was added to DJIA. This will introduce survival bias.

1.3 Data Ingestion

The current DJIA list consists of symbols MMM, AXP, AAPL, BA, CAT, CVX, CSCO, KO, DOW, XOM, GS, HD, IBM, INTC, JNJ, JPM, MCD, MRK, MSFT, NKE, PFE, PG, RTX, TRV , UNH, VZ, V, WMT, WBA, DIS

Details of each symbol and the companies associated are detailed in the components section of https://en.wikipedia.org/wiki /Dow_Jones_Industrial_Average

The weekly data for each DJIA stock will be obtained from Yahoo Finance. The steps are detailed below

1.3.1 MMM (3M Company)

3M Company was added to DJIA as Minnesota Mining & Manufacturing Company on Aug 9, 1976.

Minnesota Mining & Manufacturing Company was renamed as 3M Company on Jan 27, 2003.

MMM weekly stock prices from 1999 to 2019 can be download from the link

https://query1.finance.yahoo.com/v7/finance/download/MMM?period1=915408000&period2=1578009600&interval=1wk&events=history

1.3.2 AXP (American Express Company)

American Express Company was added to DJIA on Aug 30, 1982.

AXP weekly stock prices from 1999 to 2019 can be download from the link

https://query1.finance.yahoo.com/v7/finance/download/AXP?period1=915408000&period2=1578009600&interval=1wk&events=history

1.3.3 AAPL (Apple Inc.)

Apple Inc. was added to DJIA on Mar 19, 2015.

AAPL weekly stock prices from Mar 19, 2015, to 2019 can be downloaded from the link. But to better align with Weekly data we take the prices from the following week i.e., Mar 23, 2015.

https://query1.finance.yahoo.com/v7/finance/download/AAPL?period1=1427068800&period2=1578009600&interval=1wk&events=history

1.3.4 BA (The Boeing Company)

The Boeing Company was added to DJIA on Mar 12, 1987.

BA weekly stock prices from 1999 to 2019 can be download from the link

https://query1.finance.yahoo.com/v7/finance/download/BA?period1=915408000&period2=1578009600&interval=1wk&events=history

1.3.5 CAT (Caterpillar Inc.)

Caterpillar Inc. was added to DJIA on May 6, 1991.

CAT weekly stock prices from 1999 to 2019 can be download from the link

https://query1.finance.yahoo.com/v7/finance/download/CAT?period1=915408000&period2=1578009600&interval=1wk&events=history

1.3.6 CVX (Chevron Corporation)

Chevron Corporation was added to DJIA on Oct 30, 1985.

CVX weekly stock prices from 1999 to 2019 can be download from the link

https://query1.finance.yahoo.com/v7/finance/download/CVX?period1=915408000&period2=1578009600&interval=1wk&events=history

1.3.7 CSCO (Cisco Systems, Inc.)

Cisco Systems, Inc. was added to DJIA on Jun 8, 2009.

CSCO weekly stock prices from Jun 8, 2009, to 2019 can be download from the link

https://query1.finance.yahoo.com/v7/finance/download/CSCO?period1=1244419200&period2=1578009600&interval=1wk&events=history

1.3.8 KO (The Coca-Cola Company)

The Coca-Cola Company was added to DJIA on May 26, 1932, like Coca-Cola, Drug Inc, and dropped on Nov 20, 1935.

The Coca-Cola Company was again added back on Mar 12, 1987.

The stock is also called Knock Out for its symbol KO.

KO weekly stock prices from 1999 to 2019 can be download from the link

https://query1.finance.yahoo.com/v7/finance/download/KO?period1=915408000&period2=1578009600&interval=1wk&events=history

1.3.9 DOW (Dow Chemical Company)

Dow Chemical Company was added to DJIA on Nov 20, 1935, like DuPont.

On Sep 1, 2017, DuPont merged with the Dow Chemical Company under the name DowDuPont.

On Apr 2, 2019, DowDuPont spun off DuPont and was replaced by Dow Chemical Company.

DOW weekly stock prices from 1999 to 2019 can be download from the link

https://query1.finance.yahoo.com/v7/finance/download/DOW?period1=915408000&period2=1578009600&interval=1wk&events=history

1.3.10 XOM (Exxon Mobil Corporation)

Exxon Mobil Corporation was added to DJIA on Oct 1, 1928, as Standard Oil (NJ).

Jan 27, 2003, upon merging with Mobil, Exxon Corporation changed its name to Exxon Mobil Corporation.

Aug 9, 1976, Standard Oil (NJ) changed its name to Exxon Corporation.

XOM is the largest direct descendant of John D. Rockefeller's Standard Oil

XOM weekly stock prices from 1999 to 2019 can be download from the link

https://query1.finance.yahoo.com/v7/finance/download/XOM?period1=915408000&period2=1578009600&interval=1wk&events=history

1.3.11 GS (The Goldman Sachs Group, Inc.)

The Goldman Sachs Group, Inc. was added to DJIA on Sep 23, 2013.

GS weekly stock prices from Sep 23, 2013, to 2019 can be download from the link

https://query1.finance.yahoo.com/v7/finance/download/GS?period1=1379894400&period2=1578009600&interval=1wk&events=history

1.3.12 HD (The Home Depot)

The Home Depot was added to DJIA on Nov 1, 1999.

HD weekly stock prices from Nov 1, 1999, to 2019 can be download from the link

https://query1.finance.yahoo.com/v7/finance/download/HD?period1=941414400&period2=1578009600&interval=1wk&events=history

1.3.13 INTC (Intel Corporation)

Intel Corporation was added to DJIA on Nov 1, 1999.

INTC weekly stock prices from Nov 1, 1999, to 2019 can be download from the link

https://query1.finance.yahoo.com/v7/finance/download/INTC?period1=941414400&period2=1578009600&interval=1wk&events=history

1.3.14 IBM (International Business Machines Corporation)

International Business Machines Corporation on May 26, 1932.

IBM weekly stock prices from 1999 to 2019 can be download from the link

https://query1.finance.yahoo.com/v7/finance/download/IBM?period1=915408000&period2=1578009600&interval=1wk&events=history

1.3.15 JNJ (Johnson & Johnson)

Johnson & Johnson was added to DJIA on Mar 17, 1997.

JNJ weekly stock prices from 1999 to 2019 can be download from the link

https://query1.finance.yahoo.com/v7/finance/download/JNJ?period1=915408000&period2=1578009600&interval=1wk&events=history

1.3.16 JPM (JPMorgan Chase & Co.)

JPMorgan Chase & Co. was added to DJIA on May 6th, 1991.

Jan 27, 2003, J.P. Morgan & Company changed its name to JPMorgan Chase & Co.

JPM weekly stock prices from 1999 to 2019 can be download from the link

https://query1.finance.yahoo.com/v7/finance/download/JPM?period1=915408000&period2=1578009600&interval=1wk&events=history

1.3.17 MCD (McDonald's Corporation)

McDonald's Corporation was added to DJIA on Oct 30, 1985.

MCD weekly stock prices from 1999 to 2019 can be download from the link

https://query1.finance.yahoo.com/v7/finance/download/MCD?period1=915408000&period2=1578009600&interval=1wk&events=history

1.3.18 MRK (Merck & Co., Inc.)

Merck & Co., Inc. was added to DJIA on Jun 29, 1979.

MRK weekly stock prices from 1999 to 2019 can be download from the link

https://query1.finance.yahoo.com/v7/finance/download/MRK?period1=915408000&period2=1578009600&interval=1wk&events=history

1.3.19 MSFT (Microsoft Corporation)

Microsoft Corporation was added to DJIA on Nov 1, 1999.

MSFT weekly stock prices from Nov 1, 1999, to 2019 can be download from the link

https://query1.finance.yahoo.com/v7/finance/download/MSFT?period1=941414400&period2=1578009600&interval=1wk&events=history

1.3.20 NKE (Nike, Inc.)

Nike, Inc. was added to DJIA on Sep 23, 2013.

NKE weekly stock prices from Sep 23, 2013, to 2019 can be download from the link

https://query1.finance.yahoo.com/v7/finance/download/NKE?period1=1379894400&period2=1578009600&interval=1wk&events=history

1.3.21 PFE (Pfizer Inc.)

Pfizer Inc. was added to DJIA on Apr 8, 2004.

PFE weekly stock prices from Apr 8, 2004, to 2019 can be download from the link

https://query1.finance.yahoo.com/v7/finance/download/PFE?period1=1081382400&period2=1578009600&interval=1wk&events=history

1.3.22 PG (The Procter & Gamble Company)

The Procter & Gamble Company added to DJIA on May 26, 1932.

PG weekly stock prices from 1999 to 2019 can be download from the link

https://query1.finance.yahoo.com/v7/finance/download/PG?period1=915408000&period2=1578009600&interval=1wk&events=history

1.3.23 RTX (Raytheon Technologies)

United Aircraft was added to DJIA on Jul 18, 1930.

Apr 3rd, 2020, Raytheon Technologies was created after United Technologies merged with Raytheon Company.

Aug 9, 1976, United Aircraft changed its name to United Technologies Corporation.

Mar 4, 1939, United Aircraft is added back to DJIA.

Aug 15, 1933, United Aircraft was dropped off DJIA.

PG weekly stock prices from 1999 to 2019 can be download from the link

https://query1.finance.yahoo.com/v7/finance/download/RTX?period1=915408000&period2=1578009600&interval=1wk&events=history

1.3.24 TRV (The Travelers Companies, Inc.)

The Travelers Companies, Inc. was added to DJIA on Jun 8, 2009.

Travelers Inc. existed with a long history from 1863. Numerous mergers lead to Travelers losing its identity and former company stock may not reflect the current company so data from Jun 8, 2009, will be used.

TRV weekly stock prices from Jun 8, 2009, to 2019 can be download from the link

https://query1.finance.yahoo.com/v7/finance/download/TRV?period1=1244419200&period2=1578009600&interval=1wk&events=history

1.3.25 UNH (UnitedHealth Group Inc.)

UnitedHealth Group Inc. was added to DJIA on Sep 24, 2012.

UNH weekly stock prices from Sep 24, 2012, to 2019 can be download from the link

https://query1.finance.yahoo.com/v7/finance/download/UNH?period1=1348444800&period2=1578009600&interval=1wk&events=history

1.3.26 VZ (Verizon Communications, Inc.)

Verizon Communications, Inc. was added to DJIA on Apr 08, 2004.

VZ weekly stock prices from Apr 08, 2004, to 2019 can be download from the link

https://query1.finance.yahoo.com/v7/finance/download/VZ?period1=1081382400&period2=1578009600&interval=1wk&events=history

1.3.27 V (Visa Inc.)

Visa Inc. was added in DJIA at Sep 23, 2013

V weekly stock prices from Sep 23, 2013, to 2019 can be download from the link

https://query1.finance.yahoo.com/v7/finance/download/V?period1=1379894400&period2=1578009600&interval=1wk&events=history

1.3.28 WBA (Walgreens Boots Alliance)

Walgreens Boots Alliance was added to DJIA on Jun 26, 2018.

WBA weekly stock prices from Jun 26, 2018, to 2019 can be download from the link

https://query1.finance.yahoo.com/v7/finance/download/WBA?period1=1529971200&period2=1578009600&interval=1wk&events=history

1.3.29 WMT (Walmart)

Walmart was added to DJIA on Mar 17, 1997.

WMT weekly stock prices from 1999 to 2019 can be download from the link

https://query1.finance.yahoo.com/v7/finance/download/WMT?period1=915408000&period2=1578009600&interval=1wk&events=history

1.3.30 DIS (The Walt Disney Company)

The Walt Disney Company was added to DJIA on May 06, 1991.

DIS weekly stock prices from 1999 to 2019 can be download from the link

https://query1.finance.yahoo.com/v7/finance/download/DIS?period1=915408000&period2=1578009600&interval=1wk&events=history

1.4 Outcome Measurement

The outcome will be measured based on the percentage difference between next week's open vs close.

Performance of the Model will be measured in 2 ways:-

1. How well is the model outcome mimic the actual Outcome
2. How well the best-predicted outcome performed vs the actual outcome.

2 Methods and Analysis

The downloaded data starts in the first week of Jan from 4th Jan 1999 to 27th Dec 2019.

Downloading Raw Data from Yahoo Finance.

The downloaded data is stored in the "data" directory.

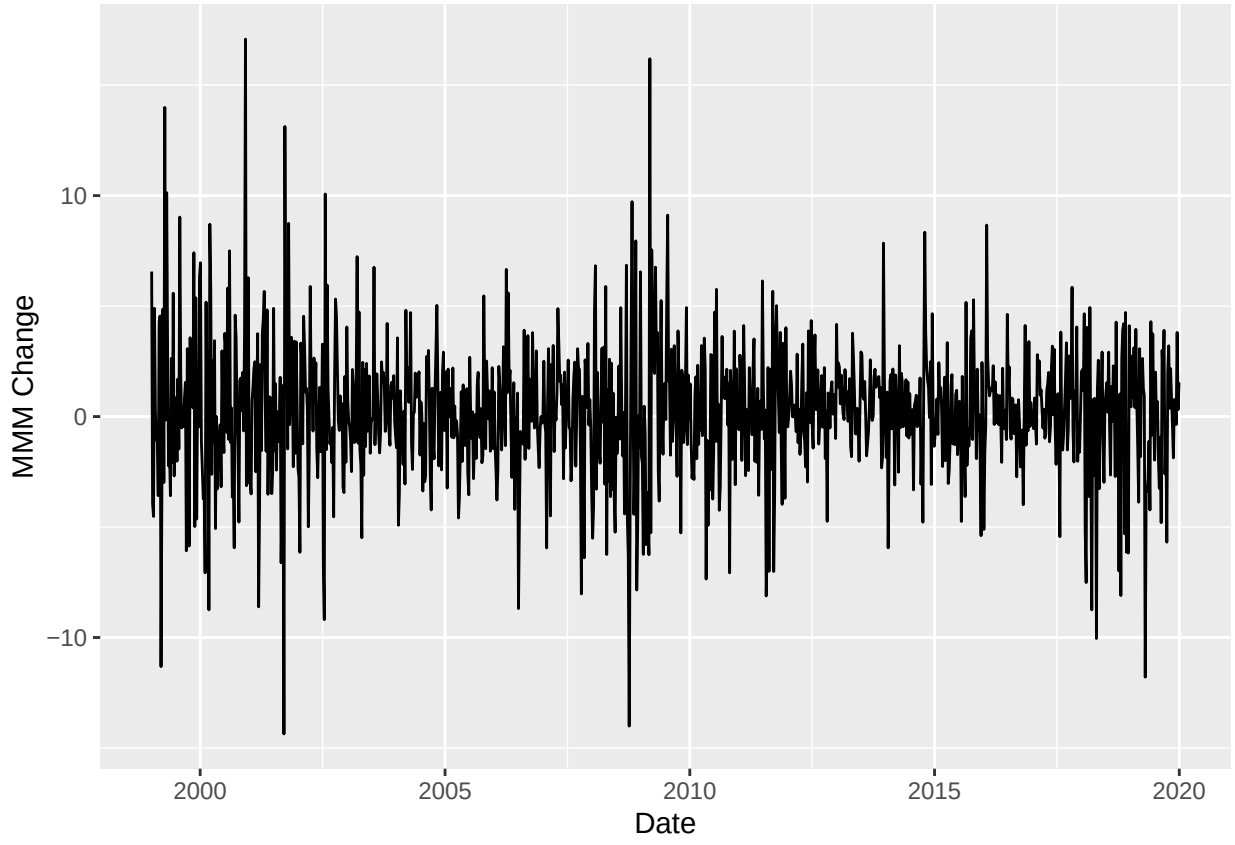
2.1 Exploratory Data Analysis

The first step is to make sure the data is clean and logical. The percentage of change every week is added to the data set. Analyze any move above or below 40% (Unusual move).

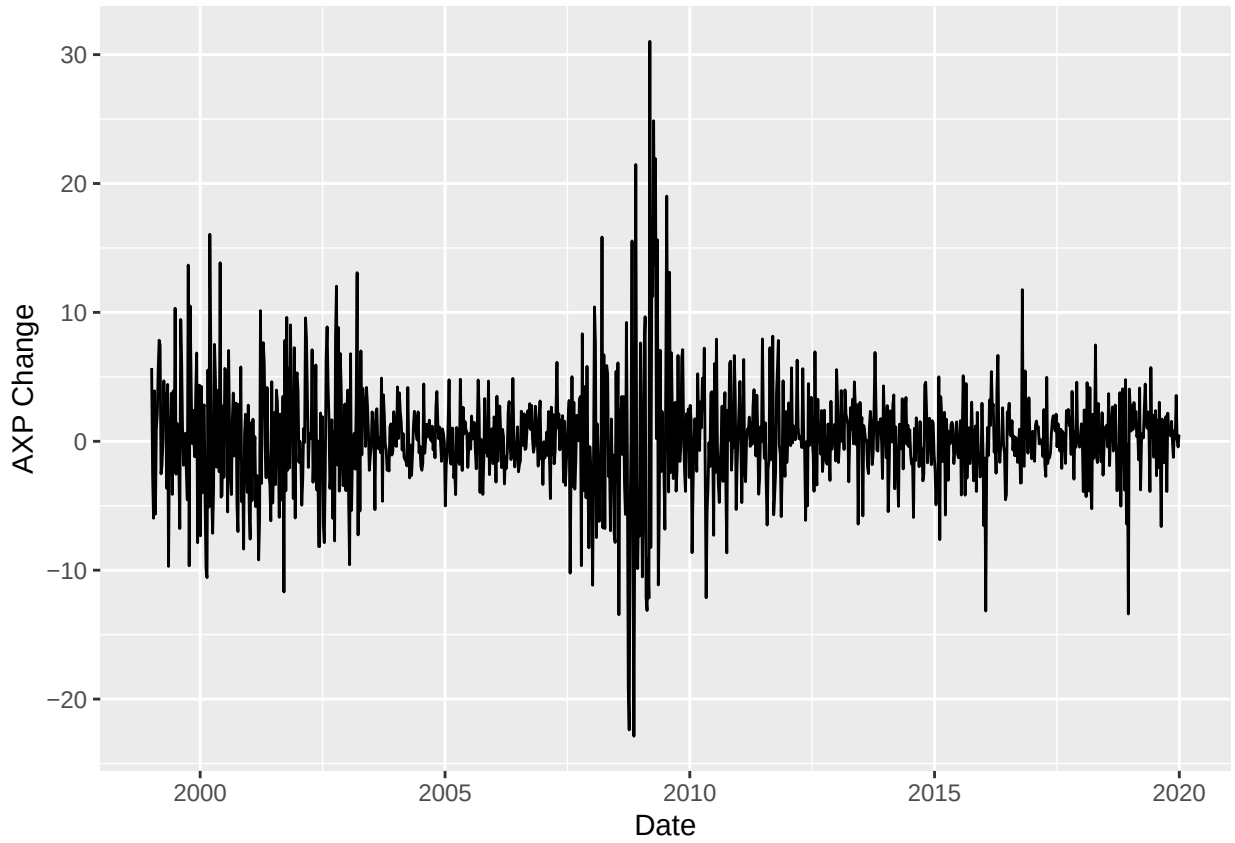
Dig deeper into every Unusual move manually. Fix the data is if there is an error.

For manual comparison, other data/chart sources like TradingView.com or google finance is used.

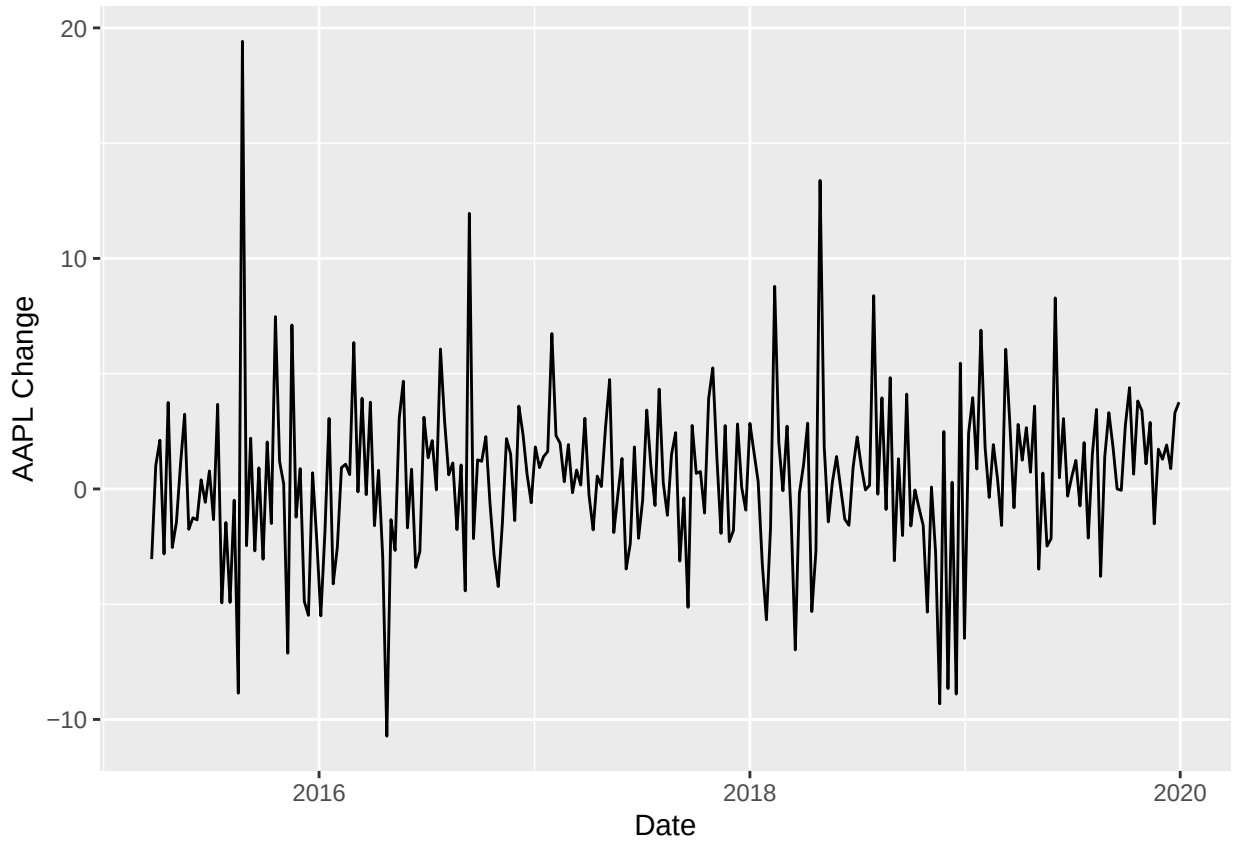
```
as.data.frame(data[,"MMM"]) %>% ggplot(aes(Date, Change, Ticker)) + geom_line() + ylab("MMM Change")
```



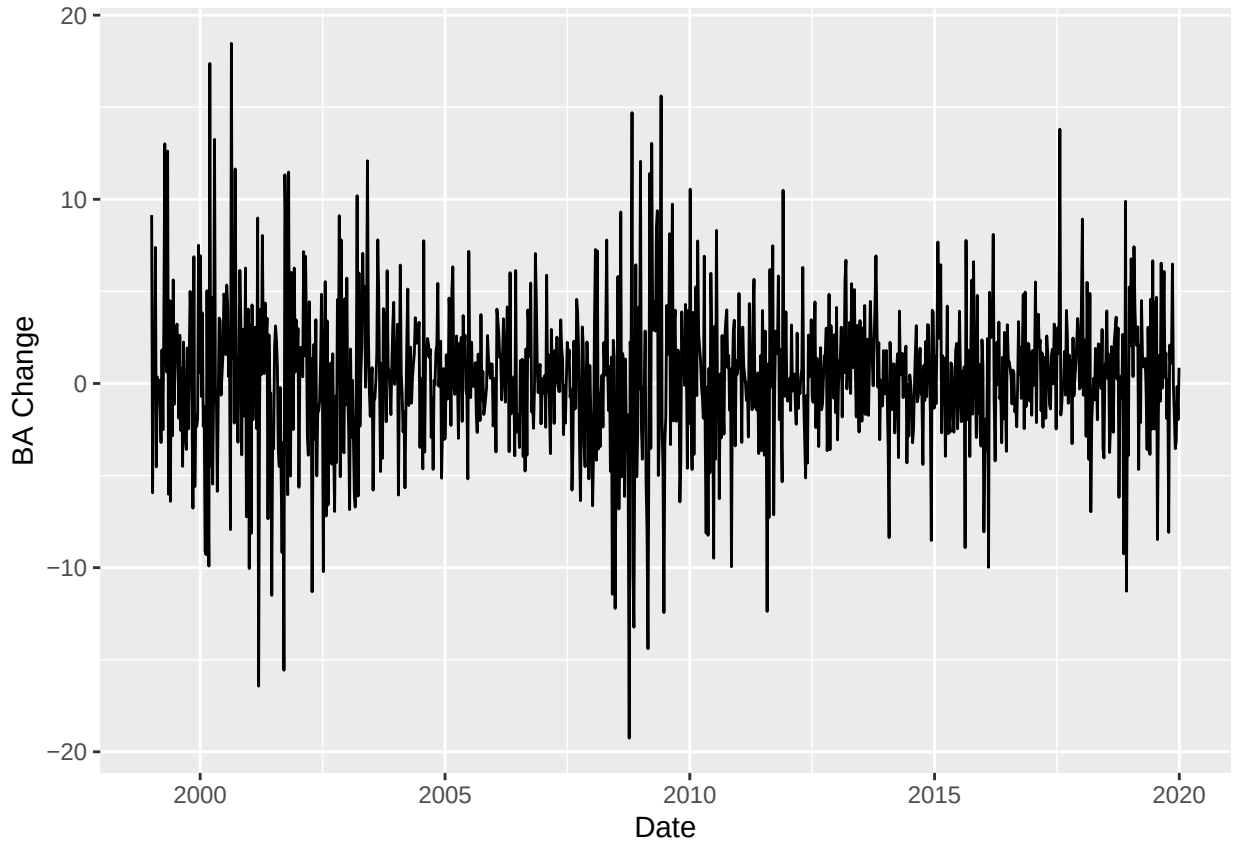

```
as.data.frame(data[,"AXP"]) %>% ggplot(aes(Date, Change)) + geom_line() + ylab("AXP Change")
```



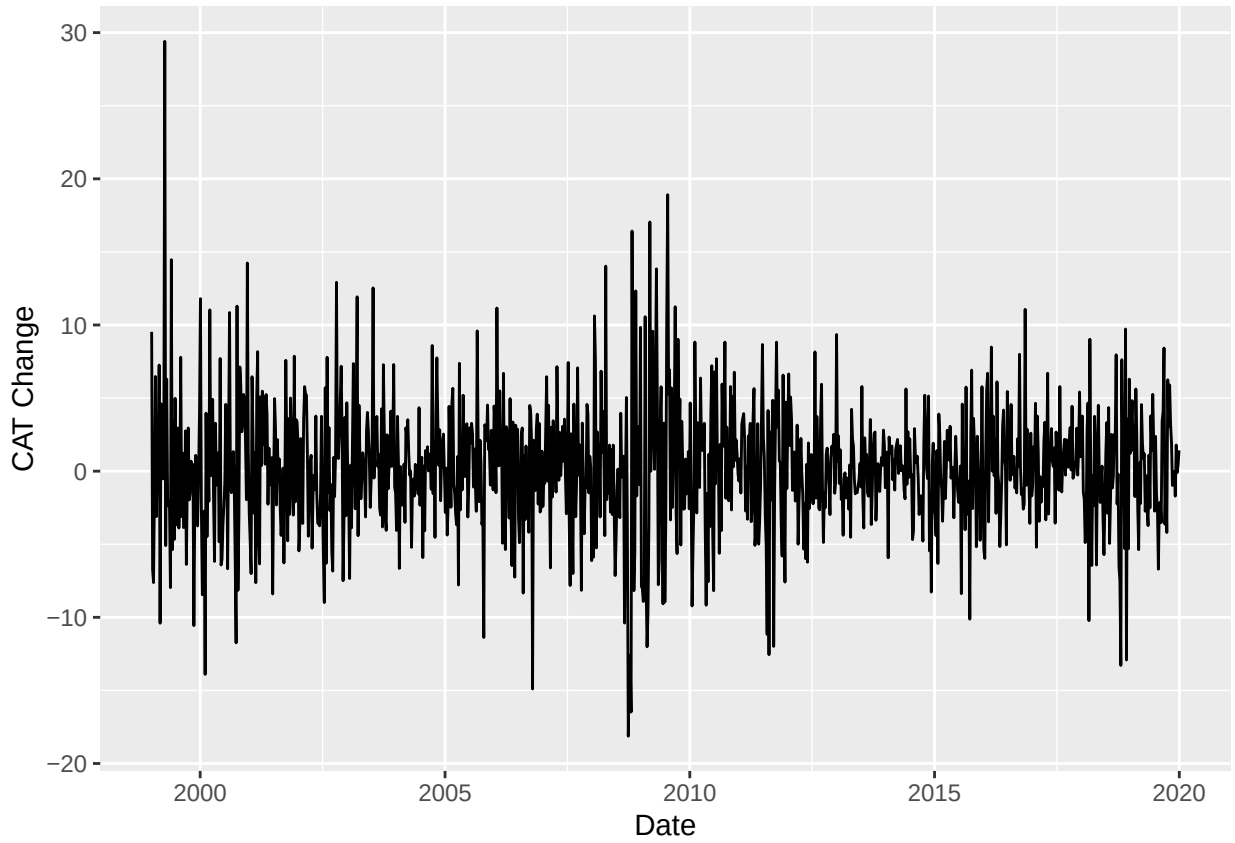

```
as.data.frame(data[,"AAPL"]) %>% ggplot(aes(Date, Change)) + geom_line() + ylab("AAPL Change")
```



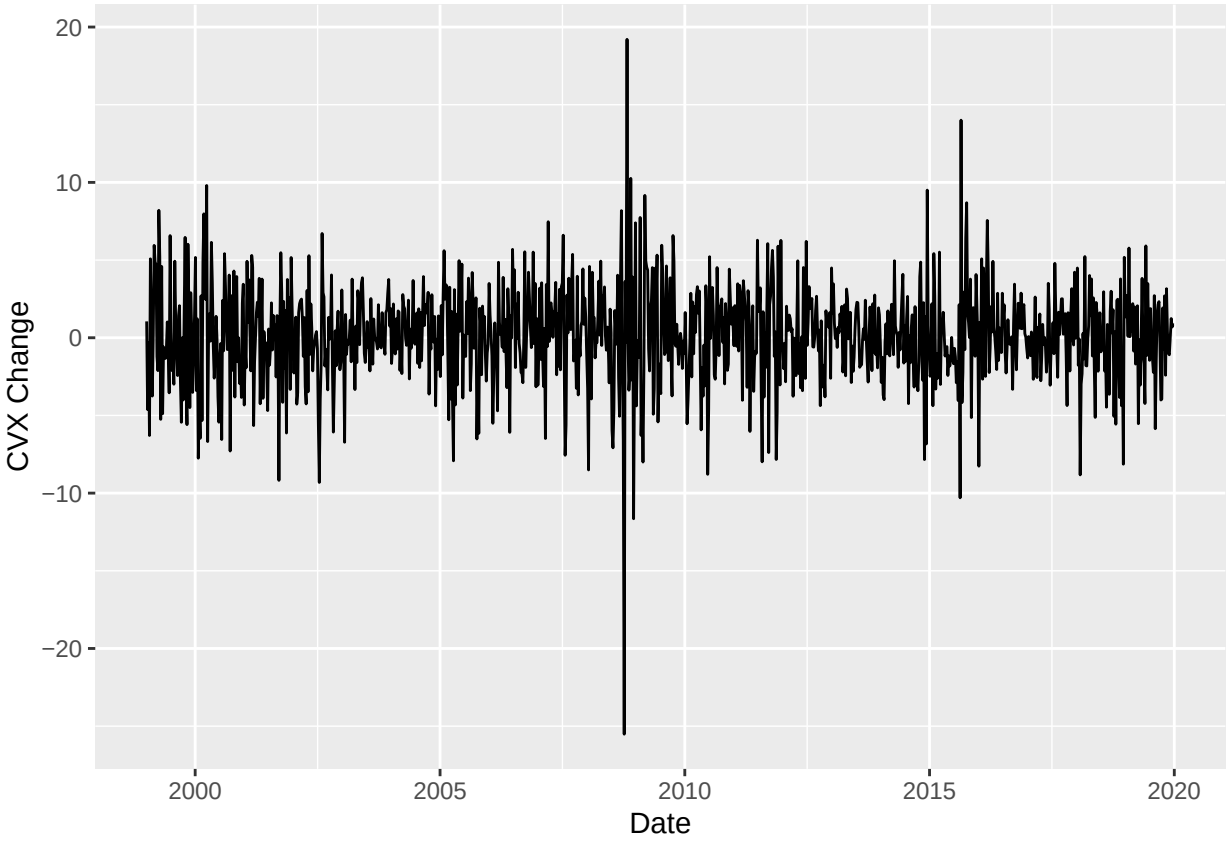

```
as.data.frame(data[,"BA"]) %>% ggplot(aes(Date, Change)) + geom_line() + ylab("BA Change")
```



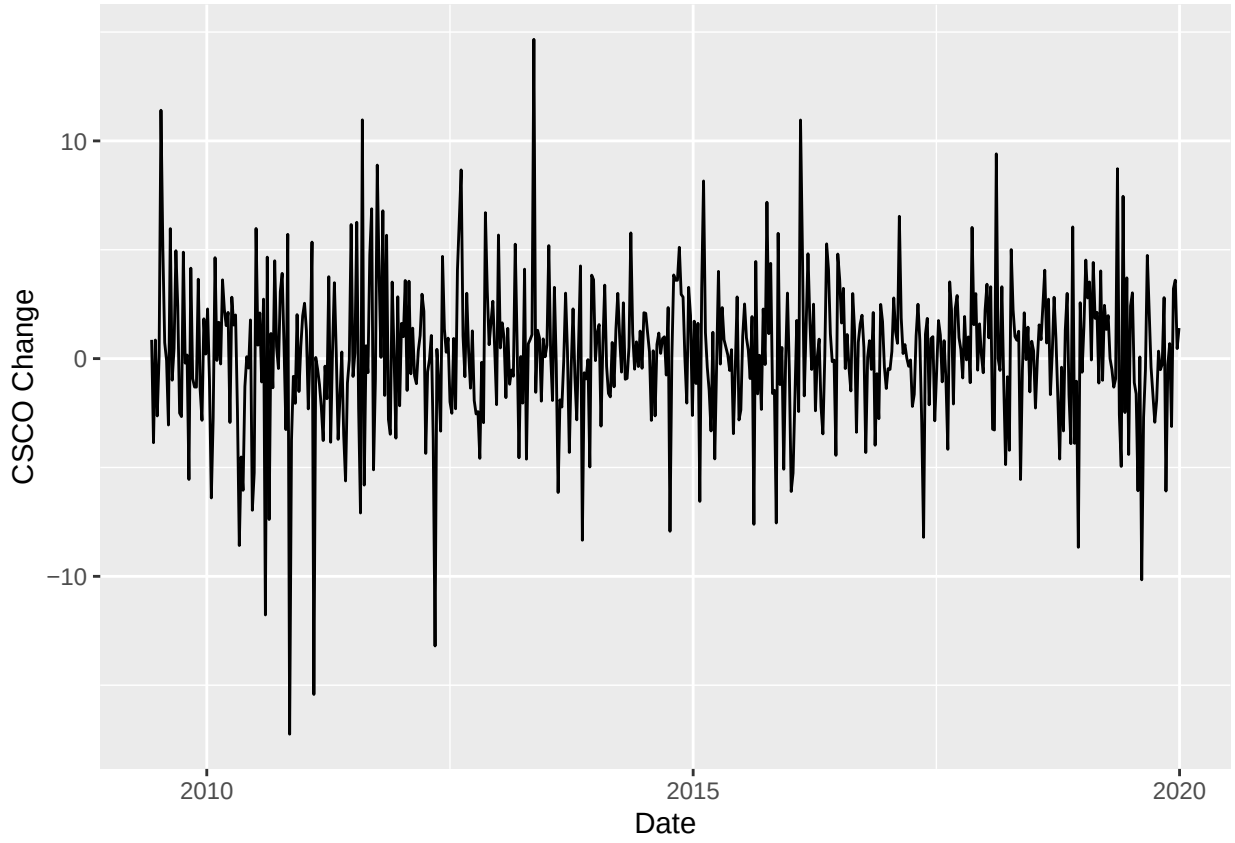

```
as.data.frame(data[,"CAT"]) %>% ggplot(aes(Date, Change)) + geom_line() + ylab("CAT Change")
```



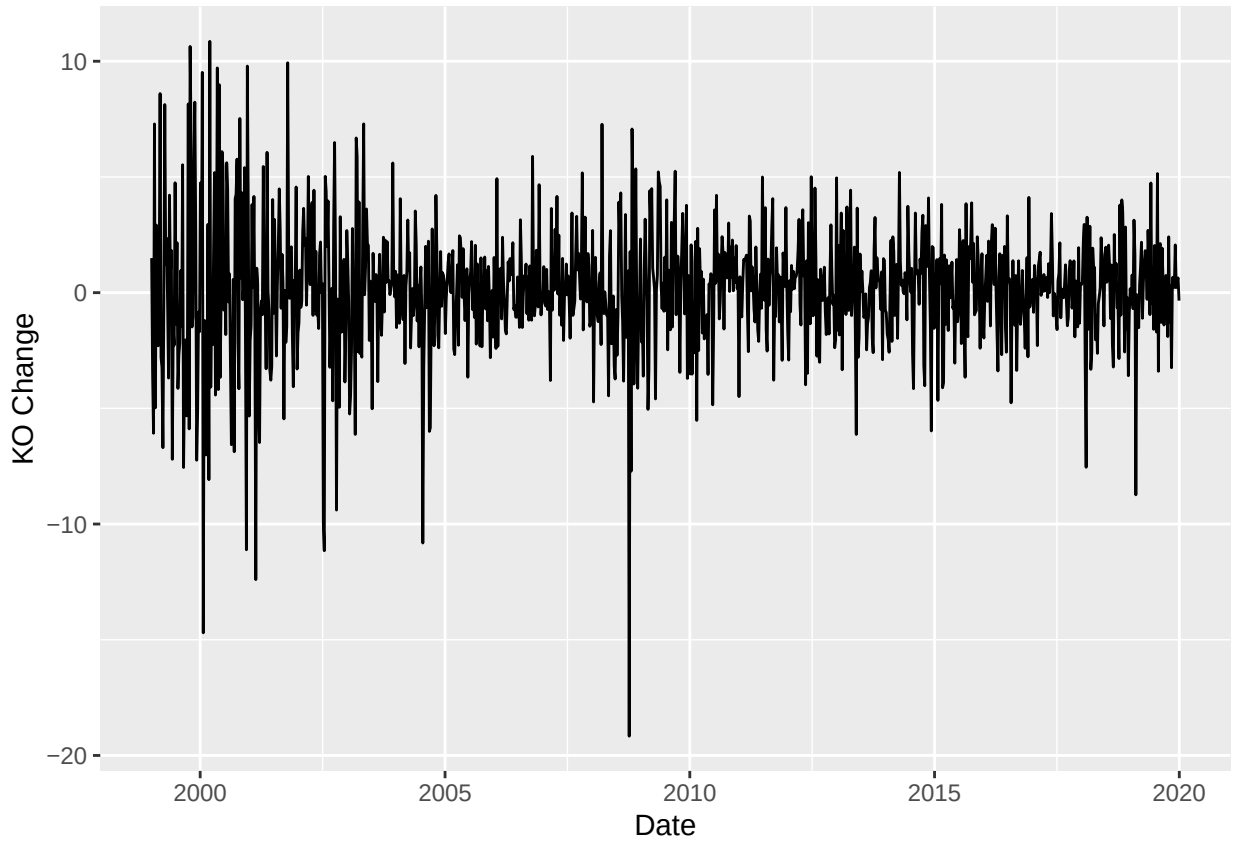

```
as.data.frame(data[,"CVX"]) %>% ggplot(aes(Date, Change)) + geom_line() + ylab("CVX Change")
```



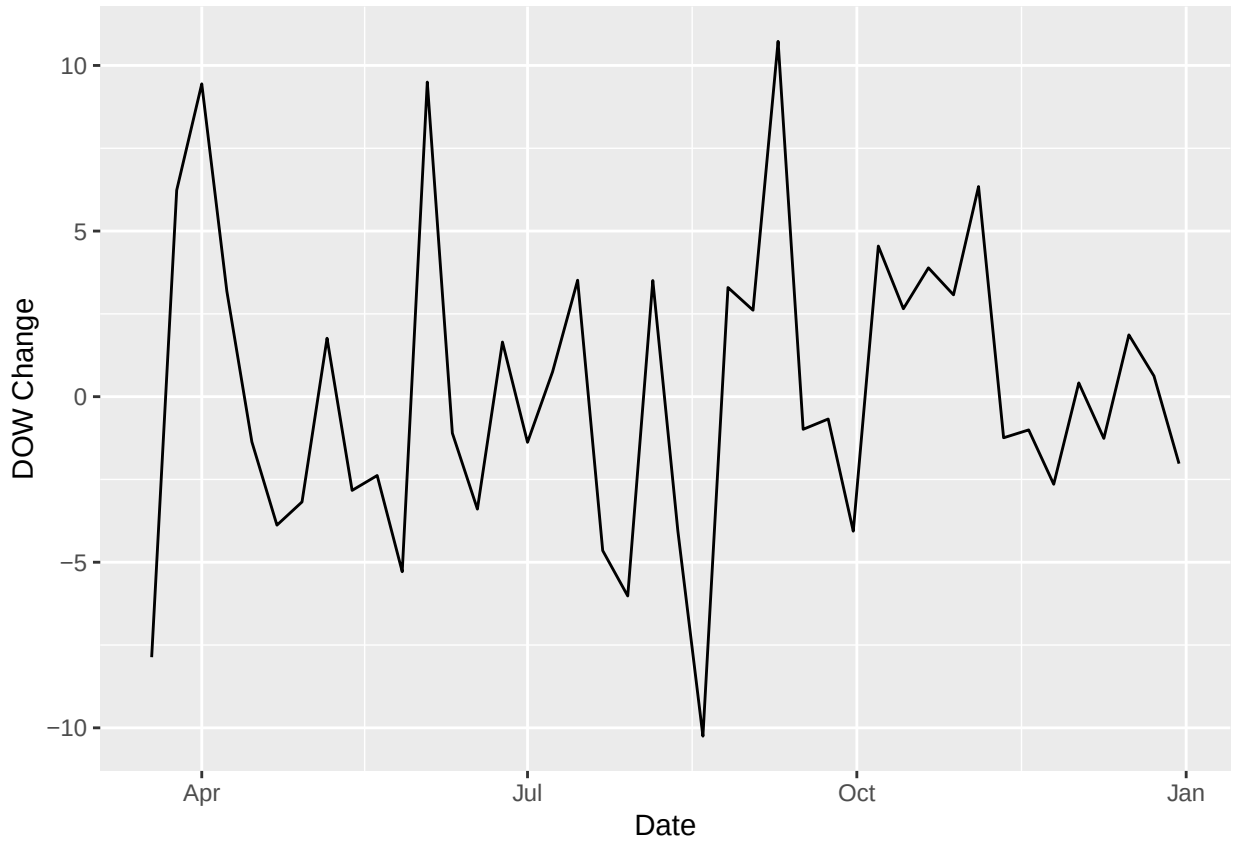

```
as.data.frame(data[,"CSCO"]) %>% ggplot(aes(Date, Change)) + geom_line() + ylab("CSCO Change")
```



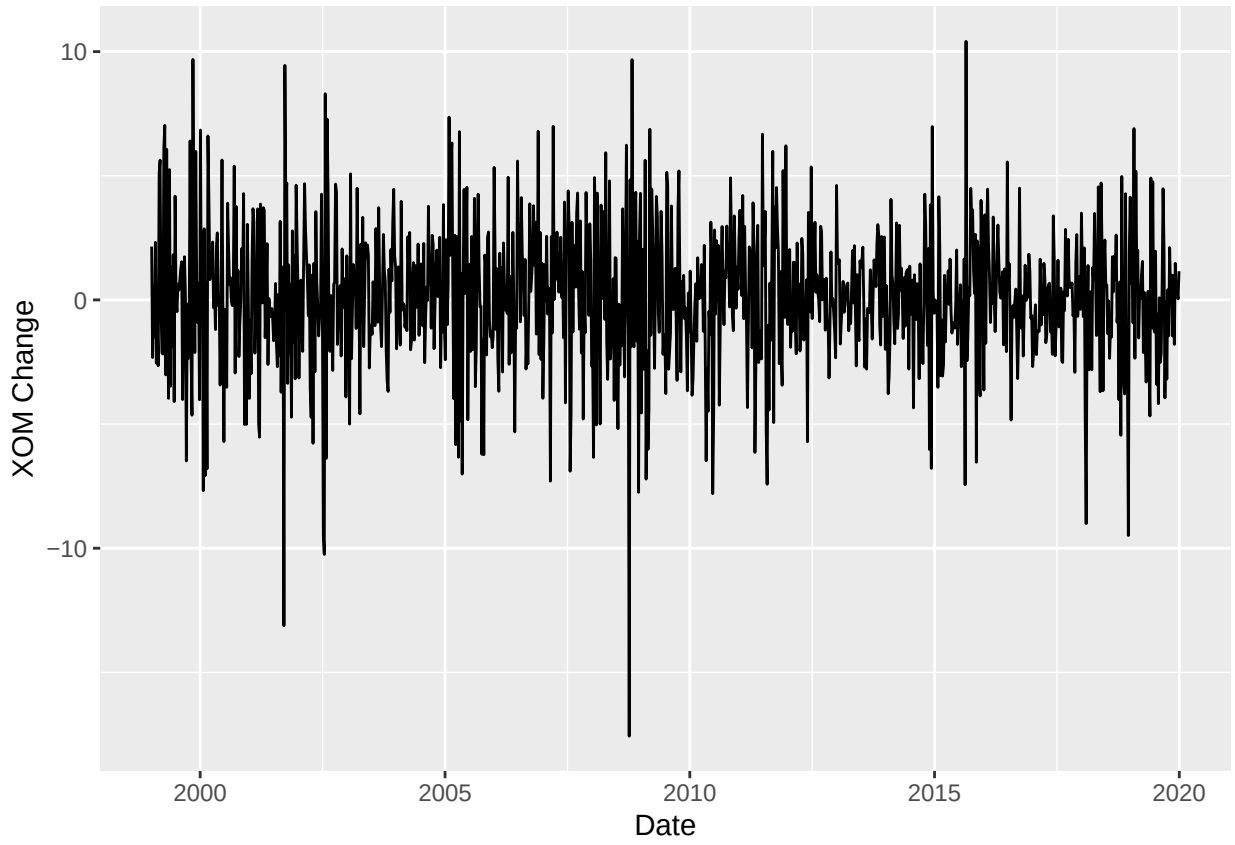

```
as.data.frame(data[,"KO"]) %>% ggplot(aes(Date, Change)) + geom_line() + ylab("KO Change")
```



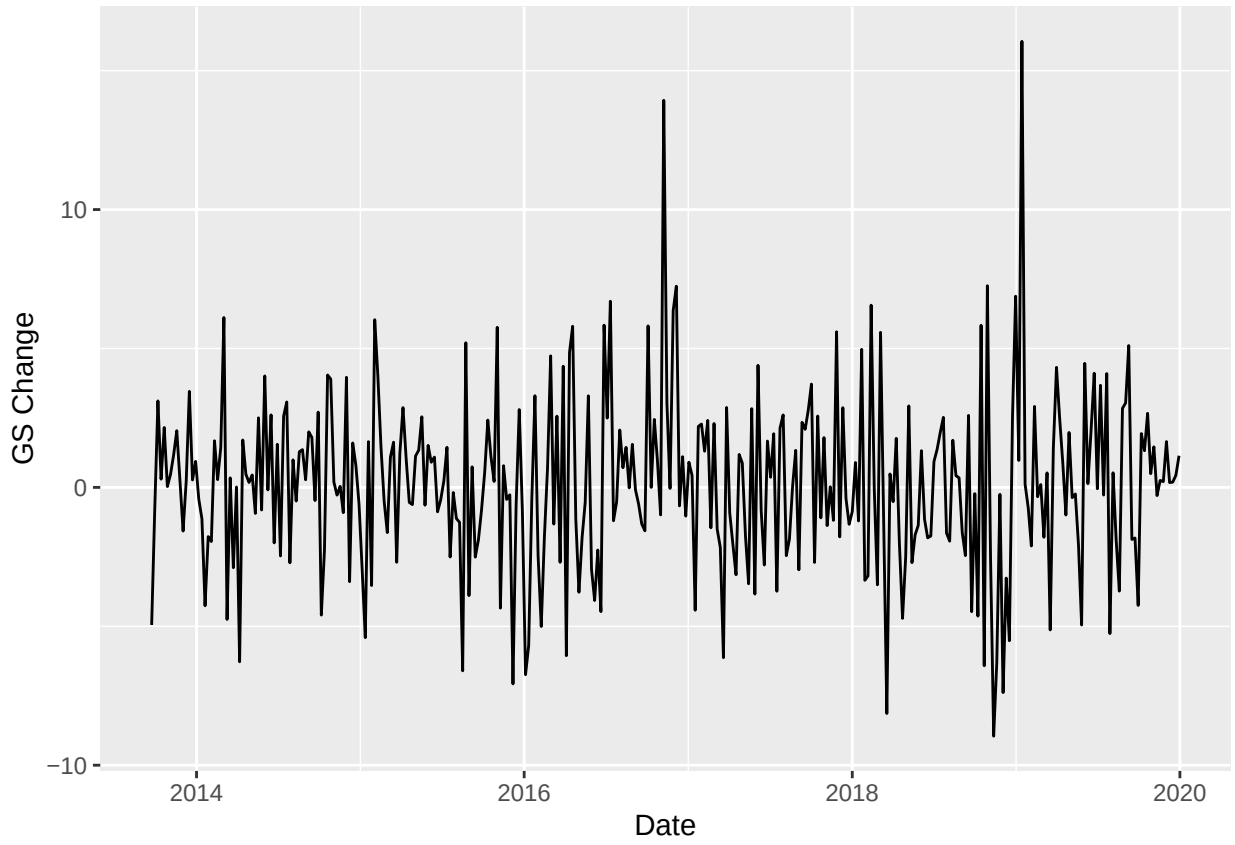

```
as.data.frame(data[,"DOW"]) %>% ggplot(aes(Date, Change)) + geom_line() + ylab("DOW Change")
```



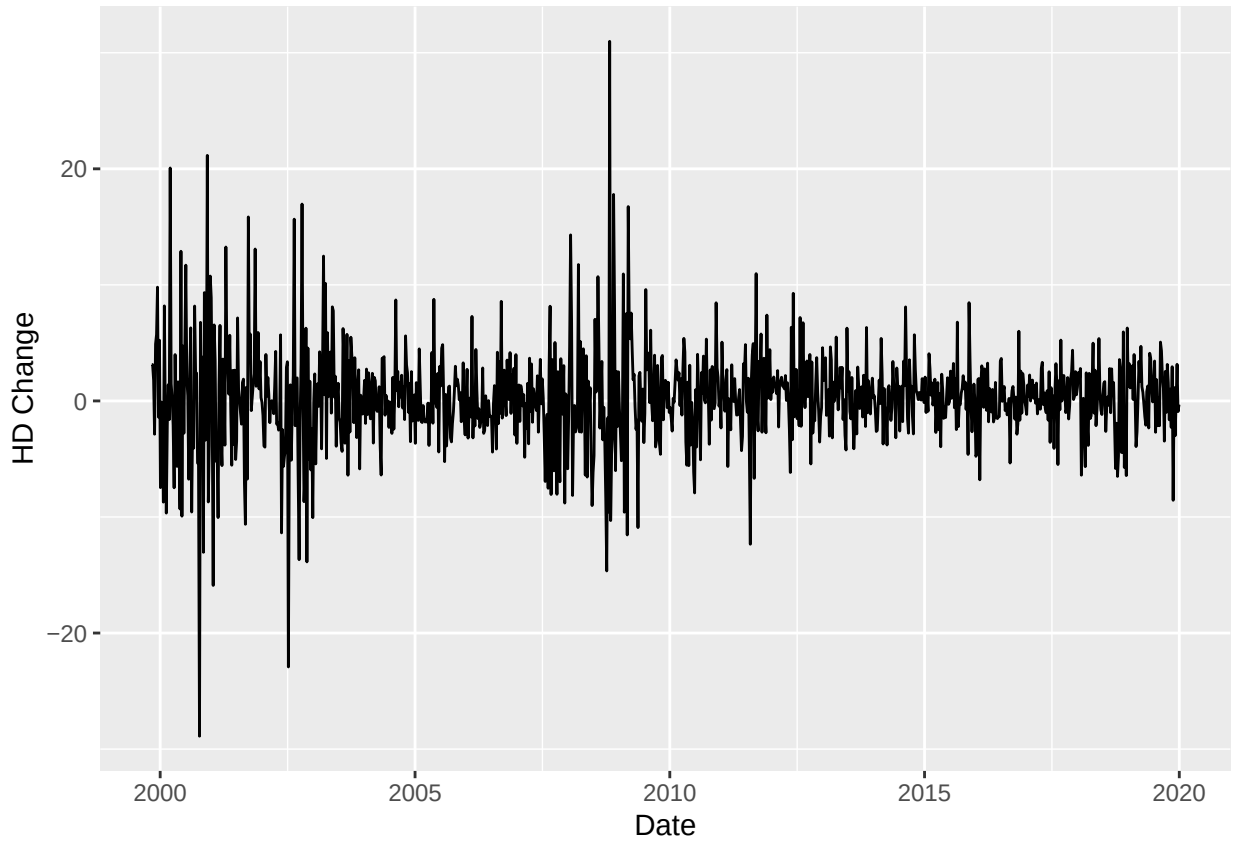

```
as.data.frame(data[,"XOM"]) %>% ggplot(aes(Date, Change)) + geom_line() + ylab("XOM Change")
```



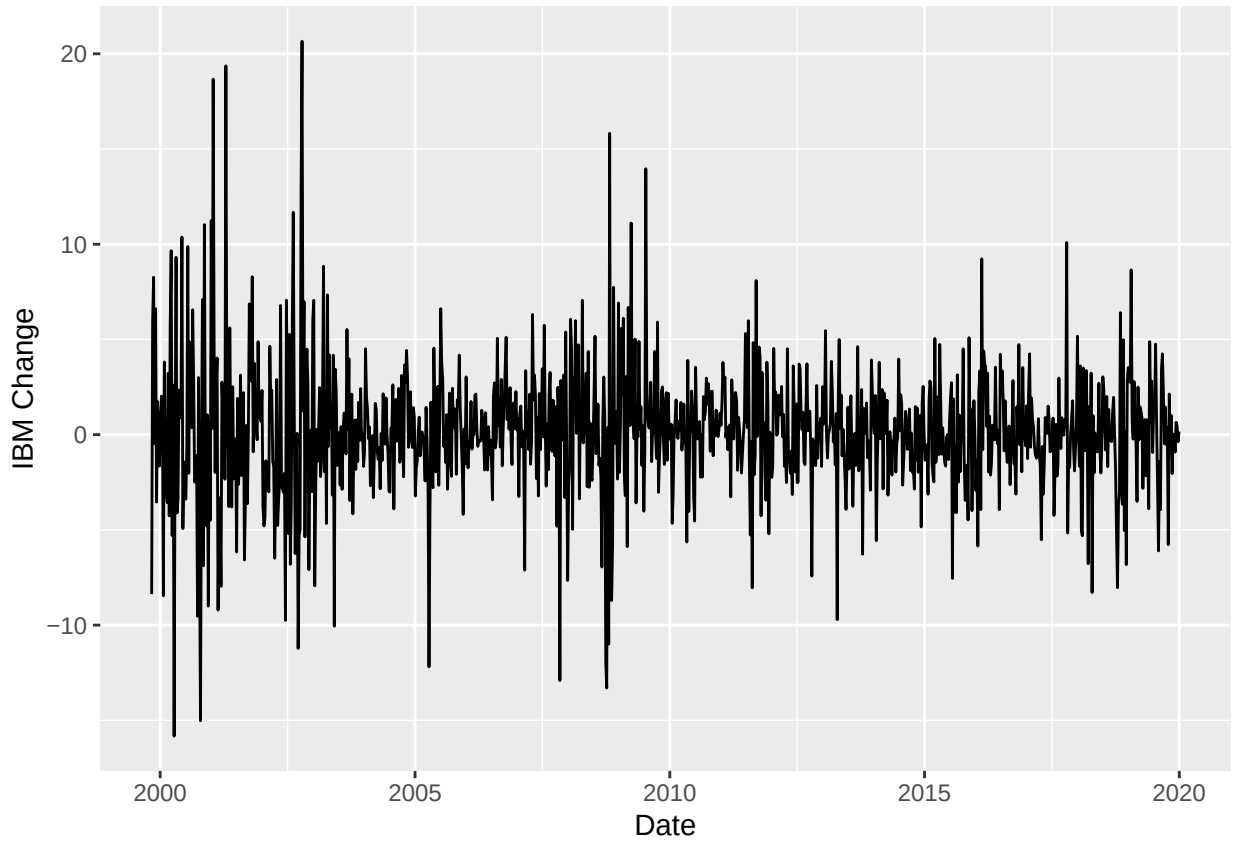

```
as.data.frame(data[,"GS"]) %>% ggplot(aes(Date, Change)) + geom_line() + ylab("GS Change")
```



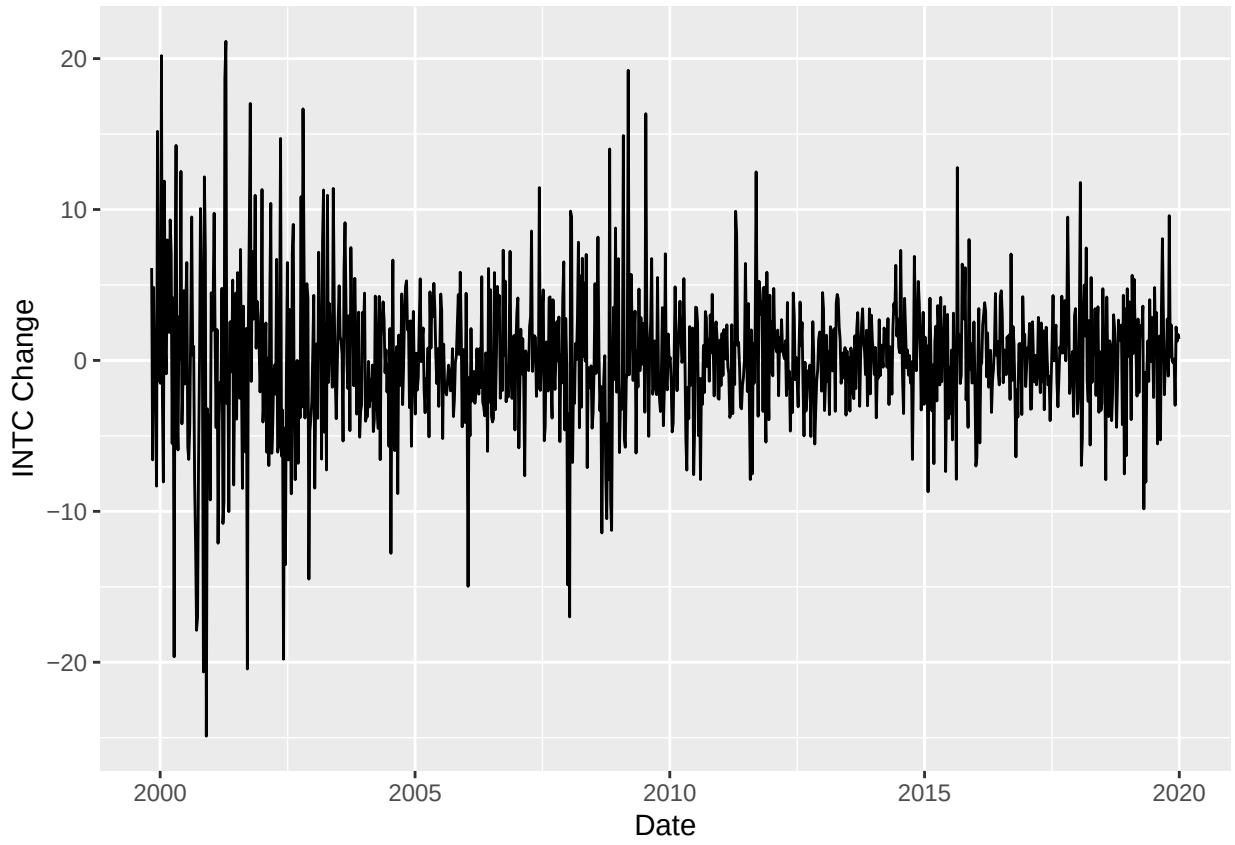

```
as.data.frame(data[,"HD"]) %>% ggplot(aes(Date, Change)) + geom_line() + ylab("HD Change")
```



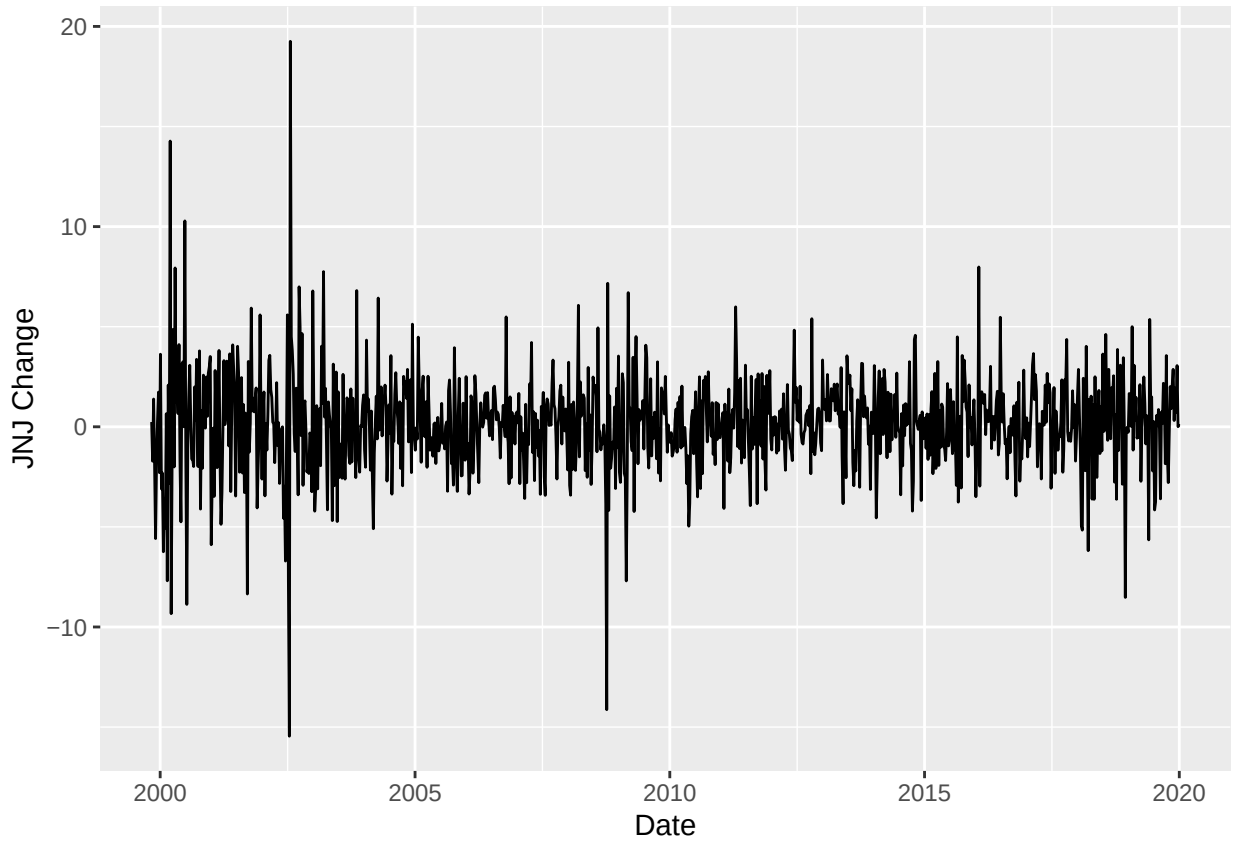

```
as.data.frame(data[,"IBM"]) %>% ggplot(aes(Date, Change)) + geom_line() + ylab("IBM Change")
```



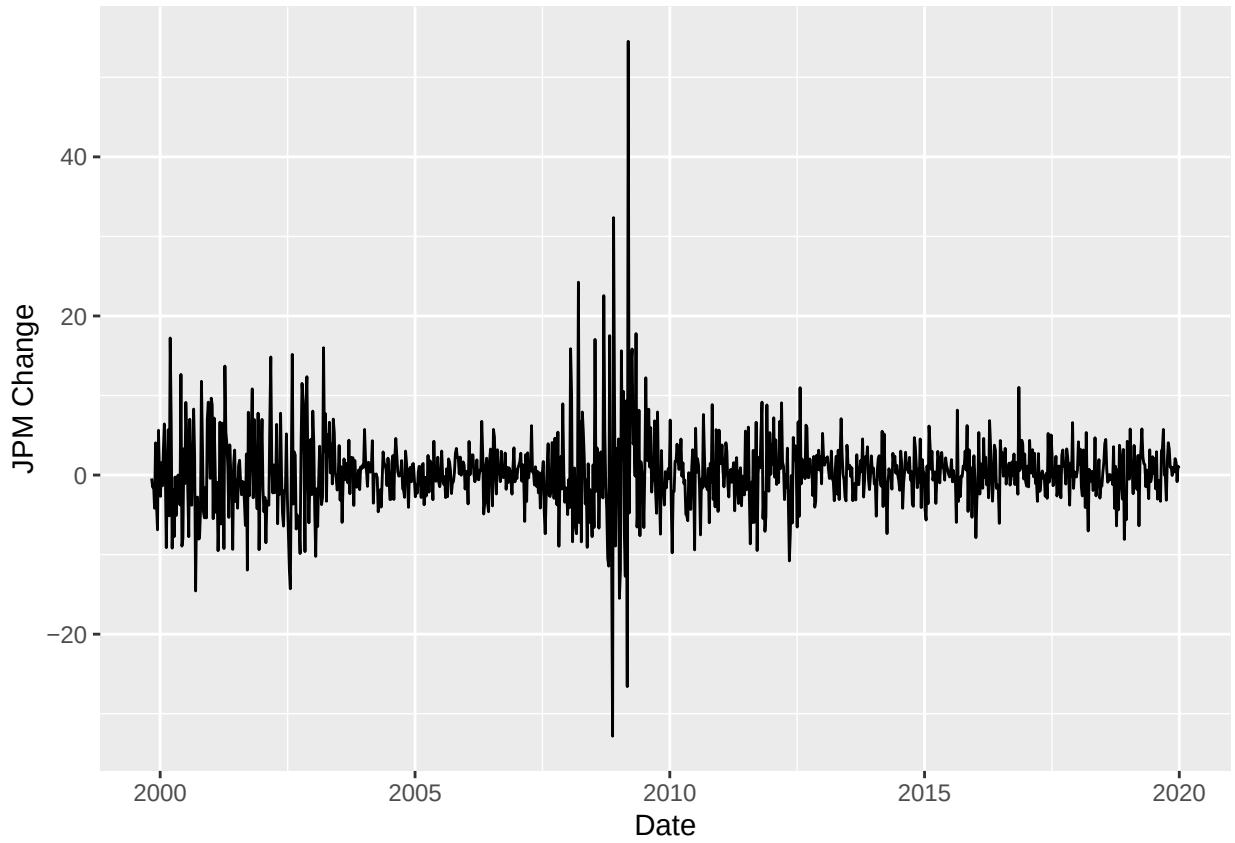

```
as.data.frame(data[,"INTC"]) %>% ggplot(aes(Date, Change)) + geom_line() + ylab("INTC Change")
```




```
as.data.frame(data[,"JNJ"]) %>% ggplot(aes(Date, Change)) + geom_line() + ylab("JNJ Change")
```



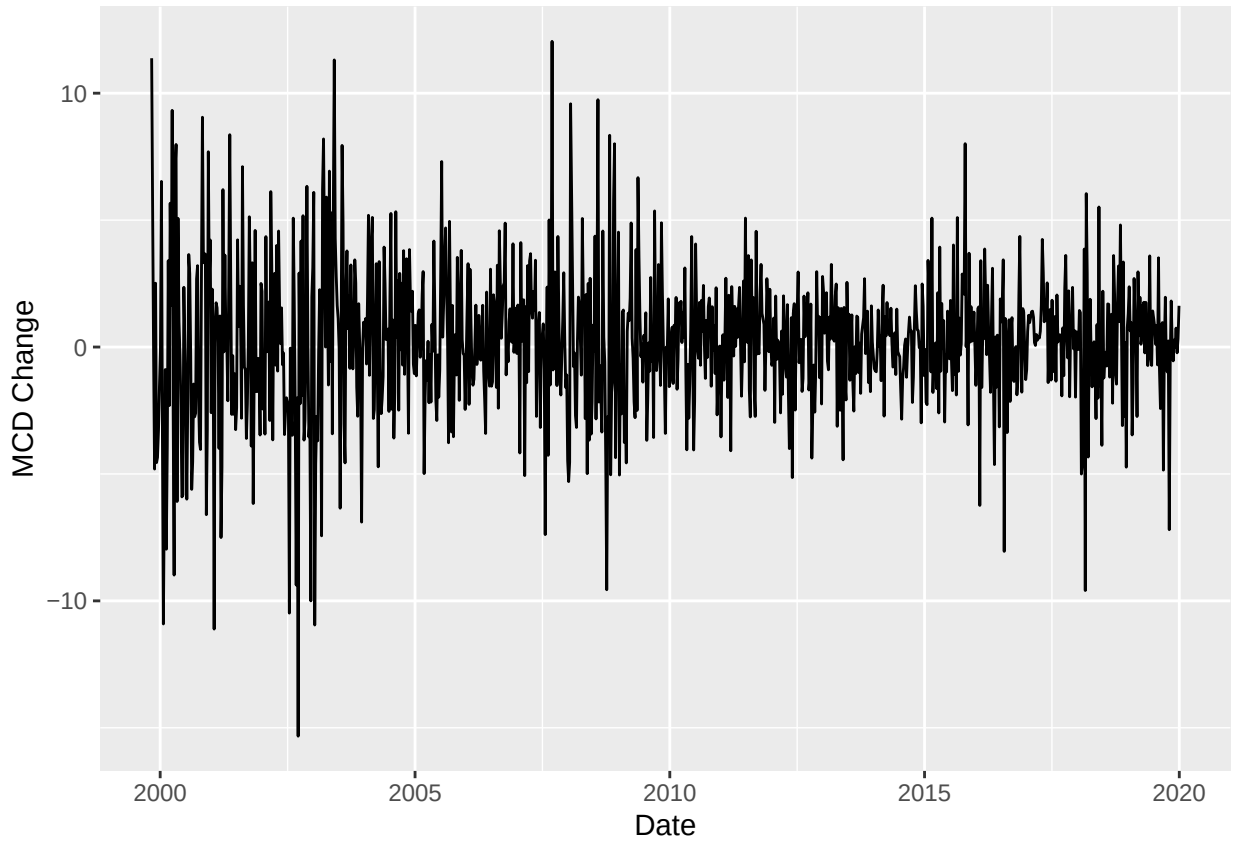

```
as.data.frame(data[,"JPM"]) %>% ggplot(aes(Date, Change)) + geom_line() + ylab("JPM Change")
```



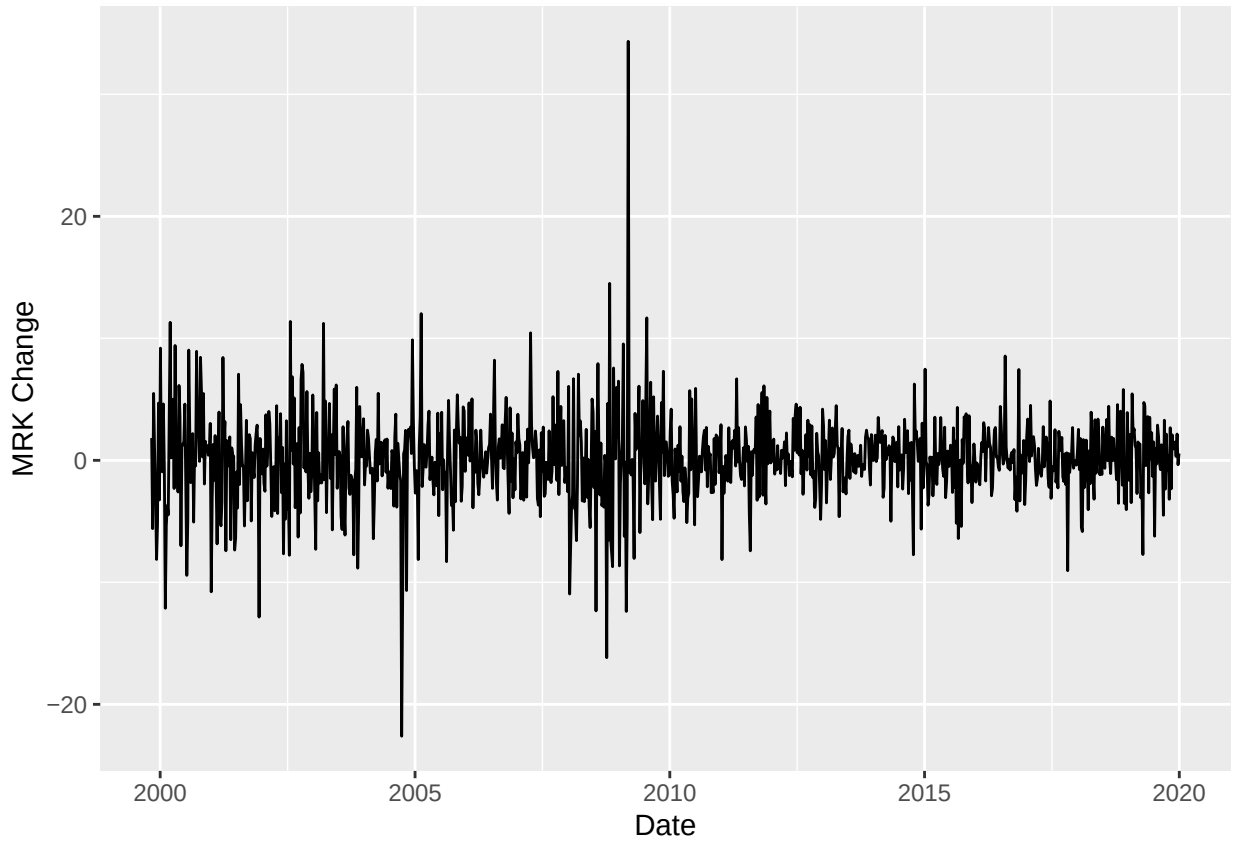

```
# JPM has a spike up of more than 40%. Unusual
head(as.data.frame(data[,"JPM"]) %>% arrange(desc(Change)))
```

```
##         Date  Open  High   Low Close Adj.Close    Volume Ticker   Change
## 1 2009-03-09 15.37 24.33 15.02 23.75  18.15491 709754000    JPM 54.52180
## 2 2008-11-24 23.92 31.93 22.50 31.66  23.90977 321245400    JPM 32.35786
## 3 2008-03-17 37.00 46.45 37.00 45.97  33.79391 345661100    JPM 24.24325
## 4 2008-09-15 38.39 48.00 34.04 47.05  35.26032 508383200    JPM 22.55796
## 5 2009-05-04 33.06 38.94 32.90 38.94  29.81940 490306200    JPM 17.78584
## 6 2008-10-27 35.10 41.25 32.52 41.25  31.15219 331939400    JPM 17.52137
```

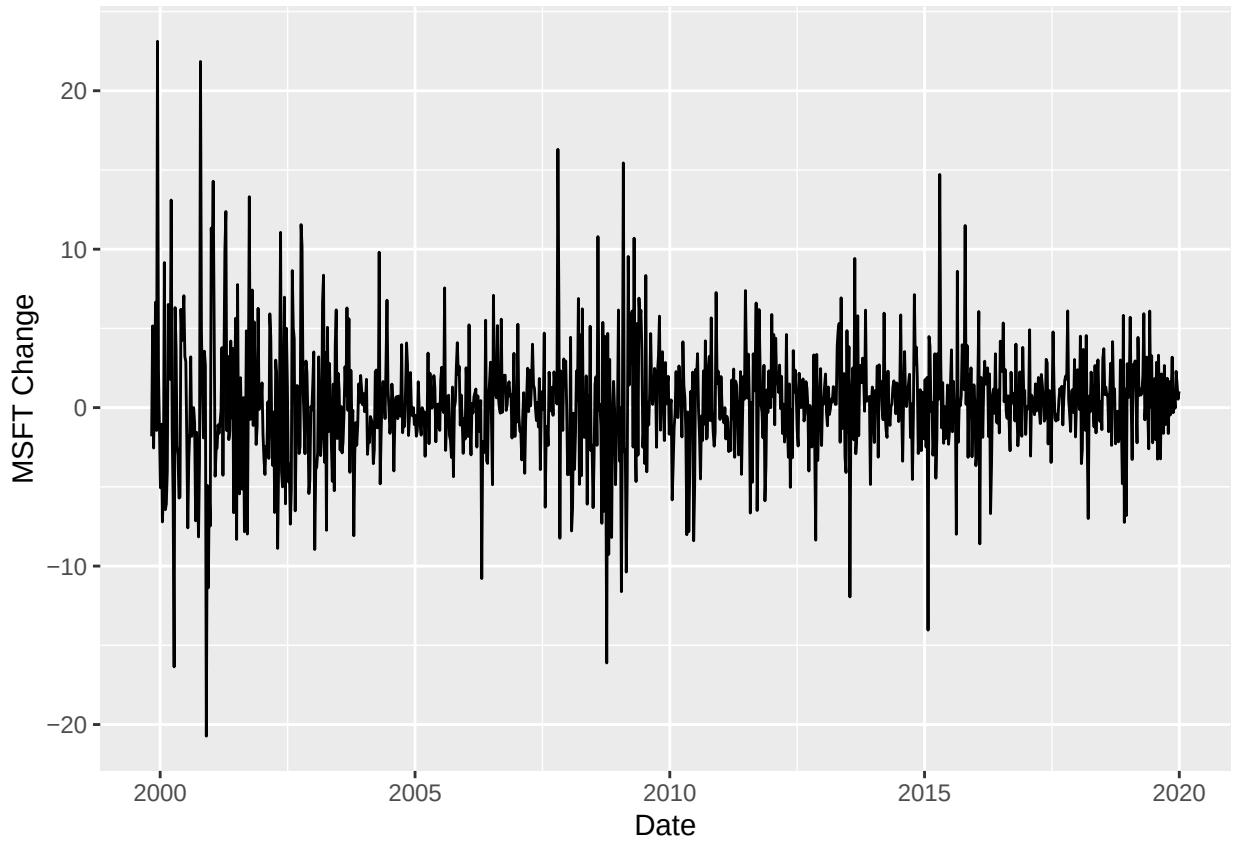
```
# 2009-03-09  15.37000  24.33000  15.02000  23.75000  18.154913 709754000    JPM  54.52179571
# Turns out it is correct when market bottomed in Mar 2009

as.data.frame(data[,"MCD"]) %>% ggplot(aes(Date, Change)) + geom_line() + ylab("MCD Change")
```



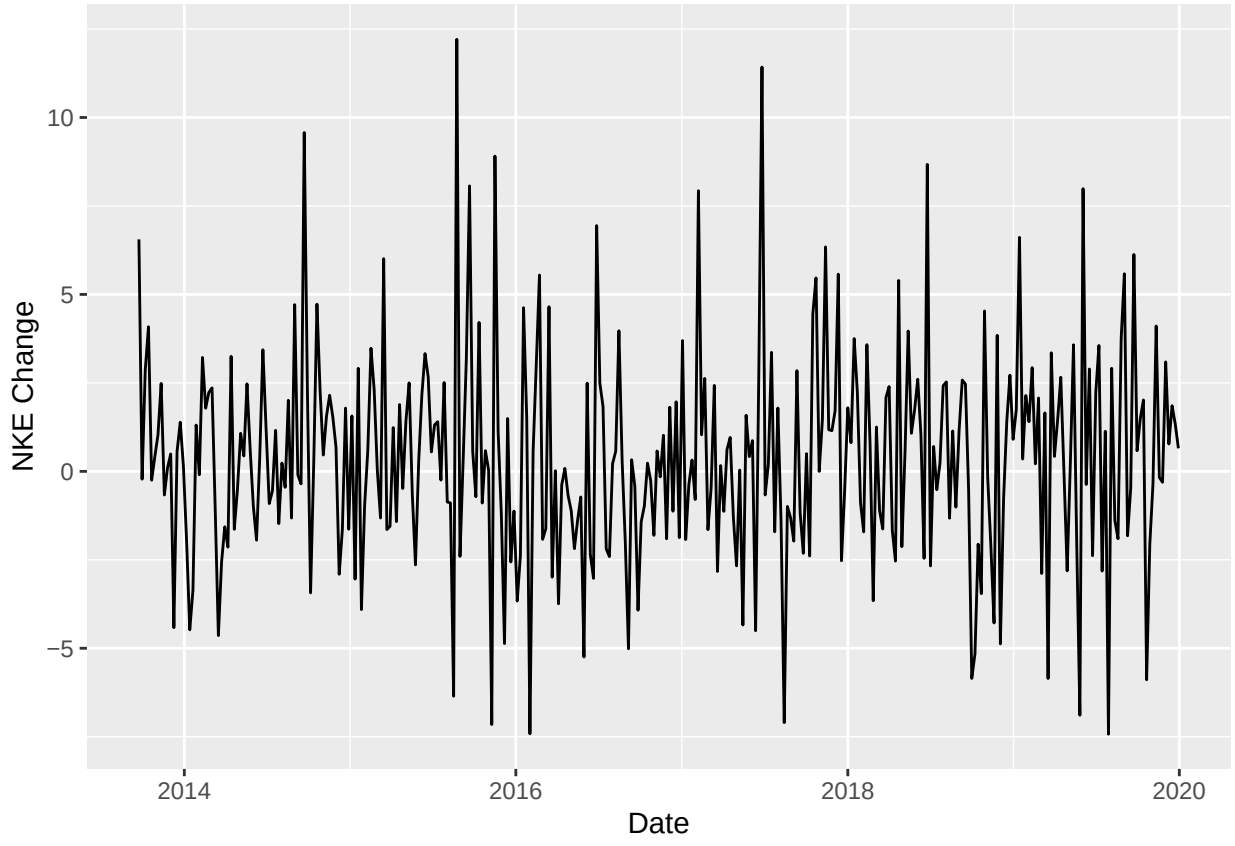

```
as.data.frame(data[,"MRK"]) %>% ggplot(aes(Date, Change)) + geom_line() + ylab("MRK Change")
```



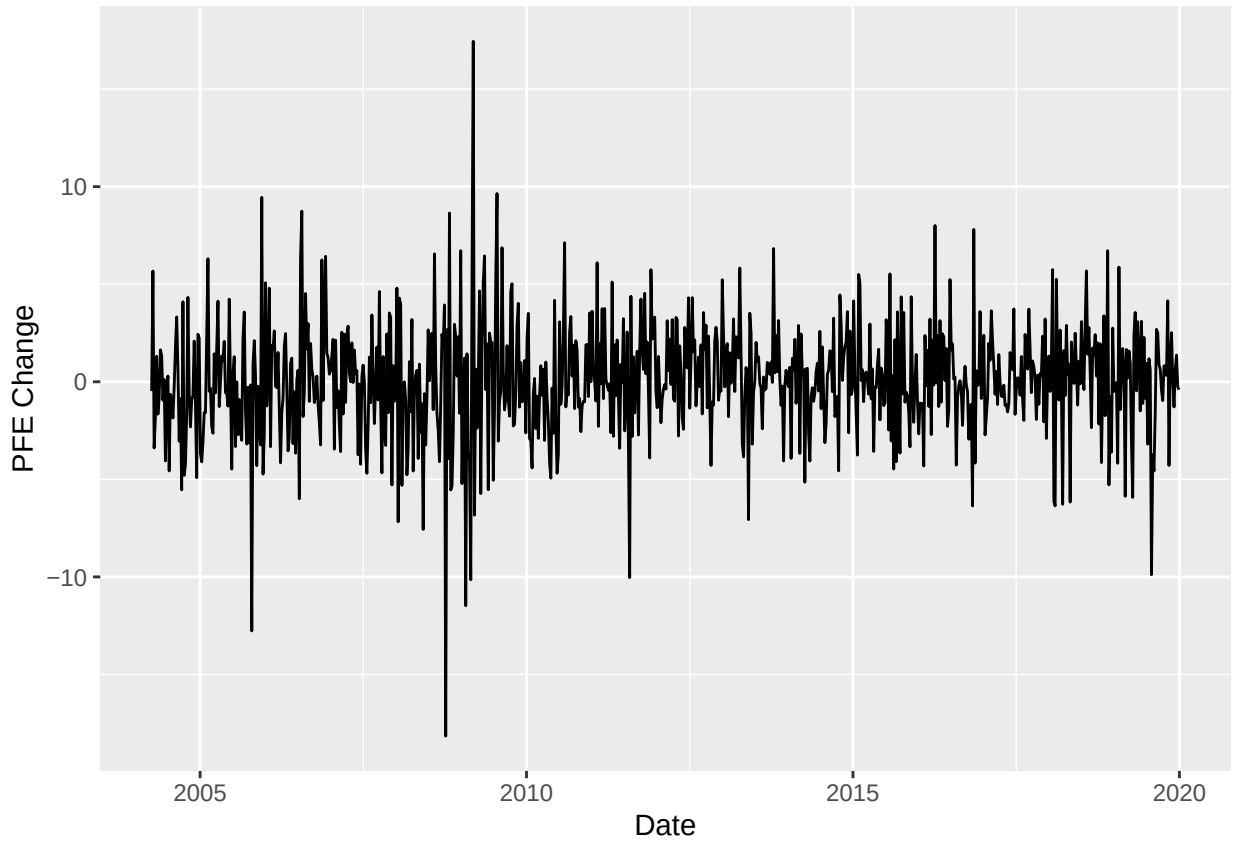

```
as.data.frame(data[,"MSFT"]) %>% ggplot(aes(Date, Change)) + geom_line() + ylab("MSFT Change")
```



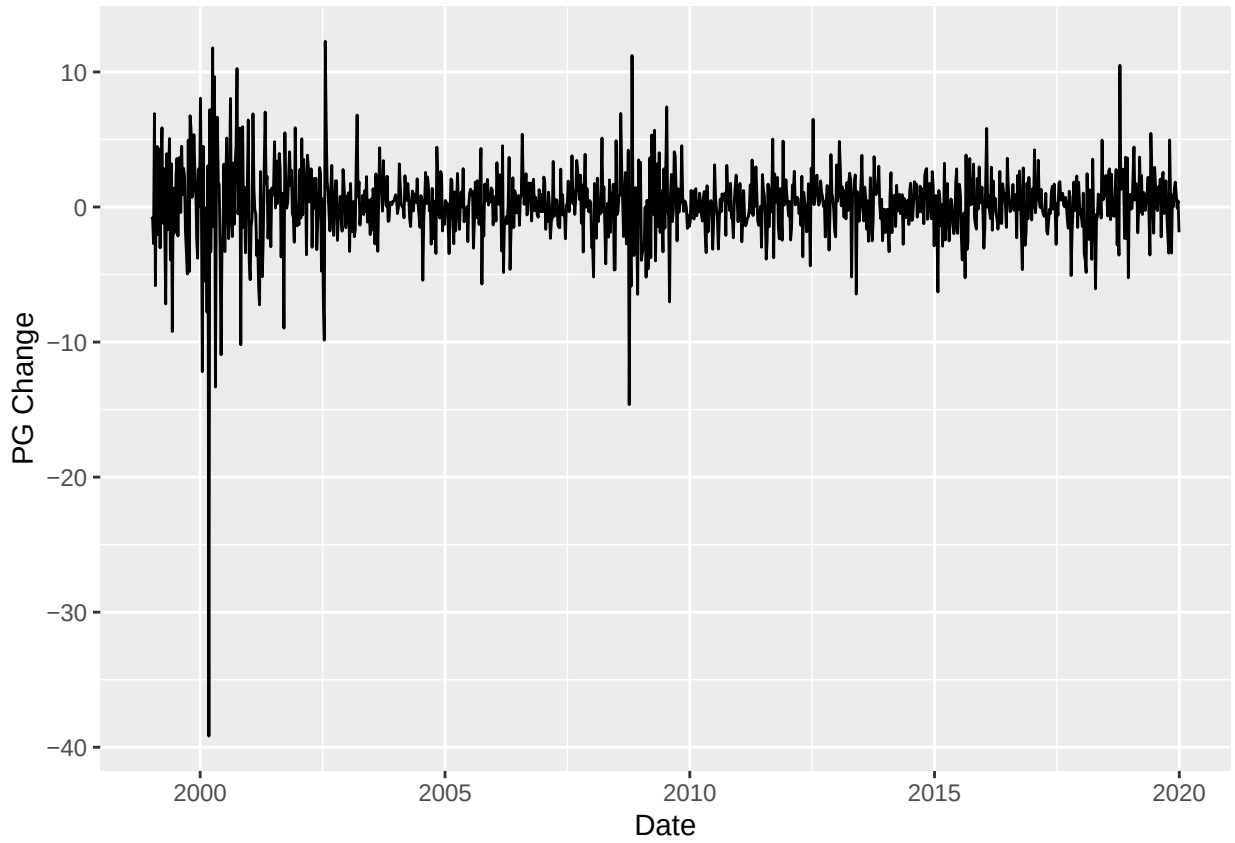

```
as.data.frame(data[,"NKE"]) %>% ggplot(aes(Date, Change)) + geom_line() + ylab("NKE Change")
```




```
as.data.frame(data[,"PFE"]) %>% ggplot(aes(Date, Change)) + geom_line() + ylab("PFE Change")
```



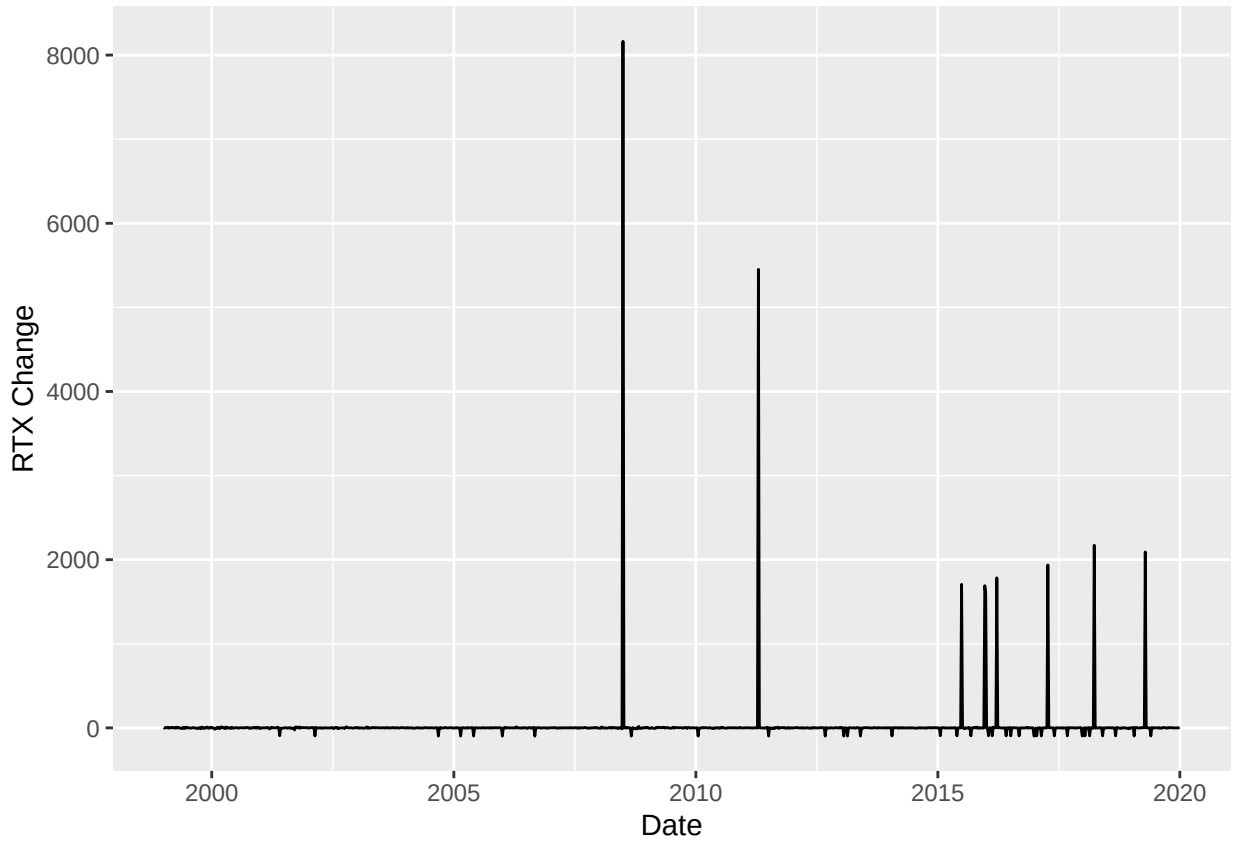

```
as.data.frame(data[,"PG"]) %>% ggplot(aes(Date, Change)) + geom_line() + ylab("PG Change")
```




```
# PG has a spike doum of nearly 40%. Unusal
head(as.data.frame(data[,"PG"]) %>% arrange(Change))
```

```
##         Date     Open     High      Low    Close Adj.Close    Volume Ticker
## 1 2000-03-06 44.18750 44.21875 26.37500 26.87500  15.58277 298318800     PG
## 2 2008-10-06 69.77000 70.46000 54.92000 59.56000  41.34907 164269800     PG
## 3 2000-04-24 34.46875 35.50000 29.56250 29.87500  17.40295  79807400     PG
## 4 2000-01-17 58.46875 59.00000 49.75000 51.34375  29.68886  44620800     PG
## 5 2000-06-05 32.00000 32.00000 27.90625 28.50000  16.60198  81385200     PG
## 6 2000-10-30 38.00000 38.43750 33.56250 34.12500  20.09478  64945200     PG
##      Change
## 1 -39.17963
## 2 -14.63379
## 3 -13.32729
## 4 -12.18600
## 5 -10.93750
## 6 -10.19737
```

```
# PG gapped doum on bad earning mid week on 2000-03-06. This is legit
# 1 2000-03-06 44.18750 44.21875 26.37500 26.87500  15.58277 298318800     PG -39.17963

as.data.frame(data[,"RTX"]) %>% ggplot(aes(Date, Change)) + geom_line() + ylab("RTX Change")
```




```
# A lot of Unusual and bizzare moves. Lets dig deeper.
head(as.data.frame(data[,"RTX"]) %>% arrange(Change) %>% select(Ticker, Date, Open, High, Low, Close, Cl
```

```
##    Ticker       Date       Open       High      Low    Close     Change
## 1     RTX 2006-09-04 2579.90991 2579.90991 39.15041 39.86155 -98.454925
## 2     RTX 2008-09-01 2456.66992 2456.66992 40.07552 40.32725 -98.358459
## 3     RTX 2010-01-18 2315.10010 2315.10010 43.42354 43.47389 -98.122160
## 4     RTX 2011-07-04 2743.62988 2743.62988 56.14852 56.85966 -97.927575
## 5     RTX 2006-01-02 1612.90002 1612.90002 34.90245 35.34298 -97.808731
## 6     RTX 2012-09-03 2031.19995 2031.19995 48.76652 49.97483 -97.539640
## 7     RTX 2013-01-21 2276.37012 2276.37012 54.33606 56.50724 -97.517660
## 8     RTX 2013-02-18 2230.39990 2230.39990 55.70799 56.94776 -97.446746
## 9     RTX 2004-09-06 1028.87000 1028.87000 29.48395 29.71051 -97.112316
## 10    RTX 2005-02-21 1064.93994 1064.93994 31.03839 31.78729 -97.015110
## 11    RTX 2013-05-27 1965.45996 1965.45996 59.66646 59.72310 -96.961368
## 12    RTX 2005-05-30 1026.57996 1026.57996 33.36060 33.43298 -96.743266
## 13    RTX 2014-01-20 1951.56995 1951.56995 70.33984 70.35872 -96.394763
## 14    RTX 2019-05-27 1916.51001 1916.51001 78.99937 79.48395 -95.852672
## 15    RTX 2019-01-21 1714.64001 1714.64001 69.63499 72.88232 -95.749410
## 16    RTX 2002-02-18  509.64002  509.64002 21.41598 21.93203 -95.696564
## 17    RTX 2017-01-16 1586.12000 1586.12000 68.84834 69.72310 -95.604172
## 18    RTX 2016-12-26 1555.54004 1555.54004 68.83575 68.98679 -95.565091
## 19    RTX 2017-01-02 1587.52002 1587.52002 69.22593 70.83071 -95.538279
## 20    RTX 2017-02-20 1556.03003 1556.03003 70.30837 70.77407 -95.451626
## 21    RTX 2017-09-04 1478.12000 1478.12000 68.65953 68.94273 -95.335782
## 22    RTX 2018-05-28 1674.57996 1674.57996 78.18124 79.17558 -95.271914
```

```
## 23    RTX 2018-02-19 1770.60999 1770.60999 79.21963 83.79484 -95.267459
## 24    RTX 2018-01-15 1771.27002 1771.27002 83.66898 85.52549 -95.171516
## 25    RTX 2016-09-05 1325.75000 1325.75000 64.61296 64.61925 -95.125834
## 26    RTX 2016-05-30 1257.59998 1257.59998 62.14600 63.20327 -94.974294
## 27    RTX 2015-05-25 1456.93994 1456.93994 73.25991 73.73820 -94.938831
## 28    RTX 2017-12-25 1584.81006 1584.81006 79.87414 80.28320 -94.934207
## 29    RTX 2016-07-04 1276.87000 1276.87000 62.49843 65.23600 -94.890944
## 30    RTX 2018-01-01 1597.64001 1597.64001 80.03146 82.80051 -94.817324
## 31    RTX 2017-05-29 1467.68005 1467.68005 76.14852 76.85337 -94.763615
## 32    RTX 2018-09-03 1581.17004 1581.17004 82.14600 83.07741 -94.745827
## 33    RTX 2015-09-07 1079.56006 1079.56006 57.19950 58.10573 -94.617647
## 34    RTX 2016-02-15  956.65997  956.65997 53.84518 55.52549 -94.195902
## 35    RTX 2016-01-18  860.17999  860.17999 52.47955 54.34235 -93.682444
## 36    RTX 2001-05-28  396.23001  396.23001 25.66709 26.17055 -93.395112
## 37    RTX 2015-01-19 1077.91003 1077.91003 72.93896 75.50661 -92.995092
## 38    RTX 2001-09-17   17.62114   17.71554 12.61800 13.29453 -24.553569
## 39    RTX 2000-01-24   19.66646   20.05979 16.20516 16.44116 -16.399996
## 40    RTX 2001-03-12   25.55066   25.55381 22.41976 22.47640 -12.032021
## 41    RTX 2008-06-23   43.56828   43.56828 38.42039 38.48332 -11.671245
## 42    RTX 2008-10-06   33.90812   34.28571 27.23726 29.97483 -11.599853
## 43    RTX 2011-08-01   52.42920   52.75016 45.72687 46.65828 -11.007078
## 44    RTX 1999-10-11   18.60447   18.60447 16.24449 16.55916 -10.993655
## 45    RTX 2018-12-17   74.72624   74.81435 66.74638 66.82190 -10.577734
## 46    RTX 2015-07-20   69.83637   69.86155 62.46696 62.49843 -10.507338
## 47    RTX 2002-08-12   21.36564   21.68660 18.28509 19.26998  -9.808544
## 48    RTX 1999-05-17   22.26243   22.28210 19.66646 20.17778  -9.363956
## 49    RTX 2002-07-08   21.79043   21.94462 19.58150 19.92133  -8.577622
## 50    RTX 2001-06-11   25.72373   25.73946 23.28194 23.52108  -8.562694
## 51    RTX 2001-01-01   24.46507   24.46507 22.36076 22.38043  -8.520902
## 52    RTX 2008-09-29   37.69037   37.98616 34.48080 34.52486  -8.398736
## 53    RTX 2011-08-15   46.07300   46.49465 42.26557 42.44808  -7.867775
## 54    RTX 2008-10-20   32.31592   32.90749 28.34487 29.77344  -7.867574
## 55    RTX 2010-05-03   47.36312   48.06797 41.01321 43.71303  -7.706614
```

```
# 37 data points are having an spike down greater than 90%
#1  2006-09-04 2579.90991 2579.90991 39.15041 39.86155 28.926479  17996700    RTX -98.454925
#2  2008-09-01 2456.66992 2456.66992 40.07552 40.32725 30.277899  42480900    RTX -98.358459
#3  2010-01-18 2315.10010 2315.10010 43.42354 43.47389 33.821892  29045500    RTX -98.122160
#4  2011-07-04 2743.62988 2743.62988 56.14852 56.85966 45.790684  30531000    RTX -97.927575
#5  2006-01-02 1612.90002 1612.90002 34.90245 35.34298 25.336058  20377500    RTX -97.808731
#6  2012-09-03 2031.19995 2031.19995 48.76652 49.97483 41.538990  30980900    RTX -97.539640
#7  2013-01-21 2276.37012 2276.37012 54.33606 56.50724 47.297348  28567300    RTX -97.517660
#8  2013-02-18 2230.39990 2230.39990 55.70799 56.94776 47.951168  23638200    RTX -97.446746
#9  2004-09-06 1028.87000 1028.87000 29.48395 29.71051 20.865843  20278000    RTX -97.112316
#10 2005-02-21 1064.93994 1064.93994 31.03839 31.78729 22.501221  23522400    RTX -97.015110
#11 2013-05-27 1965.45996 1965.45996 59.66646 59.72310 50.570084  23097600    RTX -96.961368
#12 2005-05-30 1026.57996 1026.57996 33.36060 33.43298 23.766071  21545600    RTX -96.743266
#13 2014-01-20 1951.56995 1951.56995 70.33984 70.35872 60.206238  30168900    RTX -96.394763
#14 2019-05-27 1916.51001 1916.51001 78.99937 79.48395 77.335411  20878500    RTX -95.852672
#15 2019-01-21 1714.64001 1714.64001 69.63499 72.88232 70.106575  50732700    RTX -95.749410
#16 2002-02-18  509.64002  509.64002 21.41598 21.93203 14.820339  30178400    RTX -95.696564
#17 2017-01-16 1586.12000 1586.12000 68.84834 69.72310 64.076569  17165700    RTX -95.604172
#18 2016-12-26 1555.54004 1555.54004 68.83575 68.98679 63.399899  12817000    RTX -95.565091
#19 2017-01-02 1587.52002 1587.52002 69.22593 70.83071 65.094505  18746700    RTX -95.538279
```

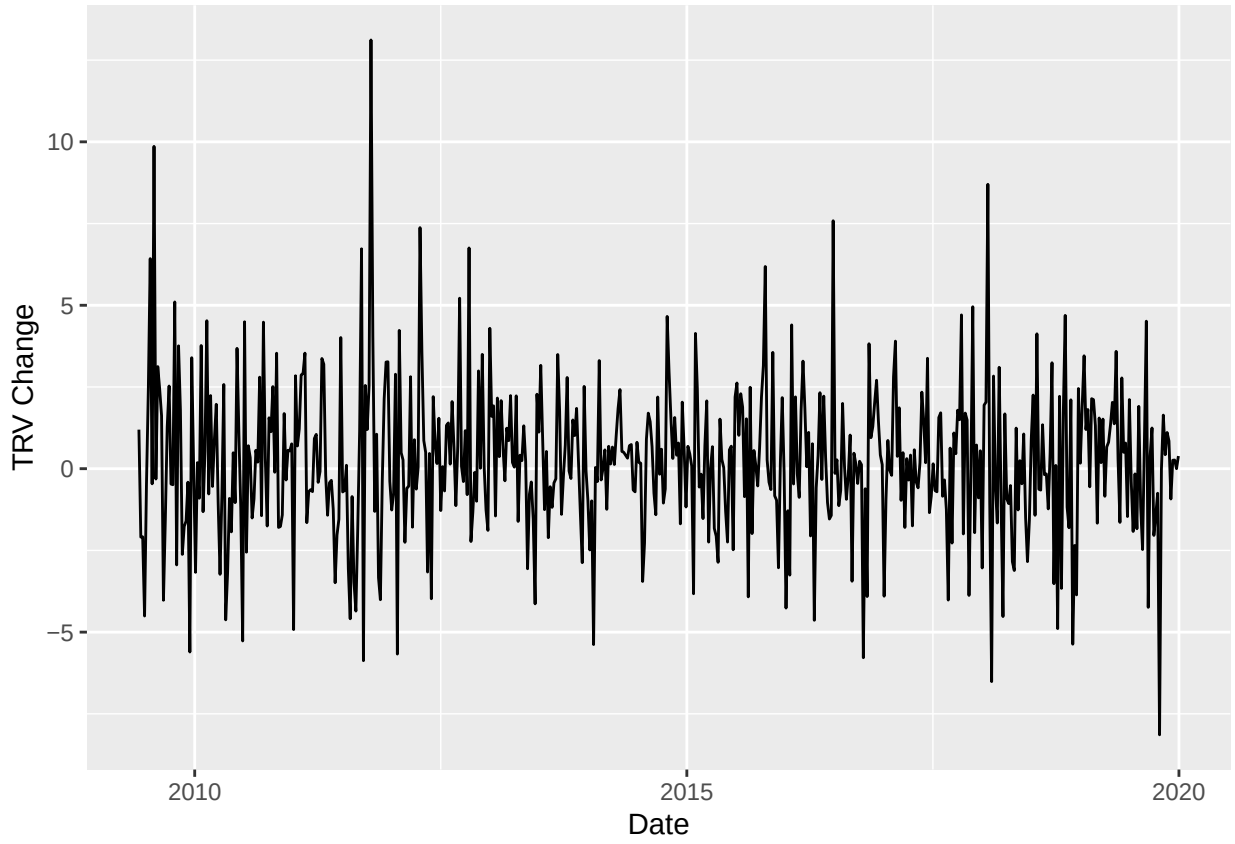
```
#20 2017-02-20 1556.03003 1556.03003 70.30837 70.77407 65.429504  17053100    RTX -95.451626
#21 2017-09-04 1478.12000 1478.12000 68.65953 68.94273 64.474068  52380400    RTX -95.335782
#22 2018-05-28 1674.57996 1674.57996 78.18124 79.17558 75.323517  18521200    RTX -95.271914
#23 2018-02-19 1770.60999 1770.60999 79.21963 83.79484 79.270821  39849400    RTX -95.267459
#24 2018-01-15 1771.27002 1771.27002 83.66898 85.52549 80.461044  24746200    RTX -95.171516
#25 2016-09-05 1325.75000 1325.75000 64.61296 64.61925 59.022083  21075200    RTX -95.125834
#26 2016-05-30 1257.59998 1257.59998 62.14600 63.20327 57.379700  20327200    RTX -94.974294
#27 2015-05-25 1456.93994 1456.93994 73.25991 73.73820 65.160629  23460900    RTX -94.938831
#28 2017-12-25 1584.81006 1584.81006 79.87414 80.28320 75.529175   9666300    RTX -94.934207
#29 2016-07-04 1276.87000 1276.87000 62.49843 65.23600 59.225140  25571700    RTX -94.890944
#30 2018-01-01 1597.64001 1597.64001 80.03146 82.80051 77.897408  24533400    RTX -94.817324
#31 2017-05-29 1467.68005 1467.68005 76.14852 76.85337 71.438110  19697500    RTX -94.763615
#32 2018-09-03 1581.17004 1581.17004 82.14600 83.07741 79.457672  19623500    RTX -94.745827
#33 2015-09-07 1079.56006 1079.56006 57.19950 58.10573 51.682026  32112200    RTX -94.617647
#34 2016-02-15  956.65997  956.65997 53.84518 55.52549 49.706249  45948500    RTX -94.195902
#35 2016-01-18  860.17999  860.17999 52.47955 54.34235 48.647114  43636200    RTX -93.682444
#36 2001-05-28  396.23001  396.23001 25.66709 26.17055 17.496731  17148500    RTX -93.395112
#37 2015-01-19 1077.91003 1077.91003 72.93896 75.50661 66.003761  37349100    RTX -92.995092
head(as.data.frame(data[,"RTX"]) %>% arrange(desc(Change)) %>% select(Ticker, Date, Open, High, Low, Cl
```

```
##    Ticker       Date     Open       High      Low      Close      Change
## 1     RTX 2008-06-30 38.48332 3180.08008 37.47640 3180.08008 8163.527972
## 2     RTX 2011-04-18 51.99497 2886.66992 51.09503 2886.66992 5451.825878
## 3     RTX 2018-03-26 77.99874 1771.87000 77.68407 1771.87000 2171.664866
## 4     RTX 2019-04-15 84.95909 1862.05005 84.37382 1862.05005 2091.701944
## 5     RTX 2017-04-10 71.10761 1448.89001 70.37130 1448.89001 1937.601818
## 6     RTX 2016-03-21 62.35368 1174.79004 61.67401 1174.79004 1784.074908
## 7     RTX 2015-06-29 70.95029 1281.63000 68.73505 1281.63000 1706.377478
## 8     RTX 2015-12-21 58.53996 1048.33997 58.33228 1048.33997 1690.810777
## 9     RTX 2015-12-28 60.40277 1024.06006 60.22656 1024.06006 1595.386006
## 10    RTX 2008-10-27 29.37697   35.00315 28.31970   34.58779   17.737792
## 11    RTX 2006-04-17 35.87162   40.87476 35.78981   40.41536   12.666672
## 12    RTX 2000-03-13 15.26117   18.36847 15.16284   17.18848   12.628874
## 13    RTX 2002-10-14 17.05475   19.42731 16.68974   19.10950   12.047968
## 14    RTX 2001-10-01 14.53744   16.83449 14.38011   16.26809   11.904760
## 15    RTX 2008-01-21 41.20831   46.70862 41.03210   45.78351   11.102633
## 16    RTX 2003-03-17 18.12461   20.13216 17.93581   20.13216   11.076384
## 17    RTX 2001-10-22 16.22089   18.01133 16.15796   17.93895   10.591654
## 18    RTX 2000-04-17 18.25047   20.33512 17.87681   20.15812   10.452596
## 19    RTX 2001-09-24 13.28194   14.83638 13.28194   14.63184   10.163463
## 20    RTX 1999-11-29 17.30648   19.43046 17.09015   18.99780    9.772726
## 21    RTX 2016-02-22 56.12964   63.27879 55.92826   61.47892    9.530212
## 22    RTX 2001-05-14 25.14160   27.53304 24.63814   27.44179    9.148941
## 23    RTX 2015-10-19 58.21271   63.49906 57.40088   63.32285    8.778382
## 24    RTX 1999-05-31 19.54846   21.23977 19.41079   21.22011    8.551304
## 25    RTX 1999-10-18 16.51982   18.09314 16.28383   17.91614    8.452385
```

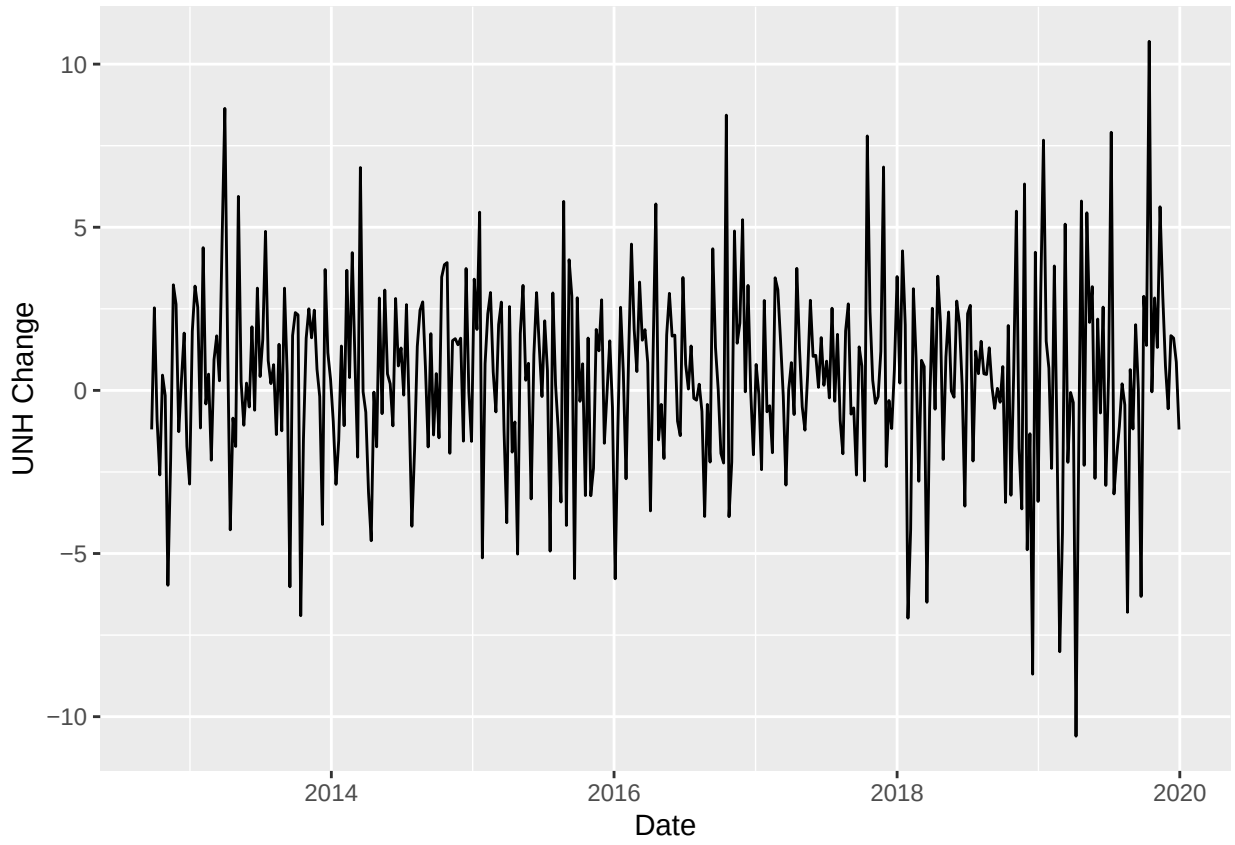
```
# 9 data point has a spike up of more than 1000%
#1  2008-06-30 38.48332 3180.08008 37.47640 3180.08008 2376.131836  61760600    RTX 8163.527972
#2  2011-04-18 51.99497 2886.66992 51.09503 2886.66992 2312.010498  36068500    RTX 5451.825878
#3  2018-03-26 77.99874 1771.87000 77.68407 1771.87000 1676.208008  42110500    RTX 2171.664866
#4  2019-04-15 84.95909 1862.05005 84.37382 1862.05005 1801.777588  14551000    RTX 2091.701944
#5  2017-04-10 71.10761 1448.89001 70.37130 1448.89001 1339.475464  14868100    RTX 1937.601818
```

```
#6  2016-03-21 62.35368 1174.79004 61.67401 1174.79004 1059.509277  21593300    RTX 1784.074908
#7  2015-06-29 70.95029 1281.63000 68.73505 1281.63000 1132.545166  30757200    RTX 1706.377478
#8  2015-12-21 58.53996 1048.33997 58.33228 1048.33997  938.470764  30846700    RTX 1690.810777
#9  2015-12-28 60.40277 1024.06006 60.22656 1024.06006  916.735535  20674000    RTX 1595.386006
```

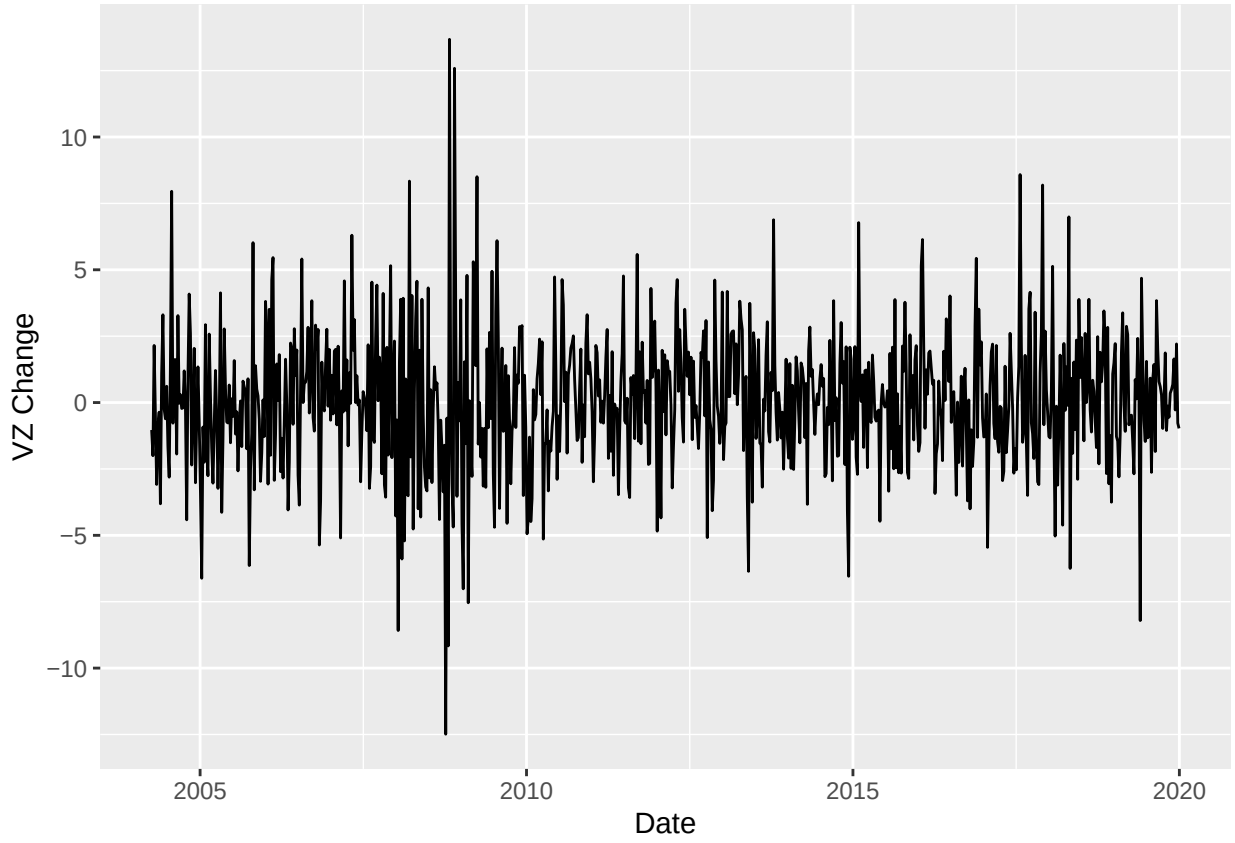
```
as.data.frame(data[,"TRV"]) %>% ggplot(aes(Date, Change)) + geom_line() + ylab("TRV Change")
```



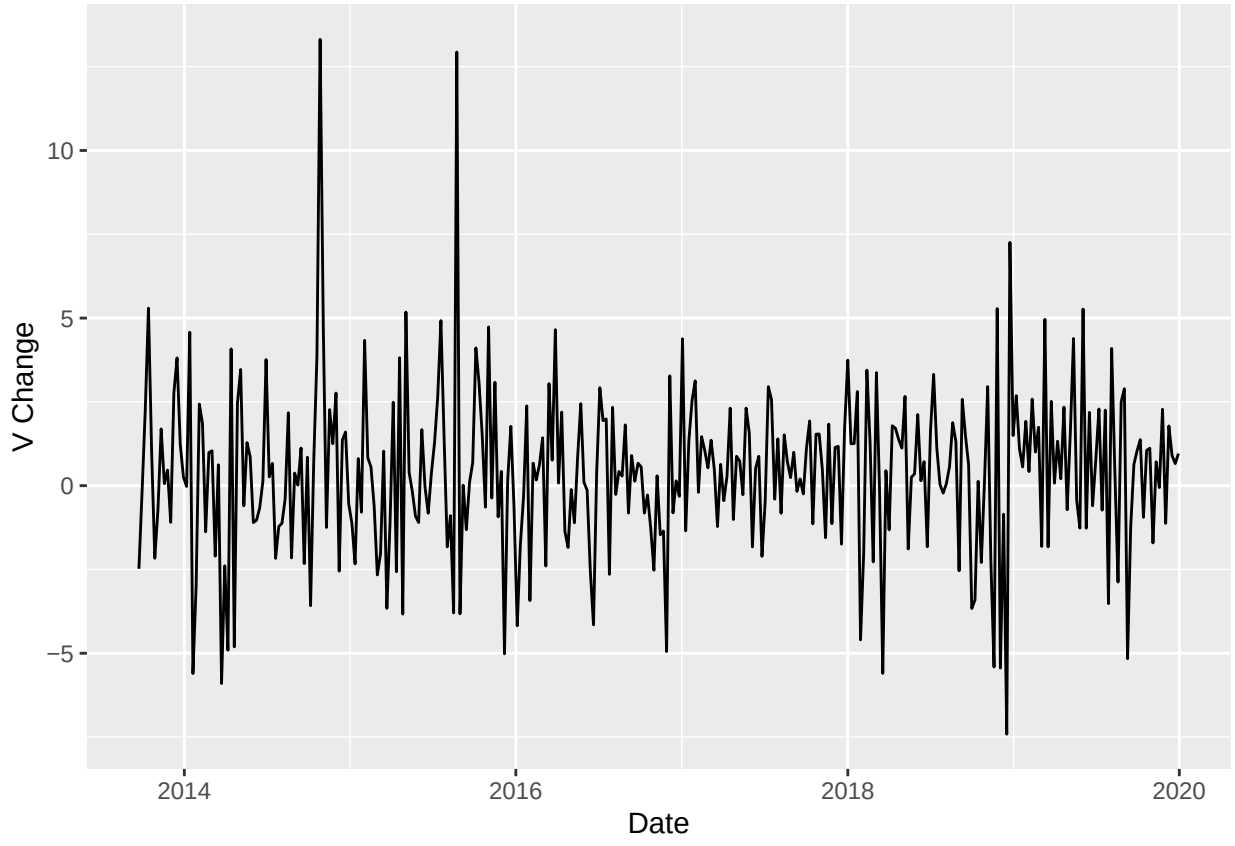

```
as.data.frame(data[,"UNH"]) %>% ggplot(aes(Date, Change)) + geom_line() + ylab("UNH Change")
```



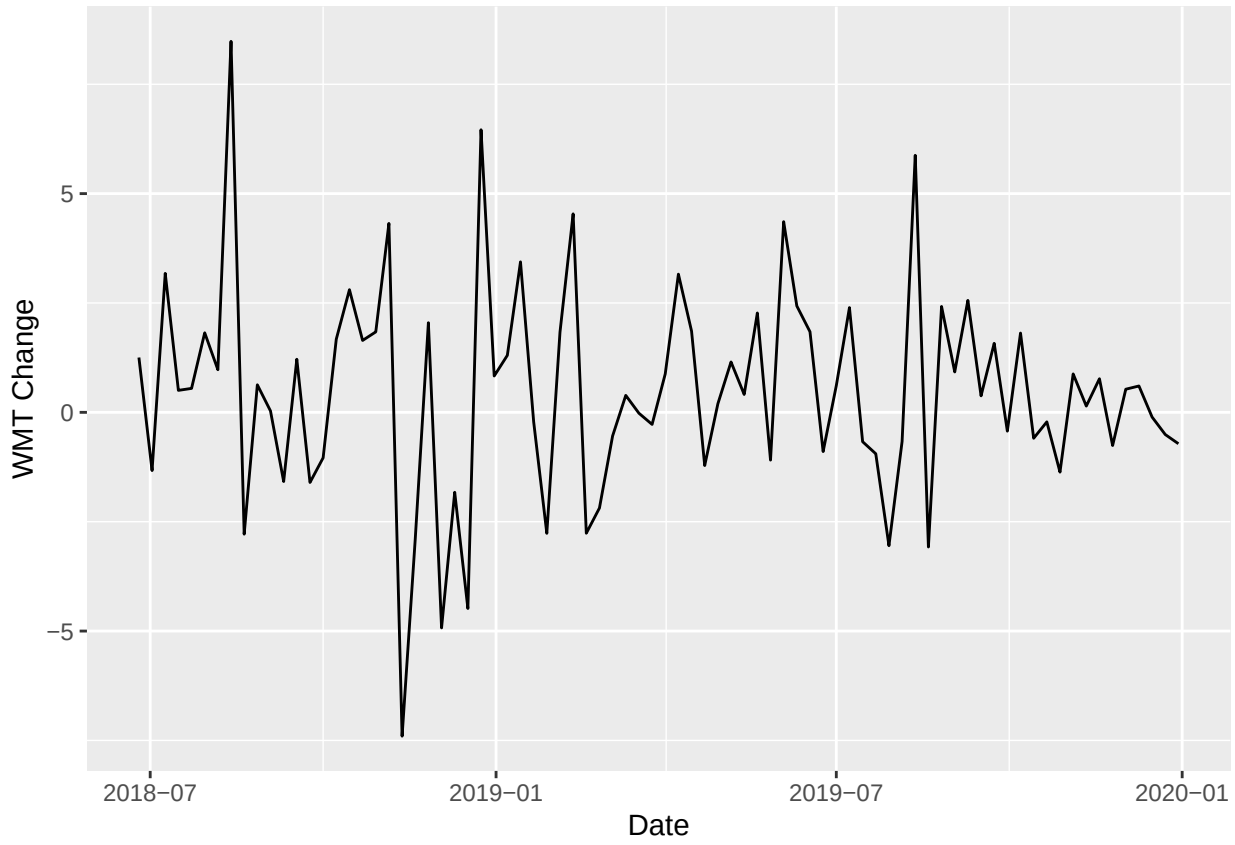

```
as.data.frame(data[,"VZ"]) %>% ggplot(aes(Date, Change)) + geom_line() + ylab("VZ Change")
```



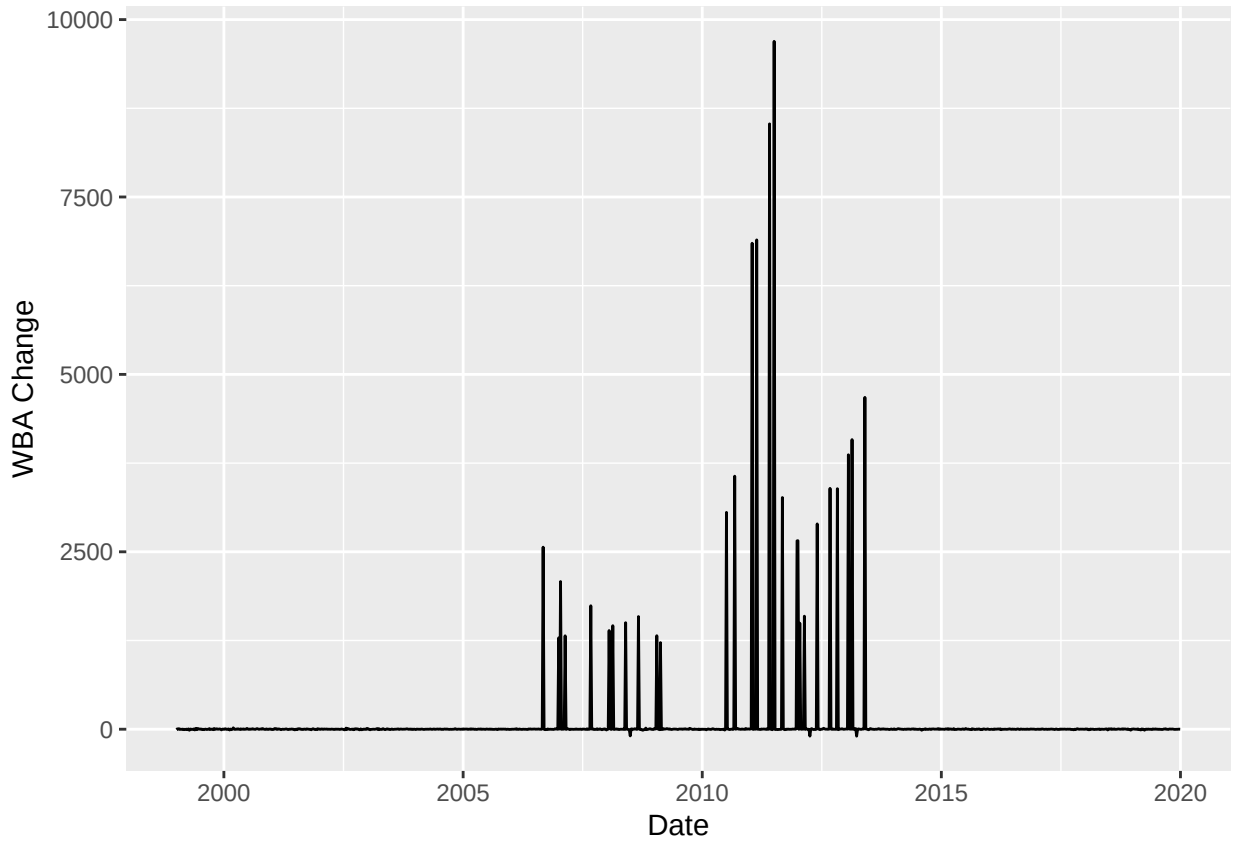

```
as.data.frame(data[,"V"]) %>% ggplot(aes(Date, Change)) + geom_line() + ylab("V Change")
```




```
as.data.frame(data[,"WMT"]) %>% ggplot(aes(Date, Change)) + geom_line() + ylab("WMT Change")
```




```
as.data.frame(data[,"WBA"]) %>% ggplot(aes(Date, Change)) + geom_line() + ylab("WBA Change")
```



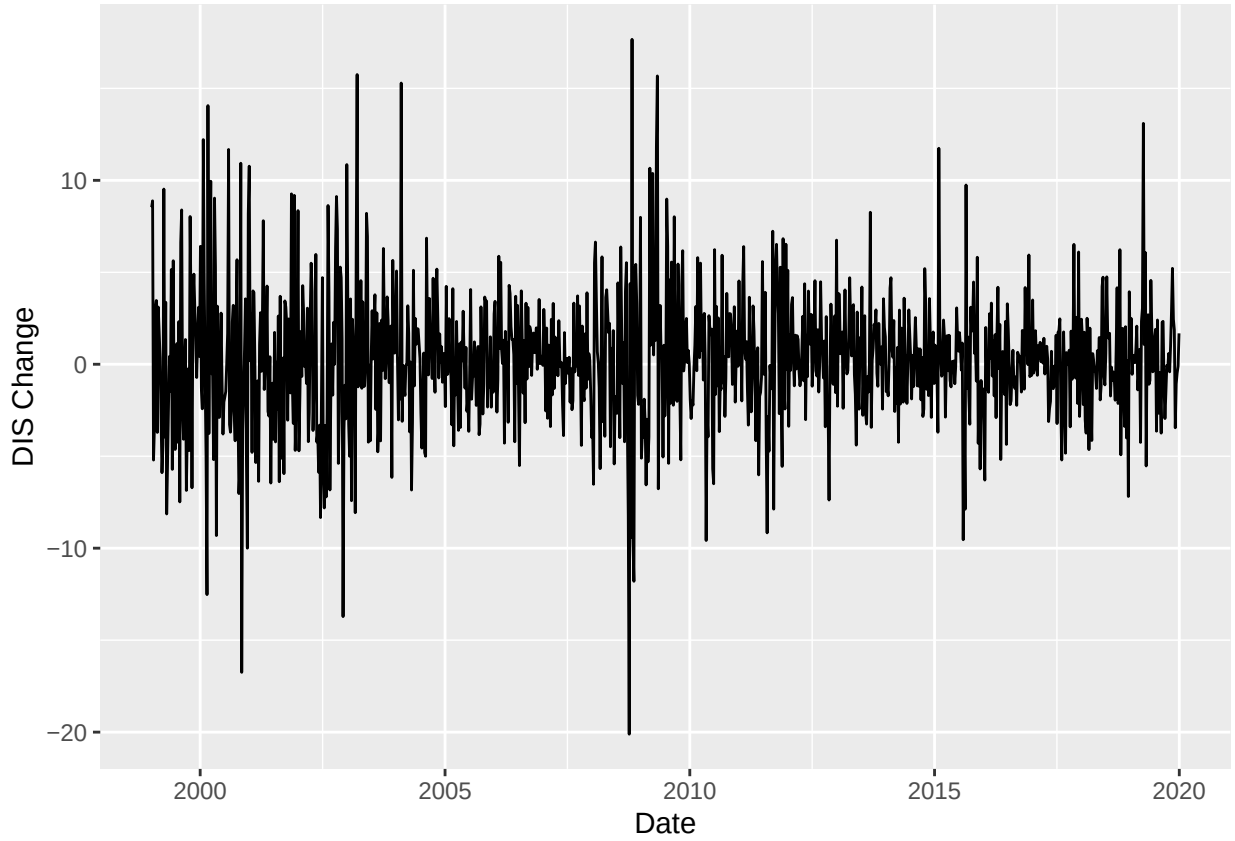

```
# 3 bad data point on spike down
#1  2013-03-25 46.4300 47.76000  1.0000  1.00000  0.842636  27628500    WBA -97.846220
#2  2012-04-02 33.5600 34.73000  1.0200  1.02000  0.834645  41351600    WBA -96.960668
#3  2008-06-30 32.9600 33.05000  2.0000  2.00000  1.525111  36779000    WBA -93.932039
head(as.data.frame(data[,"WBA"]) %>% arrange(desc(Change)) %>% select(Ticker, Date, Open, High, Low, Cl
```

```
##    Ticker       Date   Open    High     Low   Close     Change
## 1     WBA 2011-07-04  0.450 44.2600  0.4500 44.0700 9693.33333
## 2     WBA 2011-05-30  0.500 44.0600  0.5000 43.1500 8530.00040
## 3     WBA 2011-02-21  0.600 42.5400  0.6000 41.9700 6895.00017
## 4     WBA 2011-01-17  0.600 42.2000  0.6000 41.6900 6848.33317
## 5     WBA 2013-05-27  1.000 51.2500  1.0000 47.7600 4675.99980
## 6     WBA 2013-02-18  1.000 41.9900  1.0000 41.8100 4081.00010
## 7     WBA 2013-01-21  1.000 39.8100  1.0000 39.6700 3866.99980
## 8     WBA 2010-09-06  0.790 29.1200  0.7900 28.9600 3565.82266
## 9     WBA 2012-09-03  1.000 36.1900  1.0000 34.9400 3393.99990
## 10    WBA 2012-10-29  1.000 35.6900  1.0000 34.8900 3388.99990
## 11    WBA 2011-09-05  1.050 36.4500  1.0500 35.3300 3264.76210
## 12    WBA 2010-07-05  0.900 28.4100  0.9000 28.4000 3055.55556
## 13    WBA 2012-05-28  1.000 31.6400  1.0000 29.9300 2893.00000
## 14    WBA 2012-01-02  1.200 33.7000  1.2000 33.0800 2656.66683
## 15    WBA 2011-12-26  1.200 35.1900  1.2000 33.0600 2655.00008
## 16    WBA 2006-09-04  1.910 50.9900  1.9100 50.9100 2565.44503
## 17    WBA 2007-01-15  2.110 46.6000  2.1100 46.0100 2080.56863
## 18    WBA 2007-09-03  2.400 45.6900  2.4000 44.1500 1739.58342
## 19    WBA 2012-02-20  2.000 34.8600  2.0000 33.8600 1593.00005
```

```
## 20    WBA 2008-09-01  2.050 37.2400  2.0500 34.5900 1587.31707
## 21    WBA 2008-05-26  2.250 36.2000  2.2500 36.0200 1500.88889
## 22    WBA 2012-01-16  2.100 33.8900  2.1000 33.4800 1494.28571
## 23    WBA 2008-02-18  2.400 37.4600  2.4000 37.4100 1458.75000
## 24    WBA 2008-01-21  2.300 35.6600  2.3000 34.2800 1390.43474
## 25    WBA 2009-01-19  1.900 27.3500  1.9000 26.9300 1317.36842
## 26    WBA 2007-02-19  3.200 46.4900  3.2000 45.3200 1316.25000
## 27    WBA 2007-01-01  3.280 46.6900  3.2800 45.5000 1287.19512
## 28    WBA 2009-02-16  1.900 27.2500  1.9000 25.0800 1220.00000
## 29    WBA 2000-03-13 23.375 28.4375 23.3125 27.9375   19.51872
## 30    WBA 2008-10-27 21.920 26.0800 21.2800 25.4600   16.14963
## 31    WBA 2002-07-22 30.250 34.9900 30.2000 34.9900   15.66943
## 32    WBA 2012-07-16 30.540 35.0000 30.2700 34.6000   13.29403
## 33    WBA 2009-09-28 33.730 38.7500 33.6400 38.0500   12.80759
## 34    WBA 1999-05-31 23.250 26.0000 22.6875 26.0000   11.82796
## 35    WBA 2002-12-30 28.770 32.1600 28.7000 32.1600   11.78311
```

```
# 28 bad data points in spike up greater than 1000%.
#1  2011-07-04  0.450 44.2600  0.4500 44.0700  35.36953 23212100    WBA 9693.33333
#2  2011-05-30  0.500 44.0600  0.5000 43.1500  34.63117 24426700    WBA 8530.00040
#3  2011-02-21  0.600 42.5400  0.6000 41.9700  33.55104 26158700    WBA 6895.00017
#4  2011-01-17  0.600 42.2000  0.6000 41.6900  33.18947 23169900    WBA 6848.33317
#5  2013-05-27  1.000 51.2500  1.0000 47.7600  40.47093 19501900    WBA 4675.99980
#6  2013-02-18  1.000 41.9900  1.0000 41.8100  35.23062 20187200    WBA 4081.00010
#7  2013-01-21  1.000 39.8100  1.0000 39.6700  33.20824 19536500    WBA 3866.99980
#8  2010-09-06  0.790 29.1200  0.7900 28.9600  22.94061 21086700    WBA 3565.82266
#9  2012-09-03  1.000 36.1900  1.0000 34.9400  29.01172 37422400    WBA 3393.99990
#10 2012-10-29  1.000 35.6900  1.0000 34.8900  28.97020 19209700    WBA 3388.99990
#11 2011-09-05  1.050 36.4500  1.0500 35.3300  28.52729 33170700    WBA 3264.76210
#12 2010-07-05  0.900 28.4100  0.9000 28.4000  22.35734 44954400    WBA 3055.55556
#13 2012-05-28  1.000 31.6400  1.0000 29.9300  24.66038 26259100    WBA 2893.00000
#14 2012-01-02  1.200 33.7000  1.2000 33.0800  26.89282 33590000    WBA 2656.66683
#15 2011-12-26  1.200 35.1900  1.2000 33.0600  26.87657 27462600    WBA 2655.00008
#16 2006-09-04  1.910 50.9900  1.9100 50.9100  38.23864 15307500    WBA 2565.44503
#17 2007-01-15  2.110 46.6000  2.1100 46.0100  34.62063 12978100    WBA 2080.56863
#18 2007-09-03  2.400 45.6900  2.4000 44.1500  33.40829 17974200    WBA 1739.58342
#19 2012-02-20  2.000 34.8600  2.0000 33.8600  27.70696 26020500    WBA 1593.00005
#20 2008-09-01  2.050 37.2400  2.0500 34.5900  26.45701 33758200    WBA 1587.31707
#21 2008-05-26  2.250 36.2000  2.2500 36.0200  27.46725 20723000    WBA 1500.88889
#22 2012-01-16  2.100 33.8900  2.1000 33.4800  27.21801 28197200    WBA 1494.28571
#23 2008-02-18  2.400 37.4600  2.4000 37.4100  28.45269 25690500    WBA 1458.75000
#24 2008-01-21  2.300 35.6600  2.3000 34.2800  26.00370 44686200    WBA 1390.43474
#25 2009-01-19  1.900 27.3500  1.9000 26.9300  20.69978 34820500    WBA 1317.36842
#26 2007-02-19  3.200 46.4900  3.2000 45.3200  34.16058 14911000    WBA 1316.25000
#27 2007-01-01  3.280 46.6900  3.2800 45.5000  34.23689 13657200    WBA 1287.19512
#28 2009-02-16  1.900 27.2500  1.9000 25.0800  19.35882 38485200    WBA 1220.00000
```

```
as.data.frame(data[,"DIS"]) %>% ggplot(aes(Date, Change)) + geom_line() + ylab("DIS Change")
```

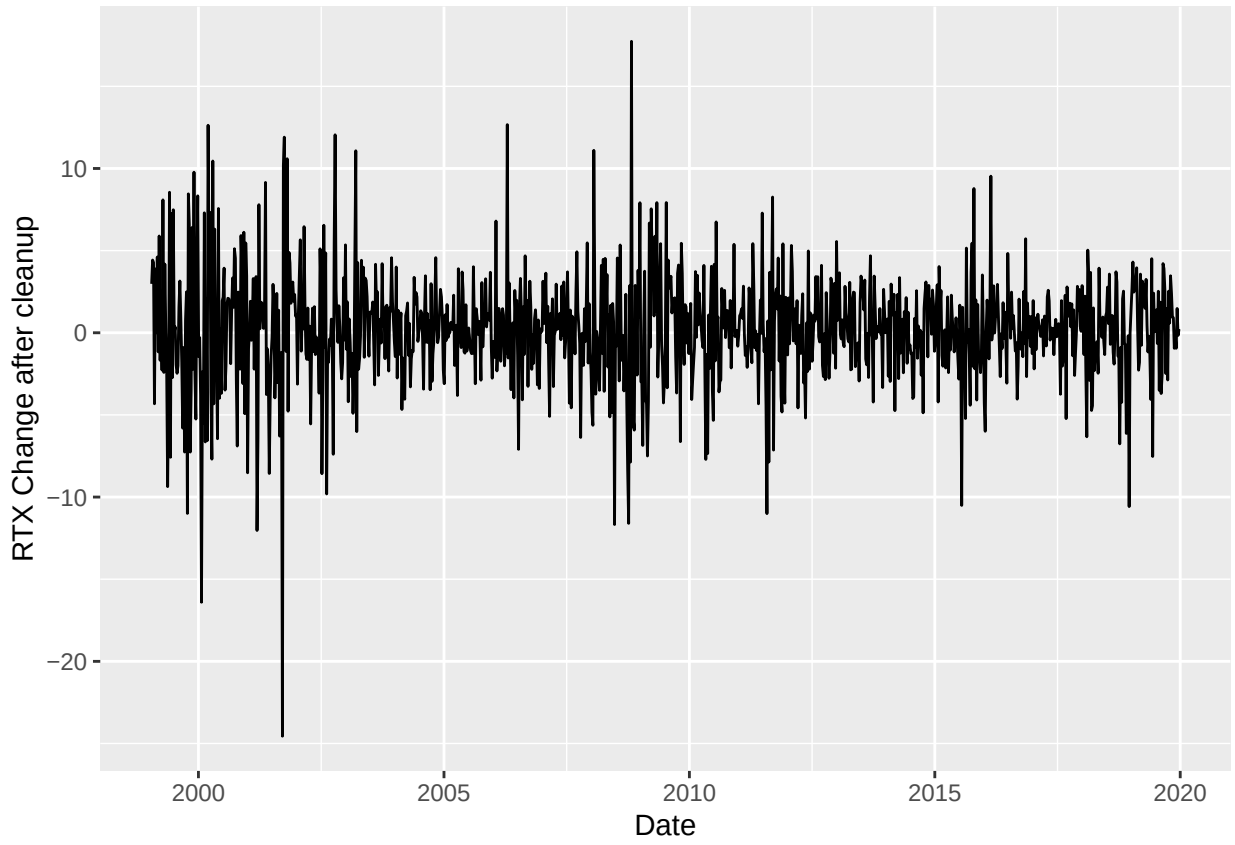


2.2 Raw Data Cleanup

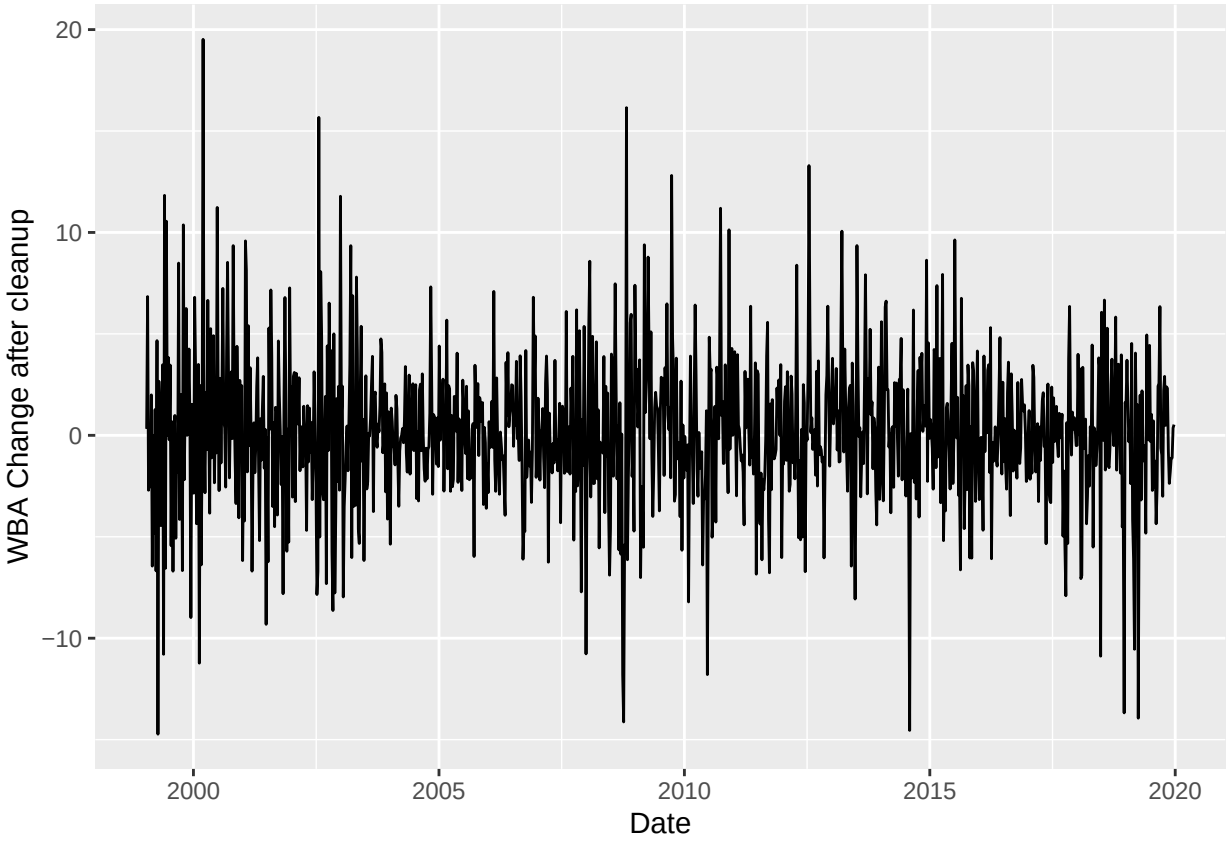
Turns out RTX and WBA data are dirty. Manually found out the correct values for each date. An update is applied during load using modules.

Added an update module to fix all the bad prices just after the CSV file load. RTX and WBA change chart looks like this after the fix.

```
as.data.frame(data[,"RTX"]) %>% ggplot(aes(Date, Change)) + geom_line() + ylab("RTX Change after cleanu
```




```
as.data.frame(data[,"WBA"]) %>% ggplot(aes(Date, Change)) + geom_line() + ylab("WBA Change after cleanup
```



Raw data is now sanitized, let's analyze the Raw data.

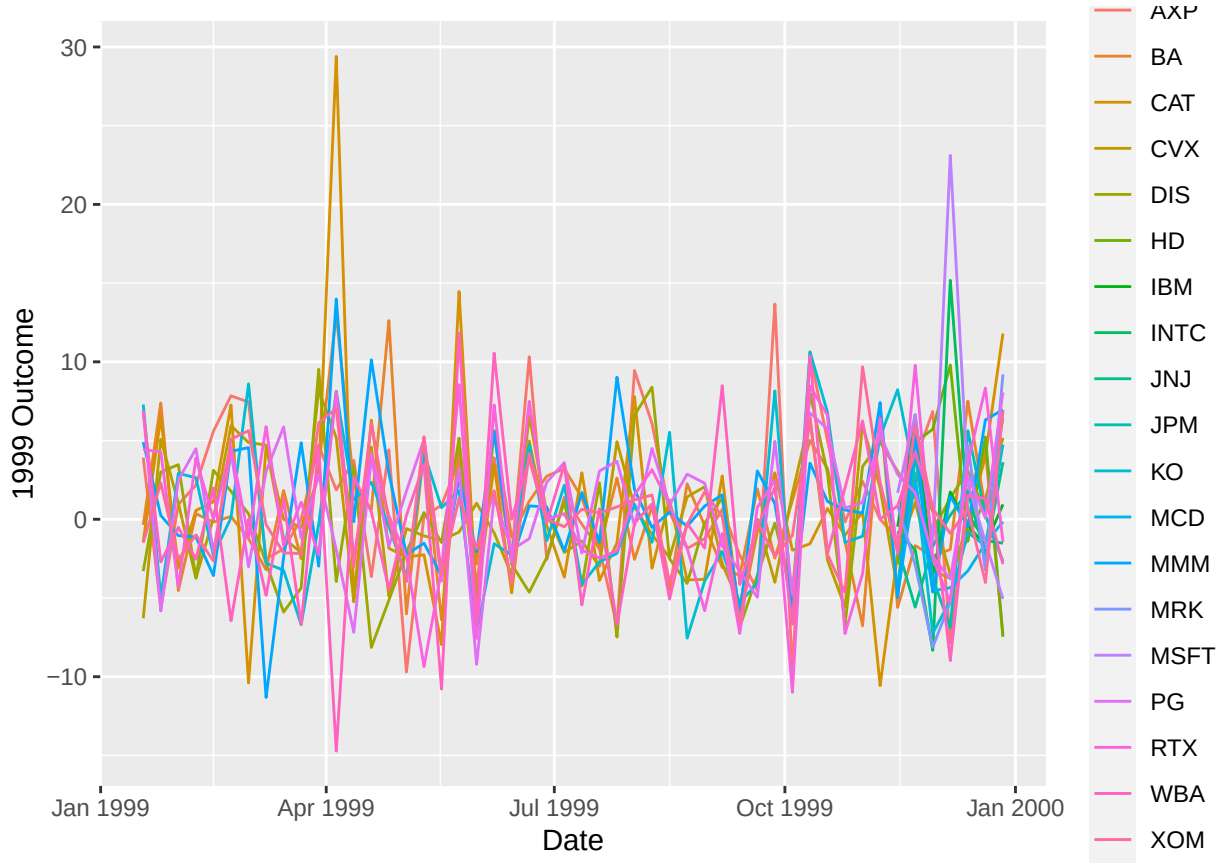
2.3 Data Analysis Strategies

2.4 Outcome Analysis

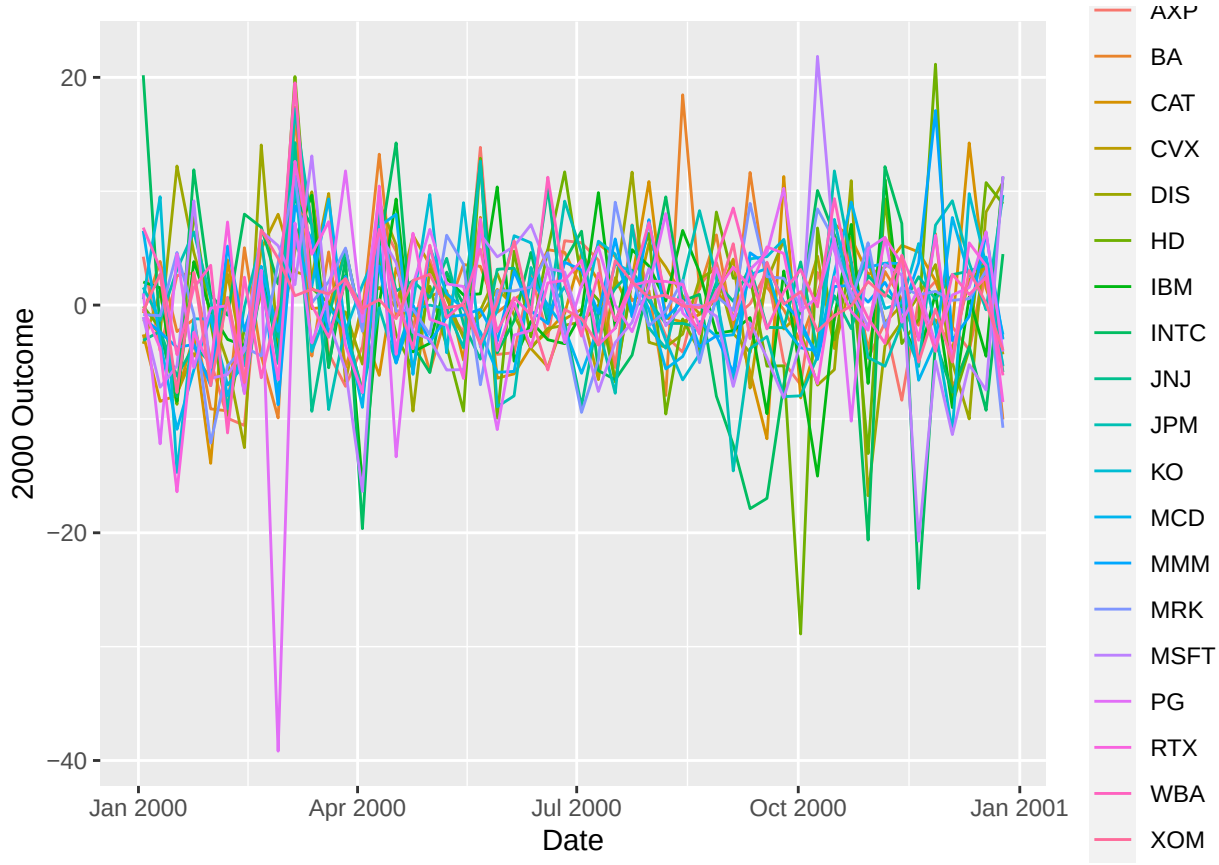
2.4.1 Check Outcomes

Outcomes per Year.

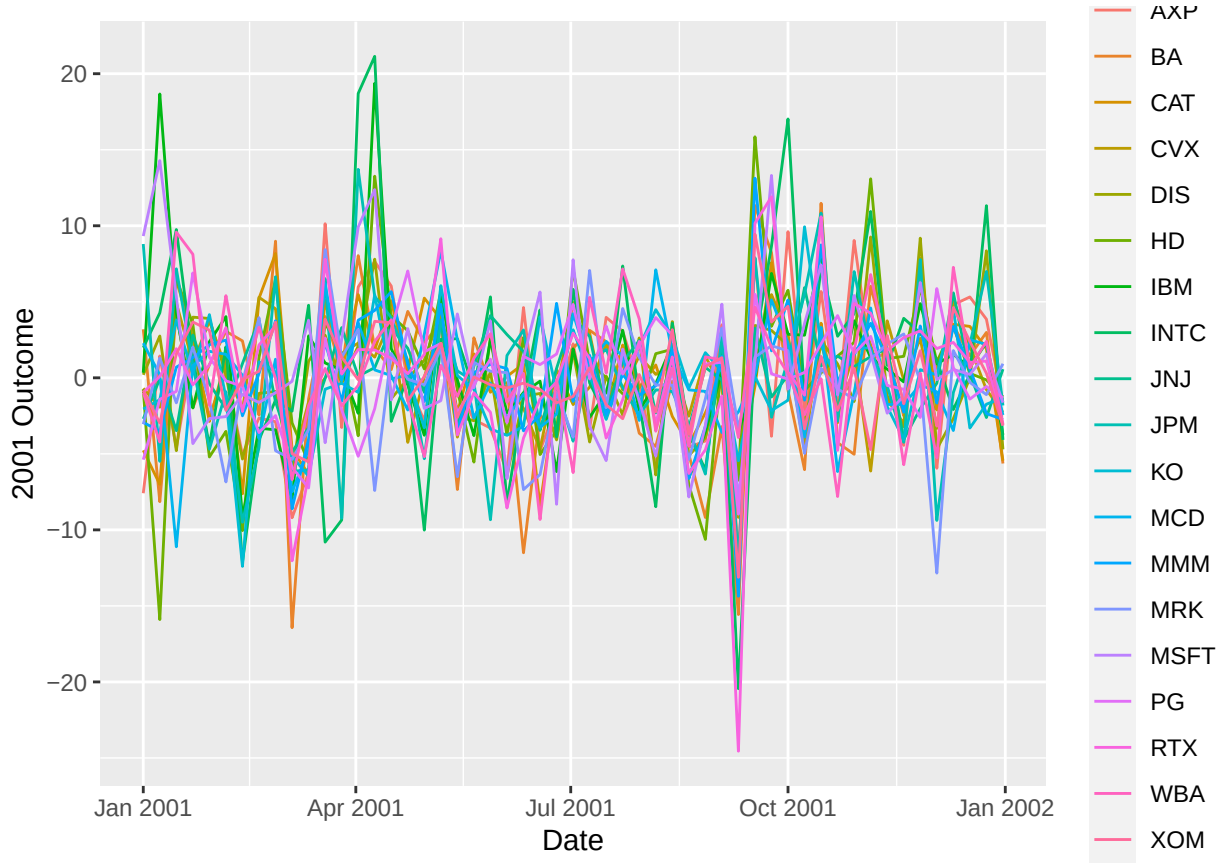
```
dedata %>% filter(year(Date) == 1999) %>% ggplot(aes(Date, Outcome, color=Ticker)) + geom_line() + ylab
```



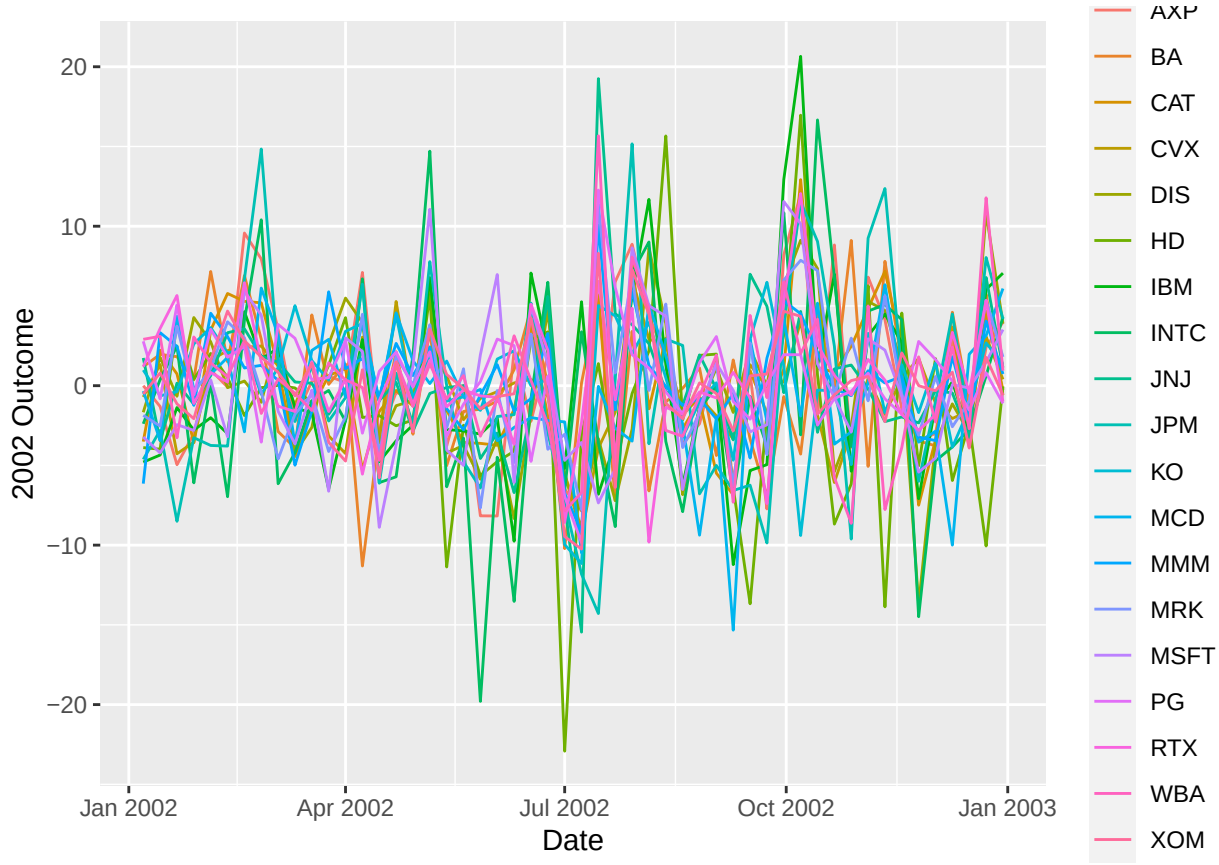

```
dedata %>% filter(year(Date) == 2000) %>% ggplot(aes(Date, Outcome, color=Ticker)) + geom_line() + ylab
```



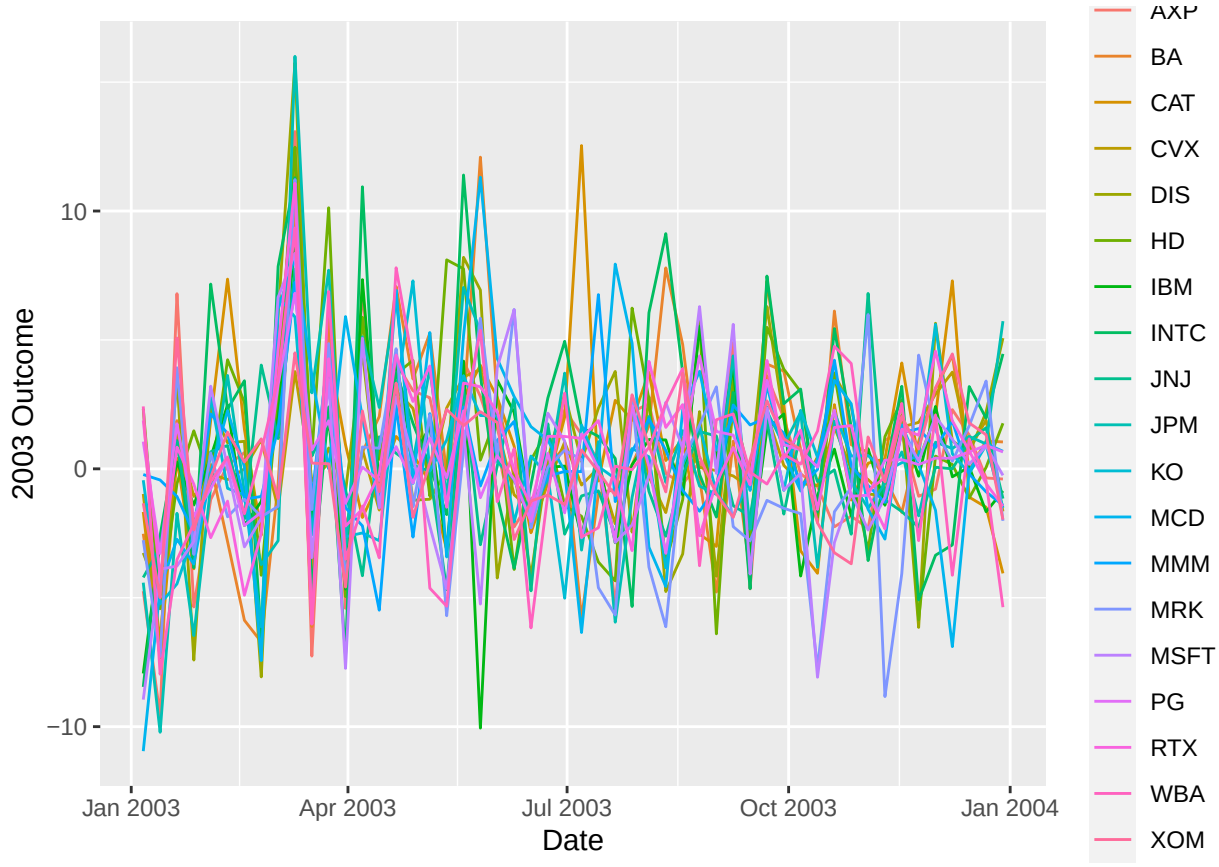

```
dedata %>% filter(year(Date) == 2001) %>% ggplot(aes(Date, Outcome, color=Ticker)) + geom_line() + ylab
```



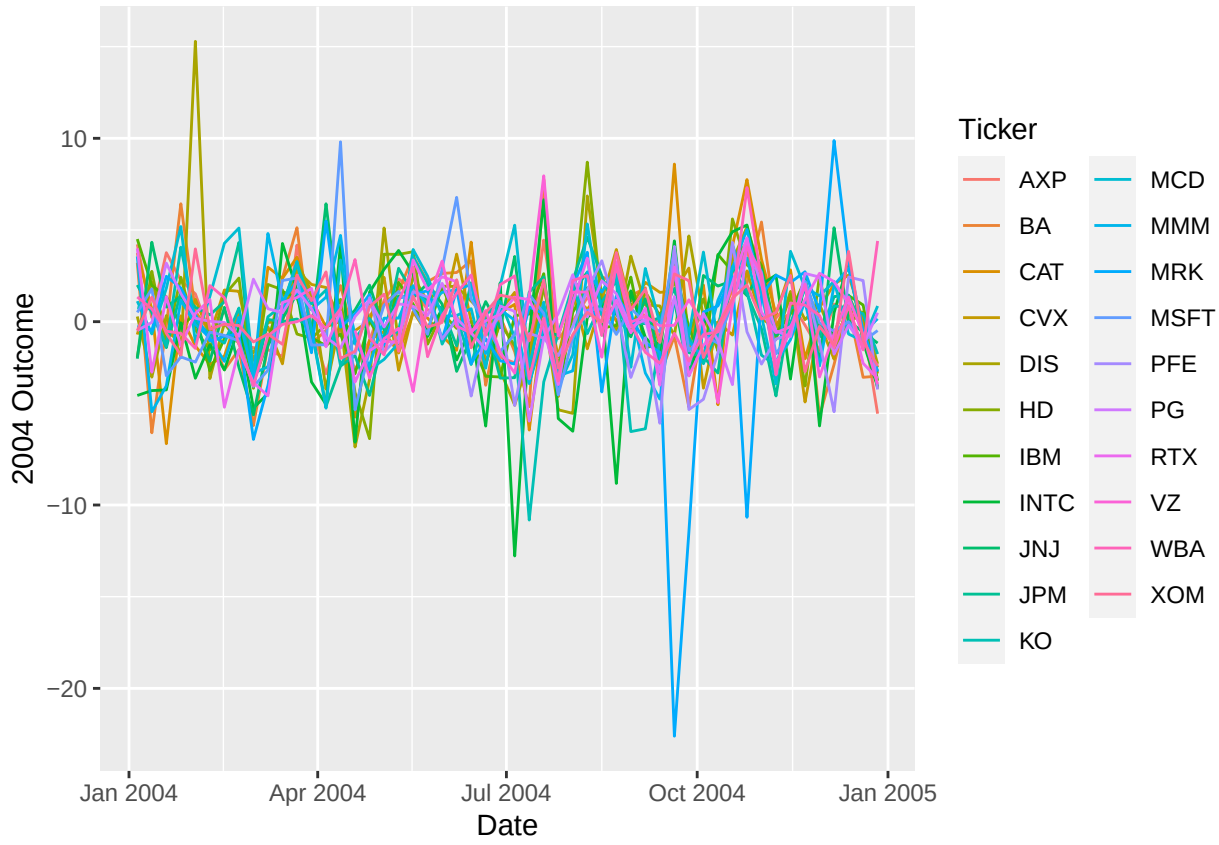

```
dedata %>% filter(year(Date) == 2002) %>% ggplot(aes(Date, Outcome, color=Ticker)) + geom_line() + ylab
```



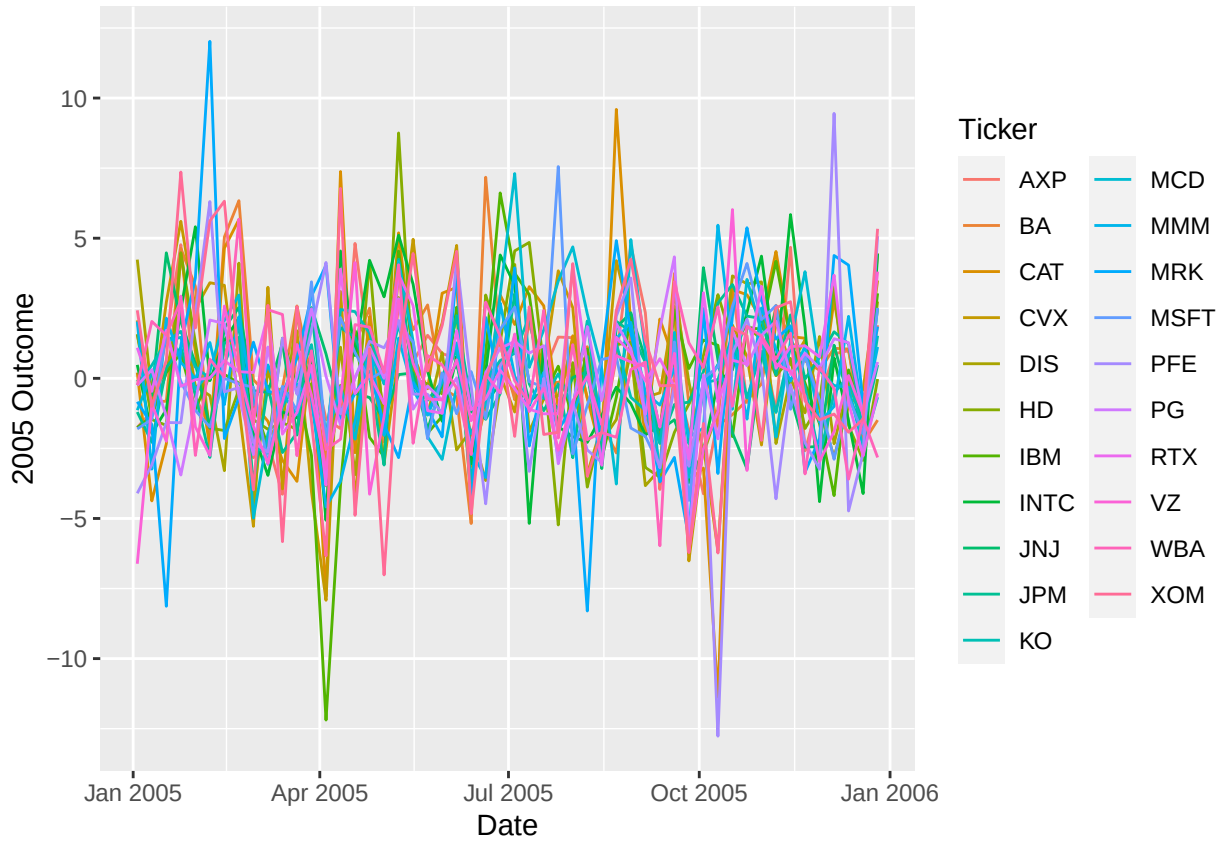

```
dedata %>% filter(year(Date) == 2003) %>% ggplot(aes(Date, Outcome, color=Ticker)) + geom_line() + ylab
```




```
dedata %>% filter(year(Date) == 2004) %>% ggplot(aes(Date, Outcome, color=Ticker)) + geom_line() + ylab
```

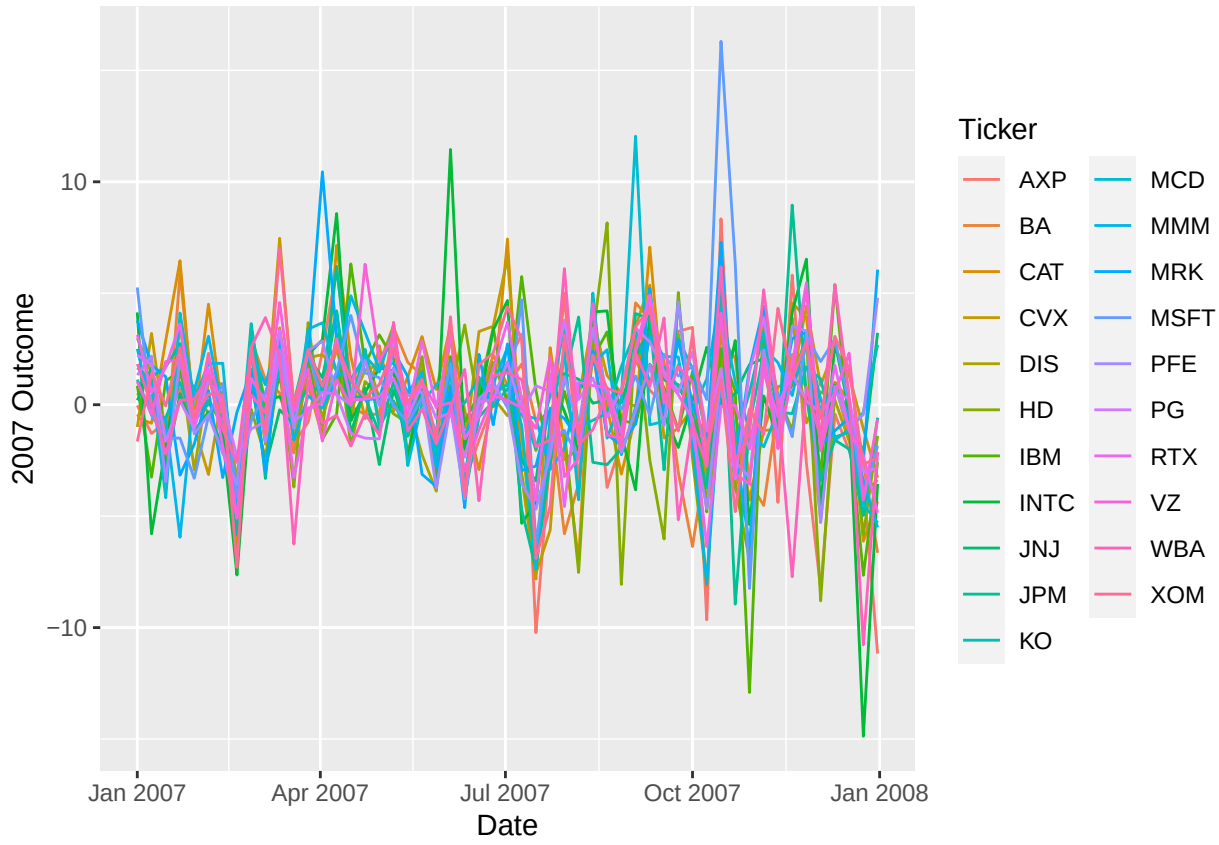



```
dedata %>% filter(year(Date) == 2005) %>% ggplot(aes(Date, Outcome, color=Ticker)) + geom_line() + ylab
```

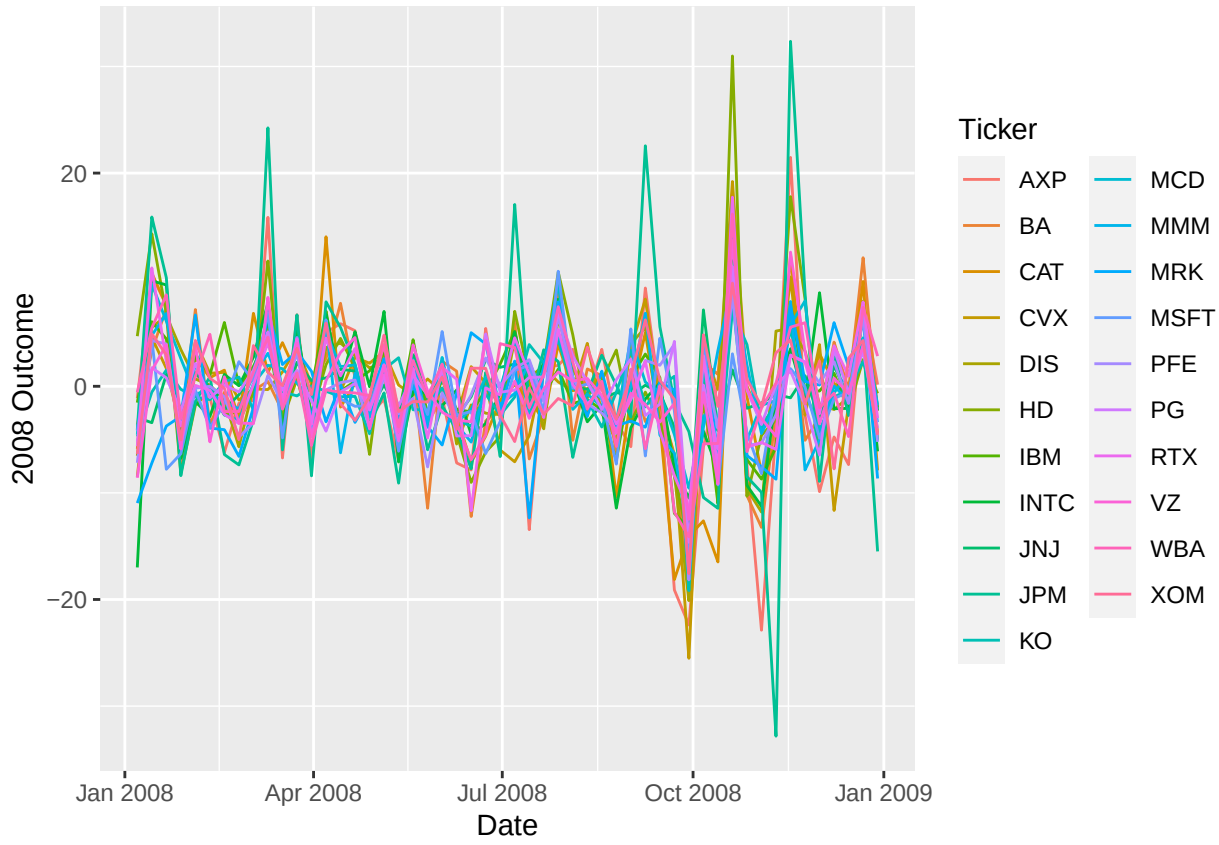



```
dedata %>% filter(year(Date) == 2006) %>% ggplot(aes(Date, Outcome, color=Ticker)) + geom_line() + ylab
```

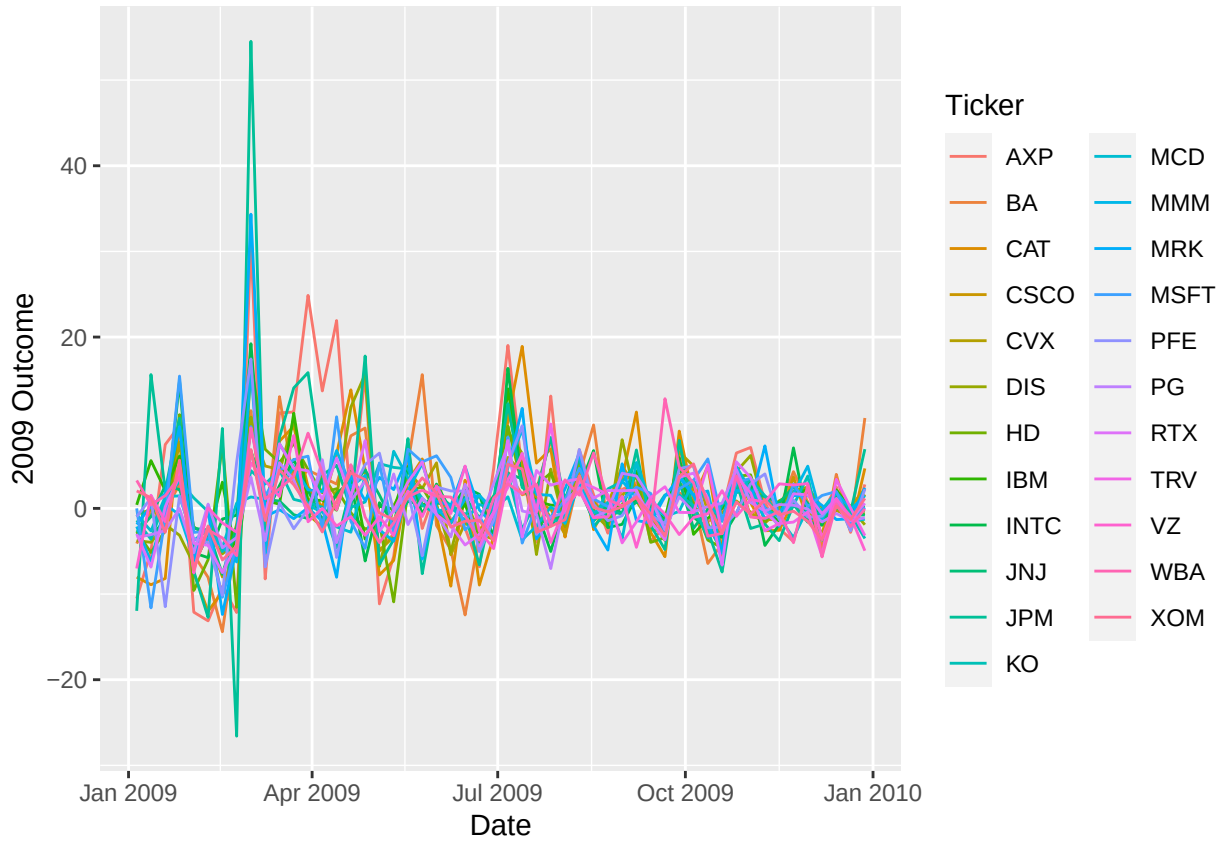

```
dedata %>% filter(year(Date) == 2007) %>% ggplot(aes(Date, Outcome, color=Ticker)) + geom_line() + ylab
```



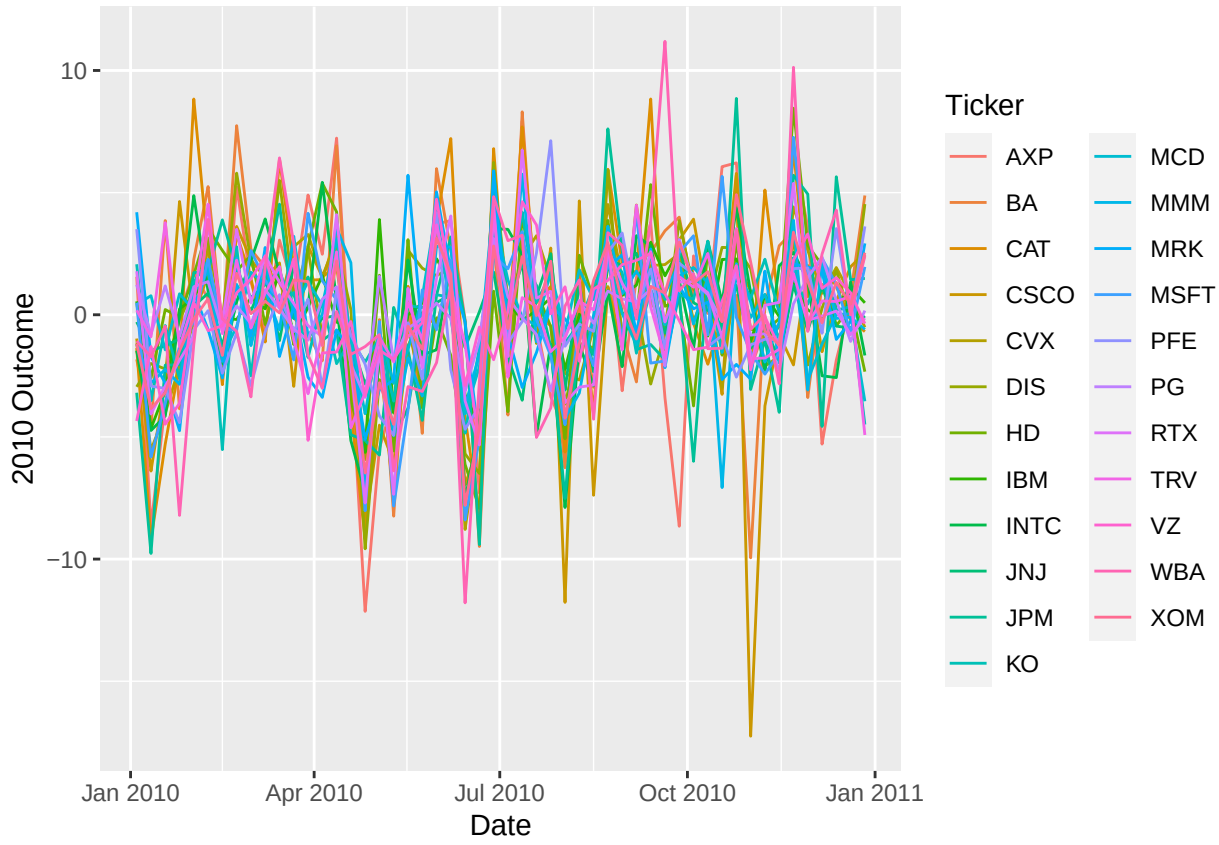

```
dedata %>% filter(year(Date) == 2008) %>% ggplot(aes(Date, Outcome, color=Ticker)) + geom_line() + ylab
```



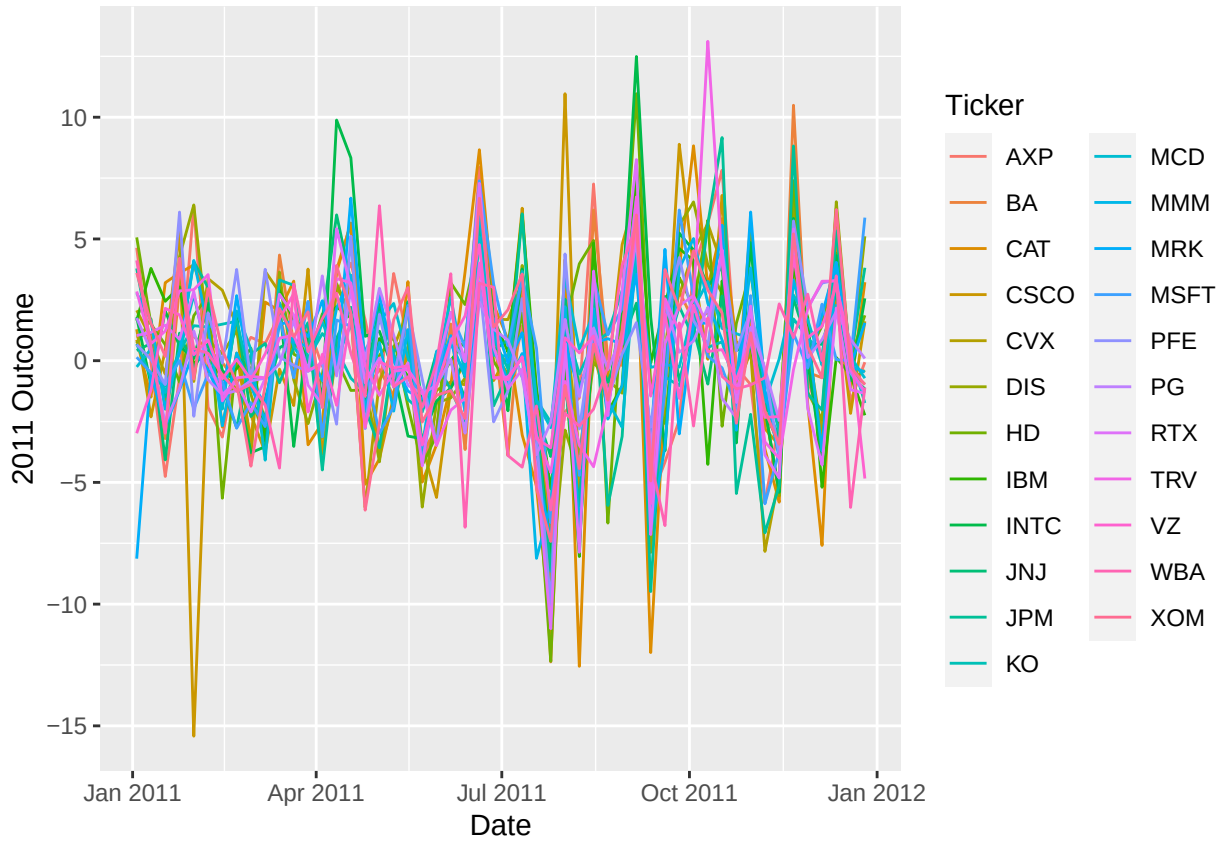

```
dedata %>% filter(year(Date) == 2009) %>% ggplot(aes(Date, Outcome, color=Ticker)) + geom_line() + ylab
```



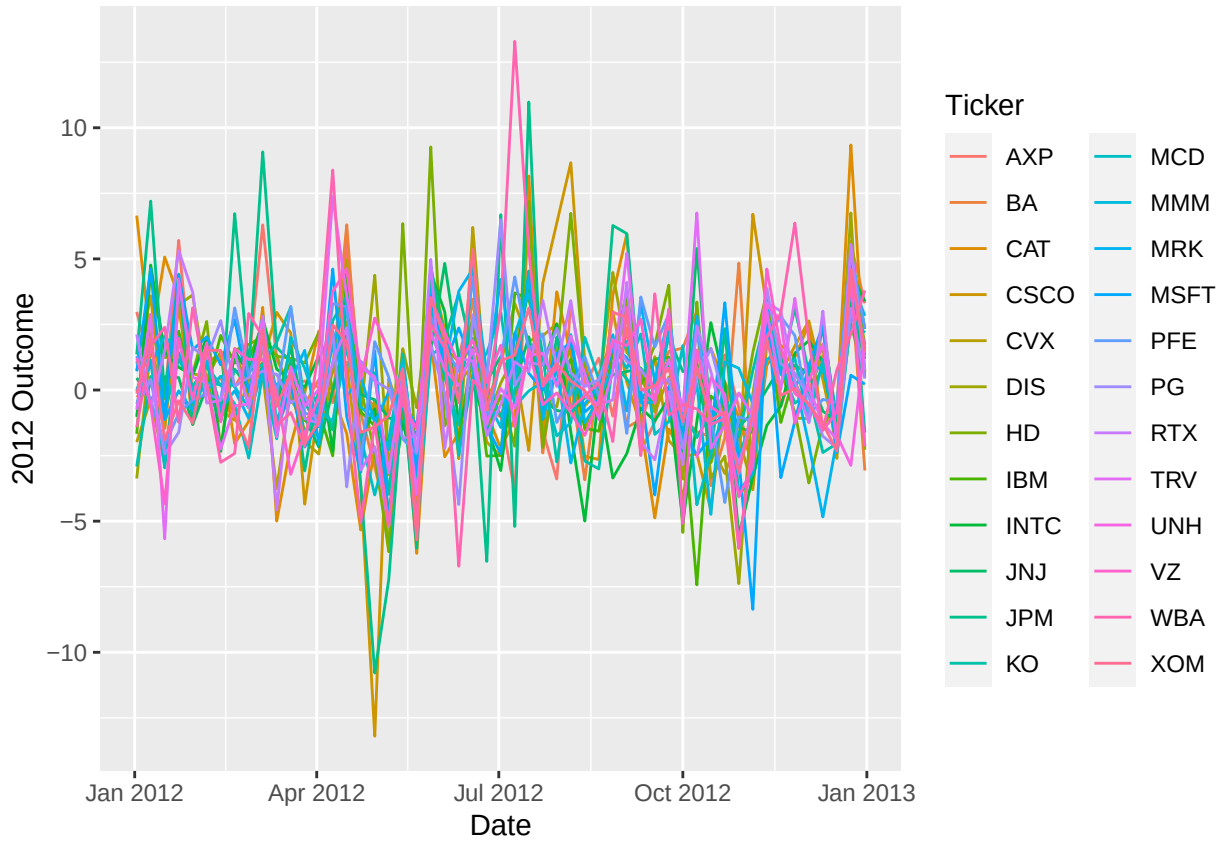

```
dedata %>% filter(year(Date) == 2010) %>% ggplot(aes(Date, Outcome, color=Ticker)) + geom_line() + ylab
```



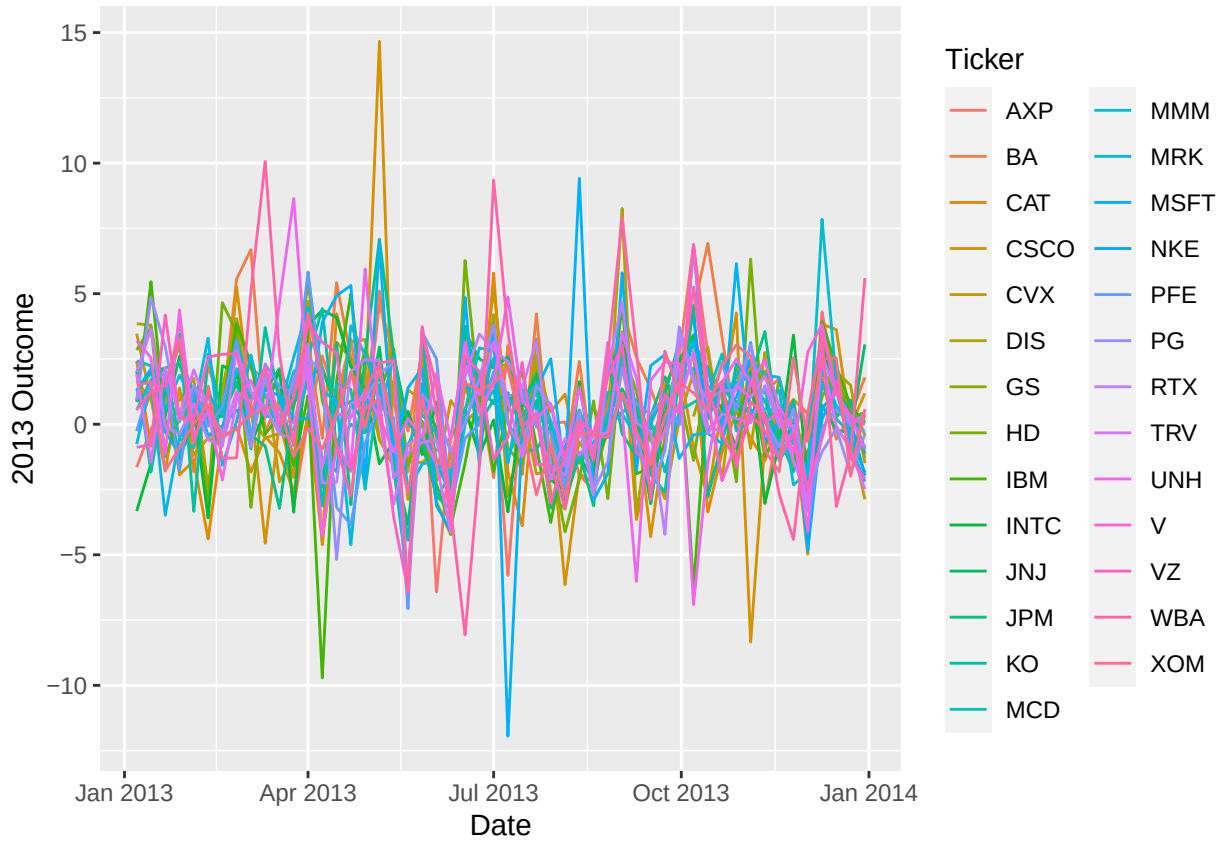

```
dedata %>% filter(year(Date) == 2011) %>% ggplot(aes(Date, Outcome, color=Ticker)) + geom_line() + ylab
```



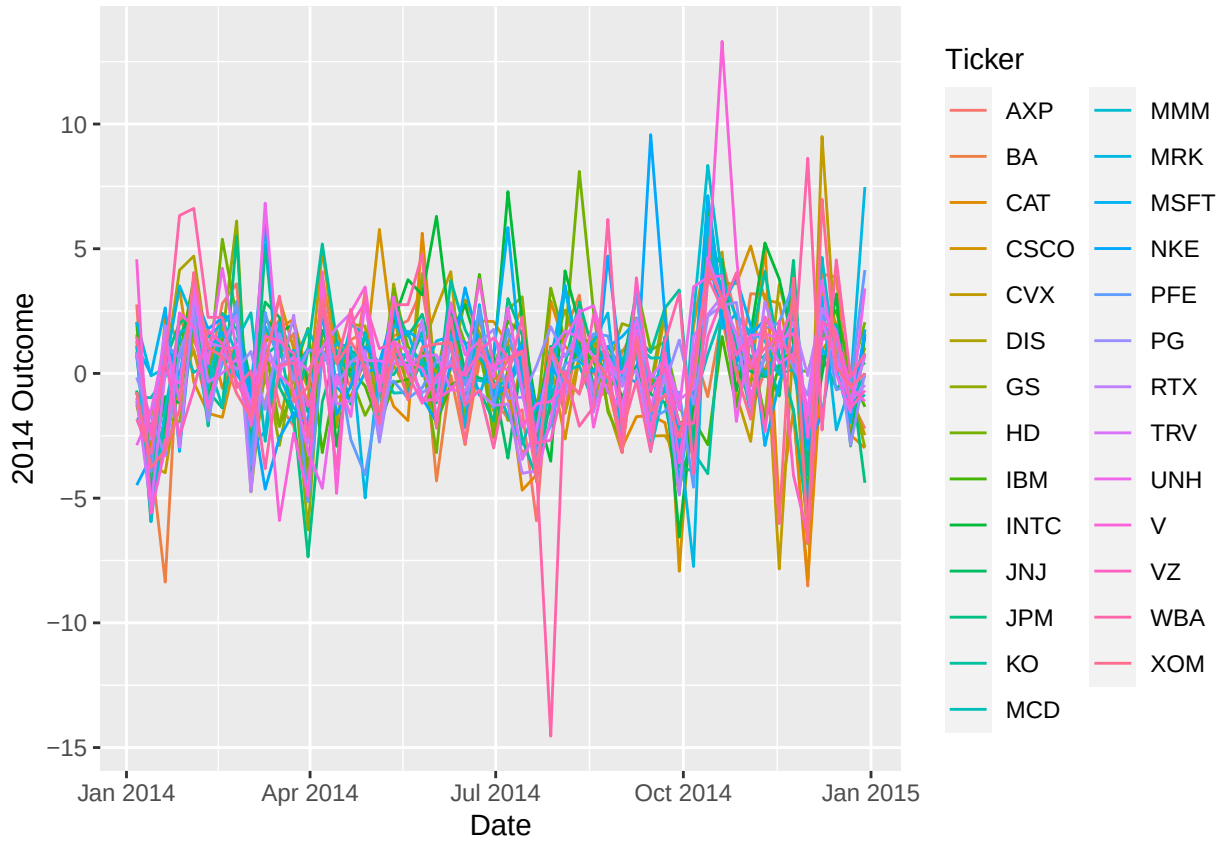

```
dedata %>% filter(year(Date) == 2012) %>% ggplot(aes(Date, Outcome, color=Ticker)) + geom_line() + ylab
```



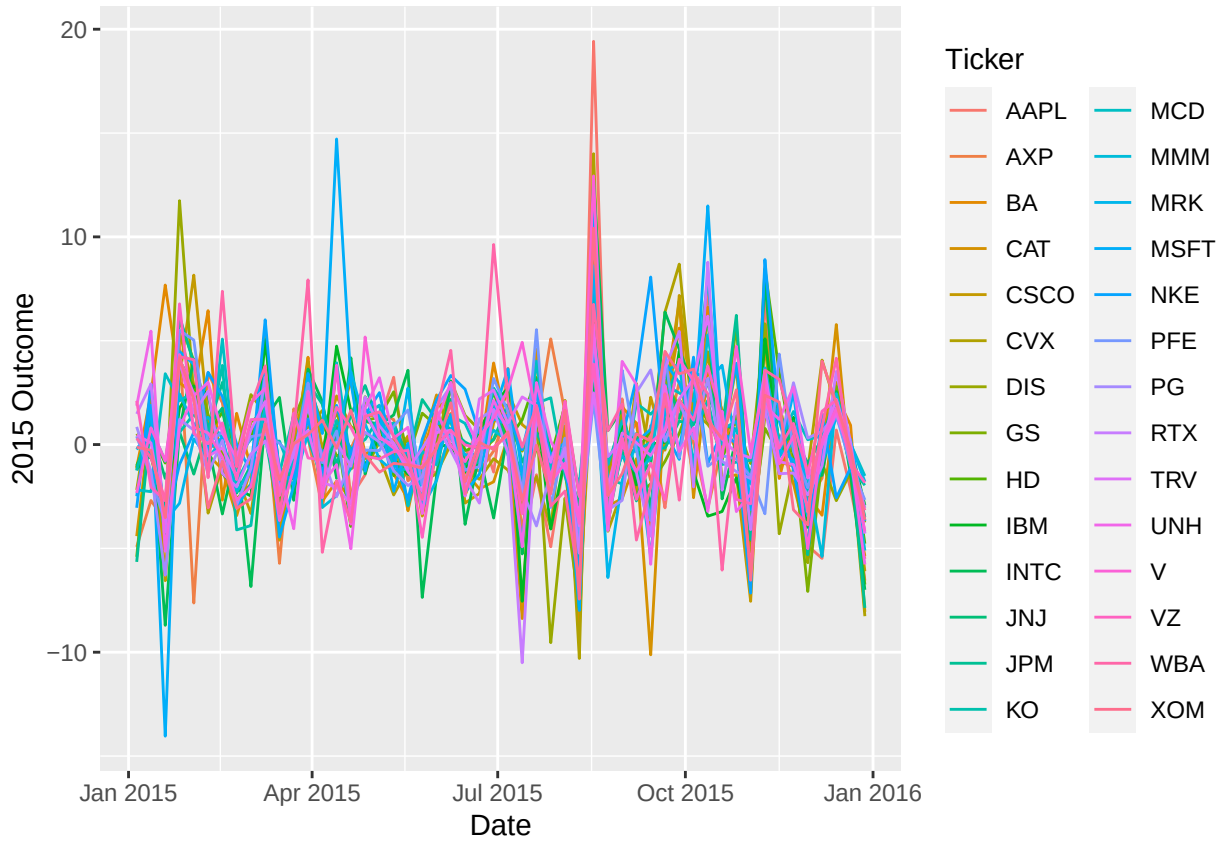

```
dedata %>% filter(year(Date) == 2013) %>% ggplot(aes(Date, Outcome, color=Ticker)) + geom_line() + ylab
```



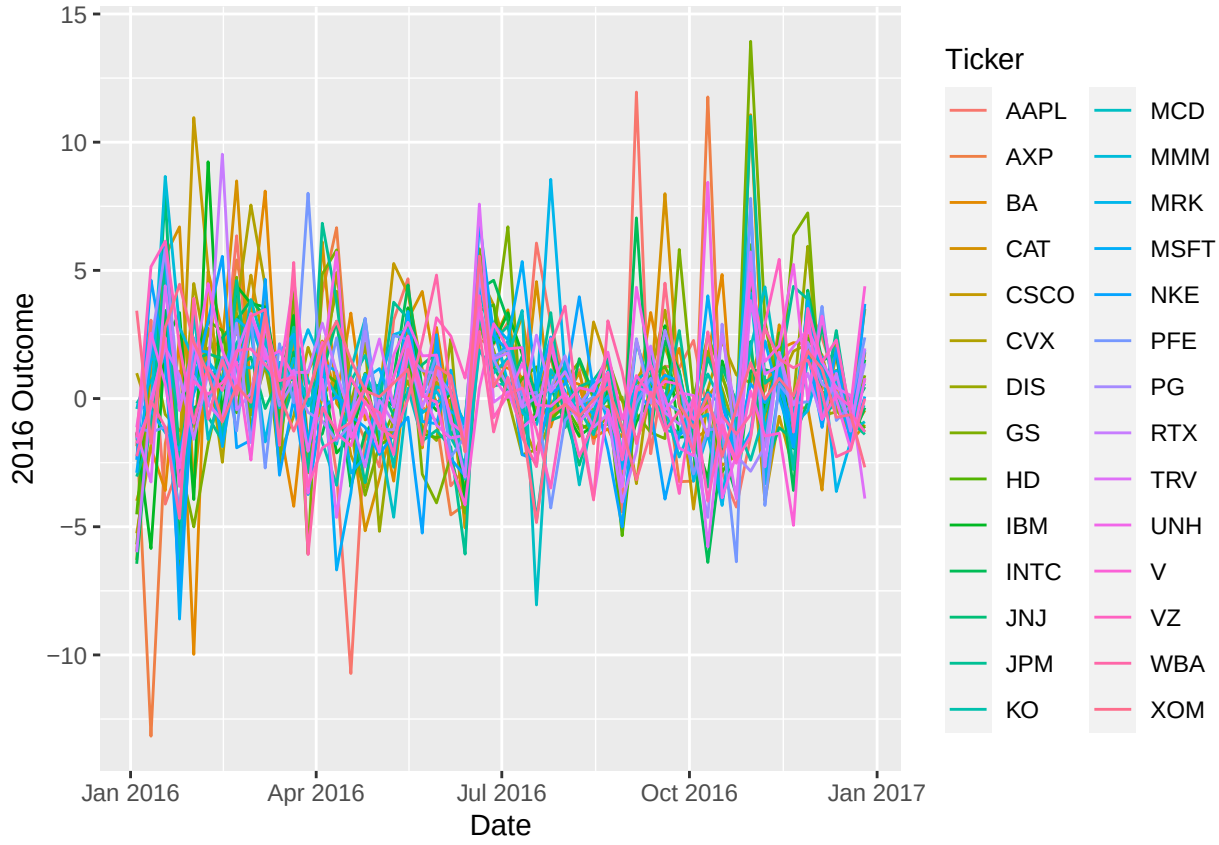

```
dedata %>% filter(year(Date) == 2014) %>% ggplot(aes(Date, Outcome, color=Ticker)) + geom_line() + ylab
```



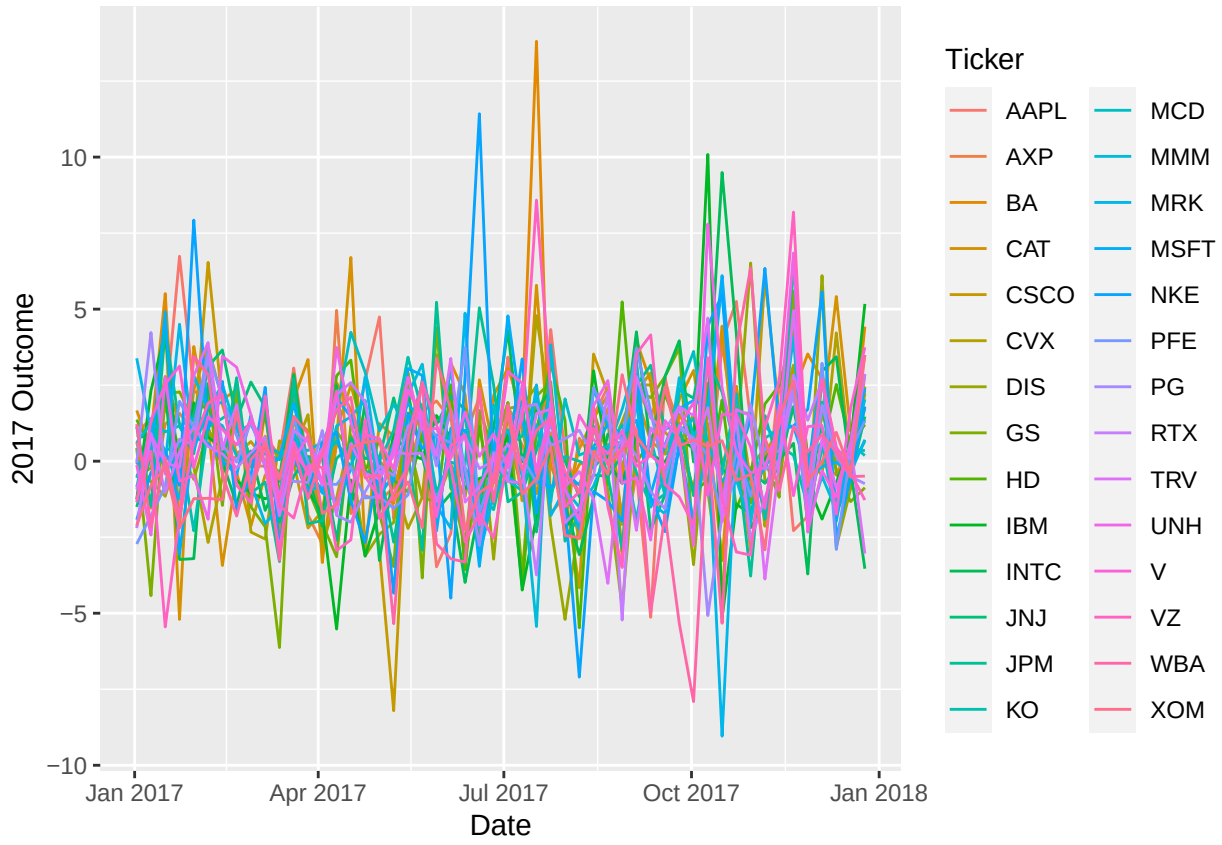

```
dedata %>% filter(year(Date) == 2015) %>% ggplot(aes(Date, Outcome, color=Ticker)) + geom_line() + ylab
```



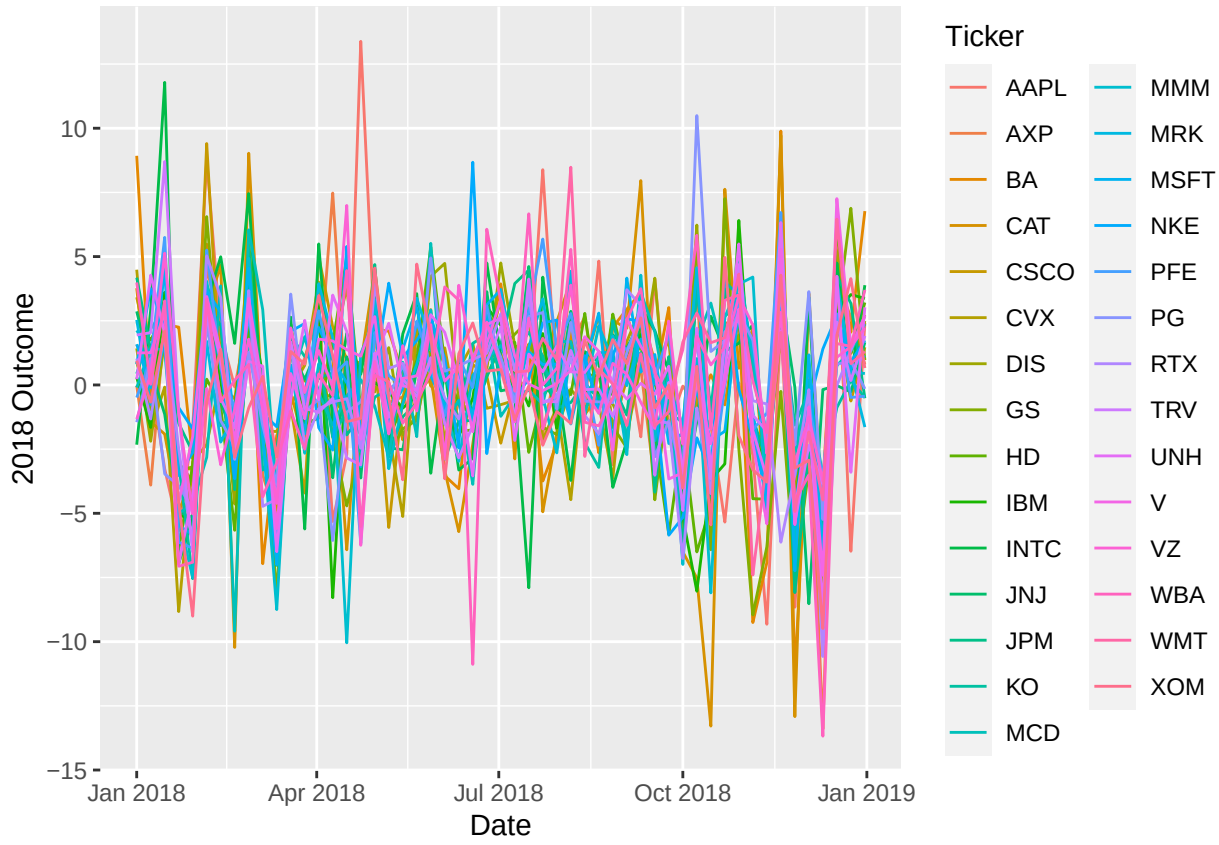

```
dedata %>% filter(year(Date) == 2016) %>% ggplot(aes(Date, Outcome, color=Ticker)) + geom_line() + ylab
```



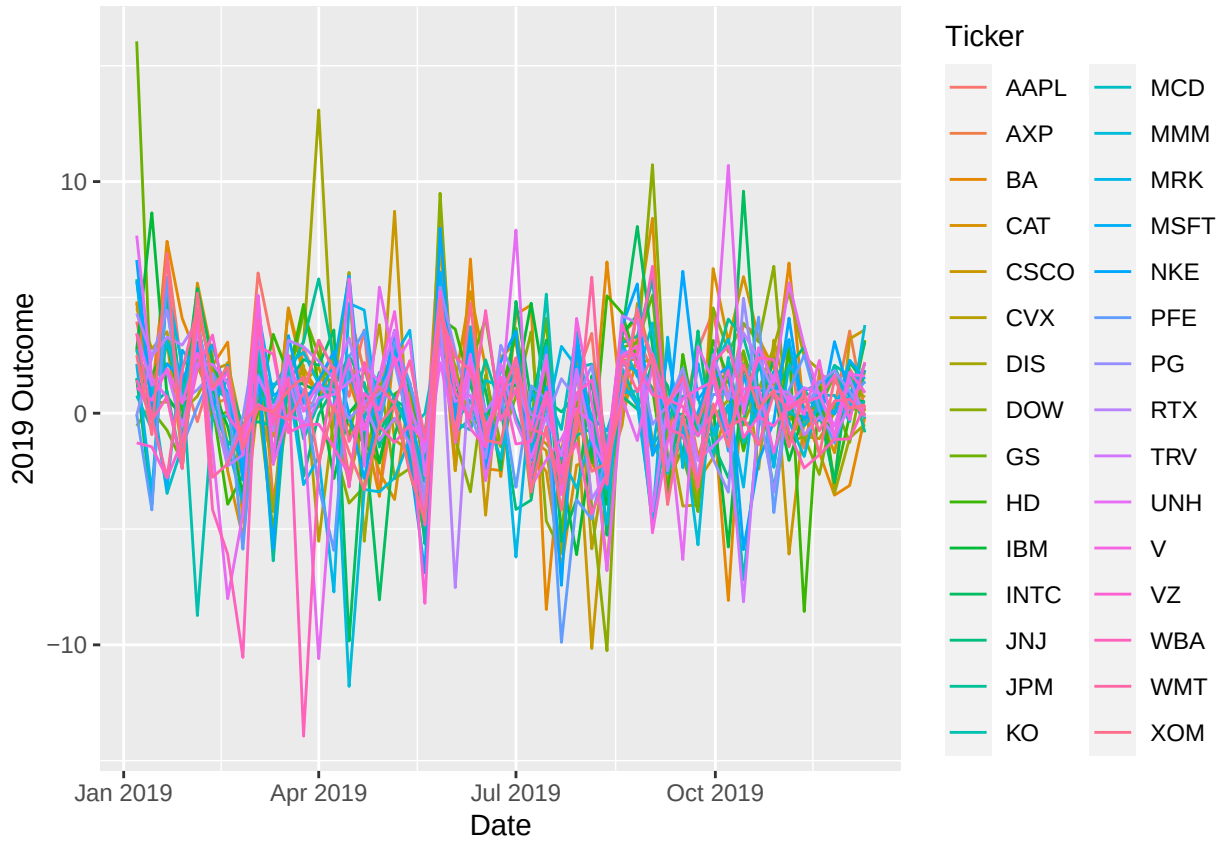

```
dedata %>% filter(year(Date) == 2017) %>% ggplot(aes(Date, Outcome, color=Ticker)) + geom_line() + ylab
```




```
dedata %>% filter(year(Date) == 2018) %>% ggplot(aes(Date, Outcome, color=Ticker)) + geom_line() + ylab
```




```
dedata %>% filter(year(Date) == 2019) %>% ggplot(aes(Date, Outcome, color=Ticker)) + geom_line() + ylab
```



Looking over the years, on an average,

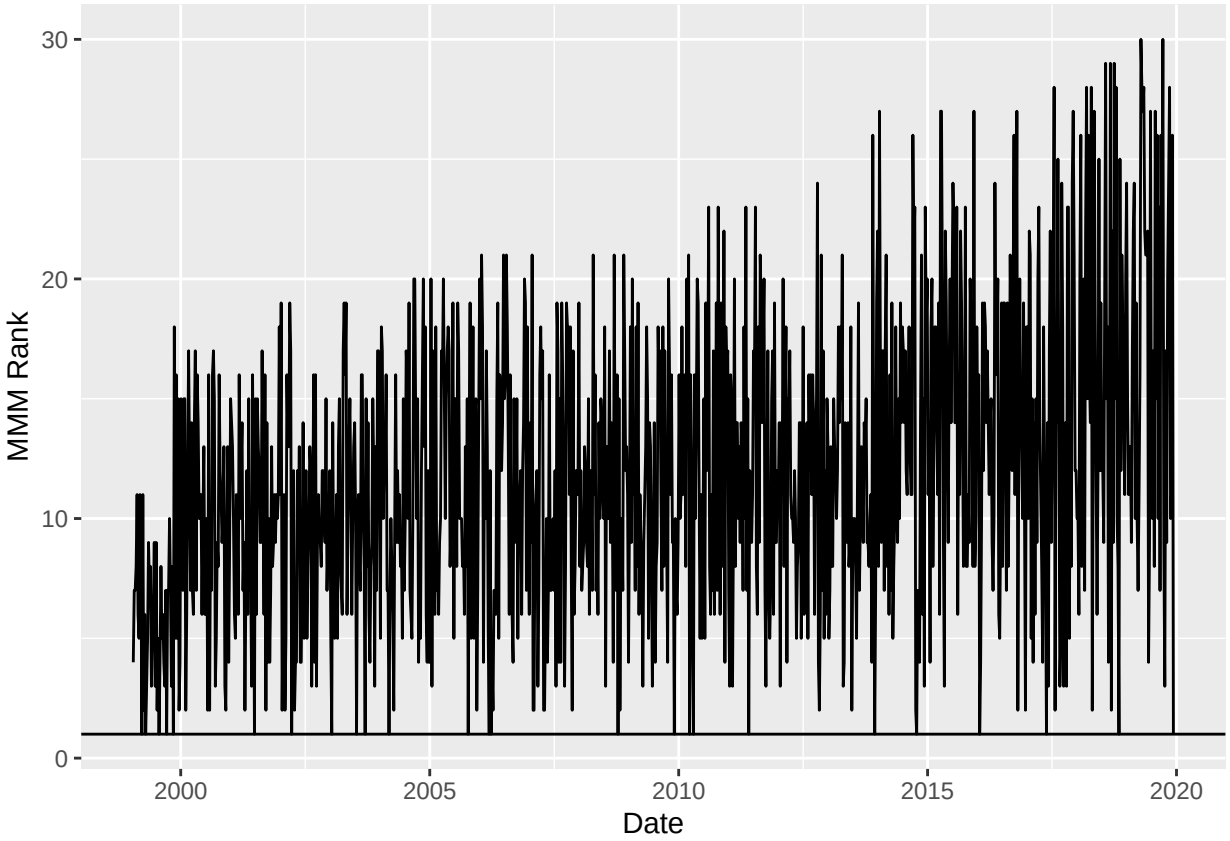
1. If a stock falls by 5% to 10% in a week there is an immediate pullback.
2. If a stock rises 10% or more in a week, there is a high likelihood that next week will be lackluster.

2.5 Features

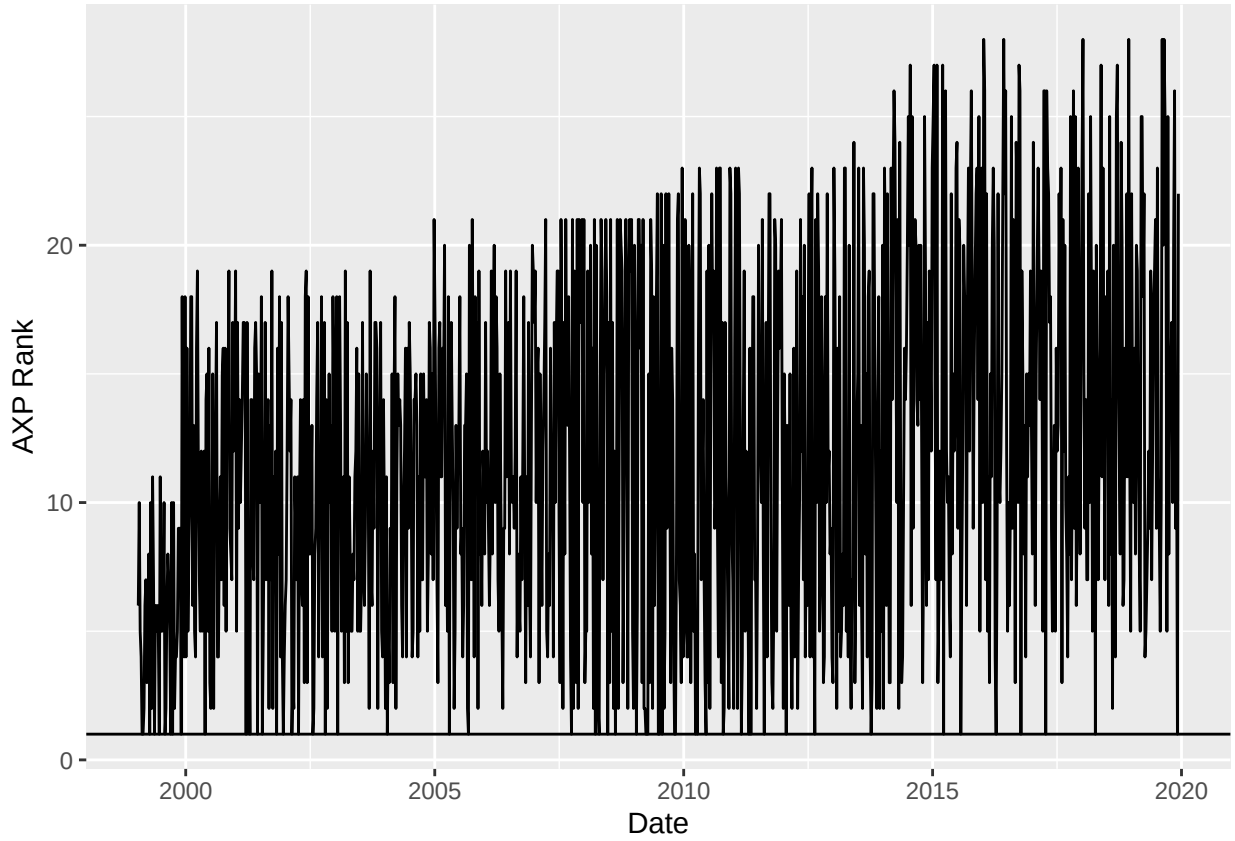
2.5.1 Feature: Today's Rank, yesterday's Rank ...

Check out each Ticker Rank history.

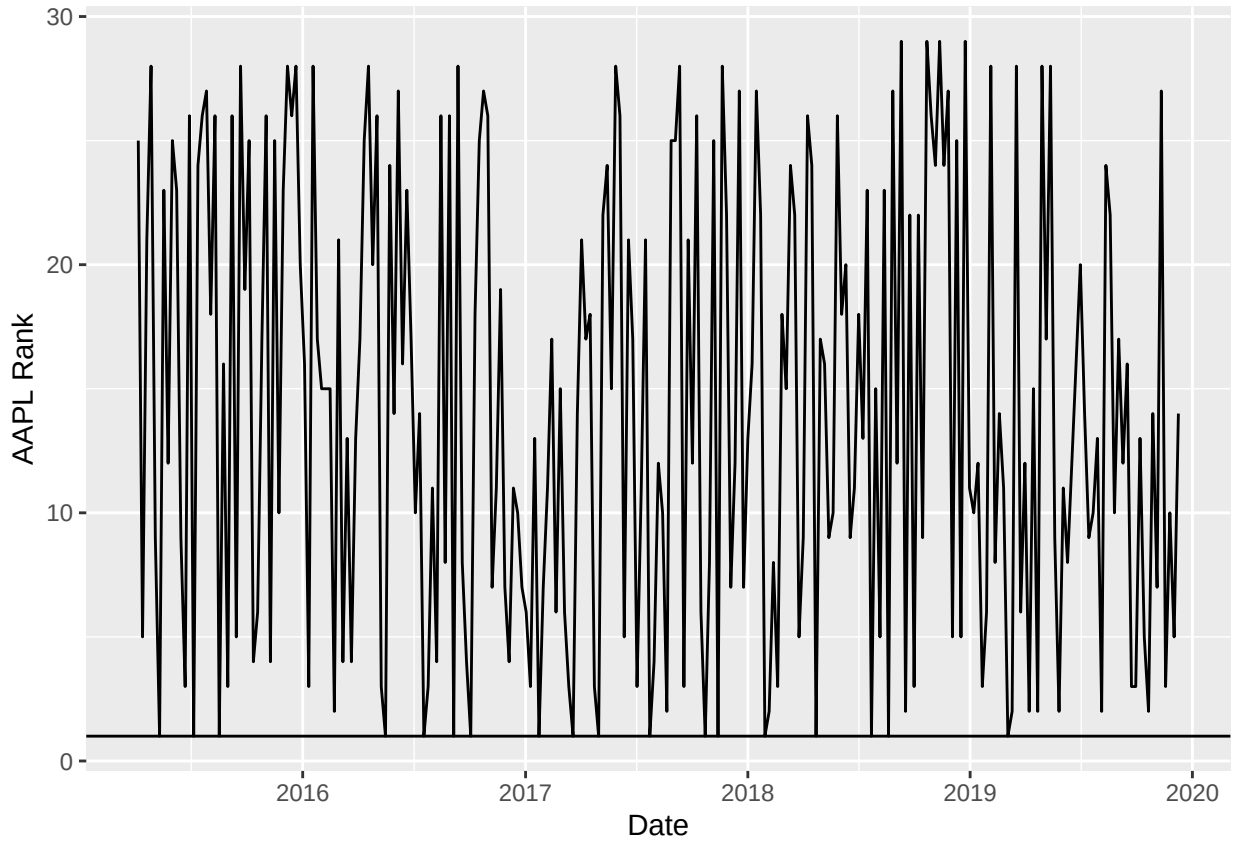
```
# Check Ranks for each Ticker lifespan
dedata %>% filter(Ticker == "MMM") %>% ggplot(aes(Date, Rank)) + geom_line() + ylab("MMM Rank") + geom_
```



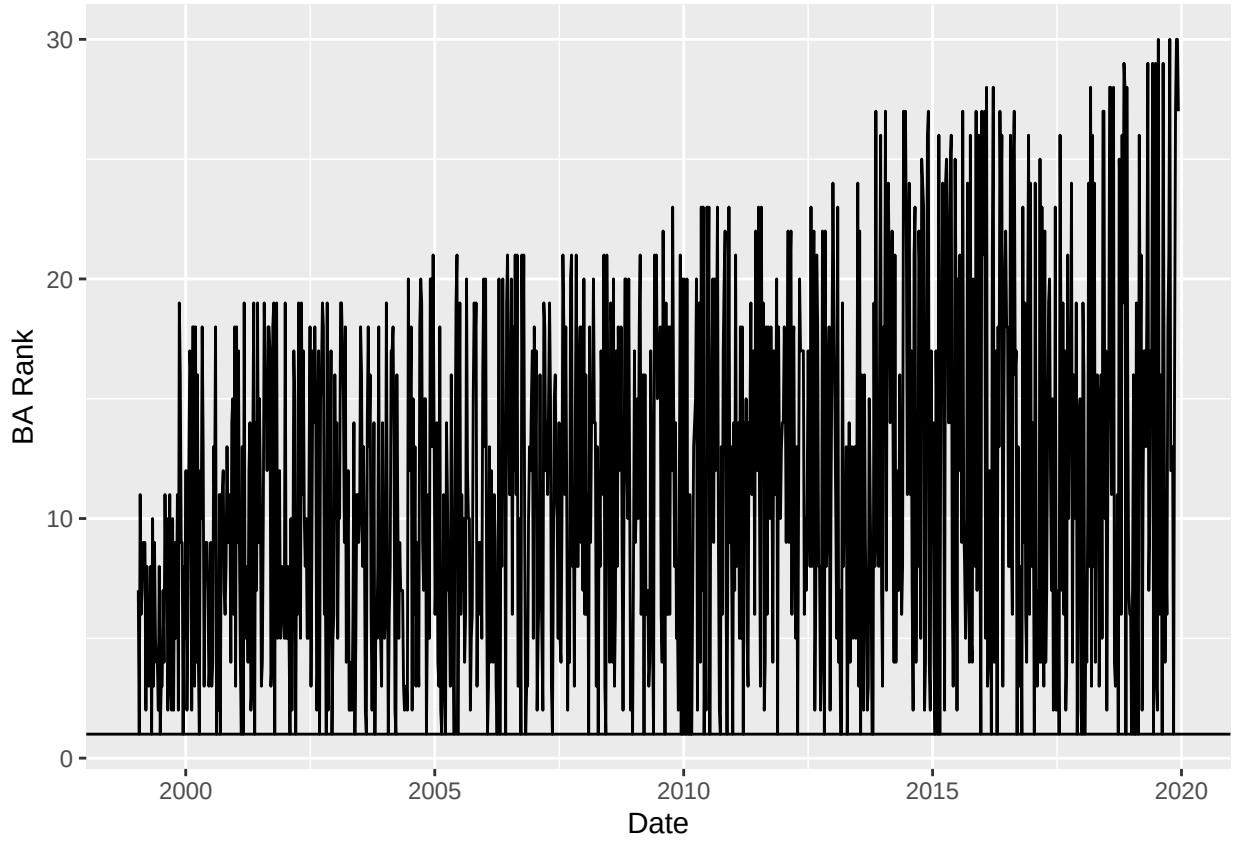

```
dedata %>% filter(Ticker == "AXP") %>% ggplot(aes(Date, Rank)) + geom_line() + ylab("AXP Rank") + geom_
```



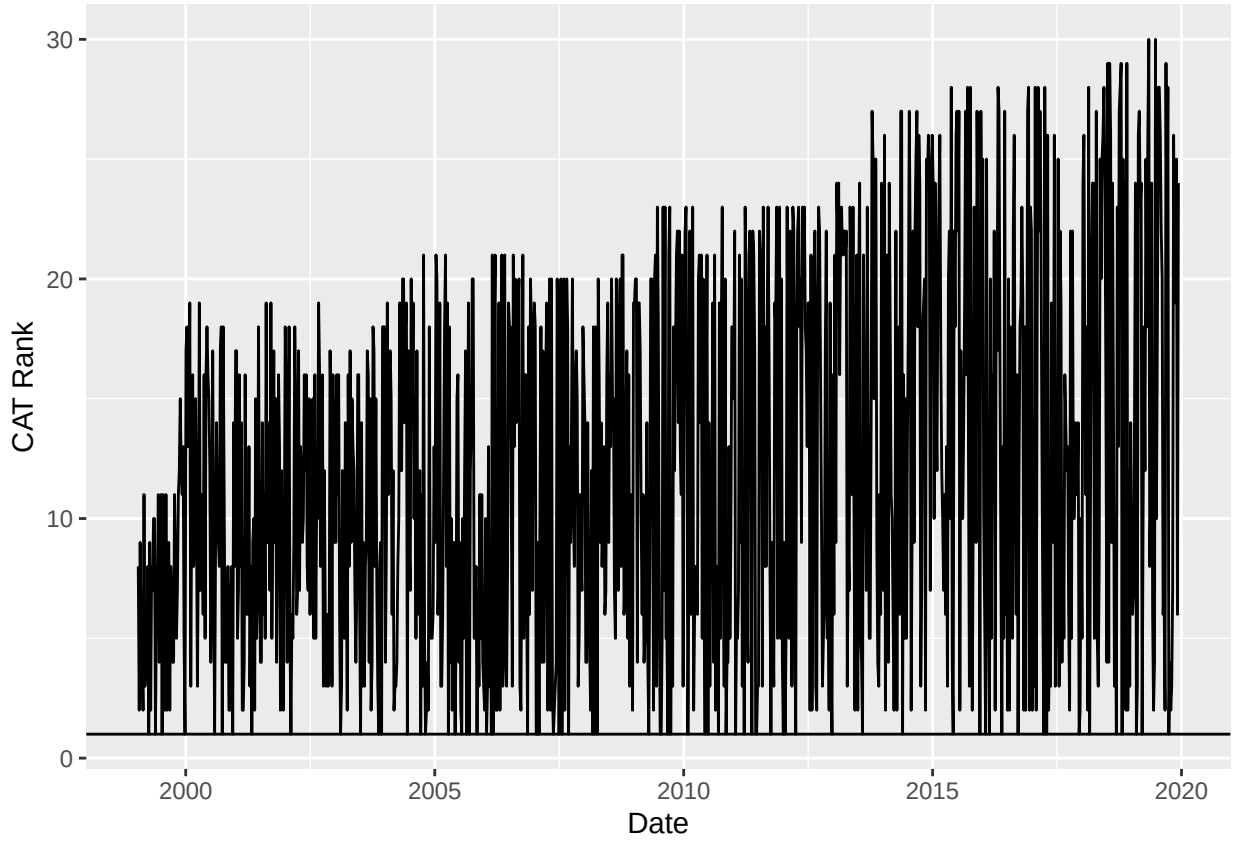

```
dedata %>% filter(Ticker == "AAPL") %>% ggplot(aes(Date, Rank)) + geom_line() + ylab("AAPL Rank") + geo
```



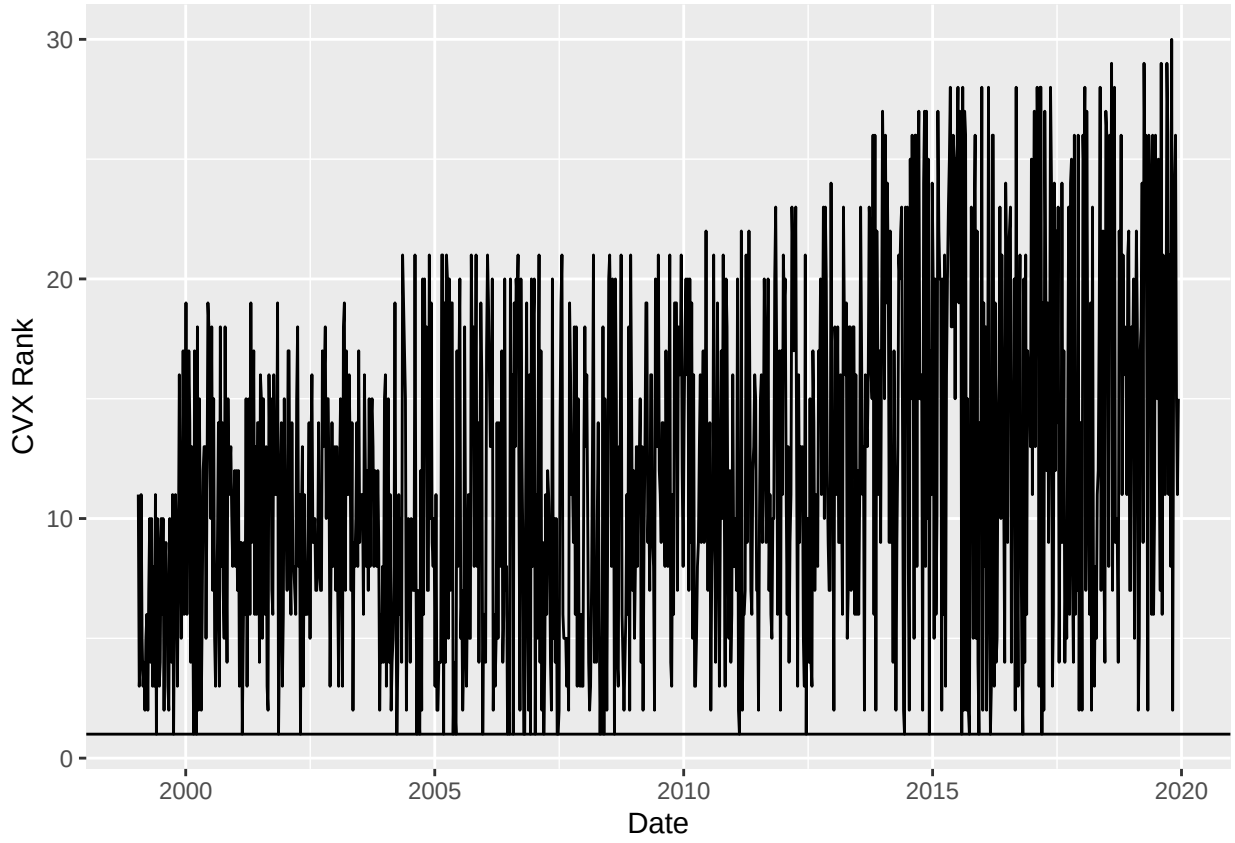

```
dedata %>% filter(Ticker == "BA") %>% ggplot(aes(Date, Rank)) + geom_line() + ylab("BA Rank") + geom_ab
```



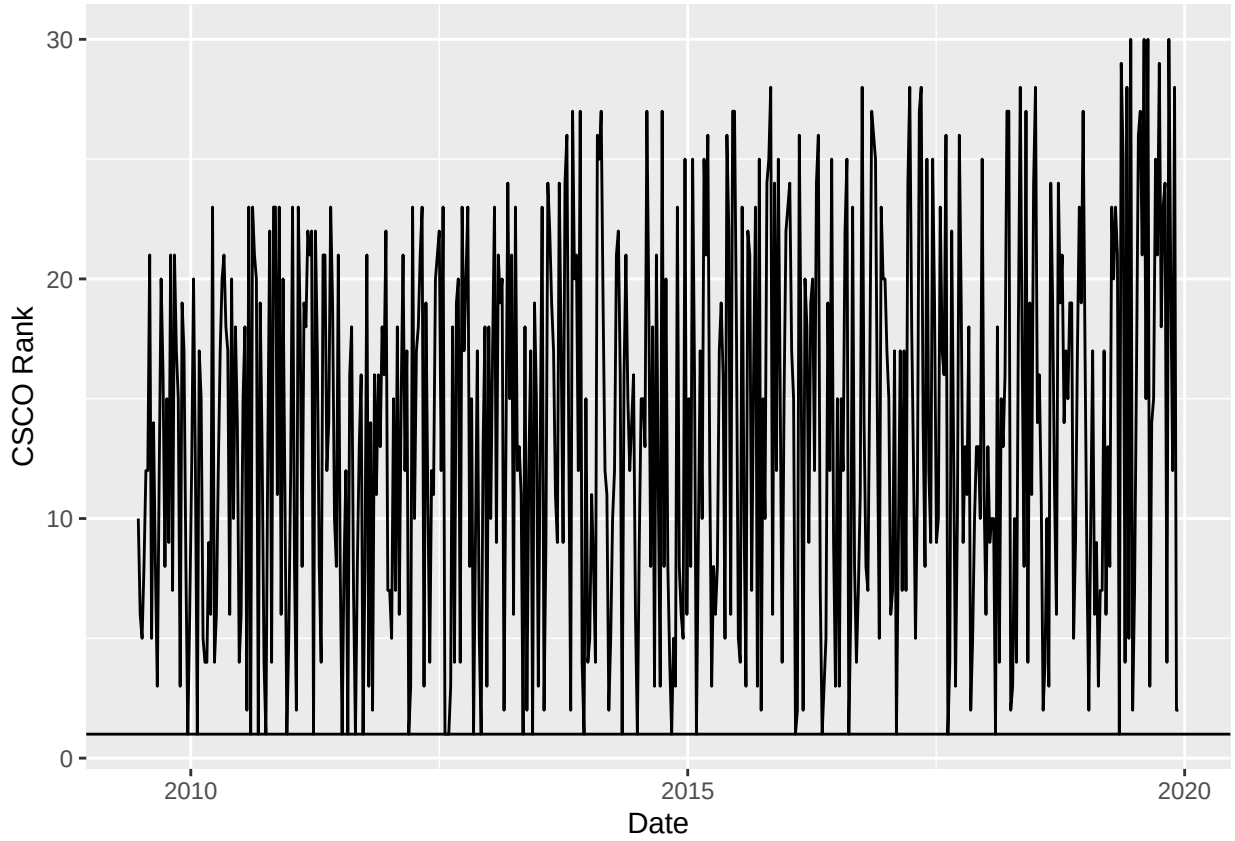

```
dedata %>% filter(Ticker == "CAT") %>% ggplot(aes(Date, Rank)) + geom_line() + ylab("CAT Rank") + geom_
```



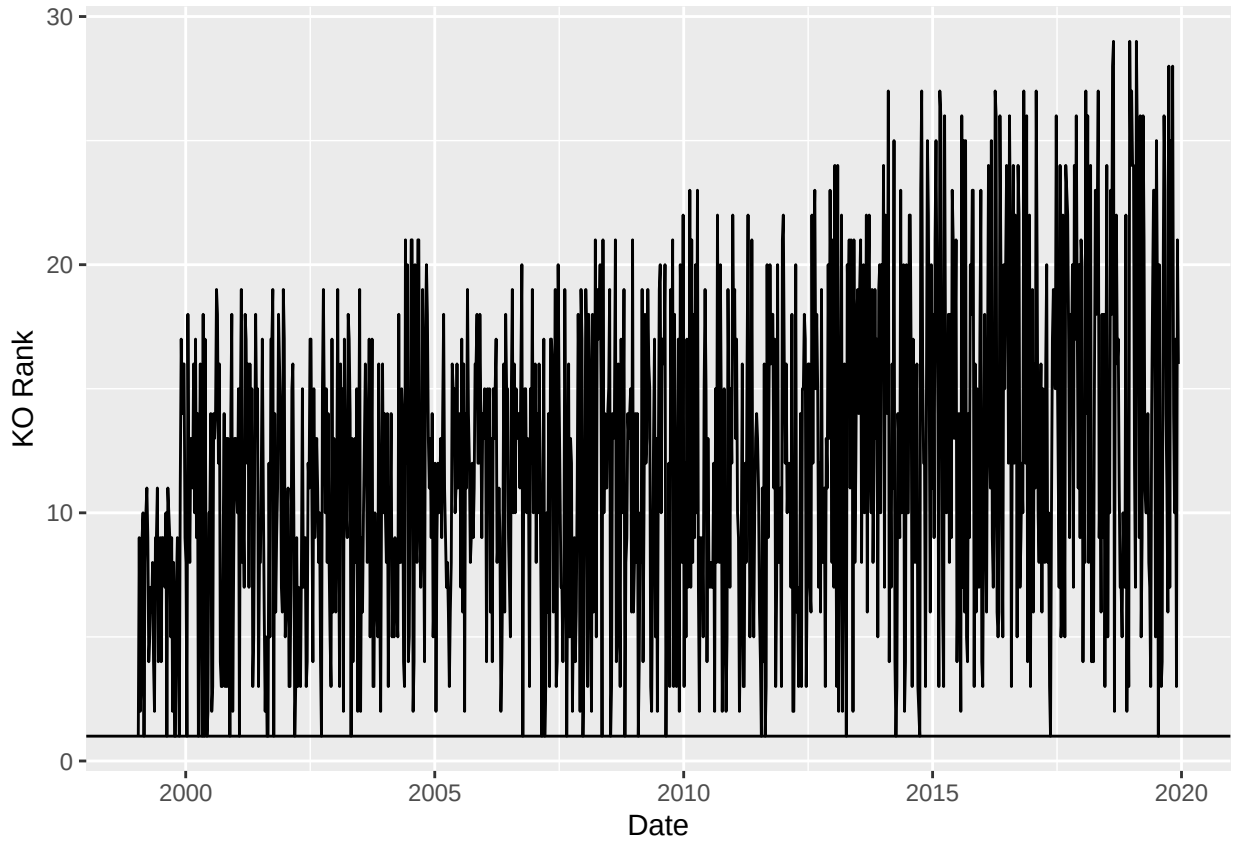

```
dedata %>% filter(Ticker == "CVX") %>% ggplot(aes(Date, Rank)) + geom_line() + ylab("CVX Rank") + geom_
```



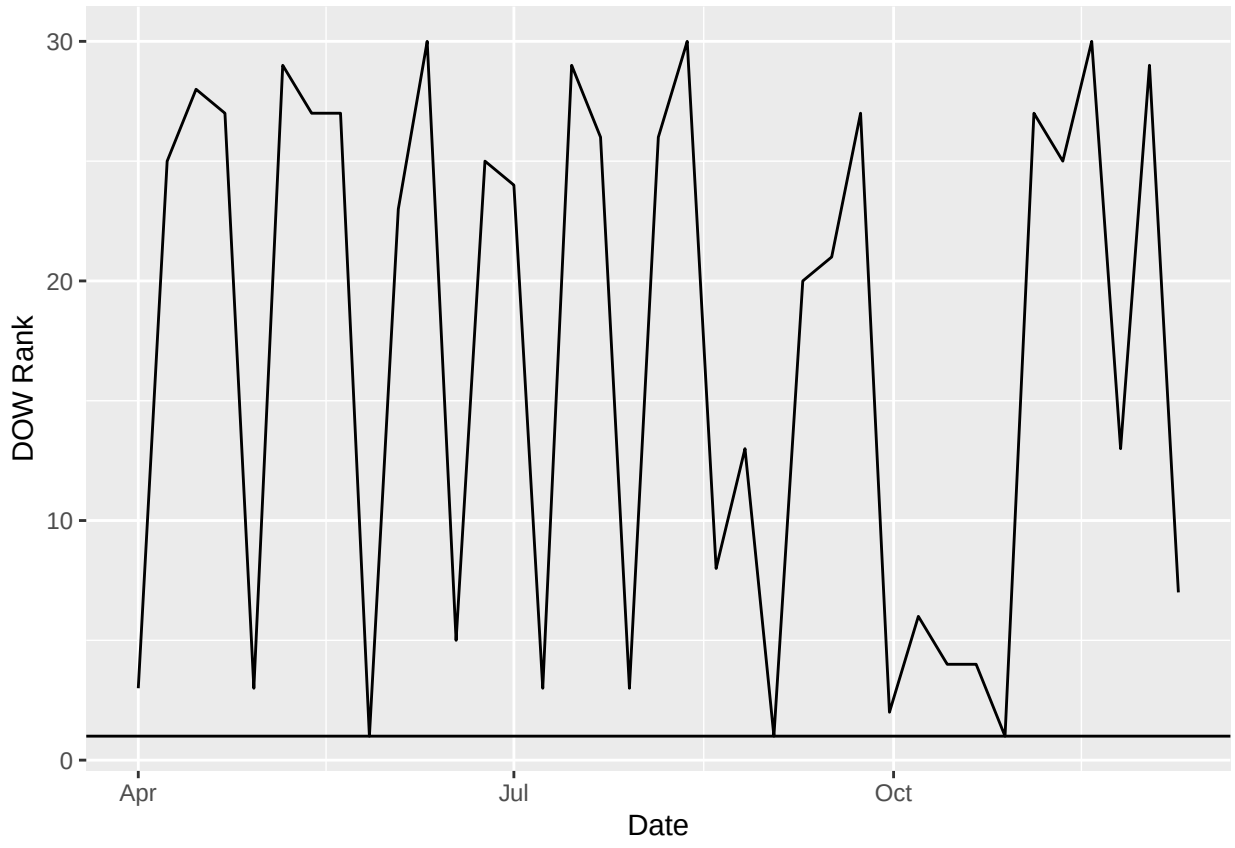

```
dedata %>% filter(Ticker == "CSCO") %>% ggplot(aes(Date, Rank)) + geom_line() + ylab("CSCO Rank") + geo
```



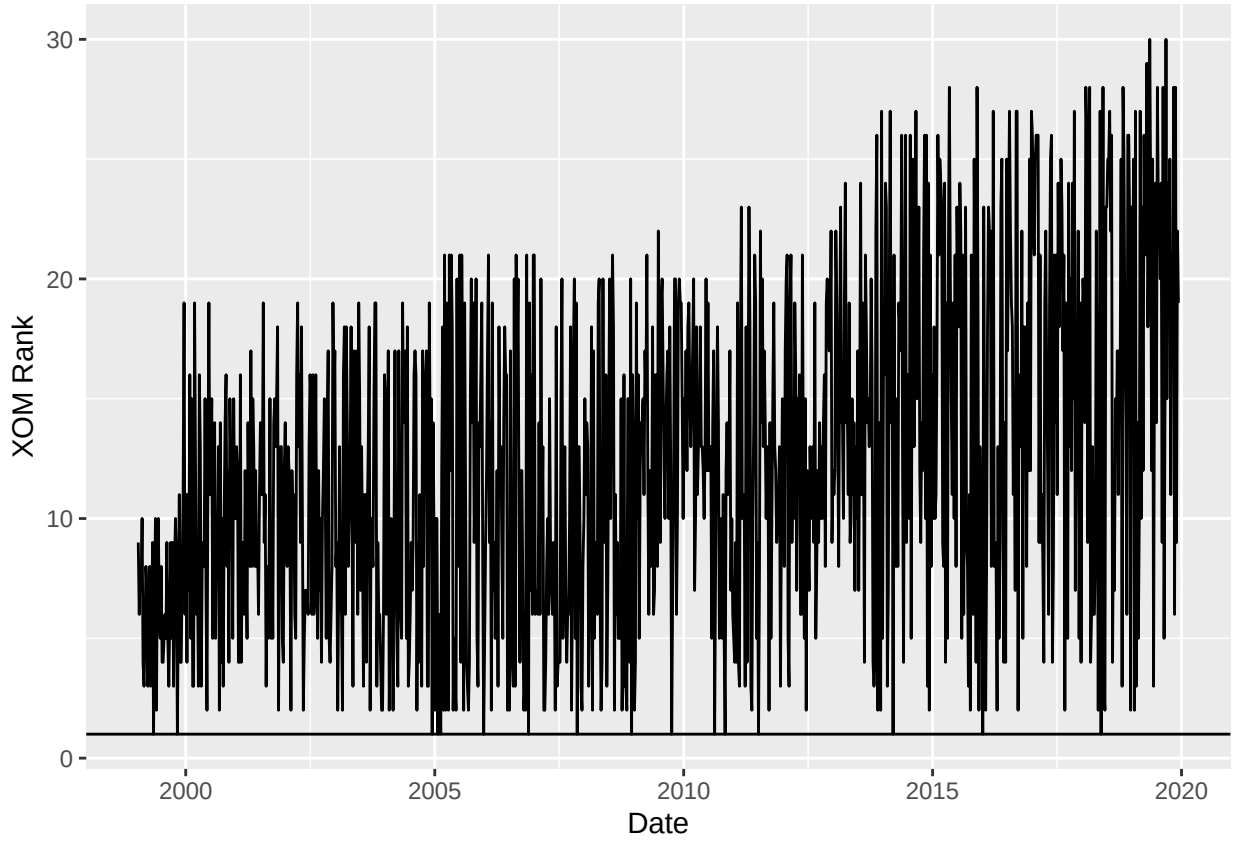

```
dedata %>% filter(Ticker == "KO") %>% ggplot(aes(Date, Rank)) + geom_line() + ylab("KO Rank") + geom_ab
```



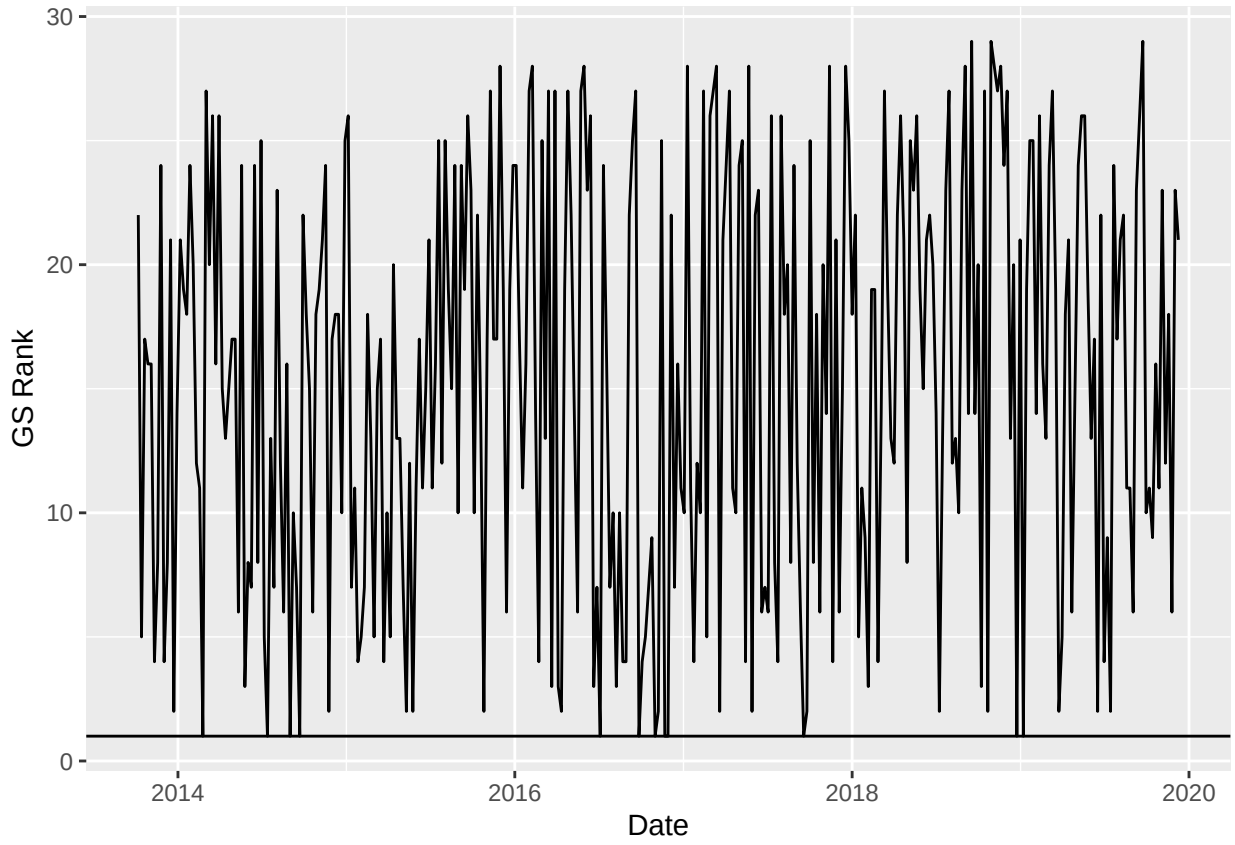

```
dedata %>% filter(Ticker == "DOW") %>% ggplot(aes(Date, Rank)) + geom_line() + ylab("DOW Rank") + geom_
```



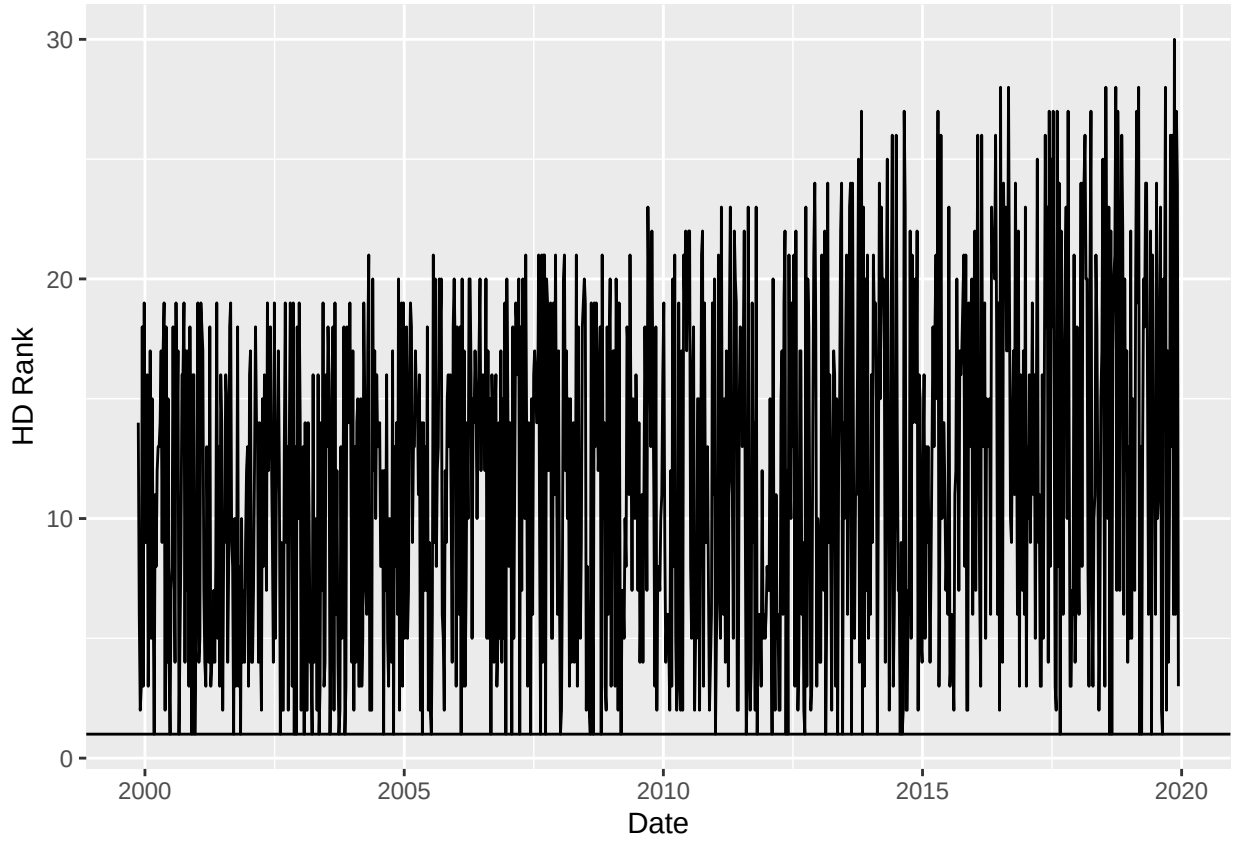

```
dedata %>% filter(Ticker == "XOM") %>% ggplot(aes(Date, Rank)) + geom_line() + ylab("XOM Rank") + geom_
```



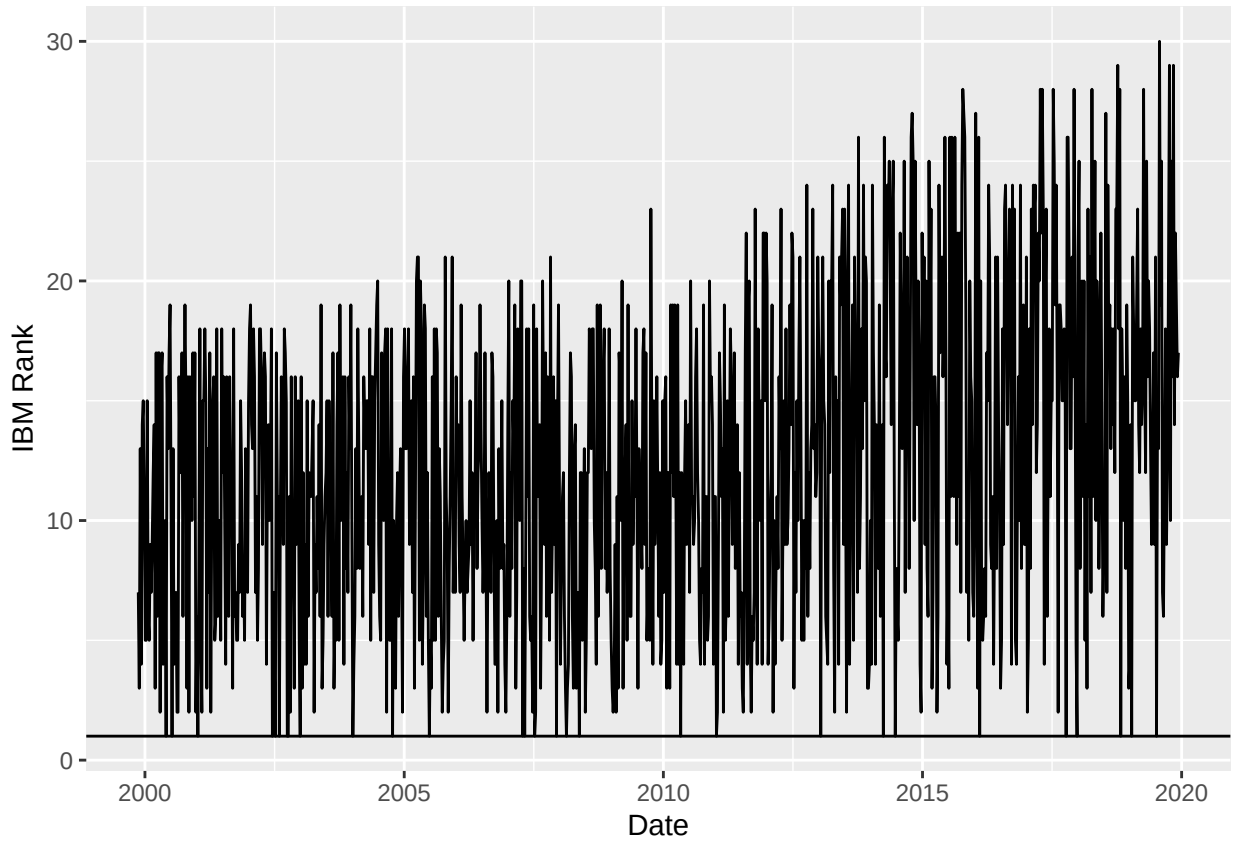

```
dedata %>% filter(Ticker == "GS") %>% ggplot(aes(Date, Rank)) + geom_line() + ylab("GS Rank") + geom_ab
```



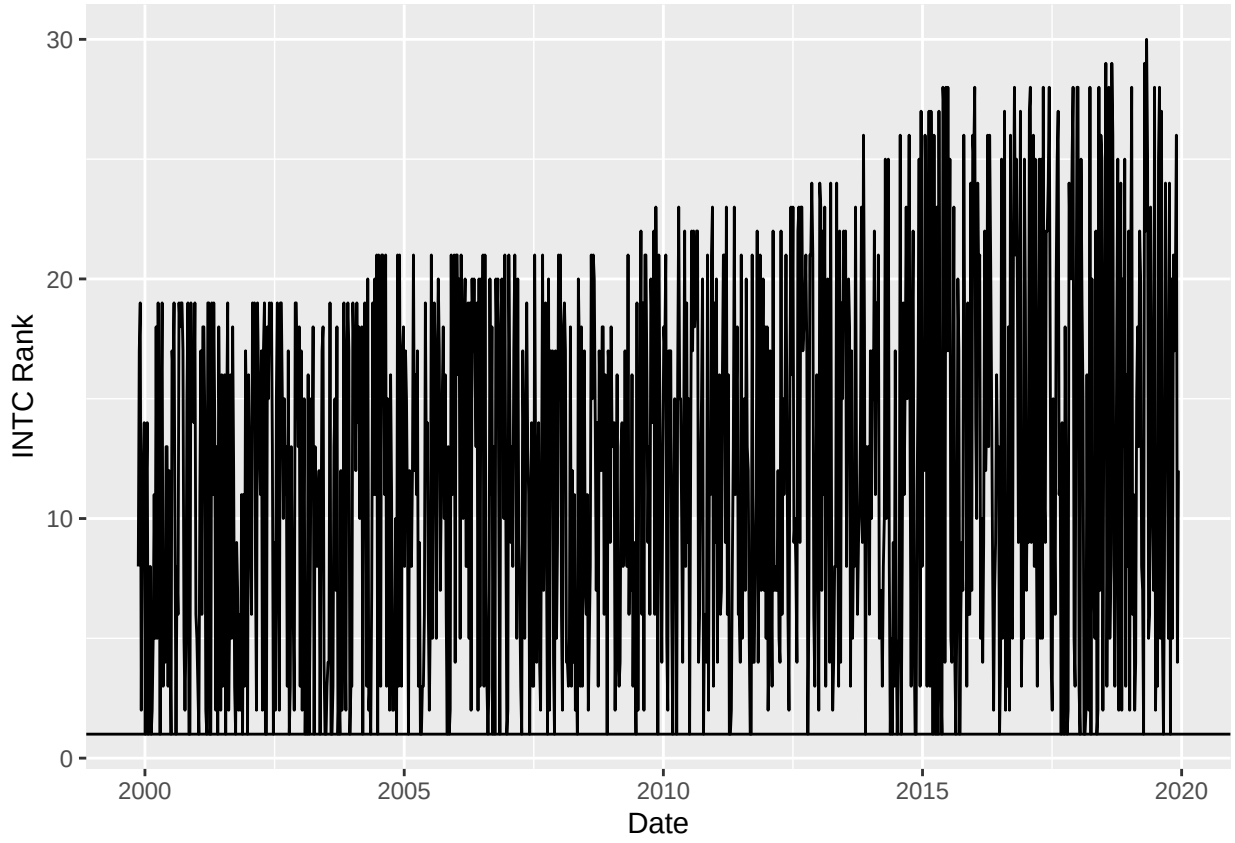

```
dedata %>% filter(Ticker == "HD") %>% ggplot(aes(Date, Rank)) + geom_line() + ylab("HD Rank") + geom_ab
```



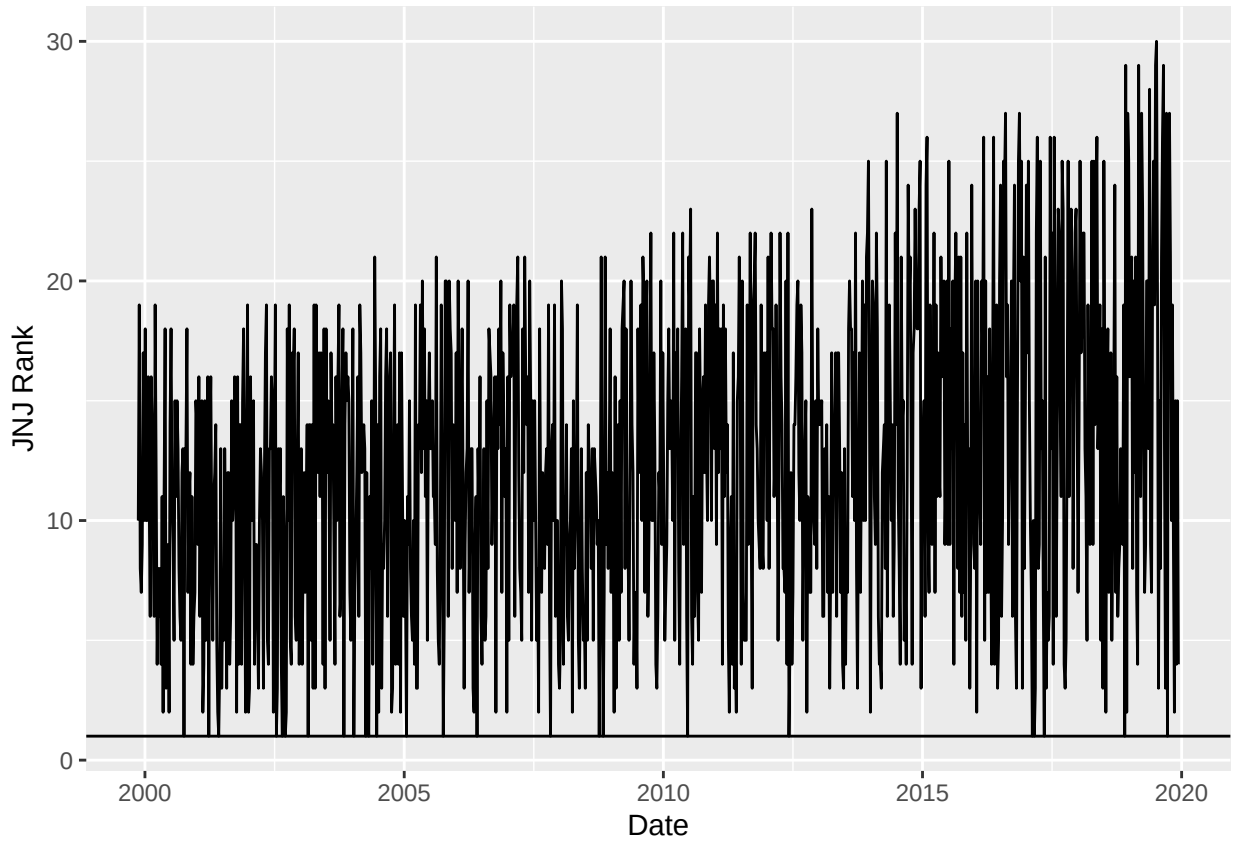

```
dedata %>% filter(Ticker == "IBM") %>% ggplot(aes(Date, Rank)) + geom_line() + ylab("IBM Rank") + geom_
```



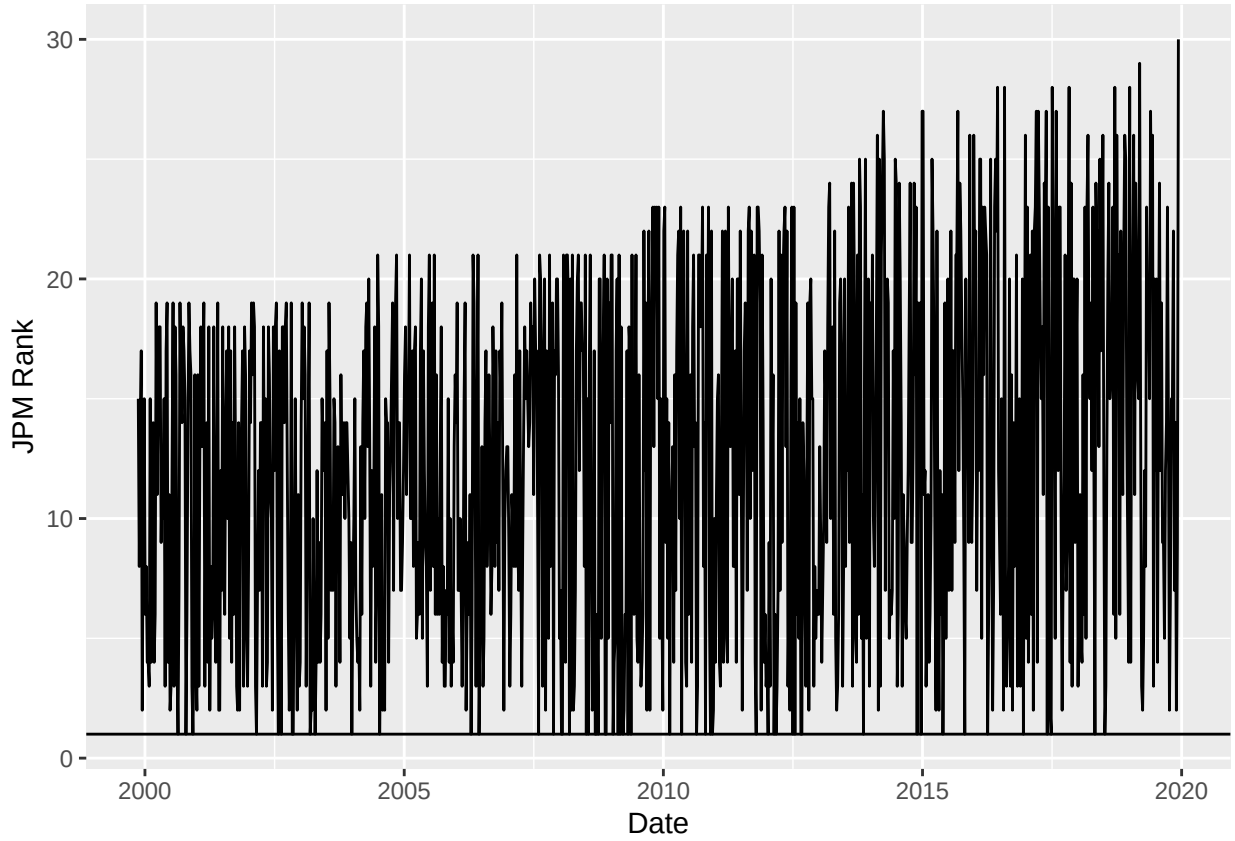

```
dedata %>% filter(Ticker == "INTC") %>% ggplot(aes(Date, Rank)) + geom_line() + ylab("INTC Rank") + geo
```



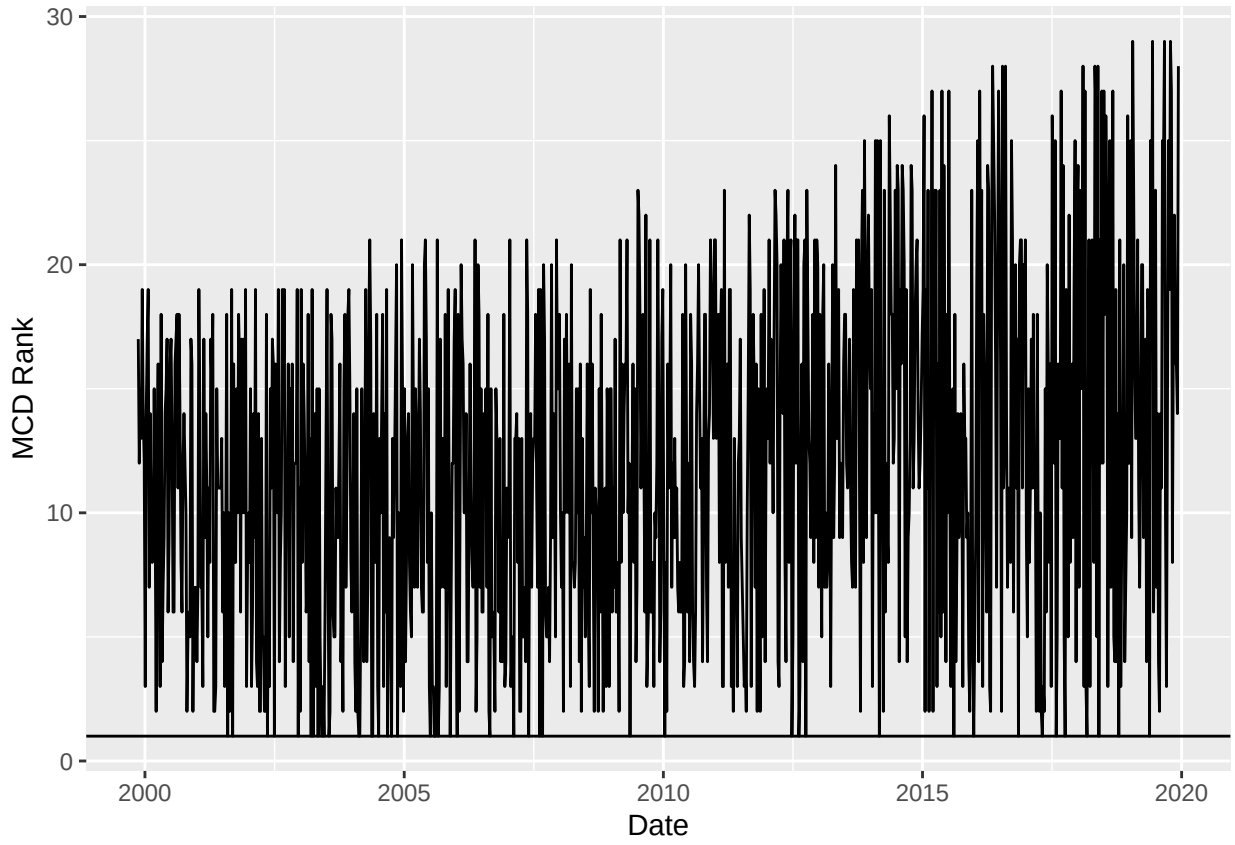

```
dedata %>% filter(Ticker == "JNJ") %>% ggplot(aes(Date, Rank)) + geom_line() + ylab("JNJ Rank") + geom_
```



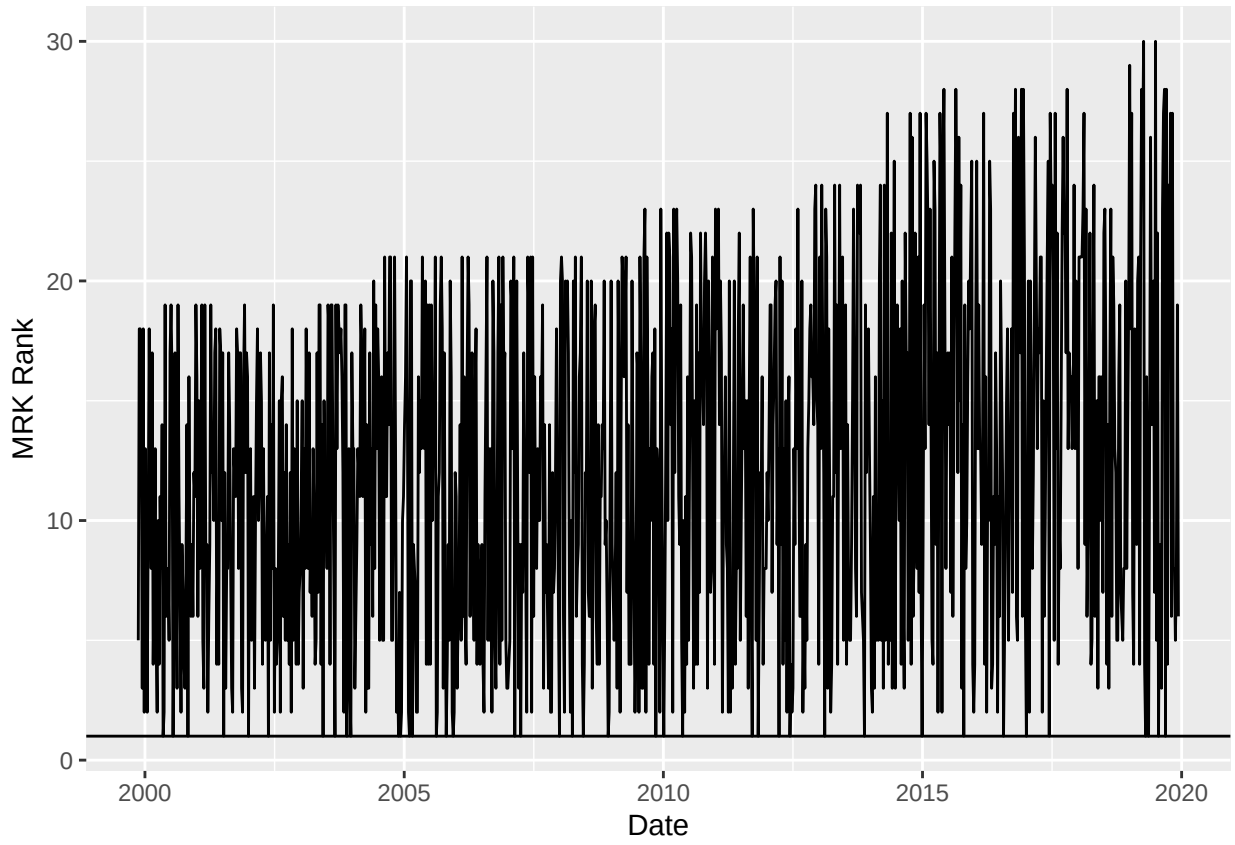

```
dedata %>% filter(Ticker == "JPM") %>% ggplot(aes(Date, Rank)) + geom_line() + ylab("JPM Rank") + geom_
```



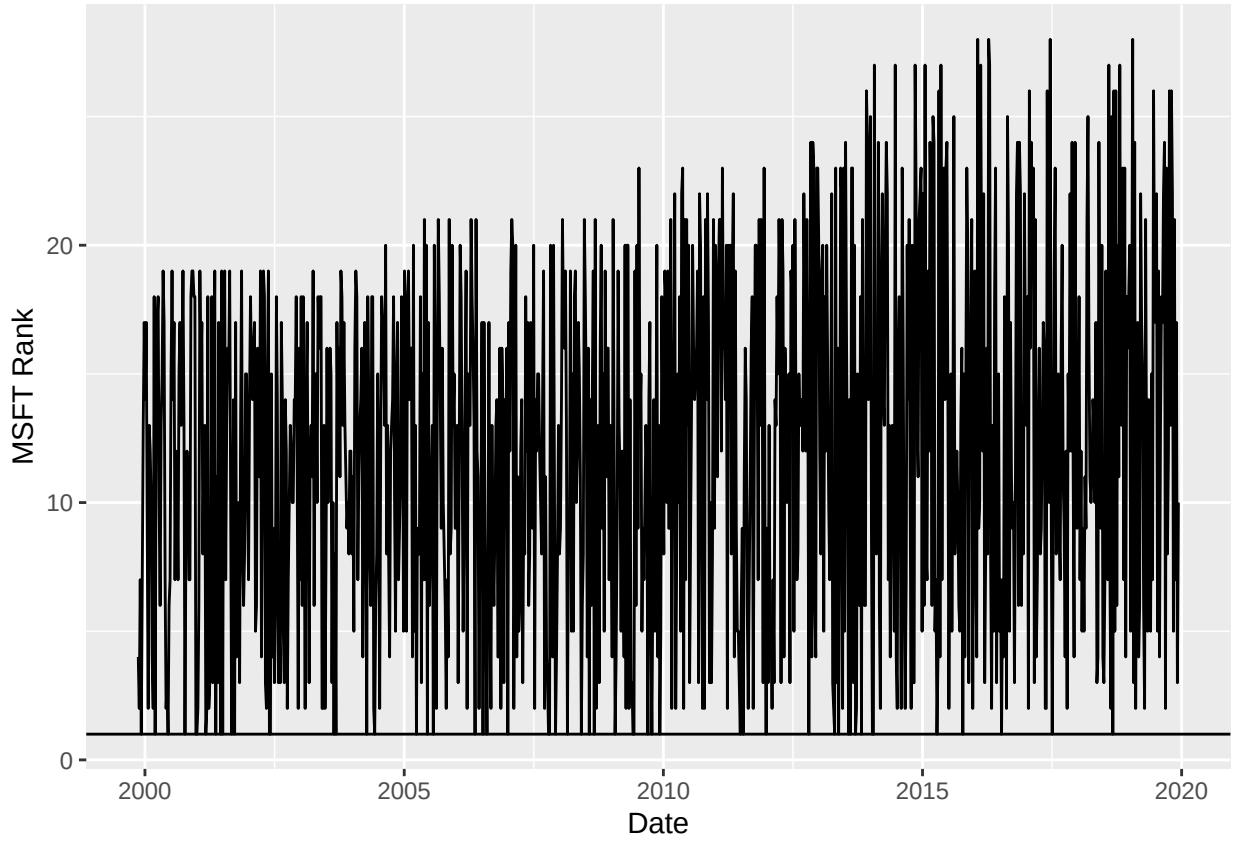

```
dedata %>% filter(Ticker == "MCD") %>% ggplot(aes(Date, Rank)) + geom_line() + ylab("MCD Rank") + geom_
```



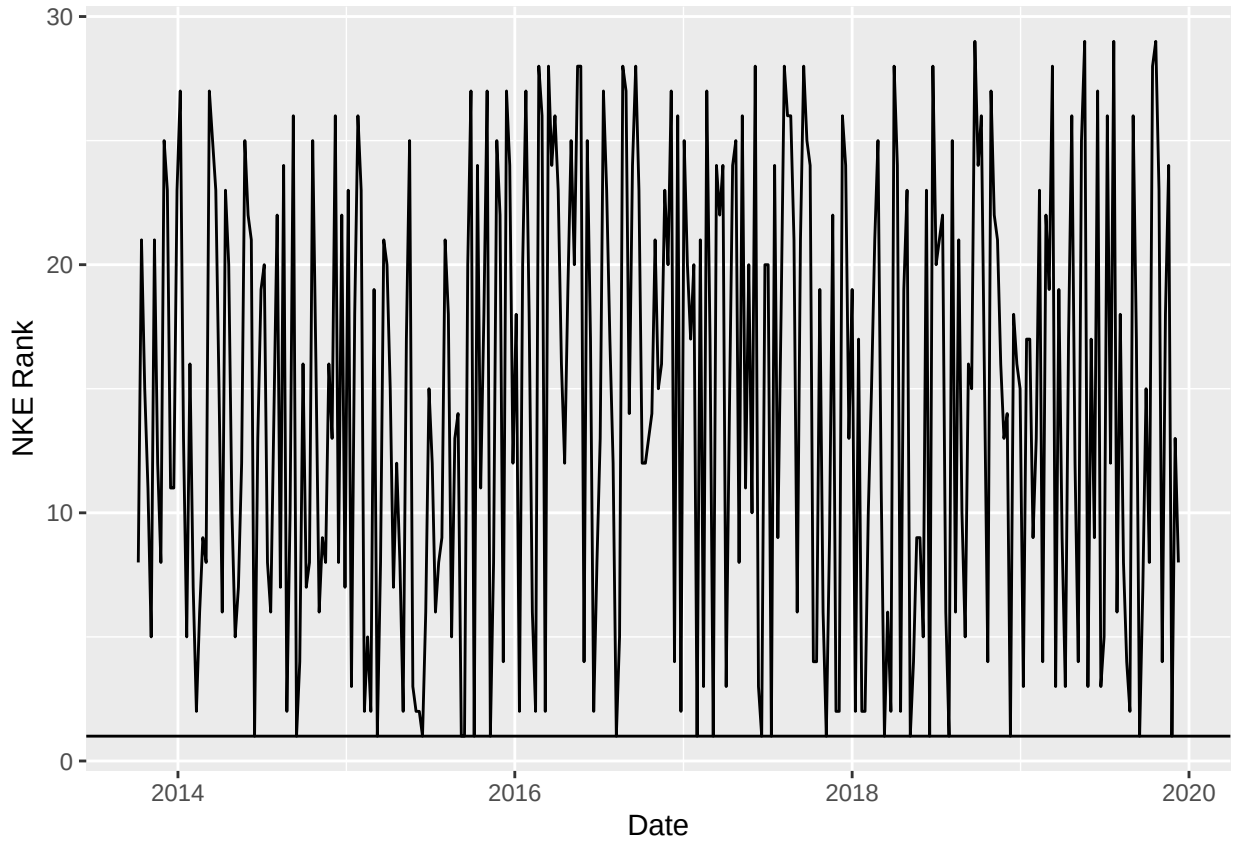

```
dedata %>% filter(Ticker == "MRK") %>% ggplot(aes(Date, Rank)) + geom_line() + ylab("MRK Rank") + geom_
```



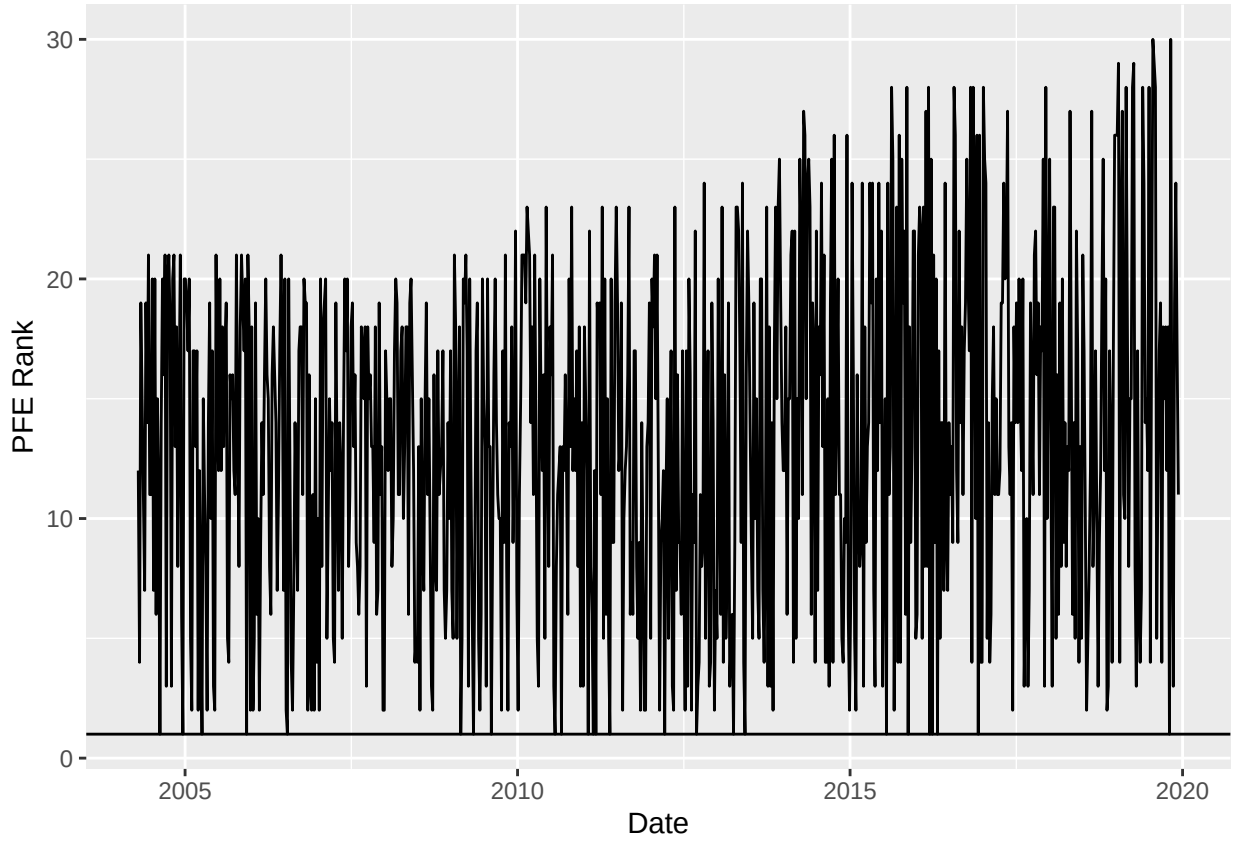

```
dedata %>% filter(Ticker == "MSFT") %>% ggplot(aes(Date, Rank)) + geom_line() + ylab("MSFT Rank") + geo
```



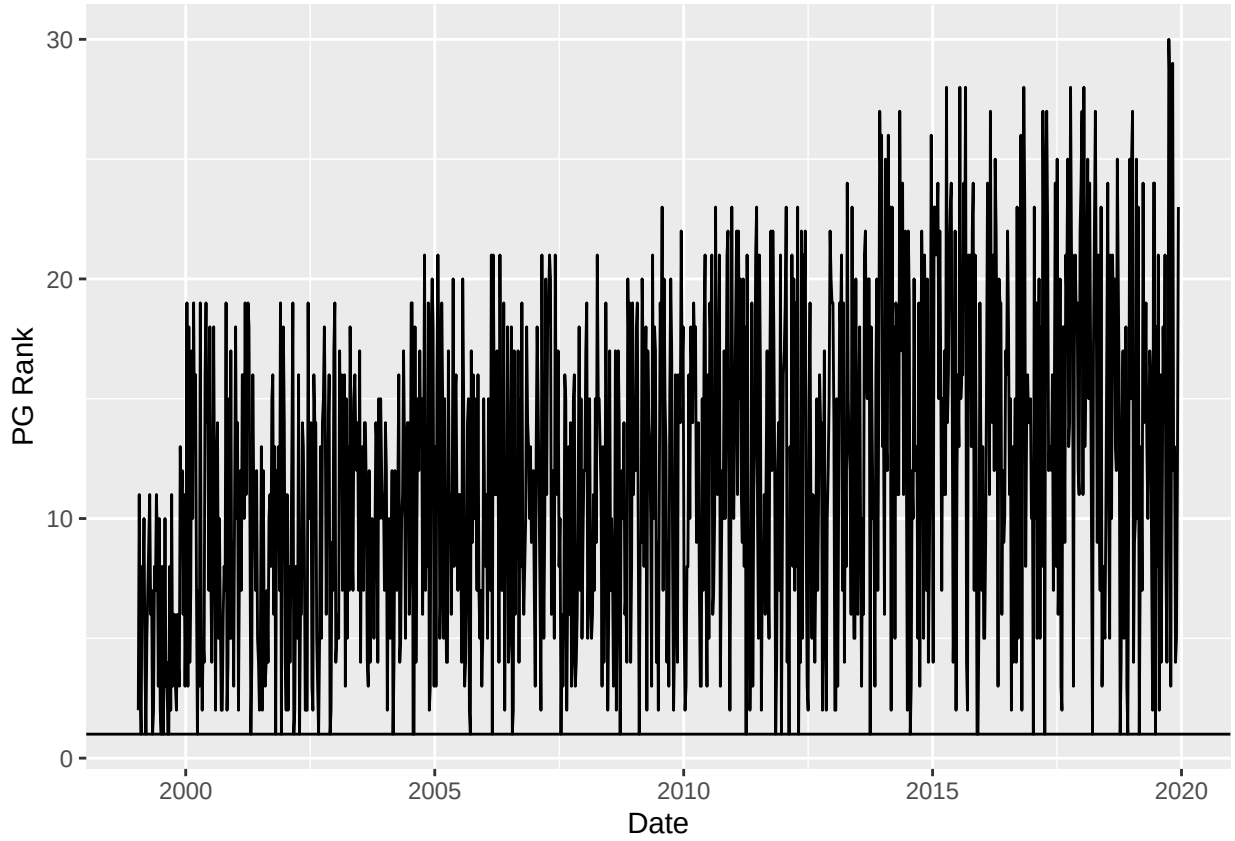

```
dedata %>% filter(Ticker == "NKE") %>% ggplot(aes(Date, Rank)) + geom_line() + ylab("NKE Rank") + geom_
```



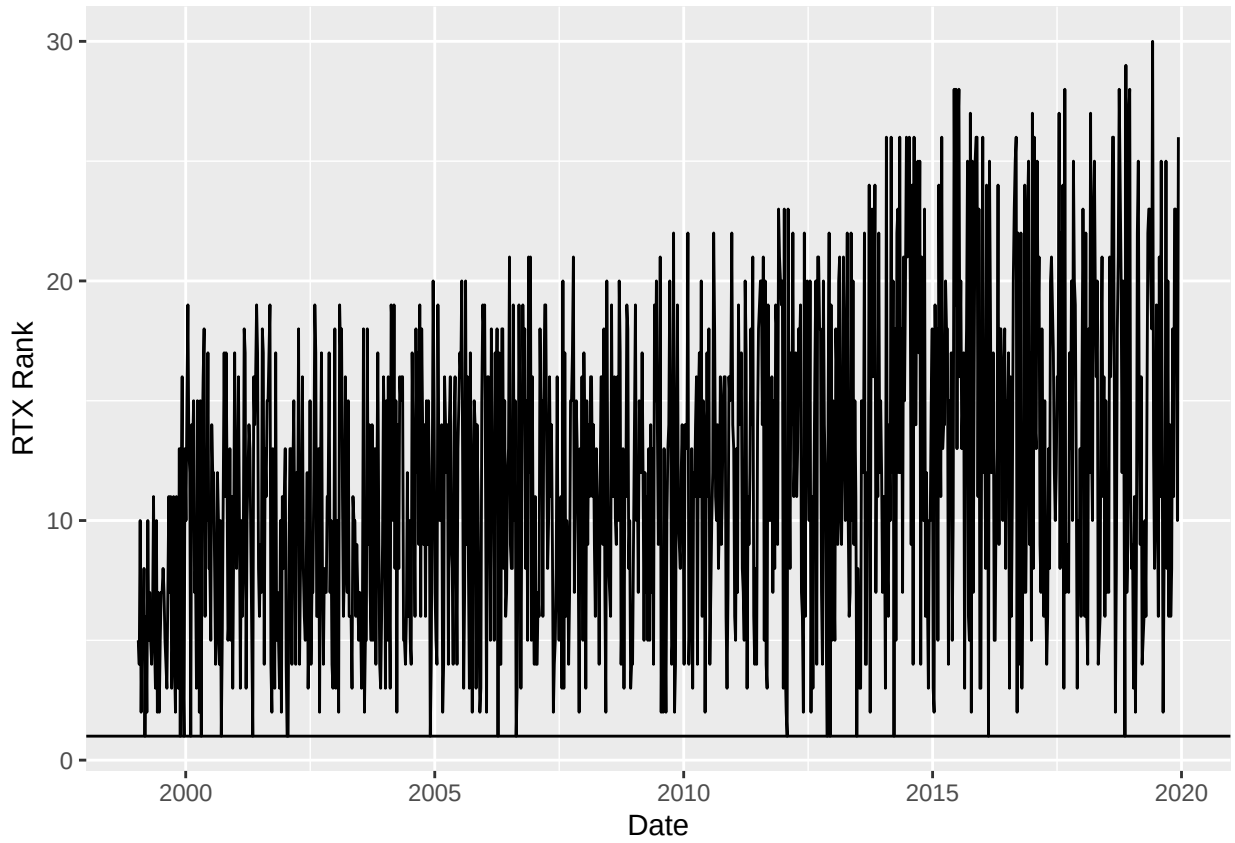

```
dedata %>% filter(Ticker == "PFE") %>% ggplot(aes(Date, Rank)) + geom_line() + ylab("PFE Rank") + geom_
```



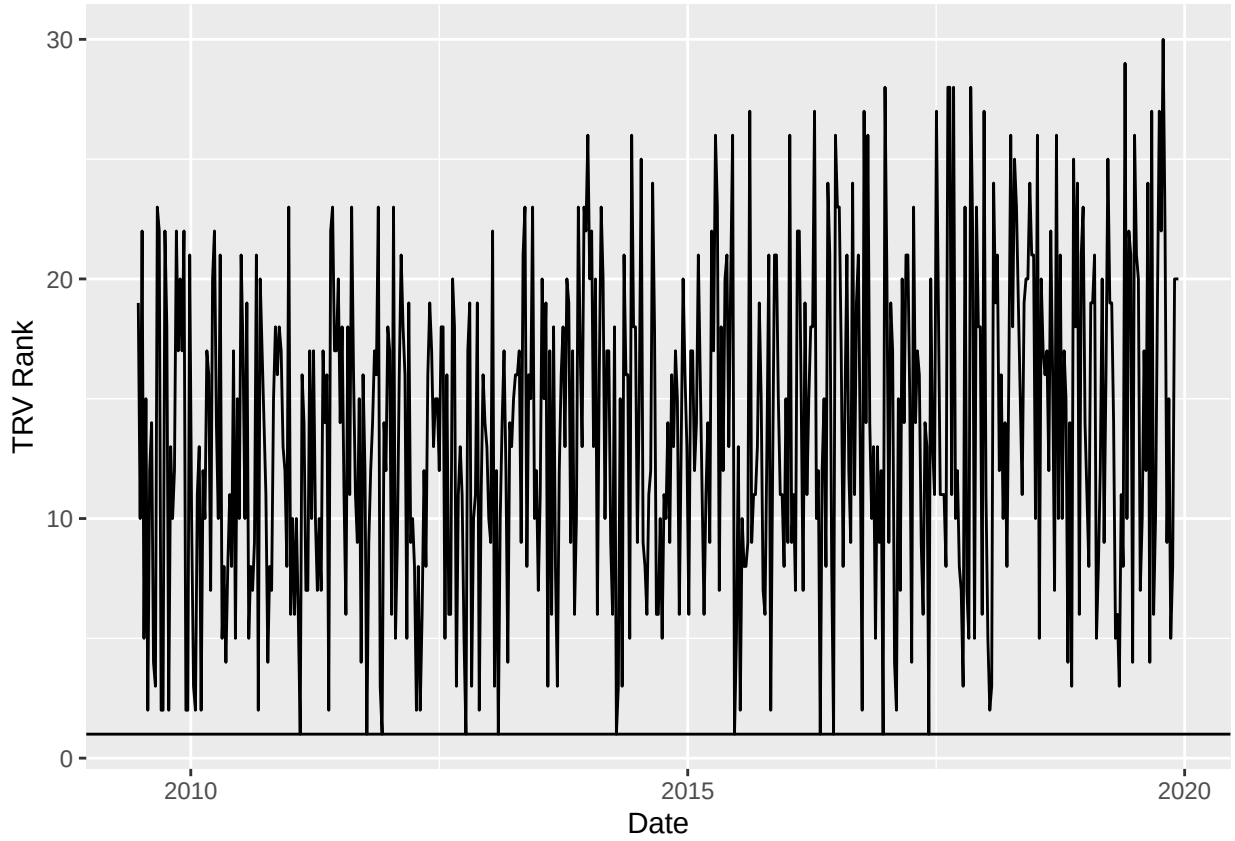

```
dedata %>% filter(Ticker == "PG") %>% ggplot(aes(Date, Rank)) + geom_line() + ylab("PG Rank") + geom_ab
```



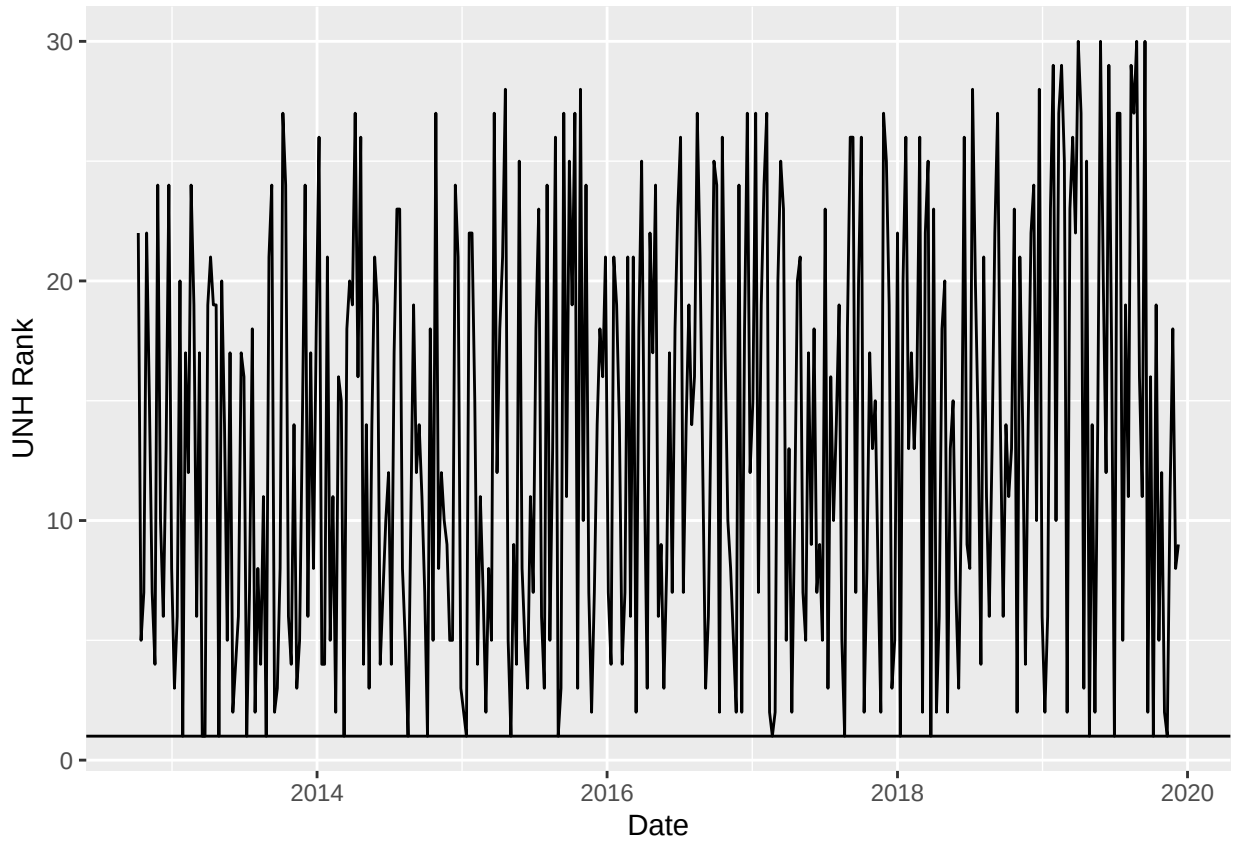

```
dedata %>% filter(Ticker == "RTX") %>% ggplot(aes(Date, Rank)) + geom_line() + ylab("RTX Rank") + geom_
```



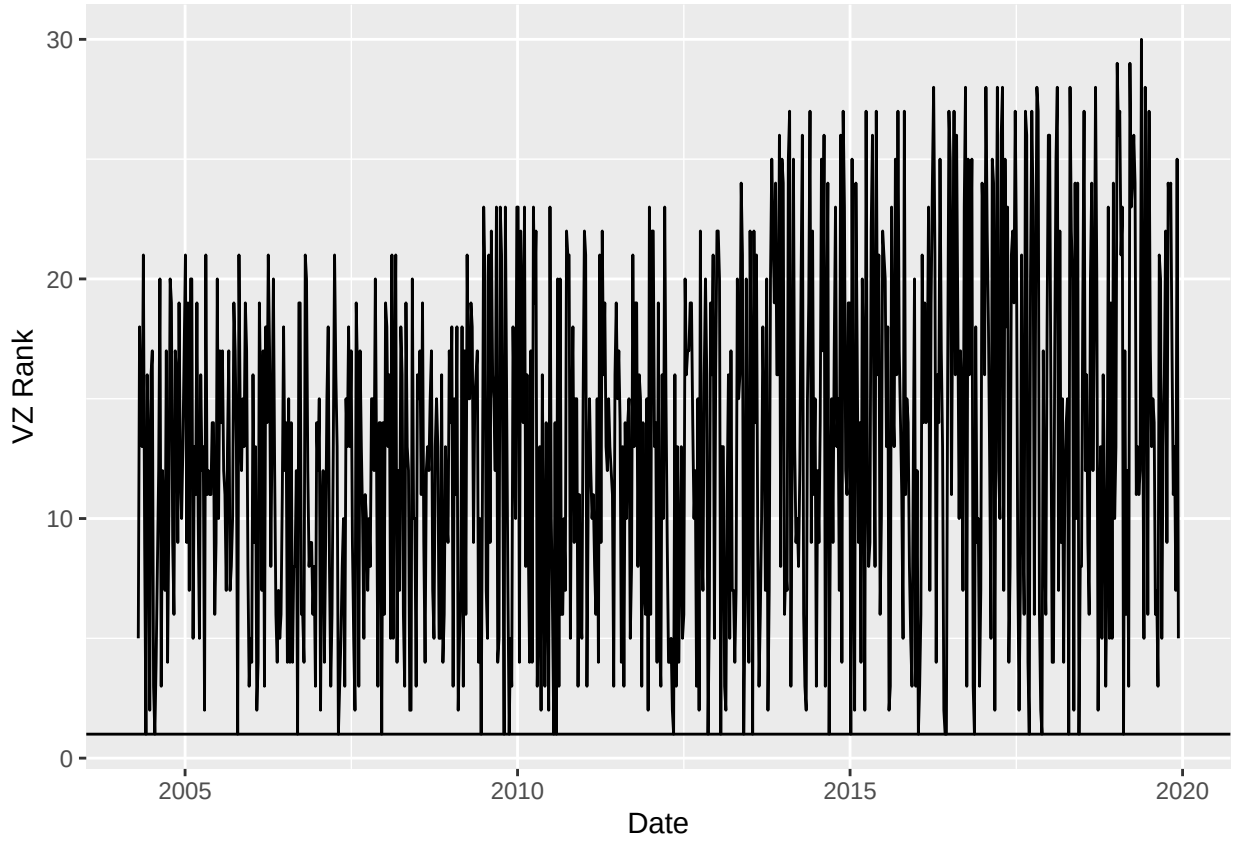

```
dedata %>% filter(Ticker == "TRV") %>% ggplot(aes(Date, Rank)) + geom_line() + ylab("TRV Rank") + geom_
```



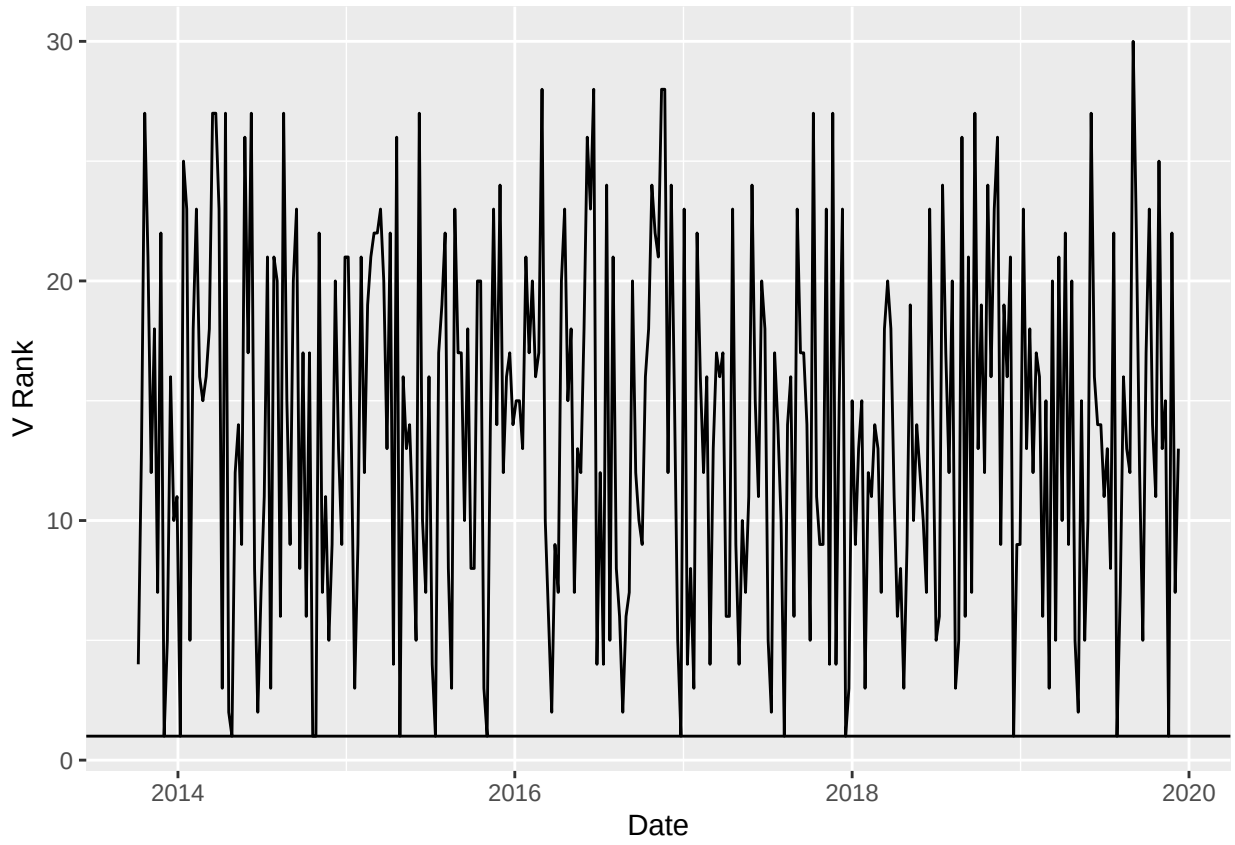

```
dedata %>% filter(Ticker == "UNH") %>% ggplot(aes(Date, Rank)) + geom_line() + ylab("UNH Rank") + geom_
```



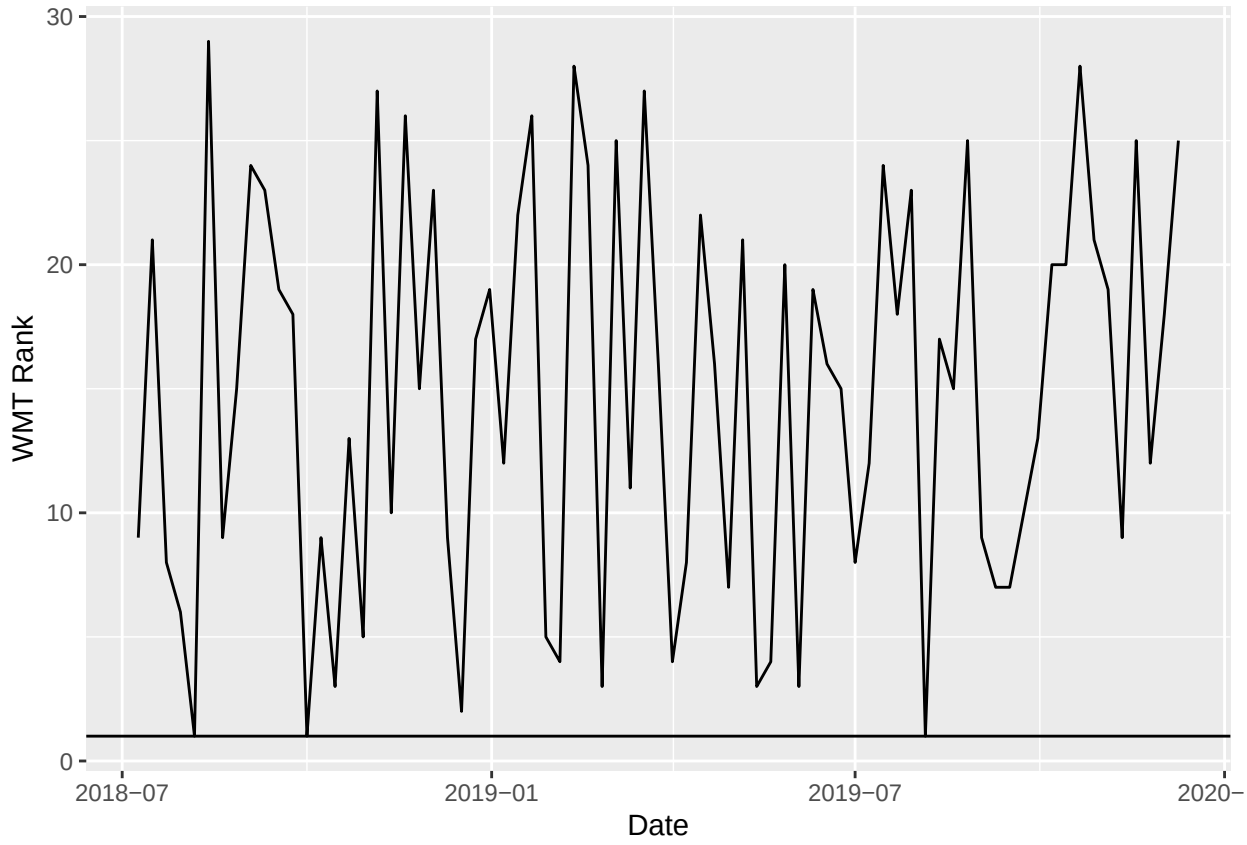

```
dedata %>% filter(Ticker == "VZ") %>% ggplot(aes(Date, Rank)) + geom_line() + ylab("VZ Rank") + geom_ab
```



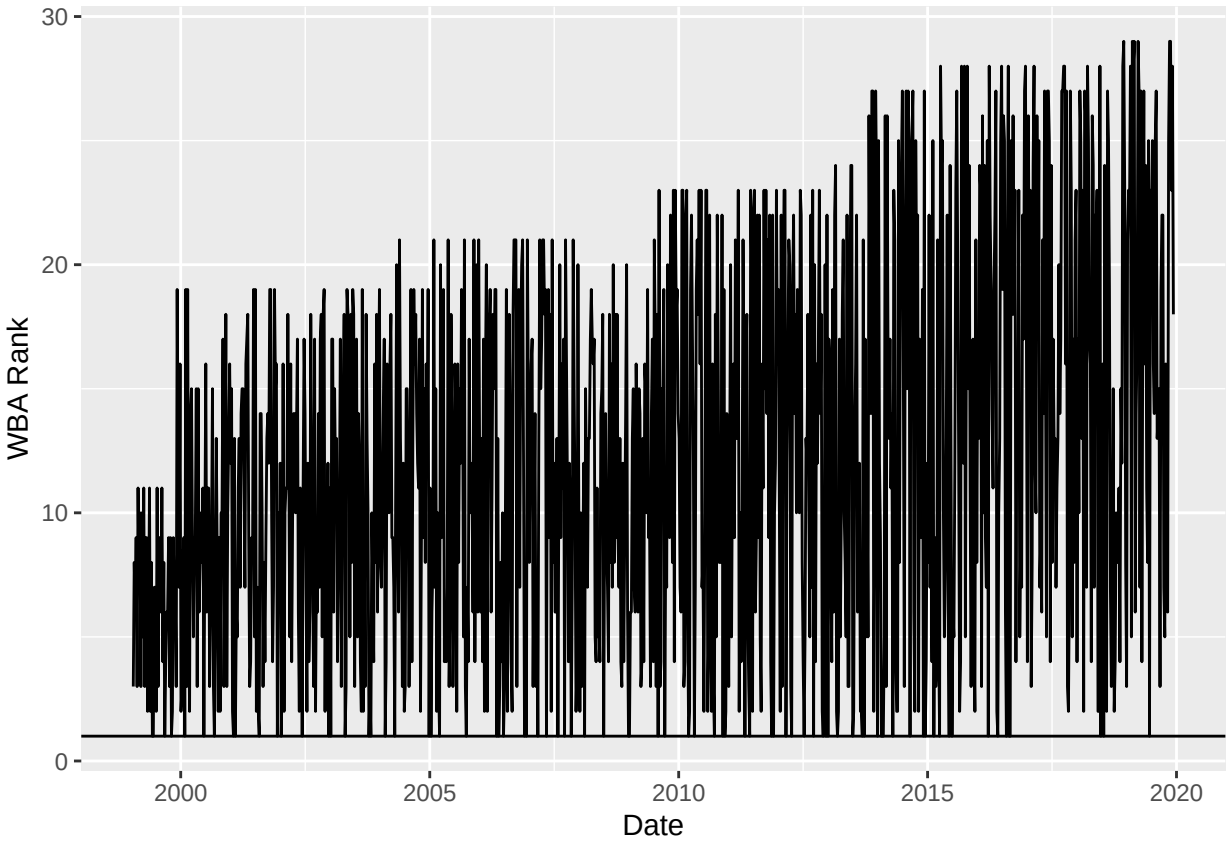

```
dedata %>% filter(Ticker == "V") %>% ggplot(aes(Date, Rank)) + geom_line() + ylab("V Rank") + geom_abli
```



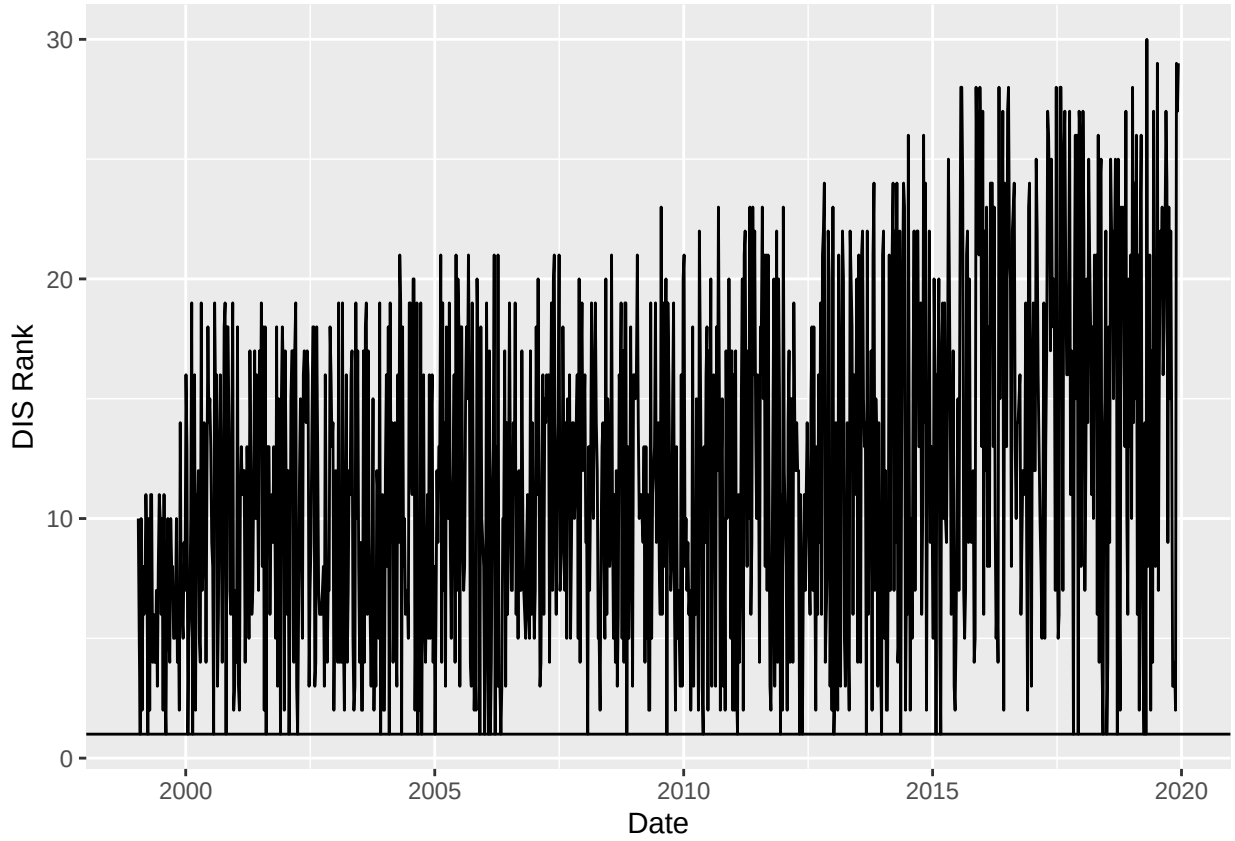

```
dedata %>% filter(Ticker == "WMT") %>% ggplot(aes(Date, Rank)) + geom_line() + ylab("WMT Rank") + geom_
```




```
dedata %>% filter(Ticker == "WBA") %>% ggplot(aes(Date, Rank)) + geom_line() + ylab("WBA Rank") + geom_
```




```
dedata %>% filter(Ticker == "DIS") %>% ggplot(aes(Date, Rank)) + geom_line() + ylab("DIS Rank") + geom_
```



In the above charts, the objective was to find persistence in the ranks of a stock based on the common belief that a strong stock remains strong. But none of the stock held the rank 1 for any series of weeks. The same is true, that the worst week stocks also doesn't have any series. So, it stands the reason that current week rank plays a role in next week's performance.

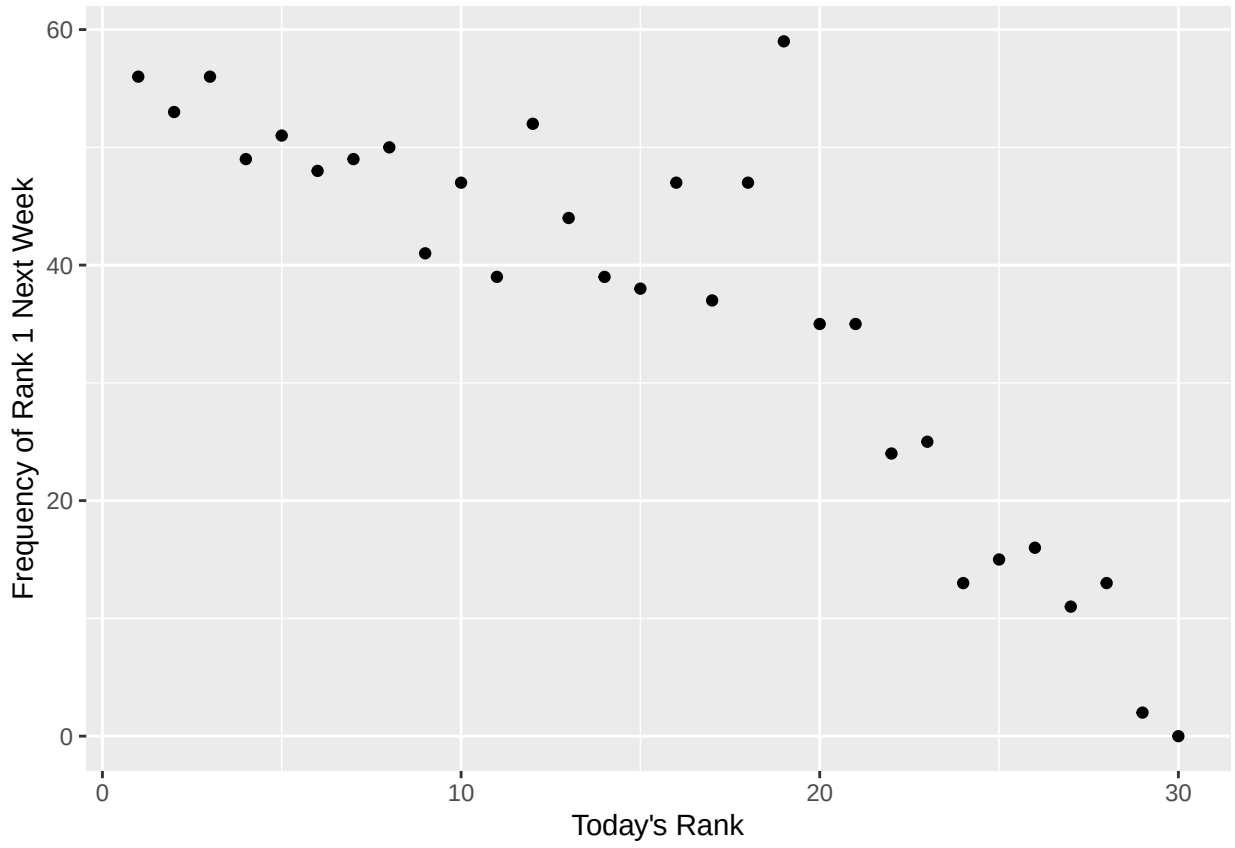
Rank (Next Day Rank) vs Change Rank(Today's Rank) matrix

```
t_rank_change_rank
```

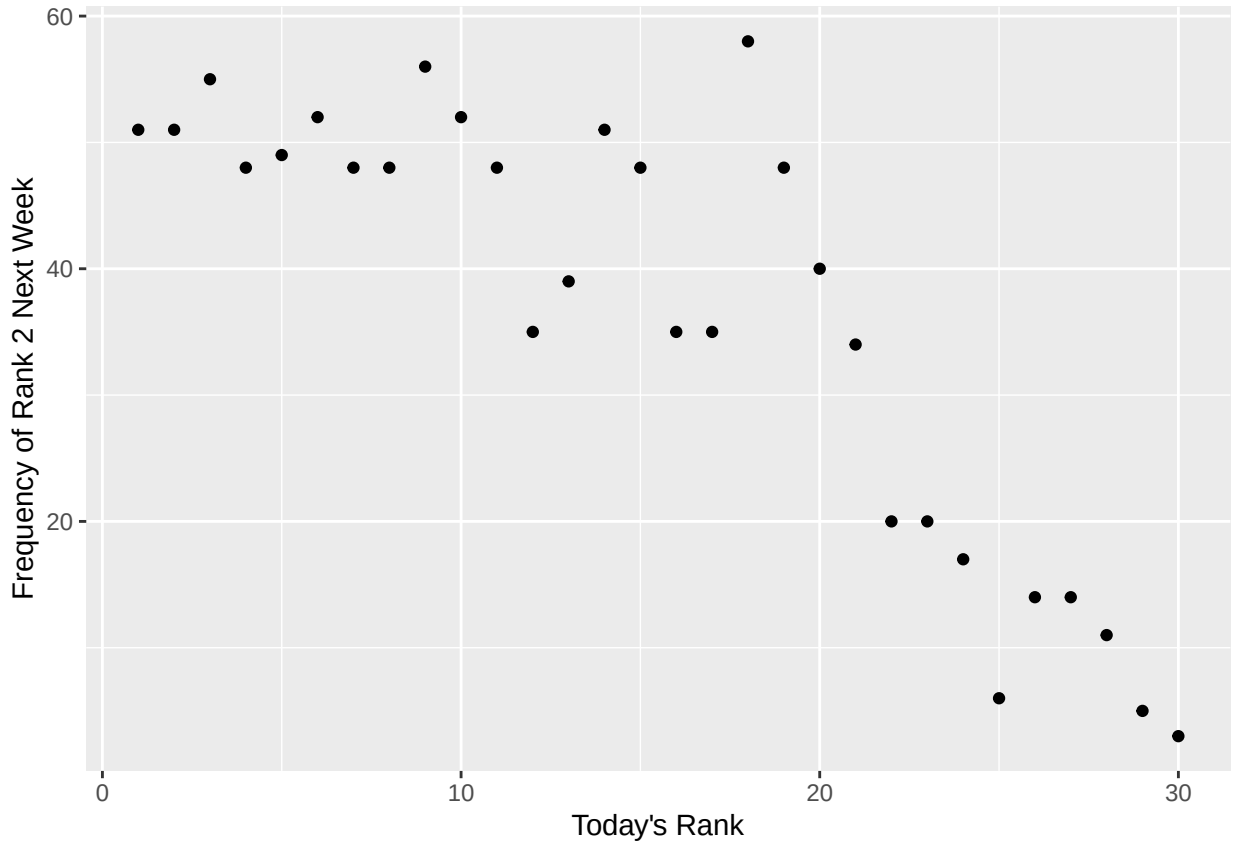
```
##            Rank
## ChangeRank  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
##          1  56 51 60 43 40 58 45 46 46 42 49 31 52 49 24 41 43 48 46 51 47 19 29
##          2  53 51 39 52 45 51 44 54 49 53 59 45 43 32 47 52 44 40 44 39 39 22 28
##          3  56 55 40 62 41 44 48 57 46 50 50 41 48 50 42 45 52 56 45 35 33 17 22
##          4  49 48 47 49 48 48 51 68 52 50 52 53 38 44 40 43 33 43 51 35 41 22 15
##          5  51 49 44 49 44 53 55 53 59 50 51 38 45 46 60 44 42 44 44 33 34 28 15
##          6  48 52 40 41 64 46 45 56 52 58 49 59 51 37 64 47 37 39 40 35 31 17 24
##          7  49 48 54 41 51 60 55 34 38 51 54 45 34 41 55 54 56 54 49 35 37 23 19
##          8  50 48 42 54 50 52 54 37 58 60 39 54 42 46 57 44 47 54 46 23 37 19 13
##          9  41 56 53 46 48 46 61 57 47 52 42 49 53 60 45 48 40 40 41 36 40 25 18
##          10 47 52 49 49 45 52 47 55 35 36 45 50 50 61 53 40 62 43 47 21 27 19 29
##          11 39 48 56 45 48 56 46 48 63 48 57 39 59 34 46 62 61 43 49 37 29 10 16
##          12 52 35 39 39 47 43 47 39 45 66 56 33 50 63 38 40 48 54 45 39 28 23 17
##          13 44 39 50 46 53 28 43 42 49 50 45 49 51 56 44 48 42 47 52 41 34 22 23
##          14 39 51 52 58 50 40 61 45 32 52 48 39 48 46 47 39 45 50 43 32 42 19 18
##          15 38 48 50 53 50 40 62 44 43 49 57 46 56 33 46 49 34 52 48 24 28 24 13
##          16 47 35 42 46 48 53 42 43 55 51 48 46 56 43 50 38 49 44 42 40 29 16 30
```

```
##          17 37 35 46 52 51 51 48 54 46 51 49 42 45 41 45 60 44 37 54 34 36 22 17
##          18 47 58 57 38 32 48 29 47 50 47 56 50 44 53 39 49 44 49 50 32 25 13 25
##          19 59 48 37 44 56 45 40 36 57 36 51 49 46 47 44 35 46 50 45 33 33 22 20
##          20 35 40 36 30 39 34 30 35 39 24 26 47 26 45 30 36 40 37 29 36 33 16 19
##          21 35 34 36 28 33 36 32 41 37 28 31 45 24 30 37 30 26 36 36 33 43 34 21
##          22 24 20 20 24 27 21 31 14 22 14 13 18 17 22 16 30 26 18 21 20 25 20 25
##          23 25 20 31 29 24 27 17 24 21 18  8 23 17 22 21 16 25 18 17 22 14 26 26
##          24 13 17 16 14 11 14 15 11 12 10 14 16 10 11 19 14 18  6 12 14 13 16 16
##          25 15  6 13 15 14  9  7 13 14  9  9 14 12  8  7 11 11 14 21 10  6 12 14
##          26 16 14 12 15 12 14 10 14 11 10 10 13 11  7 12 11 13 16 11  6 11 10 11
##          27 11 14 15 14  8  8 11 18  4  9 15  7 11 10  5 17 11  5 11  9 15 16 12
##          28 13 11 11 13 10  7 10  2  9  8  7  7  4  9  7  3  8  7  7  9  7 12  6
##          29  2  5  3  2  0  4  5  1  1  5  1  1  1  4  3  1  1  4  0  1  0  1  5
##          30  0  3  1  1  2  2  0  3  1  2  0  1  2  0  3  1  0  0  2  2  0  2  1
##            Rank
## ChangeRank 24 25 26 27 28 29 30
##          1  13 16 12 12 14  6  2
##          2  13 12 10 13 13  4  1
##          3   9  8 14 12  9  4  0
##          4  16 12 17 11 11  3  2
##          5  10 10 13 16  8  2  1
##          6  19 10 10 11  7  0  1
##          7  17  8 10 11  5  2  1
##          8  17 20  6 13  6  2  1
##          9  14  7 10 10  3  3  1
##          10 13 19 16 17 10  1  0
##          11 13  9  8 11  7  2  2
##          12 14 17 11 12  5  3  2
##          13 11 10  8 12  4  2  1
##          14 14 13  6 11  9  1  1
##          15 19 11  9  7  7  3  2
##          16 13  8 13 11  6  3  1
##          17  8 13  9  8 12  1  0
##          18 13  7 21  9 12  2  2
##          19 12 16  9 16  8  7  1
##          20 14  9 13  9  8  2  0
##          21 11 15  6  4 12  1  2
##          22 16  8 15 10  7  0  3
##          23 16 11 10  7  7  3  2
##          24 10 10 15 16 10  2  0
##          25 14  7  7 16 11  2  2
##          26 12  7 16  9  4  3  2
##          27  8 14 14 13 14  2  2
##          28 12 11  9  9 10  7  0
##          29  4  4  6  4  4  1  1
##          30  0  1  0  3  2  1  1
```

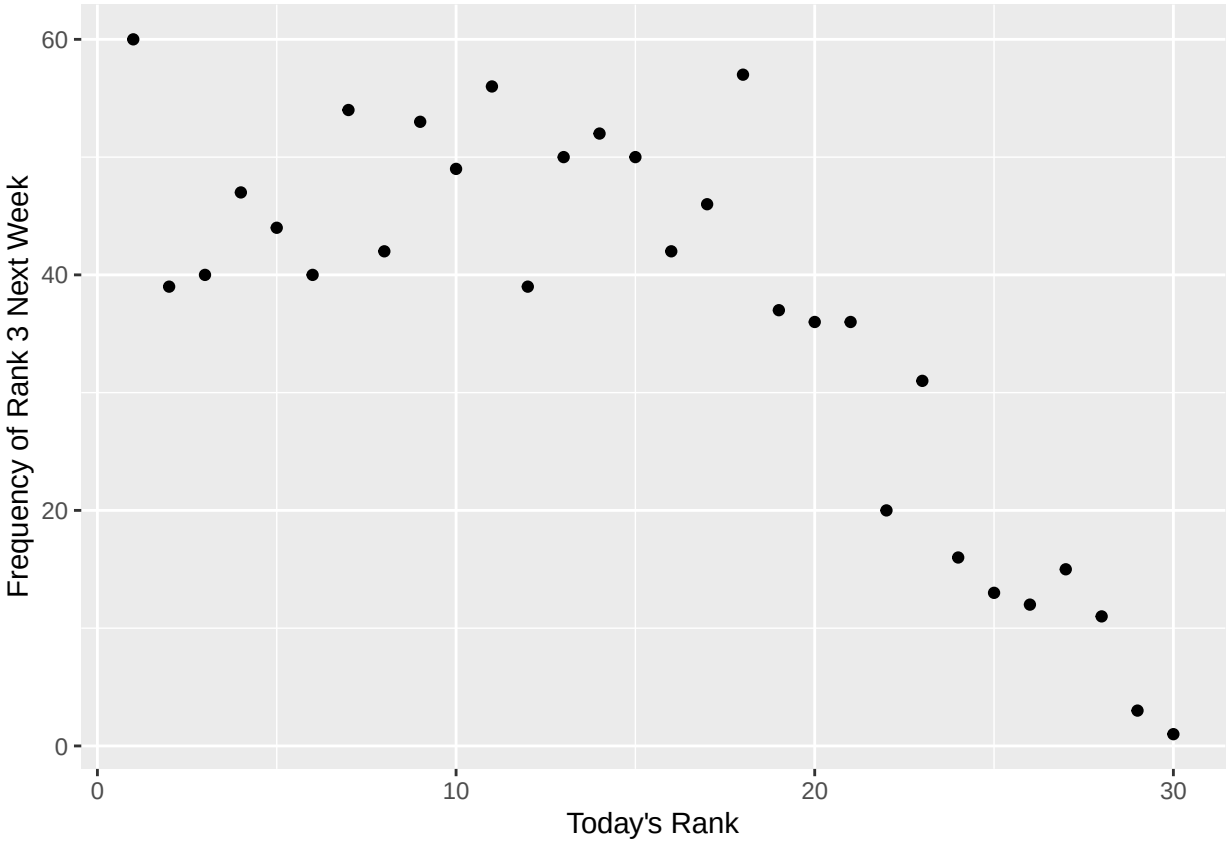
```r
# Lets check the distribution of Rank 1 vs ChangeRank
as.data.frame(t_rank_change_rank[,1]) %>%
        ggplot(aes(1:30, t_rank_change_rank[,1])) +
        geom_point() +
        ylab("Frequency of Rank 1 Next Week") +
        xlab("Today's Rank")
```




```
as.data.frame(t_rank_change_rank[,2]) %>%
        ggplot(aes(1:30, t_rank_change_rank[,2])) +
        geom_point() +
        ylab("Frequency of Rank 2 Next Week") +
        xlab("Today's Rank")
```




```
as.data.frame(t_rank_change_rank[,3]) %>%
        ggplot(aes(1:30, t_rank_change_rank[,3])) +
        geom_point() +
        ylab("Frequency of Rank 3 Next Week") +
        xlab("Today's Rank")
```



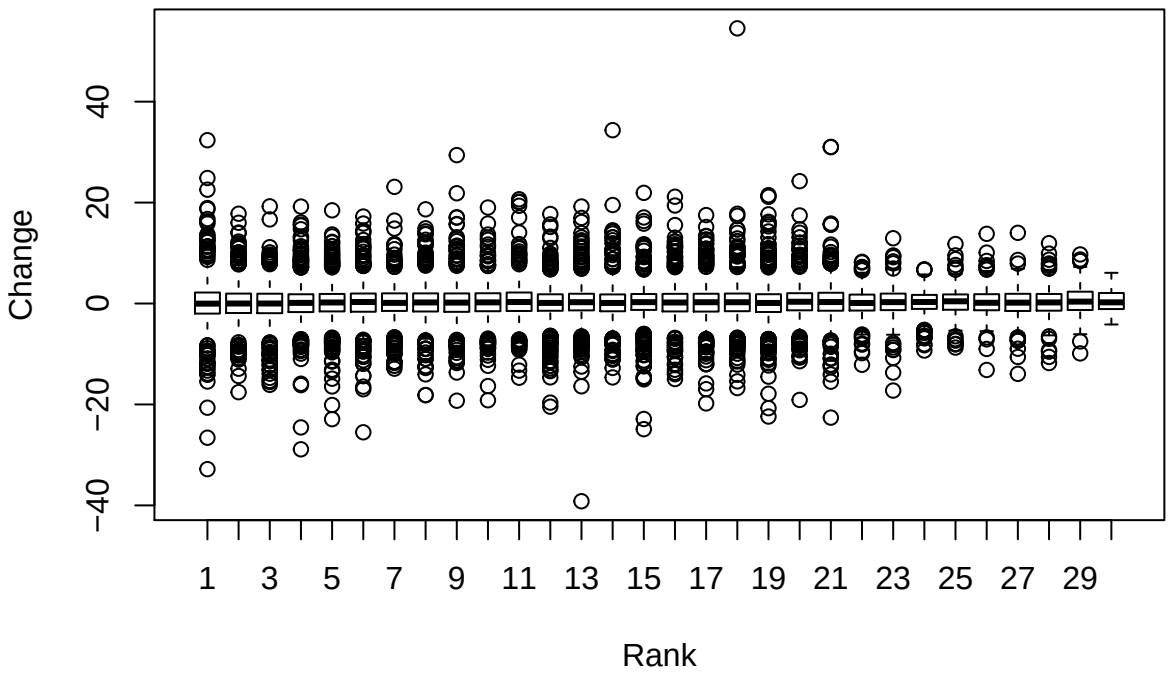
The 3 graphs clearly show that the top rank frequency comes from previous day rank being between 1-19 and the charts are negatively co-related showing that most top rank comes from higher previous day ranks.

2.5.1.1 Conclusion: Today's Rank is co-related to Next Day's Rank.

2.6 Feature: Percentage Day Change

Lets visualize how today's percentage day change effect Outcome

```
boxplot(Change~Rank,data=dedata)
```



All the boxes almost overlapped.

2.6.0.1 Conclusion: This week's change is not co-related to Outcome

2.7 Feature: Rate of Close

Lets visualize how 1 ot more week's rate of change(Close to Close change) effects Outcome.

```
cor(dedata$Outcome, dedata$ROC1)
```

```
## [1] -0.05665262
```

```
cor(dedata$Outcome, dedata$ROC2)
```

```
## [1] -0.03027169
```

```
cor(dedata$Rank, dedata$ROC1)
```

```
## [1] 0.02439916
```

```
cor(dedata$Rank, dedata$ROC2)
```

```
## [1] 0.02621931
```

2.7.0.1 Conclusion: No edge found in the distribution using this feature

3 Model Building

We will build the best models using Random Set and Timed Set.

3.0.1 Random Set

We will divide the whole dataset in a 60% train and 40% test set randomly.

We will be building a linear model, a random forest model, and an xgboost model.

We will learn from linear models which feature set does better and build specific random forest model and an xgboost model as these models take time to build.

3.0.2 Timed Set

Building a model with a randomly chosen train set does not allow an understanding of the model based on predicted outcome rank. This does not allow determining the result of buying the best-predicted outcome stock. Next, we will divide the dataset from 1999-2011 as a training set and the rest as a test set. We will call this as timed train and test set We will make new models as the random Set models will very likely contain some of the test set values. These models will answer the questions

1. If predicted rank 1 was traded what would be the profitability graph.

4 Results

The results are better than random

4.1 Linear Regression Models

4.1.1 Random Test Set

Id	Type	Rank_Day0	Rank_Day1	Rank_Day2	ROC_1Day	ROC_2Days	Cor
1	Linear Regression	Y					0.601316
2_1	Linear Regression	Y			Y		0.6020807
2_2	Linear Regression	Y				Y	0.6013089
2_3	Linear Regression	Y	Y				0.6040158
2_4	Linear Regression	Y		Y			0.6036787
3_1	Linear Regression	Y			Y	Y	0.602312
3_2	Linear Regression	Y			Y	Y	0.6048272
3_3	Linear Regression	Y			Y	Y	0.6044473
3_4	Linear Regression	Y			Y	Y	0.6040292
3_5	Linear Regression	Y			Y	Y	0.6036725
3_6	Linear Regression	Y			Y	Y	0.6061185
5_1	Linear Regression	Y	Y	Y	Y	Y	0.6071408

4.1.1.1 Best Linear Model

The model with all the 5 features has a correlation of 0.607 with the actual outcome. It is a weak co-relation but statistically significant enough.

We found from linear models that adding all the features gave the best model so we will build Random Forest and XGBoost will all feature. A single feature model will also be built in each case so that we can verify that our expectation is inline.

4.1.2 Timed Test Set

Id	Type	Rank_Day0	Rank_Day1	Rank_Day2	ROC_1Day	ROC_2Days	Cor
5_1	Linear Regression	Y	Y	Y	Y	Y	0.6902775

4.1.2.1 Total Profitability per Rank

Rank	Profit
1	125.
2	200.
3	136.
4	210.
5	211.
6	145.
7	182.
8	82.4
9	43.5
10	71.0
11	104.
12	61.4
13	256.
14	148.
15	164.
16	54.5
17	50.2
18	86.7
19	22.8
20	106.
21	63.8
22	19.1
23	35.6
24	63.0
25	59.8
26	96.9
27	-0.188
28	0.473
29	16.4
30	-14.6

4.1.2.2 Rank 1 profitability per Year

Id	Year	Profit
1	2012	31.1
2	2013	14.8
3	2014	18.6
4	2015	-49.2
5	2016	20.0
6	2017	30.0
7	2018	14.8
8	2019	44.5
	Total	124.6

Year	Profit
2012	31.1
2013	14.8
2014	18.6
2015	-49.2
2016	20.0
2017	30.0
2018	14.8
2019	44.5
Total	124.6

4.2 Random Forest Models

4.2.1 Random Test Set

Id	Type	Rank_Day0	Rank_Day1	Rank_Day2	ROC_1Day	ROC_2Days	Cor
1	Random Forest	Y					0.6337799
5_1	Random Forest	Y	Y	Y	Y	Y	0.62619

4.2.2 Timed Test Set

Id	Type	Rank_Day0	Rank_Day1	Rank_Day2	ROC_1Day	ROC_2Days	Cor
5_1	Random Forest	Y	Y	Y	Y	Y	0.6536283

4.2.2.1 Total Profitability per Rank

Rank	Profit
1	154.
2	202.
3	130.
4	161.
5	159.
6	195.
7	144.
8	65.0

Rank	Profit
9	95.7
10	166.
11	196.
12	35.7
13	176.
14	-0.790
15	106.
16	102.
17	92.1
18	83.1
19	127.
20	10.4
21	-0.665
22	85.4
23	39.7
24	90.5
25	1.14
26	75.5
27	8.83
28	71.3
29	7.46
30	22.0

4.2.2.2 Rank 1 profitability per Year

Year	Profit
2012	18.4
2013	37.3
2014	27.1
2015	6.63
2016	14.1
2017	26.2
2018	15.9
2019	8.57
Total	154.23

4.3 XGBoost Models

4.3.1 Random Test Set

Id	Type	Rank_Day0	Rank_Day1	Rank_Day2	ROC_1Day	ROC_2Days	Cor
1	XGBoost	Y					0.6339214
5_1	XGBoost	Y	Y	Y	Y	Y	0.6512258

4.3.2 Timed Test Set

Id	Type	Rank_Day0	Rank_Day1	Rank_Day2	ROC_1Day	ROC_2Days	Cor
5_1	XGBoost	Y	Y	Y	Y	Y	0.6770446

4.3.2.1 Total Profitability per Rank

Rank	Profit
1	128.
2	193.
3	112.
4	127.
5	163.
6	170.
7	177.
8	114.
9	94.1
10	122.
11	187.
12	53.3
13	190.
14	83.7
15	61.4
16	82.9
17	104.
18	47.5
19	-6.53
20	81.7
21	77.9
22	119.
23	27.7
24	85.1
25	71.3
26	44.8
27	23.8
28	19.0
29	59.6
30	-14.3

4.3.2.2 Rank 1 profitability per Year

Year	Profit
2012	9.93
2013	24.5
2014	5.67
2015	-7.08
2016	30.6
2017	9.53
2018	26.2
2019	29.1
Total	128.39

5 Conclusion

There are many ways to use the best model

1. Rank 1 Trades - Take the stock with the highest predicted long move for the next week.
2. Weighted approach - Take position in all stocks but weight them in proportion to the amount of expected outcome. This also means that you take a short position on a negative expected outcome.

In this paper, the results of taking the Rank 1 trade are evaluated and it is surprising that in most of the years it is profitable.

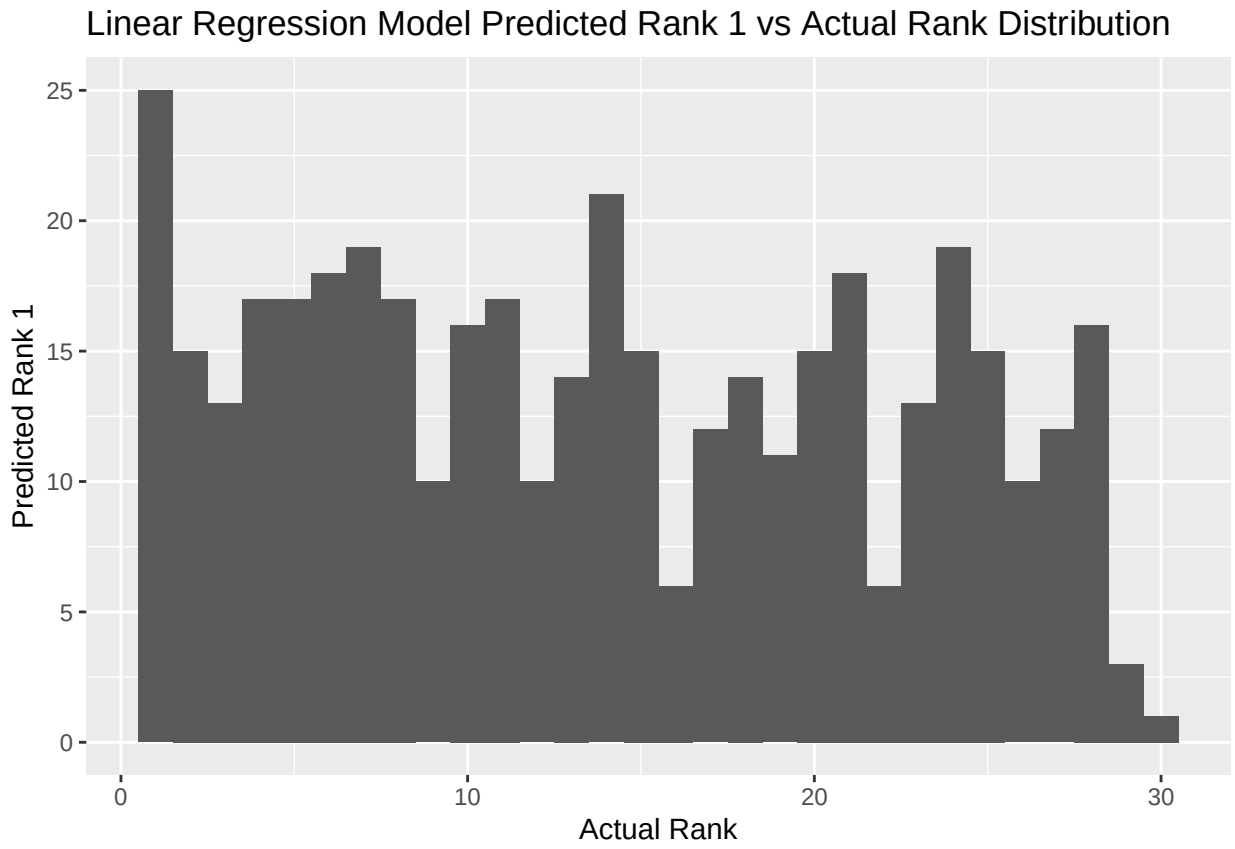
Overall, Model yearly picture vs DJIA actual yearly return in this period

Year	LinearR	RamdomF	XGBoost	DJIA
2012	31.1	18.4	9.93	7.26
2013	14.8	37.3	24.5	26.50
2014	18.6	27.1	5.67	7.52
2015	-49.2	6.63	-7.08	-2.23
2016	20.0	14.1	30.6	13.42
2017	30.0	26.2	9.53	25.08
2018	14.8	15.9	26.2	-5.63
2019	44.5	8.57	29.1	22.34
Total	124.6	154.23	128.39	94.26

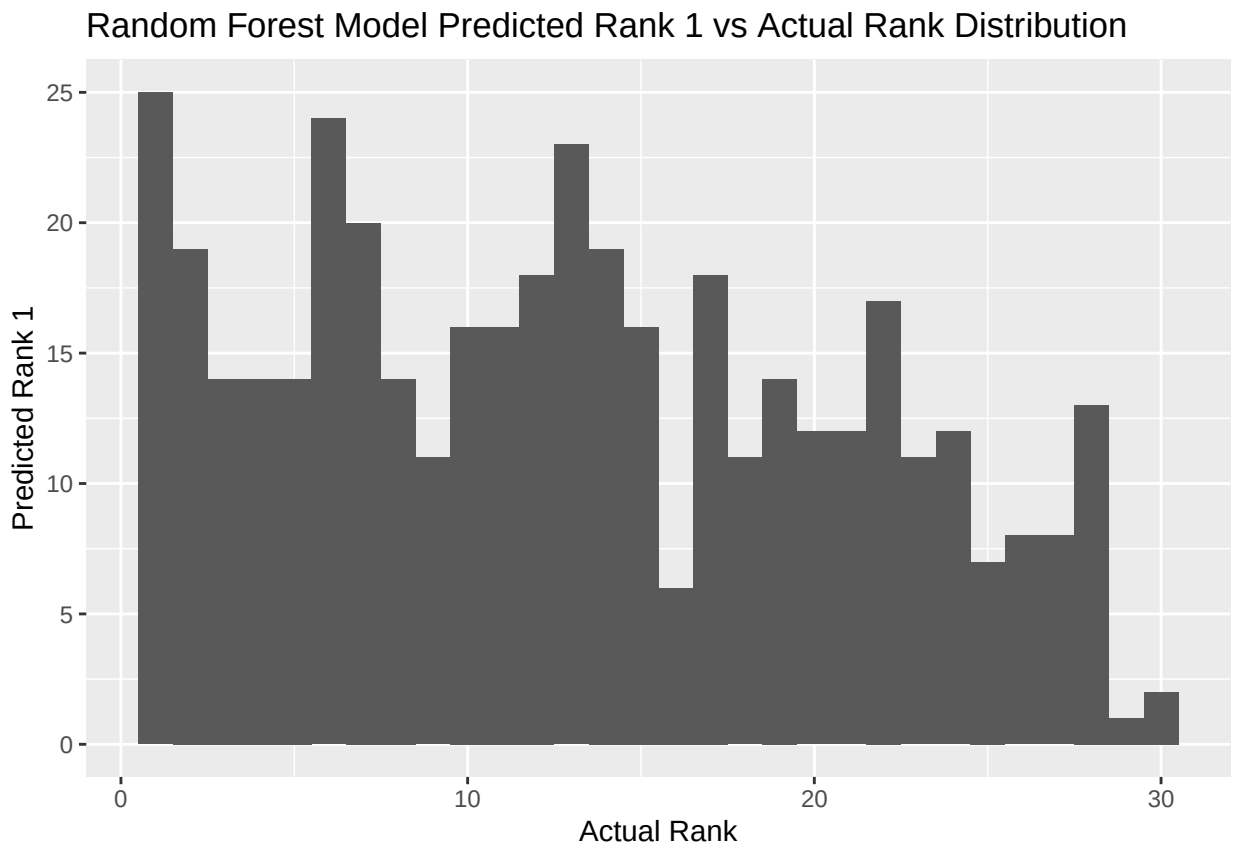
Random Forest Model has beaten DJIA 7 out of the last 8 years(Except 2019).

As far as accuracy of the measurement is concerned, this is how the predicted rank 1 vs actual rank on the next week is distributed

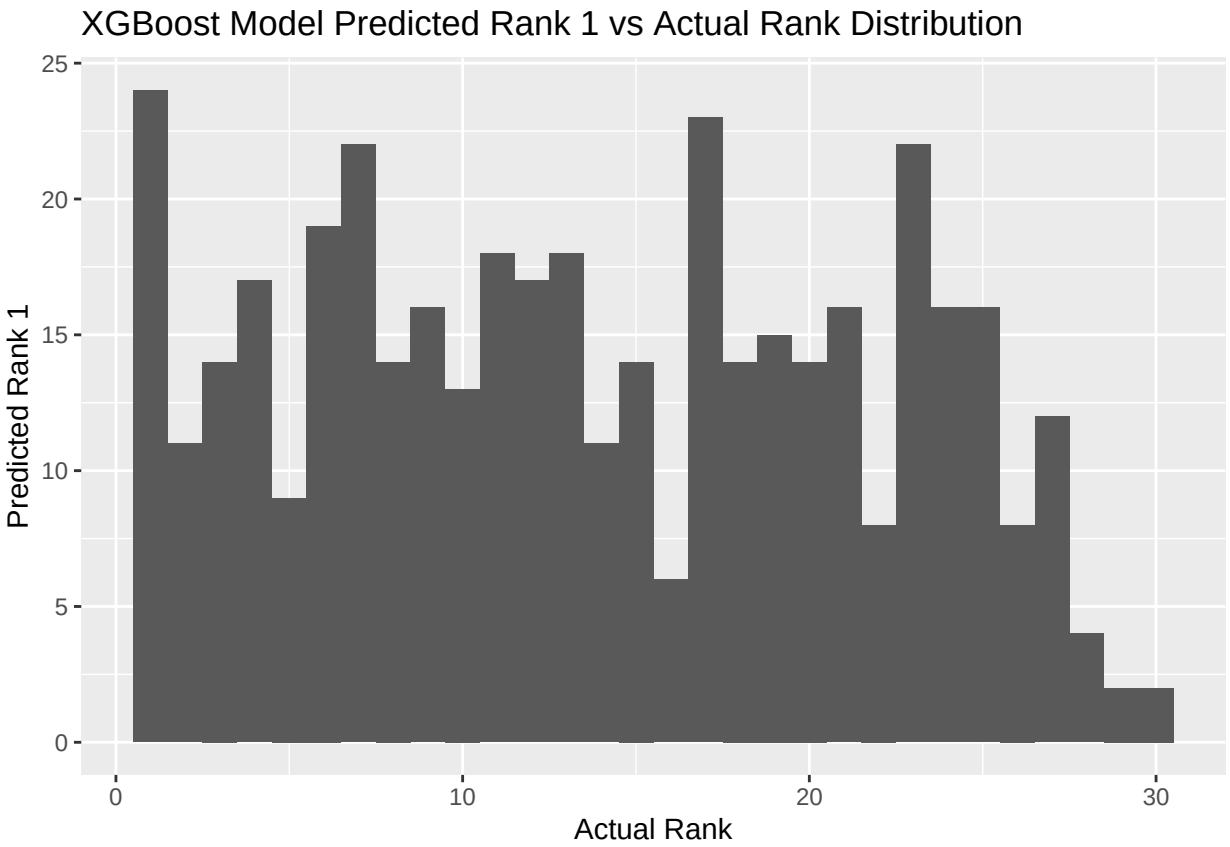
```
rank1_res %>% ggplot(aes(Rank)) +
  geom_histogram(bins = 30) +
  ylab("Predicted Rank 1") +
  xlab("Actual Rank") +
  ggtitle("Linear Regression Model Predicted Rank 1 vs Actual Rank Distribution")
```




```
rank1_rf %>% ggplot(aes(Rank)) +
  geom_histogram(bins = 30) +
  ylab("Predicted Rank 1") +
  xlab("Actual Rank") +
  ggtitle("Random Forest Model Predicted Rank 1 vs Actual Rank Distribution")
```




```
rank1_xgb %>% ggplot(aes(Rank)) +
  geom_histogram(bins = 30) +
  ylab("Predicted Rank 1") +
  xlab("Actual Rank") +
  ggtitle("XGBoost Model Predicted Rank 1 vs Actual Rank Distribution")
```


XGBoost Model Predicted Rank 1 vs Actual Rank Distribution
Predicted Rank 1
Actual Rank
0
5
10
15
20
25
0
10
20
30