Course Overview and Moving Averages

MGMT 638: Data-Driven Investments: Equity

Kerry Back, Rice University

Course goals

- Get exposure to quantitative investment strategies
- Learn the types of risk adjustments people do to analyze all investment strategies - information ratios, attribution analysis, …
- Learn to **execute** python code to backtest and analyze strategies

I am not promising you will be proficient at **writing** python code by the end of the course, but ChatGPT can help.

Course materials

- Slides, assignments, and links to notebooks at mgmt638.kerryback.com
- Submit assignments on Canvas
- Three versions of slides: html, pdf, and Jupyter notebook
- The notebook is more inclusive. It contains all of the code to do all of the analysis that is presented in the html and pdf slides.
- Notebooks are set to open on Google Colab.

Grading

- Grades will be based on individual weekly assignments (80%) and class participation (20%).
- We will do work similar to the assignments in class each week, so there will be an opportunity for coaching.

Predictors

- Past prices
  - moving averages, support and resistance, ...
  - momentum and reversals
  - pairs trading (error correction)
- Corporate fundamentals: ratios and growth rates
- Corporate actions (earnings, dividends, 8ks)
- Corporate insider trades
- Other trades - institutional, retail, short selling
- Social media sentiment
- Proprietary data (satellite images, phone location data, ...)

Backtesting

- Idea + data $\rightarrow$ backtest
- Can we reasonably backtest the strategy "buy electric car companies whose name starts with T?"
- We can backtest a more general strategy with two parameters: type of company, first letter of name
- We can backtest in a loop, updating once per year for example:
    - Find the type of company and the first letter of name that did best in the past n years
    - Buy that company and hold for a year
    - Update each year: find the new best company/first letter and hold it for a year

Tests and other considerations

- Past average return
- Sharpe ratio
- CAPM alpha and information ratio
- Fama-French alpha and information ratio
- Attribution analysis
- Maximum drawdown
- Tracking error relative to a benchmark
- Correlation with other strategies
- Turnover and ransactions costs (including shorting fees)

Universe of stocks

- Large cap or small cap or mid cap or some of all?
- Industries: do we want to bet on industries or match industry weights to a benchmark?
- Value vs growth, etc.
- Our goal could be to find the best possible strategy without any constraints or we might be constrained to find the best strategy within mid-cap energy, for example.
- Different strategies may work better or worse depending on the universe of stocks we can consider.

Example for today

- Do moving average strategies work?
- Get adjusted closing prices from Yahoo Finance
- Adjusted for splits and dividends
- % change is total return, including dividends

In [2]:
```
data.head()
```

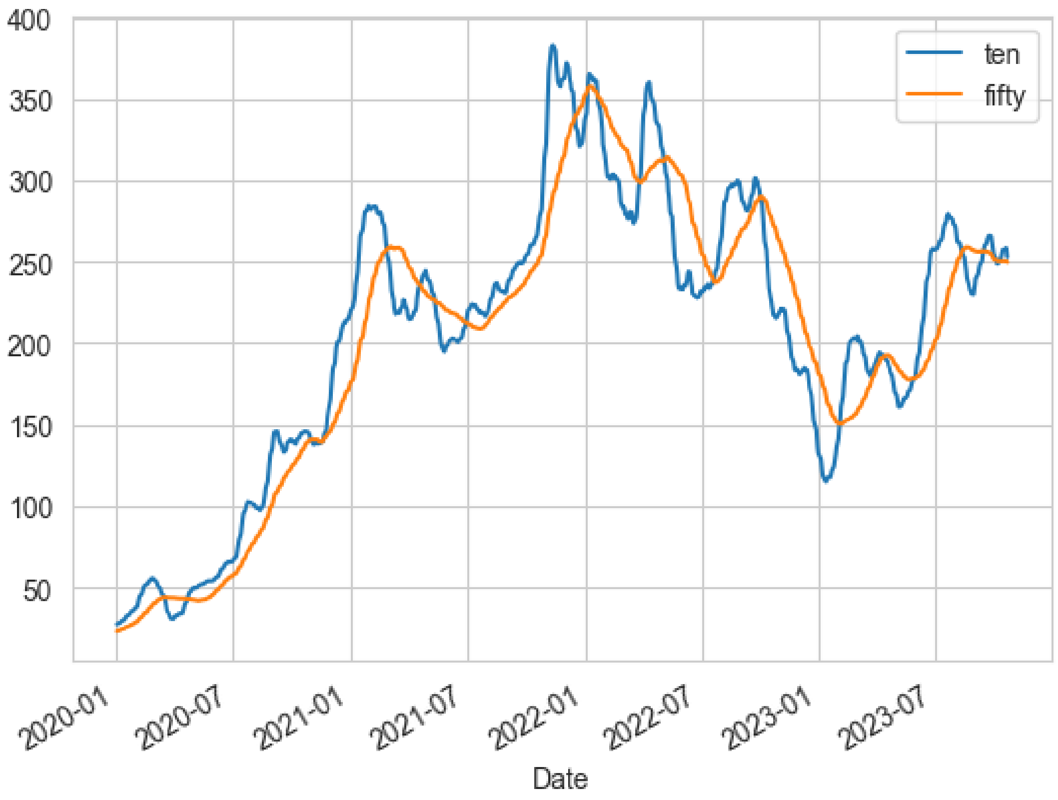
Out[2]:

	closeadj
Date	
2010-06-29	1.592667
2010-06-30	1.588667
2010-07-01	1.464000
2010-07-02	1.280000
2010-07-06	1.074000

Compute and plot moving averages

- Compute average of adjusted closing price over prior n days
- Do 10 day and 50 day as illustration
- Plot from 2020 on only so we can see detail better

In [4]:
```
_ = data[["ten", "fifty"]].loc["2020-01-01":].plot()
```



Compute returns

- Buy and hold return = percent change in adjusted closing price
- Moving average strategy:
    - Long all money in account when 10 day > 50 day
    - Zero position (and no interest for simplicity) otherwise

In [6]:
```python
data.head()
```

Out[6]:

	closeadj	ten	fifty	buy_hold	long	mvg_avg
Date						
2010-09-09	1.381	1.351	1.322	-0.009	True	-0.009
2010-09-10	1.345	1.357	1.318	-0.026	True	-0.026
2010-09-13	1.381	1.360	1.313	0.027	True	0.027
2010-09-14	1.408	1.366	1.311	0.019	True	0.019
2010-09-15	1.465	1.375	1.314	0.041	True	0.041

What tests do we want to do?

- Start by calculating average returns - multiply by 252 to annualize
- Then look at plot of compound returns - log scale works better for long time period
- Compute Sharpe ratios
- CAPM alphas, ...

Mean returns

In [7]:

```python
buy_hold = 252*data.buy_hold.mean()
mvg_avg = 252*data.mvg_avg.mean()

print(f"buy and hold mean return is {buy_hold:.2%} annualized")
print(f"moving average mean return is {mvg_avg:.2%} annualized")
```

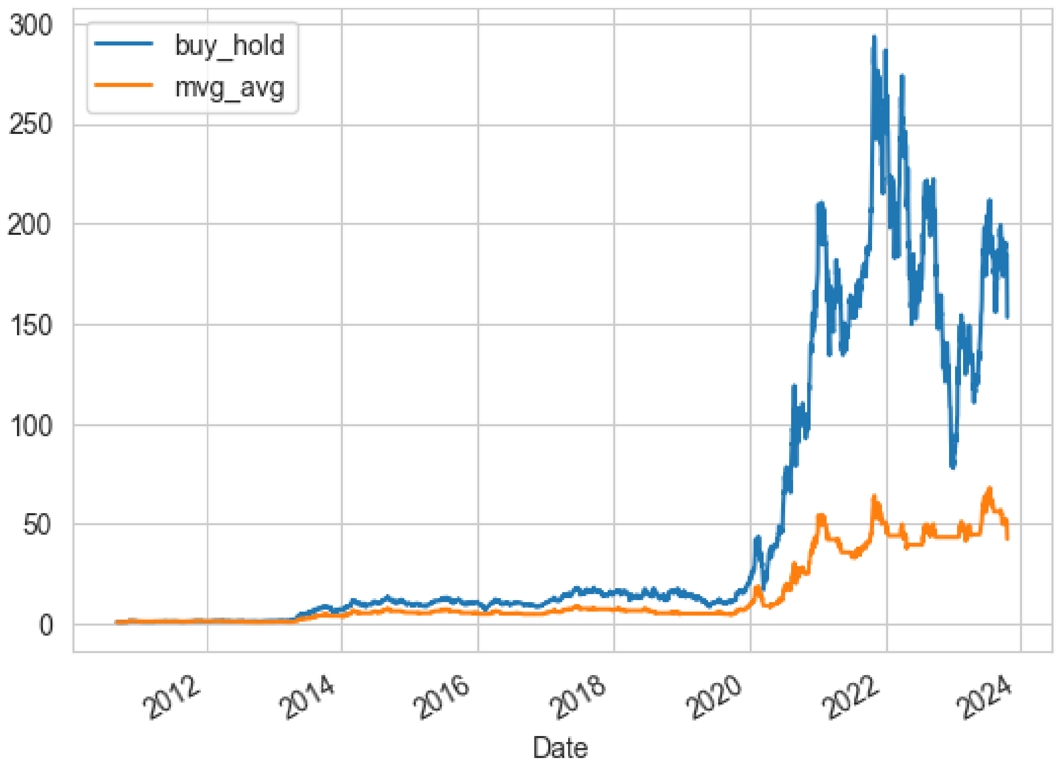
```
buy and hold mean return is 54.43% annualized
moving average mean return is 37.46% annualized
```

Compound return plots

- We will plot the compound return (how much your money grows to starting from $1).
- First in a normal scale and then in a log scale.

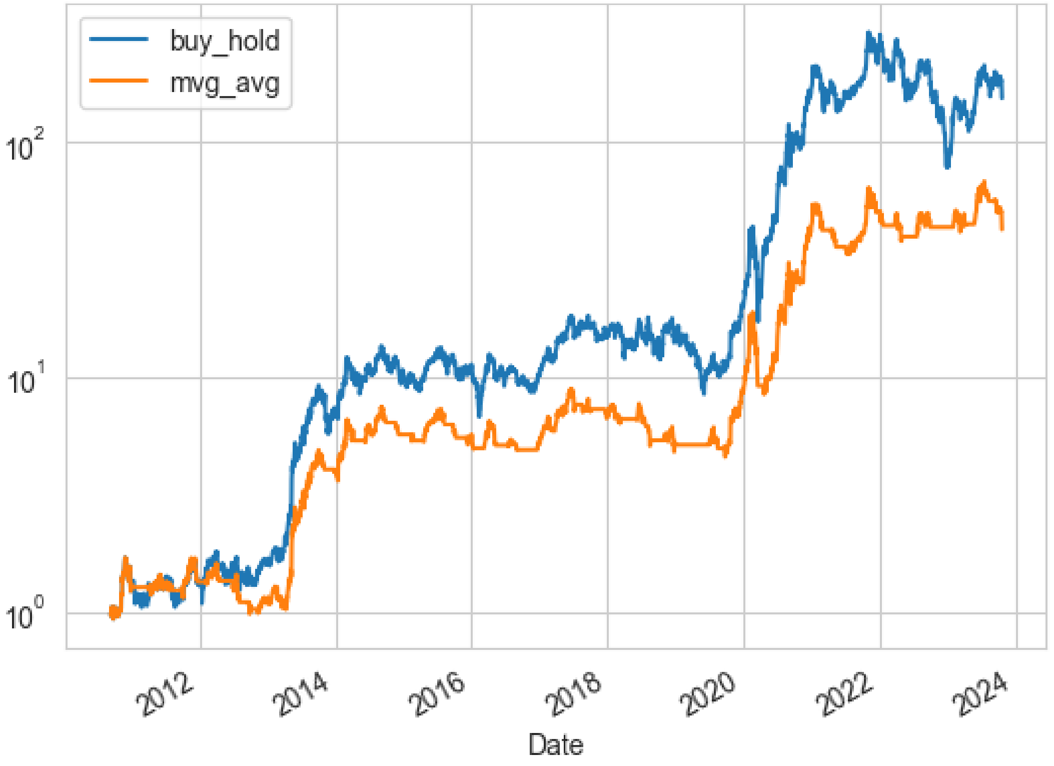
In [8]:

```
_ = (1+data[["buy_hold", "mvg_avg"]]).cumprod().plot()
```



In [9]:

```
_ = (1+data[["buy_hold", "mvg_avg"]]).cumprod().plot(
    logy=True
)
```



Sharpe ratios

- Sharpe ratio is expected return minus risk-free rate / standard deviation
- We'll skip the risk-free rate
- Annualize mean return by multiplying by 252
- Annualize variance by multiplying by 252
- $\Rightarrow$ annualize standard deviation by multiplying by $\sqrt{252}$
- $\Rightarrow$ annualize Sharpe ratio by multiplying by $\sqrt{252}$

In [11]:

```python
print(f"Buy and hold Sharpe ratio is {buy_hold_sharpe:.2%} annualized")
print(f"Moving average Sharpe ratio is {mvg_avg_sharpe:.2%} annualized")
```

```
Buy and hold Sharpe ratio is 95.92% annualized
Moving average Sharpe ratio is 88.76% annualized
```

Multiple stocks

- We can get data for multiple stocks from Yahoo Finance by passing a list of tickers.
- Here is an example.

In [13]:
```
data.head(7)
```

Out[13]:

date	ticker	closeadj
2000-01-03	CVX	17.508469
	F	13.405162
	MSFT	36.205597
	PG	28.428749
	WMT	43.717701
2000-01-04	CVX	17.508469
	F	12.957256

- Then we can do everything we did for a single ticker by running the code in a groupby object.
- Portfolio returns:
    - Instead of looking at returns for each stock individually, we can compare portfolios.
    - We will equal weight each day for simplicity.
    - Neither of the strategies is buy and hold - we have to trade each day to get back to equal weights.
    - For the moving average strategy, we will equal weight the stocks for which the 10 day > 50 day moving average (which could be no stocks or all stocks or anything in between).

In [16]:

```
rets.head()
```

Out[16]:

	eq_wtd	mvg_avg
date		
2000-01-03	NaN	0.0
2000-01-04	-0.024771	0.0
2000-01-05	-0.001449	0.0
2000-01-06	0.013458	0.0
2000-01-07	0.051980	0.0

In [17]:
```
rets.tail()
```

Out[17]:

date	eq_wtd	mvg_avg
2023-10-16	0.010296	0.004630
2023-10-17	0.004664	0.002299
2023-10-18	0.000899	0.000413
2023-10-19	-0.004877	0.000946
2023-10-20	-0.006354	-0.005492

Mean returns

In [18]:

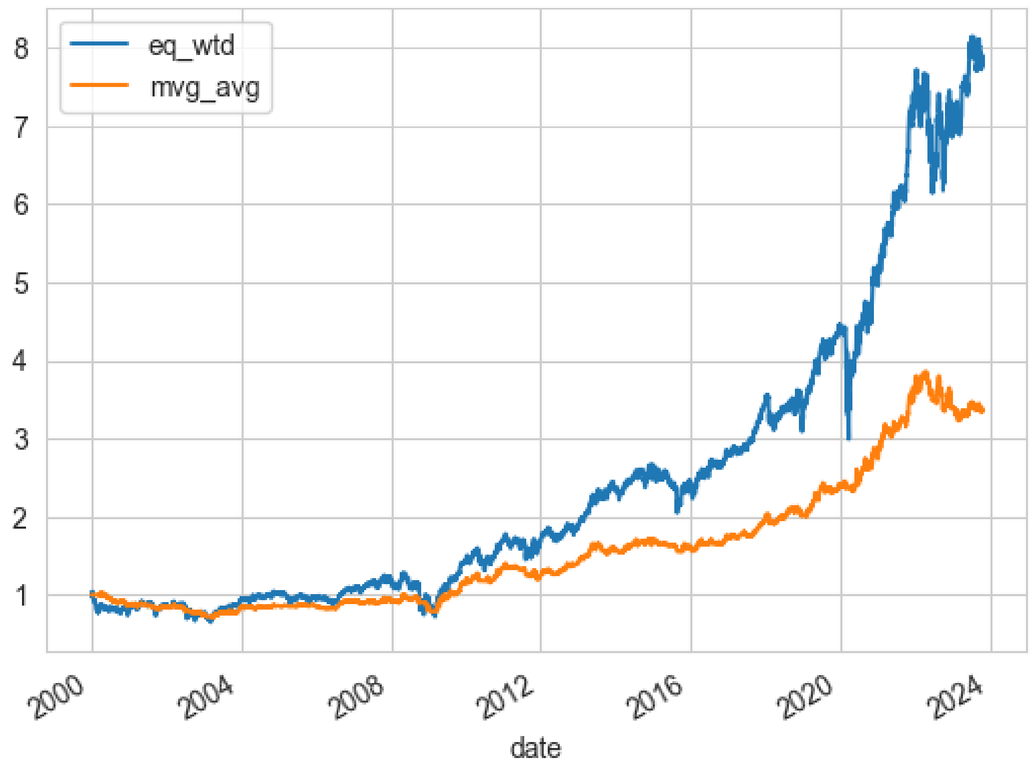
```
eq_wtd = 252*rets.eq_wtd.mean()
mvg_avg = 252*rets.mvg_avg.mean()

print(f"equally weighted mean return is {eq_wtd:.2%} annualized")
print(f"moving average mean return is {mvg_avg:.2%} annualized")
```

```
equally weighted mean return is 10.60% annualized
moving average mean return is 5.65% annualized
```

Compound returns

In [19]:

```python
_ = (1+rets).cumprod().plot()
```



In [20]:

```
_ = (1+rets).cumprod().plot(logy=True)
```

Sharpe ratios

In [22]:
```
print(f"Equally weighted Sharpe ratio is {eq_wtd_sharpe:.2%} annualized")
print(f"Moving average Sharpe ratio is {mvg_avg_sharpe:.2%} annualized")
```

```
Equally weighted Sharpe ratio is 35.35% annualized
Moving average Sharpe ratio is 29.48% annualized
```

Exercise

Look at different sets of stocks and different moving averages and test strategies.