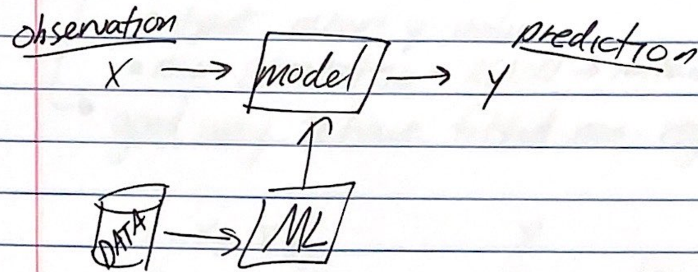
03-01

The ML problem



supervised, regression, learning,

supervised → provide example X, Y

regression → numerical prediction

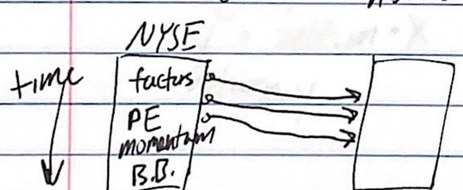
learning → train with data

- lin reg — parametric
- KNN — instance based (keep older data)
- decision trees / forest

Stocks:

X — historical data

Y — historical price



check forward price corresponding to past data

Problem we will focus on:

out of sample orders and testing

03-02

KNN → group w/closest to the new figure, output mean y value
- non-parametric
- KNN → instance based
- good way to have fitted non-regression line

X_1 X_2 X_3	Y
Xtrain / Xtest	Ytrain / Ytest
·data features	·price

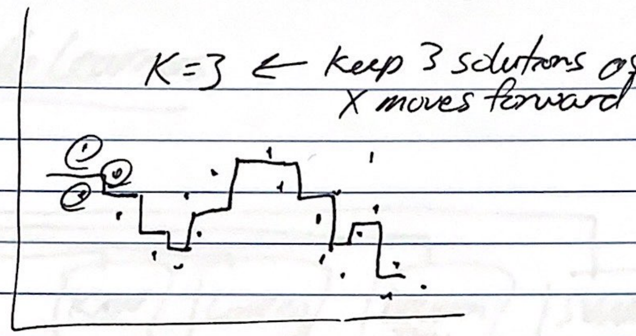
constructor → obj of learner
learner.train
y = learner.query(Xtest)

API example for linreg

```
class LinRegLearner::
  def __init__():
    pass
  def train (X, Y):
    self.m, self.b = fav_linreg(X, Y)    OLS
  def query (X)
    y = self.m·X + self.b
    return y
```

03-03

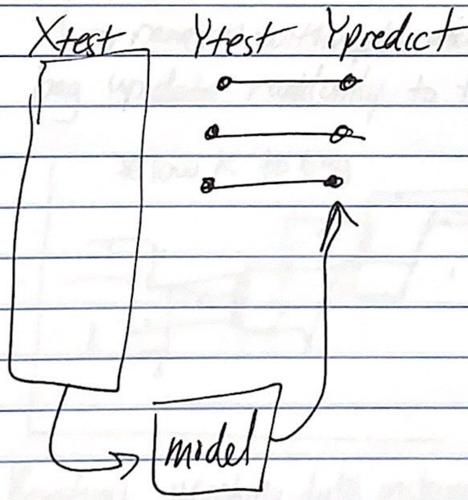
KNN



more K → straight mean of all values
less K → curve fit

data in 20% chunks
↳ cross validation can accidentally look forward
· always roll

<u>correlation</u> np.corrcoef()



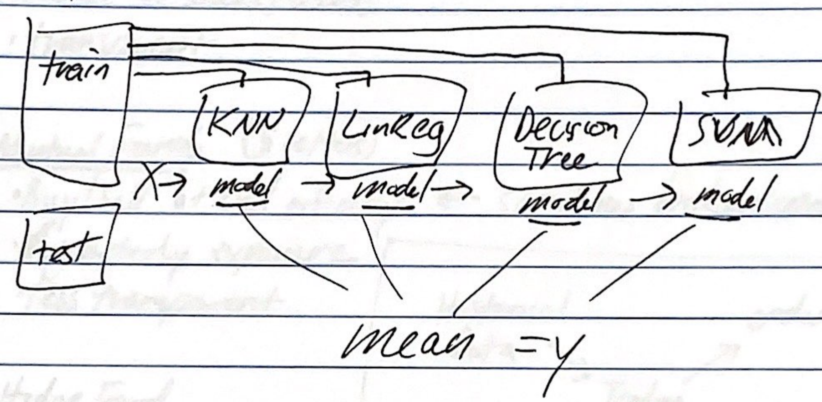
<u>overfitting</u>:
in-sample error decreasing
OOS error increasing

polynomial and KNN overfit at opposite n values

03-04

Ensemble Learners

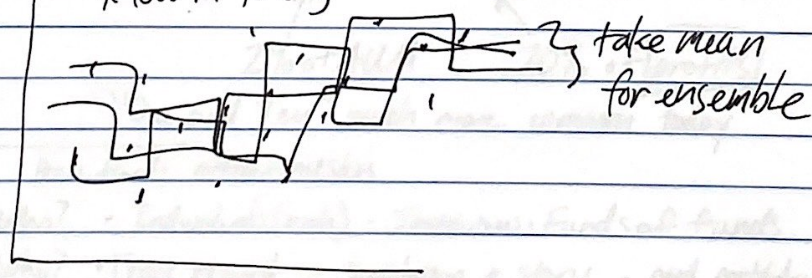
Data



why? lower error
less overfitting ← biases uncorrelated

Bootstrap aggregating $\boxed{\text{bagging}}$
→ random with replacement to make bags
bag up data randomly to train models → ensemble

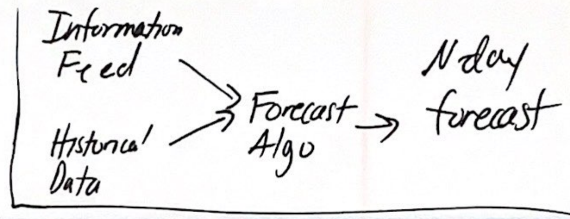
*low K to bag



Boosting weighing data instances that have been modeled poorly before creating next bag
→ AdaBoost
- this will overfit ~~before~~ worse than bagging

Boosting and Bagging are only wrappers for existing learners
- reduce error, reduce overfittage

02-01

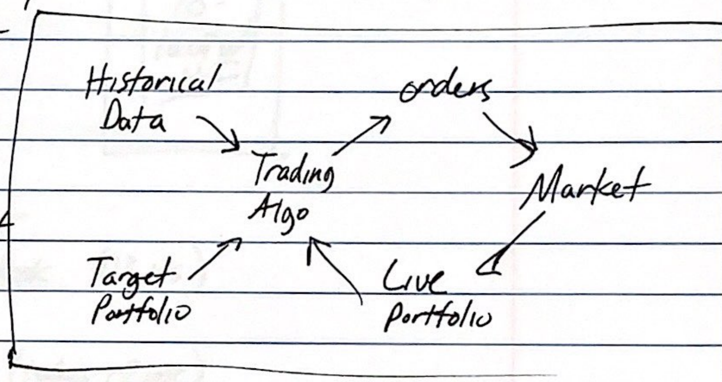


Types of funds

ETFs (4 or 3 letters)
- tradable like stocks ← very liquid
- basket of stocks/assets
- transparent

Mutual Fund (5 letters)
- Buy/sell at end of day ← sometimes broker specific
- quarterly exposure
- less transparent

Hedge Fund
- buy/sell by agreement
- No disclosure
- not transparent



Incentives for managers
- ETF → expense ratio of AUM (.01% : 1 BIP)
- Mutual Fund → expense ratio, higher than ETF ~.5% to 3%~
- Hedge Funds → "Two and Twenty"

2% of AUM   20% of profits

"One and Ten" much more common today

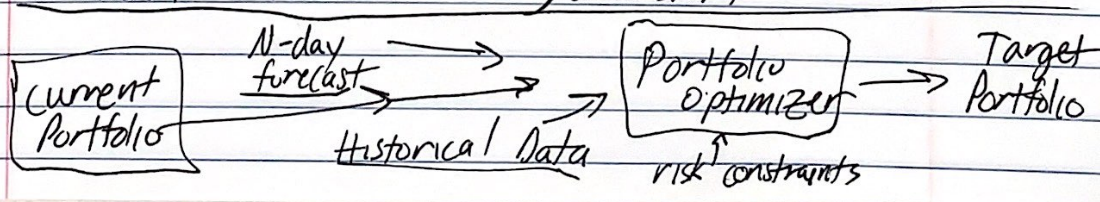
How funds attract investors

Who? • Individuals (rich) • Institutions • Funds of funds.

Why? • Track record • simulation + story • good portfolio fit

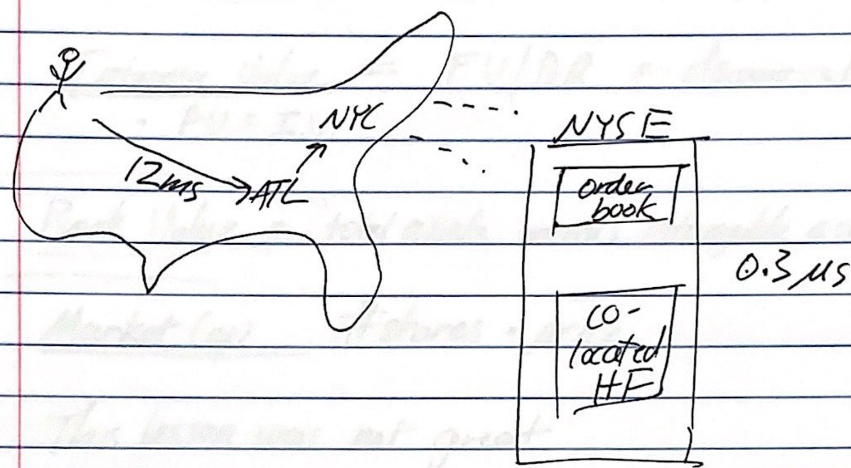
why the strategy makes sense

Hedge funds Goals / Metrics: return, volatility, risk/reward (Sharpe)
- beat a benchmark → SP500
- absolute return → long and short



02-02

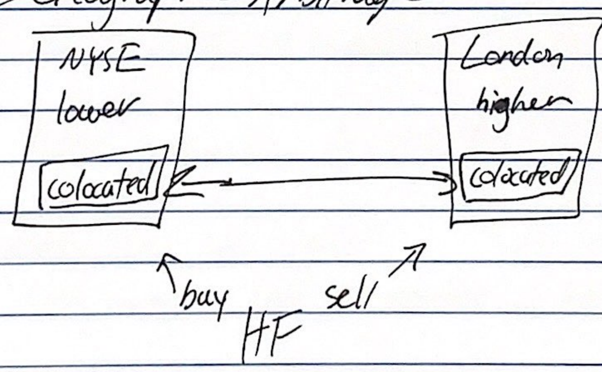
Market Mechanics



Ⓐ order book exploit

① HF observes order book (0.3 μs)
② HF buys stock
③ you click "buy" (takes 12ms)
④ price goes up meanwhile from 0.3 μs → 12ms
⑤ HF sells you stock

Ⓑ Geographic Arbitrage



02-03

Company Valuation

The value of a future dollar : $PV = FV/(1+r)^i$ ← t

Intrinsic Value = FV/DR ← discount rate

- P.V. = I.V.

Book Value = "total assets minus intangible assets and liabilities"

Market Cap #shares • price

This lesson was not great.

02-04

Capital Asset Pricing Model

you can't beat the market
$E[\alpha] = 0$
result: index fund

Portfolio Daily returns

$$\underbrace{R_p(t)}_{\text{portfolio return}} = \sum_i \underbrace{w_i}_{\text{asset weight}} \underbrace{R_i(t)}_{\text{asset return}}$$

negative weights indicate short position

with no leverage: $\sum |w_i| = 1.0$

---

market cap weighted indexes uses relative weights

$$w_i = mkt\_cap_i / \sum_j mktcap_j$$

CAPM equation:

$$\underbrace{R_i(t)}_{\text{return for a stock}} = \underbrace{\beta_i \underbrace{R_m(t)}_{\text{return for market}}}_{\text{market}} + \underbrace{\alpha_i(t)}_{\text{residual}}$$

CAPM says $E[\alpha] = 0$

α = y-intercept
β = slope

CAPM for portfolio

$$R_p(t) = \sum_i w_i (\beta_i R_m(t) + \alpha_i(t))$$

$$\beta_p = \sum_i w_i \beta_i$$

$\therefore$ $R_p(t) = \beta_p R_m(t) + \alpha_p(t)$ ← CAPM

or

$R_p(t) = \beta_p R_m(t) + \sum_i w_i \alpha_i(t)$ ← ACTIVE

Implications of CAPM

- only way to beat the market is by choosing β

Arbitrage Pricing Theory (APT)

β for one stock can be broken down into seperate factors, multiple β's

02-05

Two-Stock Scenario

A: predict +1% over market $\beta_A = \cancel{\ } 1.0$
B: predict -1% below market $\beta_B = 2.0$

$$\left.\begin{aligned} r_A &= \beta_A r_m + \alpha_A \\ r_B &= \beta_B r_m + \alpha_B \end{aligned}\right\} \quad r_m = 0\%\text{, our return is alpha}$$

can we weight so that $\beta_p = 0$

abs$(\Sigma w) = 1$ but negative positions have negative weights

CAPM for hedge funds summary:

Assuming
- information $\longrightarrow \alpha_i$ (forecast)
- β_i

CAPM enables:
minimize market risk when $\beta_p = 0 \mid w_i$

02-06 Technical Analysis

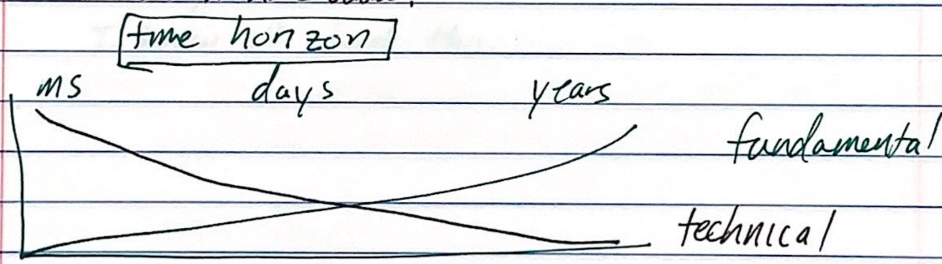
$$normed = \frac{values - mean}{values.std()} = [-1, 1]$$

- historical price and volume <u>only</u>
- computed statistics are called indicators
- indicators are heuristics

<u>Why it might work?</u>
- there is information in price
- heuristics work in other fields

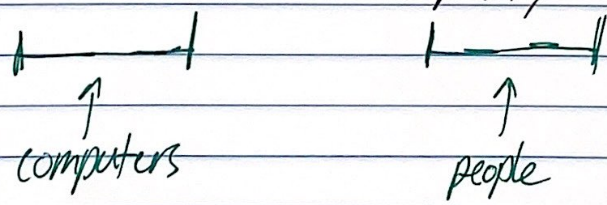
<u>When is TA effective?</u>
- individual indicators are typically <u>weak</u>
- combinations are much stronger
- look ~~vs~~ for contrasts (stock vs market)
- works better for shorter periods

When does TA have value?



← decision speed —

— decision complexity →



momentum → rate of change $= \frac{price_t}{price_{t-n}} - 1$

SMA → simple moving average, proxy for value

BB → add band $n\sigma$ ± the SMA, typically $n=2$ so 2σ
- outside crosses towards the inside

02-07

Dealing with Data

How data is aggregated?

tick data is usually aggregated into OHLCV
↳ no time requirement, just successful transactions

Why might stocks split?
- increase liquidity and availability
- use adjusted close to normalize for splits and dividends
  CAREFUL: NOT for historical filters

Dividend adjustments
- price ~~before~~ pre-div is $S+d$
- price at div is S
- adjusted price is reduced by dividend amount

Survivorship Bias
I know all about this.

02-08

Efficient Market Hypthesis

Eugene Fama
· Nobel Prize

EMH assumptions:
- large number of investors operating simultaneously
- new information arrives randomly
- prices adjust quickly
- price reflects all information

Where does info come from?
- Price / Volume
- Fundamentals (reported quarterly)
- Exogenous (other stuff in the world)
- Company Insiders

Forms of EMH

Weak - future prices cannot be predicted by
↳ analyzing historical price
↳ TA no FA yes

Semi-Strong - prices adjust rapidly to new public information
↳ TA no FA no insider yes

Strong - prices reflect all information public and private
↳ impossible to beat the market

Is EMH correct?
unlikely strong, we see this refuted
unlikely semi-strong, we see PE ratios being predictive

02-09

The Fundamental Law of Active Portfolio Management

Buffet "Only when the tide goes out do you discover who has been swimming naked."
"Wide diversification is only necessary when investors do not know what they are doing."

Grinold's Fundamental Law...

performance = skill · $\sqrt{\text{breadth}}$

$IR = IC \cdot \sqrt{BR}$

information ratio — information coeffecient — trading opportunities

Real World:

similar performance

RenTech trades ~100k/day
Buffet holds ~120 stoks

$\boxed{IR}$ $r_p(t) = \underbrace{\beta_p r_m(t)}_{\text{market}} + \underbrace{\alpha_p(t)}_{\text{skill}}$, $IR = \frac{\text{mean}(\alpha_p(t))}{\text{stdev}(\alpha_p(t))}$

information ratio

↳ sharpe ratio of excess returns, part due to skill

$\boxed{IC}$ information coeffecient - correlation of forecasts to returns
$\boxed{BR}$ breadth, number of trading opportunities

$IR = IC \cdot \sqrt{BR}$ Grinold and Kahn

performance — skill — breadth

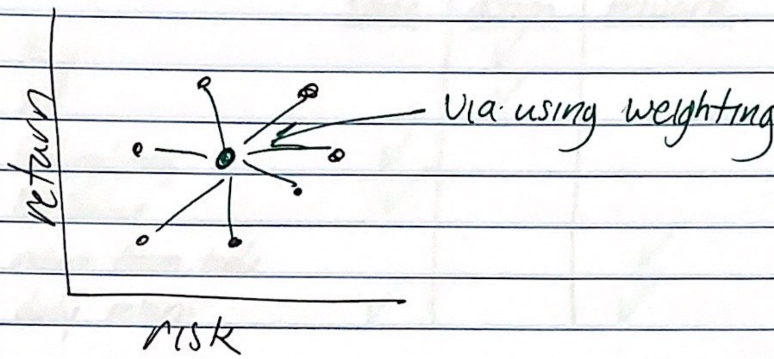
If Buffet and Simon both have same IR
Simon is $\frac{1}{1000}$ as effecient as Buffet
Buffet trades 120/yr

$IC_B \cdot \sqrt{120} = IC_S \sqrt{X}$ $IC_B = \frac{1}{1000} IC_S$

$1000 \cdot \sqrt{120} = \sqrt{X}$

02-10 Portfolio Optimization

What is Risk? → volatility, σ of historical daily returns



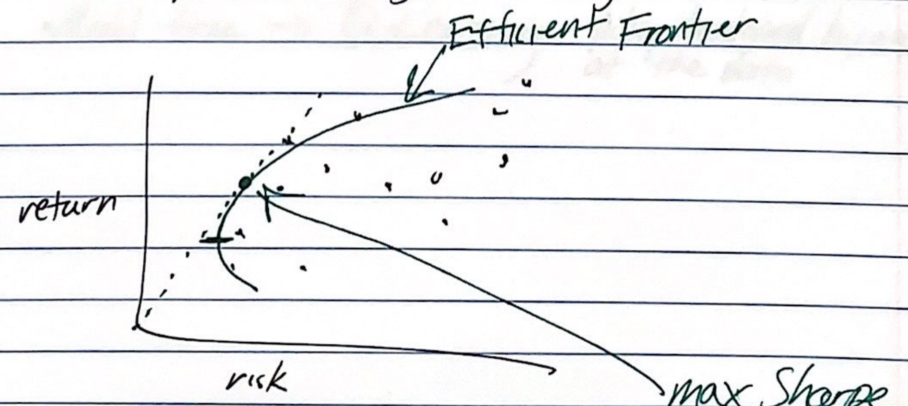
Can we do better? Harry Markowitz, Nobel prize
- blend of stock/bond is lower risk

The importance of covariance
- smoothes out equity curve and lowers volatility by blending anticorrelated assets

MVO → mean variance optimization
- anticorrelation in short term
- inputs: expected return, σ, covariance, target return

Outputs: asset weights for portfolio that minimize risk

Efficient Frontier
- curve where portfolios have optimized weights for maximum risk-return target
- max sharpe is line tangent from origin



03-05

Reinforcement Learning

robotics: sense
think
act

Trading as a RL problem:

	state	action	reward
Buy		✓	
Sell		✓	
Holding Long	✓		
BB Value	✓		
return from trade			✓
daily return	✓		✓

RL problem: environment ⇒ the market
state ⇒ features, holdings
actions ⇒ buy, sell, do nothing

Markov Decision Problem

- set of states, S
- set of actions, A
- transition function $T[S, A, S']$
- Reward function $R[S, A]$

∴ Find policy $\pi(s)$ that will maximize reward

Unknown transitions and rewards

experience tuple: $\langle s_1, a_1, s_1', r_1 \rangle$ $\langle s_2, a_2, s_2', r_2 \rangle$

- Model based: build model of $T[s, a, s']$, $R[s, a]$
  ↳ value/policy iteration
- Model-free → Q-Learning: policy developed by only looking at the data

03-09
continued

Reinforcement Learning

What to optimize?

With infinite horizon, result is useless.
With finite horizon, we can gain meaningful knowledge

infinite horizon: $\sum_{i=1}^{\infty} r_i$ | finite horizon: $\sum_{i=1}^{n} r_i$

$\underbrace{\text{discounted reward}}_{\text{used in Q-Learning for ease of math application and convergent properties}}$: $\sum_{i=1}^{\infty} \lambda^{i-1} \cdot r_i \quad , 0 < \lambda \leq 1.0$

usually interest rate based

λ: each dollar in the future is worth $\lambda^i \cdot 1$ where i is future period

RL algorithms solve MDPs (Markov decision Problems)

03-06

Q-Learning (model free approach)

What is Q? is a table of many states/actions

$Q = [s, a]$ = immediate reward + discounted reward

Q represents the value of taking action $\boxed{a}$ in state $\boxed{s}$

How to use Q?

$$\underbrace{\pi(\underline{s})}_{\text{policy for state } s} = \text{argmax}_{\underline{a}}(Q[\underline{s}, \underline{a}])$$

- we are in state $\underline{s}$
- check Q table for best action a

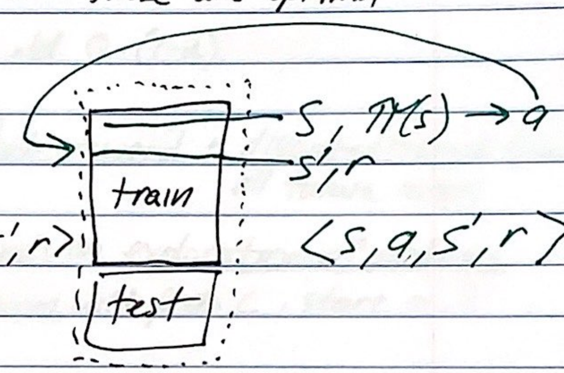
↳ finds $\underline{a}$ that maximizes the function

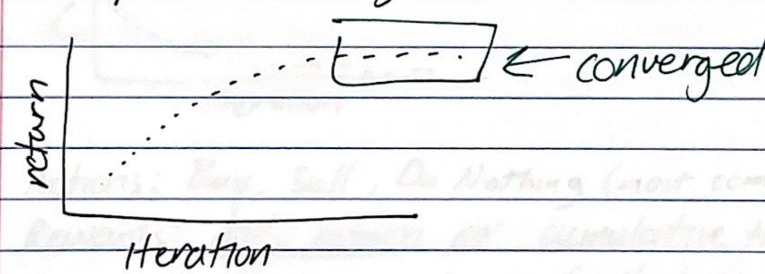
$\pi^*(s)$, $Q^*[s,a]$ ← these are optimal

Q-learning Procedure

Big Picture

- select training data
- iterate over time $\langle s, a, s', r \rangle$
- test policy π
- repeat until converge





Details on iterations

- set start time, init Q[ ]
- compute s
- select a via consulting Q, our policy
- observe r, s'
- update Q!

(compute s through observe r, s') gives us $\langle s, a, s', r \rangle$

03-06
continued

Q-Learning

Update Rule

$\alpha \rightarrow$ learning rate from 0 to 1.0 default ~ 0.2

$$Q'[s,a] = (1-\alpha)Q[s,a] + \alpha \cdot \text{improved estimate}$$

$$Q'[s,a] = (1-\alpha)Q[s,a] + \alpha(r + \lambda \cdot \text{later rewards})$$

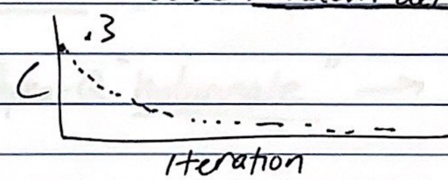
λ range from 0 to 1.0

a low λ means we value later rewards less

$$\text{later rewards} = Q[s', \text{argmax}_{a'}(Q[s',a'])]$$

new Q value from [s,a] is old $Q \cdot (1-\alpha)$
$+ \alpha \cdot$ new best estimate
↳ immediate reward + discounted reward from all future actions

- success of Q-Learning depends on exploration of actions
  - choose random action with prob. c, start ~.3



Actions: Buy, Sell, Do Nothing (most common)
Rewards: daily return or cumulative trade return
↳ gives agent more feedback for learning and convergence
State: C/SMA, BB, P/E value, holdings, return since entry
Creating the state → state is an integer
↳ discritize each factor (integer creation)
↳ combine

Discretizing → real num to int across limited scale

```
stepsize = size(data) / groups or steps
data.sorted()
for i in range(0,steps)
    threshold[i] = data[(i+1)·stepsize]
```

thresholds are far apart for sparse data, closer for crowded data

03-07

Dyna

Richard Sutton
Dyna speeds up Q-learning

Dyna-Q Big Picture

$\left.\begin{array}{l}\text{init Q-table} \\ \text{observe } s \\ \text{execute } a \text{, observe } s' \text{ and } r \\ \text{update Q with } \langle s, a, s', r\rangle \\ \text{repeat}\end{array}\right\}$ Q-Learning

expensive computationally and monetarily and time-wise

Dyna-Q

$\left.\begin{array}{l}\text{Learn model } T, R \\ \text{"halucinate experience"} \\ \text{update Q} \\ \text{repeat}\end{array}\right]$ cheap

- do this 100/200 times, then return to real world interaction

Dyna-Q "halucinate" → s, a = random
s' = infer from $T[\,]$
$r = R[s,a]$

Learning T → transition function

$T[s,a,s']$ prob $(s,a) \rightarrow s'$ (that we end up at s')

- init $T_{count} = 0.00001$ ← avoid D.B. Zero error later
- while executing, observe s, a, s'
- incriment $T_c[s,a,s']$ when we successfully get s' outcome

How to evaluate T? (in terms of T_c)

$$T[s,a,s'] = T_c[s,a,s'] \Big/ \sum_i T_c[s,a,i]$$

03-07
continued Dyna

Learning R

$R[s,a]$ = expected reward for s,a
r = immediate reward

$R'[s,a] = (1-\overset{0.2}{\alpha})R[s,a] + \alpha \cdot r$

Dyna-Q recap

EXPENSIVE

↓ backtest or live
update Q } original Q-Learning

↓ update model → $T'[s,a,s']$ and $R'[s,a]$

↓ simulation
hallucinate 100/200 experiences
update Q

CHEAP

↓ return to next "day" in backtest or live environment