finewbie$

Personal investment advisor for beginners

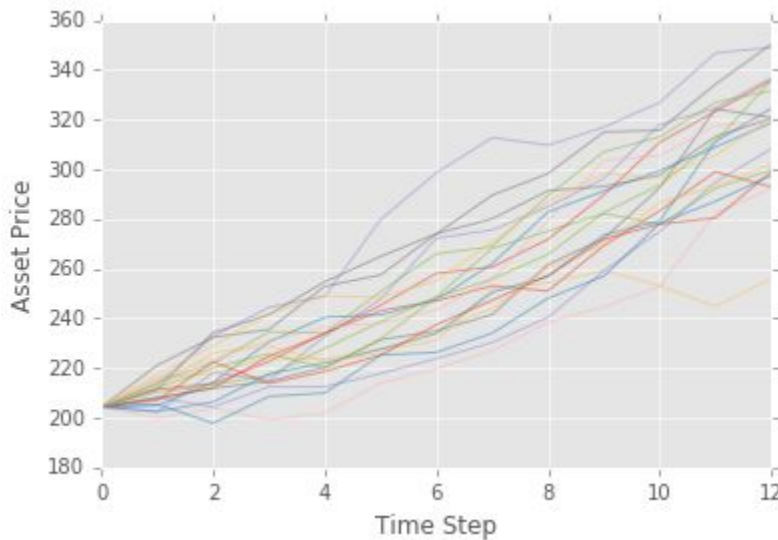Jangwon Park
Sowmya Tata
Kai Zhang

Agenda

1. Description of the model
2. Description of front-end & back-end
3. Demonstration of finewbie
4. Future improvements

Description of the model

Model

- Mean-variance stochastic optimization based on Monte-Carlo simulation
- Asset prices follow GBM

Objective function

Minimize:

$$\lambda \frac{1}{S} \sum_{s=1}^{S} (W_s - W)^2 - (1 - \lambda) W$$

Where:

λ = investor's risk attitude

S = total number of scenarios

W_s = final portfolio value under scenario s

W = mean portfolio value across **all** scenarios

Constraints

t = 0 constraint:

$$\sum_{i=1}^{N} X(i, 0) = b$$

t = T constraint: $\forall s \in S$

$$\sum_{i=1}^{N} (1 + r(i, T, s))X(i, T-1, s) = \sum_{i=1}^{N} X(i, T, s)$$

Where:

N = total number of assets

X(i,t,s) = net value of asset *i*, at time t, under scenario s (in $)

b = initial contribution ($)

r(i,t,s) = rate of return on asset *i*, at time t, under scenario s

Constraints (cont'd)

t = 1 to t = *T* - 1 constraints: $\forall s \in S$

$$\sum_{i=1}^{N} (1 + r(i,t,s))X(i,t-1,s) + p(t,s) = \sum_{i=1}^{N} X(i,t,s)$$

Where:

p(t,s) = additional contribution ($) from investor at time t, under scenario s

Constraints (cont'd)

Inequality constraints:

$$\forall s \in S, t \in \{1, ..., T\}, i \in \{1, ..., N\}$$

$$\sum_{i=1}^{N} X(i, T, s)(1 - m(i)) \geq G$$

$$X(i, 0, s) \geq init\_alloc(i)$$

$$X(i, t, s) \geq 0,$$

$$p(t, s) \geq 0,$$

$$p(t, s) \leq D,$$

Where:

$m(i)$ = management fees of asset i

G = target goal

$init\_alloc(i)$ = recommended initial allocation for asset i

D = maximum contribution from investor at each time step = investor's disposable income

Front-End

Wanted the layout to be:

- Simplistic, visually-appealing and intuitive design
- Informative by explaining terms used

Technologies used:

- Python, Flask, Bootstrap, jinja2, HTML/CSS, RSS Feeds (news), bokeh (graphing library)
- Did not have time to use JavaScript extensively for additional functionality

User Testing

Multiple user tests were conducted throughout the design process

- Risk profile questionnaire
- Layout and pages in the web app
- Final application testing
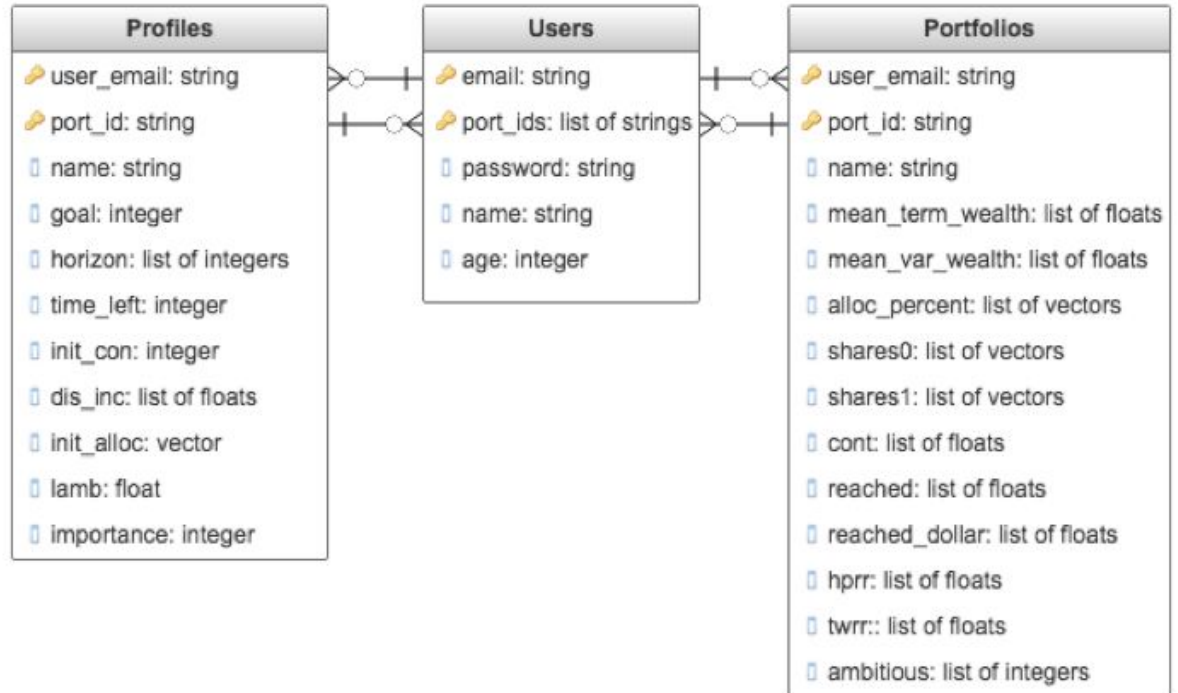- (plus random testing and feedback sessions in the common room)

Asset Selection

- ETFs trade like stocks
- Diversified
- Low management fees (Good for students)
- 5 Class
- Top 1 by Total Assets ($) (limit management fees and computation costs)
- 1% Savings Account

Asset Class	Symbol	Name	Total Assets ($MM)	YTD	Average Volume
Large Cap	SPY	SPDR S&P 500 ETF	$255,164,614.91	20.00%	64,641,232.0
Small Cap	IWM	iShares Russell 2000 ETF	$46,578,195.68	14.57%	24,111,828.0
International	VEU	Vanguard FTSE All-World ex-US ETF	$22,374,340.64	24.48%	1,721,730.0
Short-Term Fixed Income	CSJ	iShares 1-3 Year Credit Bond ETF	$11,817,006.32	1.10%	470,261.0
Long-Term Fixed Income	BLV	Vanguard Long-Term Bond ETF	$2,341,500.00	9.43%	182,234.0

Data Management

- Yahoo Finance prices (temporary data, use them once)
- SQL vs NoSQL (MongoDB)
- Flexibility
- Stores Lists
- Profiles, Users, Portfolios

Profiles

- 🔑 user_email: string
- 🔑 port_id: string
- 🔲 name: string
- 🔲 goal: integer
- 🔲 horizon: list of integers
- 🔲 time_left: integer
- 🔲 init_con: integer
- 🔲 dis_inc: list of floats
- 🔲 init_alloc: vector
- 🔲 lamb: float
- 🔲 importance: integer

Users

- 🔑 email: string
- 🔑 port_ids: list of strings
- 🔲 password: string
- 🔲 name: string
- 🔲 age: integer

Portfolios

- 🔑 user_email: string
- 🔑 port_id: string
- 🔲 name: string
- 🔲 mean_term_wealth: list of floats
- 🔲 mean_var_wealth: list of floats
- 🔲 alloc_percent: list of vectors
- 🔲 shares0: list of vectors
- 🔲 shares1: list of vectors
- 🔲 cont: list of floats
- 🔲 reached: list of floats
- 🔲 reached_dollar: list of floats
- 🔲 hprr: list of floats
- 🔲 twrr:: list of floats
- 🔲 ambitious: list of integers

Demo

Future Improvements

Business Logic:

Integer programming to avoid suggesting purchases / sales of partial shares

Reformulation of stochastic programming using weights of assets rather than $ values

Front End:

Better, interactive displays for portfolio performance

Effective communication and error messaging to the user

Back End:

Create a temporary database collection for Yahoo Prices for faster request time and better user experience (Yahoo Prices is a computationally heavy function that runs with the Optimization function in the same request)