Computer Simulations and Risk Assessment – Lecture 1

Fall 2019

Brandeis International Business School

Lecture I – Outline

General Introduction to the Course

- Instructor Info.
- Course Info
- Quant Finance Career

Introduction to Python

- Python Installation
- Intro to Python

Instructor 1: Steve Xia

- About the instructor:
  - Professor of the Practice of Finance
  - 16 Years of experience working in the investment management industry as Head of research/Portfolio Manager. Currently Senior Managing Director, Head of Quantitative Research and Analytics at Guardian
  - Worked as head of fixed income quant and leader of active asset allocation research for top US asset managers
  - Ph.D. in Engineering, and certificate in Financial Technology, MIT
  - CFA and FRM charter holder

- Office hour:

  Fridays, 9-9:30am, 12:30-1:30 pm, S-001B

- Contact info: qsxia@brandeis.edu

Meeting time and TA

- Course meeting time
  - Fridays 9:30 AM–12:20 AM, Lemberg Academic Center 180

- TA and TA Office hour:
  - Mark McAvoy and Yu Song
  - Office Hour:
    - Yu Song 2:30 -3: 30 pm on Mondays
    - Mark McAvoy 4 - 5pm on Wednesday
  - Contact info:
    - ysong4025@brandeis.edu
    - mcavoy@brandeis.edu

Course Information - Prerequisites

1. FIN 201a, or a basic knowledge of finance is essential.
2. Econ210f/Econ211f or Econ184a: A basic working knowledge of mathematical statistics is important. You need to know about random variables, probability distributions and densities. Also, a little knowledge of linear regression will be useful. A standard one semester course in math stats with calculus will cover this.
3. Fin 270a (Options and derivatives) would also be useful, but it is not required.
4. Although computer skills will be taught in the course, some enthusiasm for programming will be useful.
5. The course also assumes basic calculus equivalent to about 1 semester of calculus at the undergraduate level.

This course is designed for 2nd year IBS masters students (MA, MSF, MBAi). PhD students may also find some of the content useful as well.

Course Information – Textbooks and Readings

Optional Readings:

1. Jon Danielsson, *Financial Risk Forecasting*, Wiley, 2011. (ISBN: 978-0-470-66943-3)
2. Python for Data Analysis, Wes McKinney, O’Reilly Media, Inc., 2018.

Course Information

Grading

Grades will be based on problem sets (10%), a midterm exam (30%), a group project (15%), and a final exam (45%).

Communications

You are responsible for all announcements and materials in class. Also, much of the information in class will be sent over Latte and the class website.

Required Software

Python programming language. Follow the instructions to install Python on your own computer. Please make sure you install the exact version of software per the instruction

Course Information-Rules specific to Fin285

Exams

- Your own work.

Problem sets

- Hand in your own work.
- Can talk and assist each other.
- Use all resources

Group projects

- Own work for the group.
- Hand in one writeup per group.

Laptops: Please bring to class if you have one

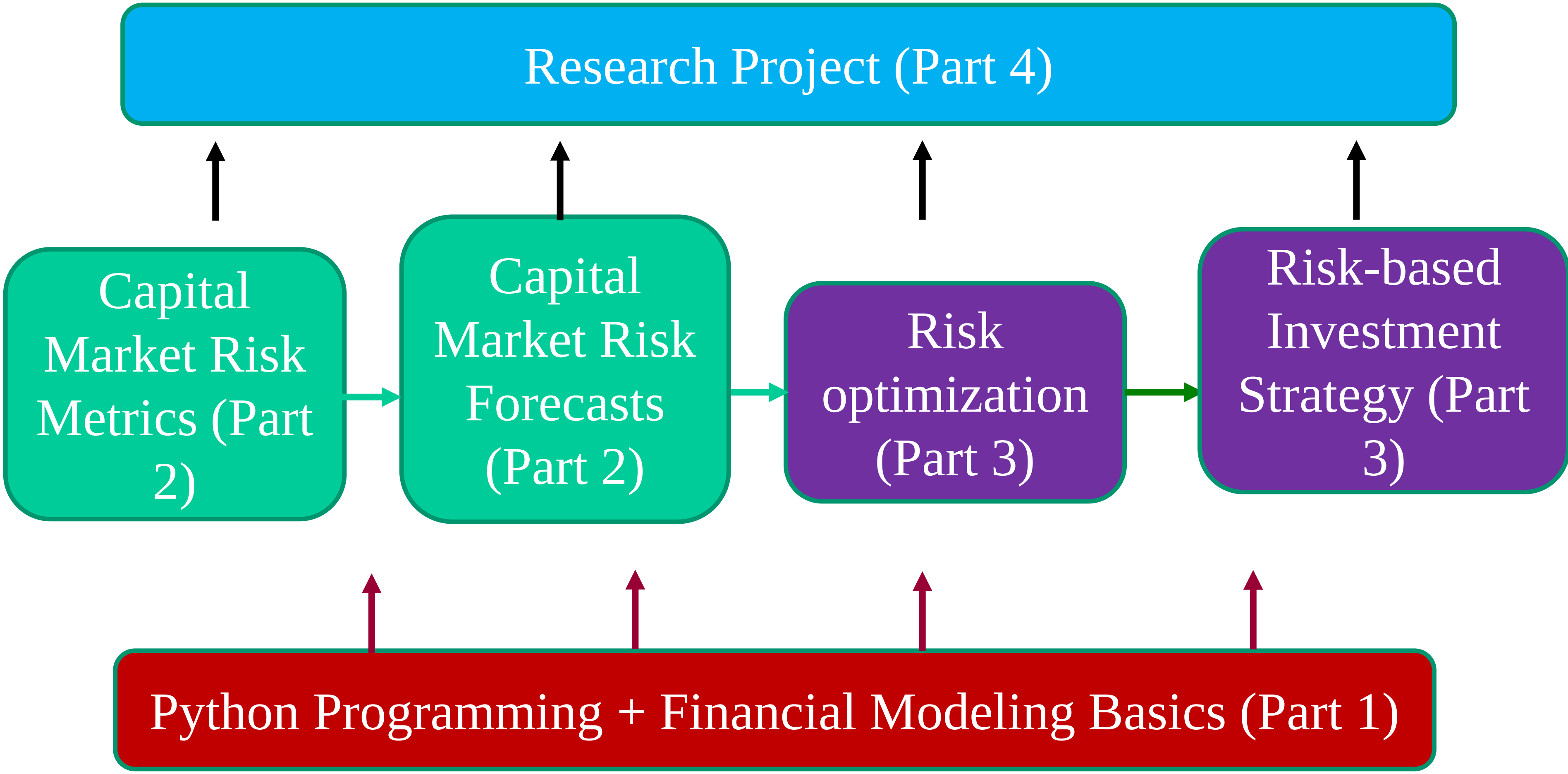
What it takes to do well in this course?

- Workload statement
  - students will spend a minimum of 9 hours of study time per week in preparation for class. I recommend 15 hours
- Class participation
  - Come to class and follow the instruction
  - Strongly encourage asking questions in class!
- Follow the lecture notes closely and study the notes and the sample codes together!
- Do the homework and project yourself!
  - Hard skill like programming is hard to master but it is why it is very valuable in the job market!

What will you learn in this course?

- Programming/Quantitative skills
- How to apply programming skills in real-life finance applications
  - Risk forecasting/management :
    - how to forecast risks for a portfolio of securities? Both in terms of standard volatility measures (Vol. covariance) and tail risks (VaR and CVaR)
  - Application:
    - Exchange traded funds construction and management
    - Risk-parity investment strategy
  - Real-life project based research how to apply some of the important quantitative analysis skills!

Course Structure



Research Project (Part 4)

Capital Market Risk Metrics (Part 2)

Capital Market Risk Forecasts (Part 2)

Risk optimization (Part 3)

Risk-based Investment Strategy (Part 3)

Python Programming + Financial Modeling Basics (Part 1)

Brandeis

Course Information - Schedule

Class Date	Text Chapters
Aug. 30, 2019 – L1	• Course Introduction/Python Installation • Introduction to Quantitative Finance Career • Python basics
Sep. 6, 2019 – L2	• Advanced Python Topics
Sep. 13, 2019 – L3	• Advanced Python Topics
Sep. 20, 2019 – L4	• Sourcing and handling Data • Stylized financial data analysis using Python
Sep. 27, 2019 – L5	• Value at Risk
Oct. 4, 2019 – L6	• Conditional Value at Risk (Expected Shortfall) + Mid-term Review
Oct. 11, 2019	• Mid-term
Oct. 18, 2019 – L7	• Modeling Volatility I
Oct. 25, 2019 – L8	• Modeling Volatility II
Nov. 1, 2019 – L9	• Practical application case Studies I
Nov. 8, 2019 – L10	• Practical application case Studies II
Nov. 15, 2019 – L11	• Back Testing + Conditional risk prediction
Nov. 22, 2019 – L12	• Research project presentation
Dec. 6, 2019 – L13	• Final Review

Homework Assignment – Lecture 1

- Homework is assigned weekly after each lecture is completed and due one week after

- Week1: Make sure you have Python installed correctly and get comfortable with the code editor Spyder

Introduction to Financial Engineering/ Quantitative Finance

- Review of quant finance career and job opportunities
- Knowledge and skills necessary for a career in quant finance: math, financial market and programming

Impact of AI/Data/Quant to Business Majors

MoF/FE/BA

- High demand but also high supply
- With popularization of quantitative investing, big data and machine learning, prospects is good

Corporate Finance

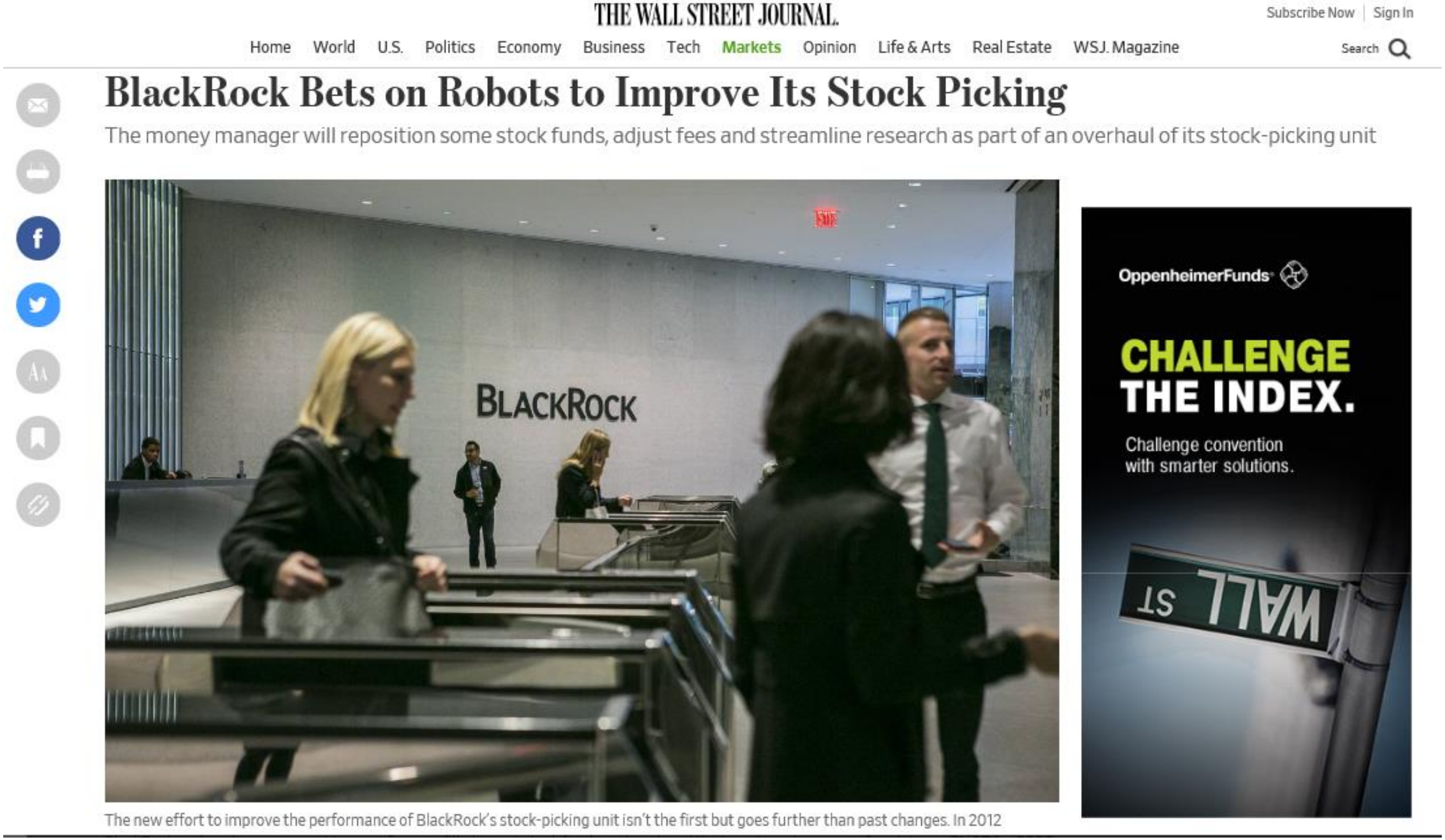
- Investment banking will continue to do well
- General corporate finance not so much

Accounting

- Ok for now but low-end services will be gradually replaced by machine learning and data driven automated services
- Only high-end services will be exempt from being replaced by

- The Industry is changing towards using machines!
- Learn some quantitative skills no matter what is your major

Why are quantitative skills important?



THE WALL STREET JOURNAL.

Subscribe Now | Sign In

Home World U.S. Politics Economy Business Tech Markets Opinion Life & Arts Real Estate WSJ. Magazine

Search

BlackRock Bets on Robots to Improve Its Stock Picking

The money manager will reposition some stock funds, adjust fees and streamline research as part of an overhaul of its stock-picking unit

The new effort to improve the performance of BlackRock's stock-picking unit isn't the first but goes further than past changes. In 2012

Quantitative finance career and job opportunities

- It is the field international students have an edge
- The coming decade is going to be a golden decade for people with quantitative skills
  - In the coming 10 years, having quantitative/programming skills are going to be one of the necessary conditions for becoming a good investment professional
  - Advancement in AI and big data technology is already changing the investment world
- The career can be rewarding both financially and professionally
  - Intellectually challenging
  - Well compensated

But there are also a lot of competitions!

- A lot of smart Scientists / Engineers switching career!
  - They most likely have better quantitative skills

- Almost every major US business school has a MoF/FE program!
  - A lot of supply of graduates of MoF/FE program
  - A lot of Chinese/Asian students studying MoF/FE

- The key to land a job: 1+1>2
  - You are competing both with MBAs and Engineering/Science Ph.ds
  - You need to have better quant/programming skills than the MBAs and better finance knowledge than the Engineering/Science Ph.Ds.

How do you become a good quantitative investor?

- Build up your hard skills!
  - Be very good at programming:
    - Python/R/Matlab/C++ etc.
    - Quantitative Modeling/Machine Learning
    - SQL /Data Science etc.
  - Understand how capital market work! – this is how you beat the scientist/engineers!
    - Stocks/Bonds/FX/Commodities how do they trade?
    - How to price them?
    - How to manage the risks?
    - Basics of constructing/rebalancing a portfolio
    - Read the financial press!

Intro to Python

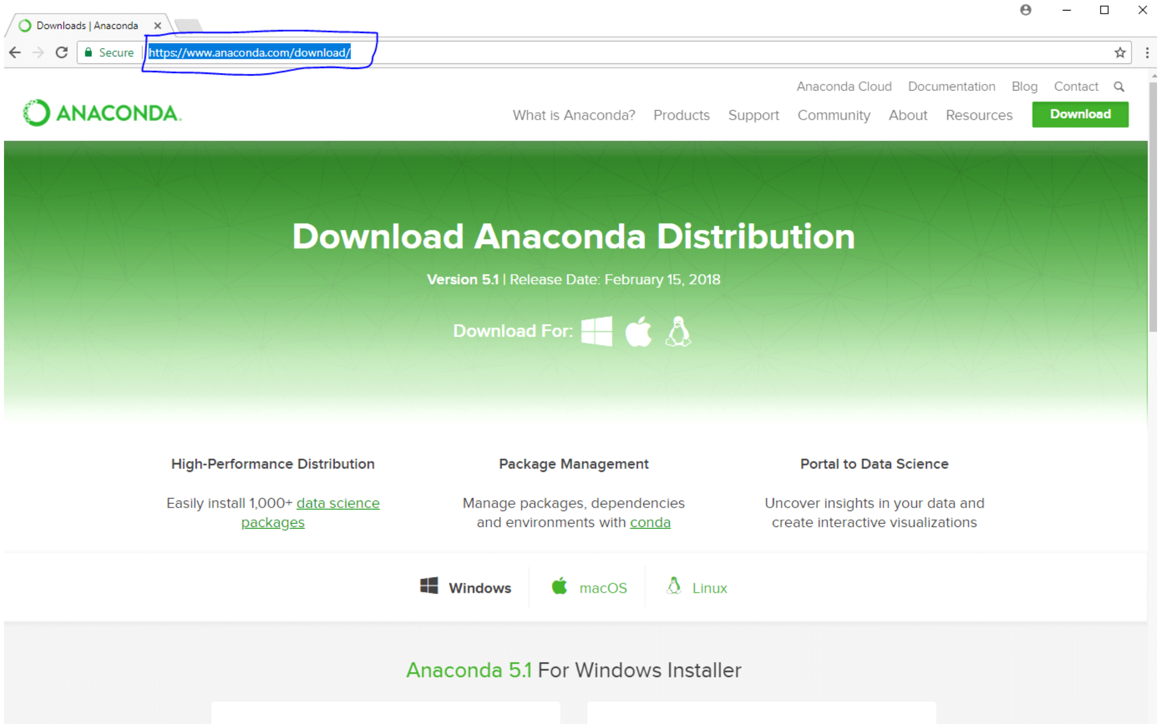
- Installation
- Python Code Editor
- Introduction to Python Programing

Introduction to Python

- Python is a high level language suitable for rapid development
  - Open source and free
- Strength:
  - Easy to use, a lot of built-in and shared capabilities/ functions
  - Very good for data science / AI/Machine learning ）
- Weakness:
  - Not as fast as machine level languages. Not strong in objected-oriented programming

Introduction to Python - installation

- Sent you the installation guide before the class
- Recommending using the Anaconda version and use the Spyder code editor. Please install the exact version as specified by the installation instruction
- Please make sure you have everything working this week if not already



Downloads | Anaconda

https://www.anaconda.com/download/

Download Anaconda Distribution

Version 5.1 | Release Date: February 15, 2018

Download For:

High-Performance Distribution

Easily install 1,000+ data science packages

Package Management

Manage packages, dependencies and environments with conda

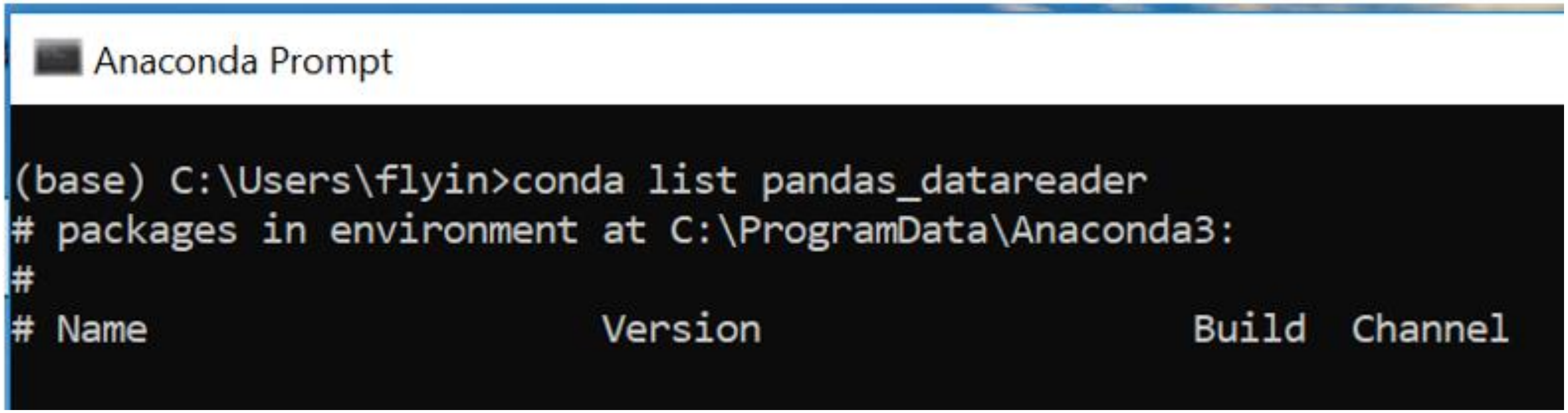
Portal to Data Science

Uncover insights in your data and create interactive visualizations

Anaconda 5.1 For Windows Installer

Package installation

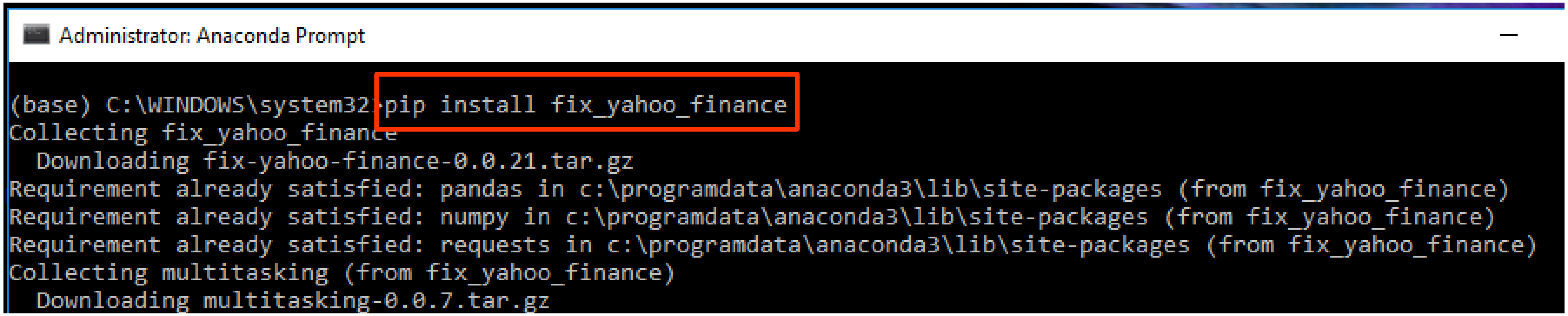
- Use *"conda list packagename"* first to check whether Anaconda has the package included already for quick installation
    - Always use 'conda install packagename' before trying the 'pip' approach for the ananconda environment
    - Try out 'conda list pandas_datareader' yourself under Conda prompt
- You may need to run the Anaconada prompt as system administrator if the folder you want to have the software installed needs such permission



```
(base) C:\Users\flyin>conda list pandas_datareader
# packages in environment at C:\ProgramData\Anaconda3:
#
# Name                    Version                   Build  Channel
```

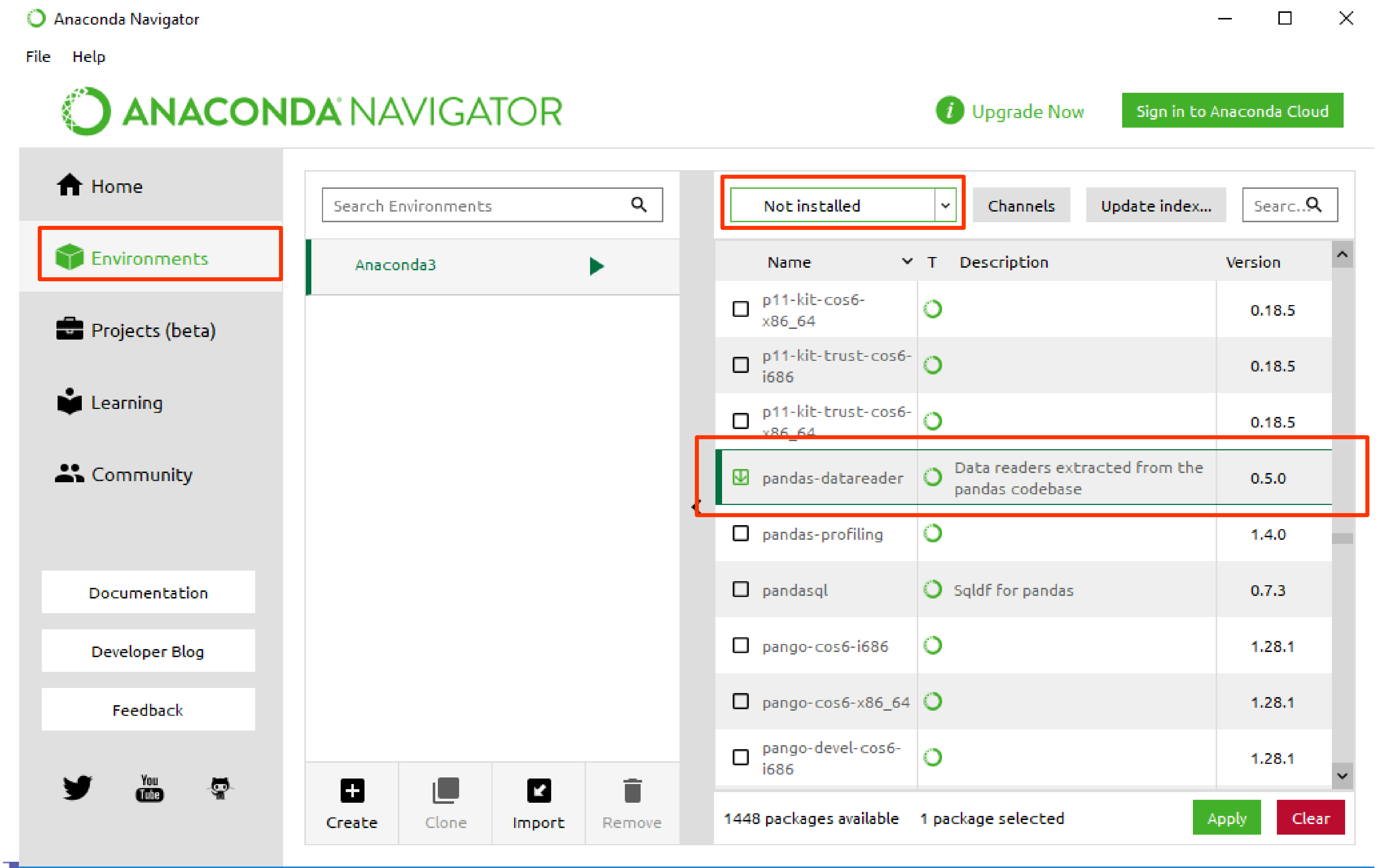
Using Anaconda Navigator to Install Package Pandas-datareader

- Try install the following packages we will need later:
  - pip install pandas_datareader (note use pip here instead of conda to install because pandas_datareader is not in the pandas environment)
  - pip install fix_yahoo_finance
  - pip install arch
- Please note on a Mac machine, the conda and pip command could be accessed by opening up a ‘terminal’



```
(base) C:\WINDOWS\system32>pip install fix_yahoo_finance
Collecting fix_yahoo_finance
  Downloading fix-yahoo-finance-0.0.21.tar.gz
Requirement already satisfied: pandas in c:\programdata\anaconda3\lib\site-packages (from fix_yahoo_finance)
Requirement already satisfied: numpy in c:\programdata\anaconda3\lib\site-packages (from fix_yahoo_finance)
Requirement already satisfied: requests in c:\programdata\anaconda3\lib\site-packages (from fix_yahoo_finance)
Collecting multitasking (from fix_yahoo_finance)
  Downloading multitasking-0.0.7.tar.gz
```

Using Anaconda Navigator to Install Package Pandas-datareader



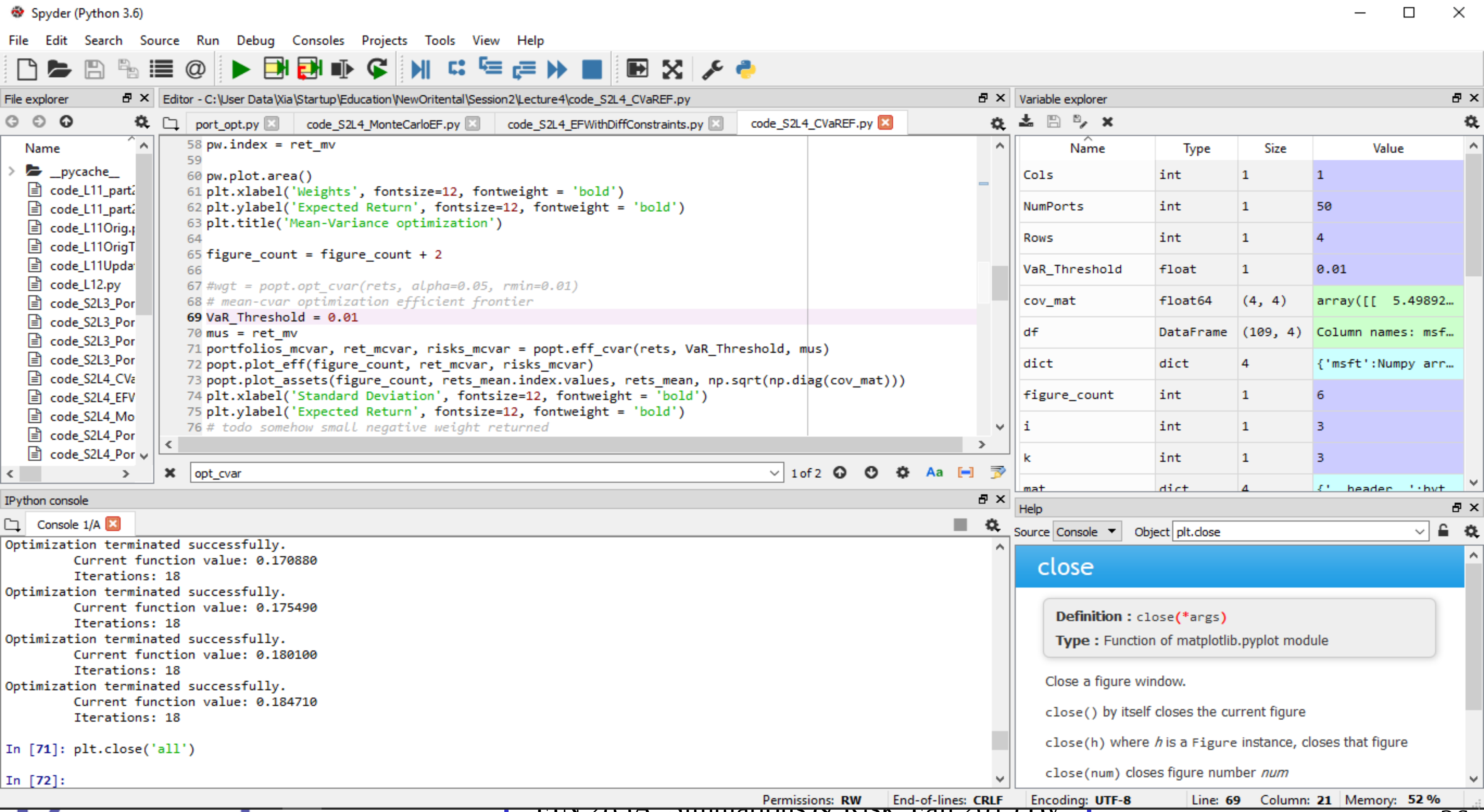
Not installed

	Name	T	Description	Version
□	p11-kit-cos6-x86_64	○		0.18.5
□	p11-kit-trust-cos6-i686	○		0.18.5
□	p11-kit-trust-cos6-x86_64	○		0.18.5
☑	pandas-datareader	○	Data readers extracted from the pandas codebase	0.5.0
□	pandas-profiling	○		1.4.0
□	pandasql	○	Sqldf for pandas	0.7.3
□	pango-cos6-i686	○		1.28.1
□	pango-cos6-x86_64	○		1.28.1
□	pango-devel-cos6-i686	○		1.28.1

1448 packages available   1 package selected

Introduction to Python editor Spyder – User Interface

- **Spyder is one of the IDEs (Integrated development environment) to code/debug your python codes**



```
pw.index = ret_mv

pw.plot.area()
plt.xlabel('Weights', fontsize=12, fontweight = 'bold')
plt.ylabel('Expected Return', fontsize=12, fontweight = 'bold')
plt.title('Mean-Variance optimization')

figure_count = figure_count + 2

#wgt = popt.opt_cvar(rets, alpha=0.05, rmin=0.01)
# mean-cvar optimization efficient frontier
VaR_Threshold = 0.01
mus = ret_mv
portfolios_mcvar, ret_mcvar, risks_mcvar = popt.eff_cvar(rets, VaR_Threshold, mus)
popt.plot_eff(figure_count, ret_mcvar, risks_mcvar)
popt.plot_assets(figure_count, rets_mean, rets_mean.index.values, np.sqrt(np.diag(cov_mat)))
plt.xlabel('Standard Deviation', fontsize=12, fontweight = 'bold')
plt.ylabel('Expected Return', fontsize=12, fontweight = 'bold')
# todo somehow small negative weight returned
```

Name	Type	Size	Value
Cols	int	1	1
NumPorts	int	1	50
Rows	int	1	4
VaR_Threshold	float	1	0.01
cov_mat	float64	(4, 4)	array([[5.49892…
df	DataFrame	(109, 4)	Column names: msf…
dict	dict	4	{'msft':Numpy arr…
figure_count	int	1	6
i	int	1	3
k	int	1	3
mat	dict	4	{'__header__':b'…

```
Optimization terminated successfully.
         Current function value: 0.170880
         Iterations: 18
Optimization terminated successfully.
         Current function value: 0.175490
         Iterations: 18
Optimization terminated successfully.
         Current function value: 0.180100
         Iterations: 18
Optimization terminated successfully.
         Current function value: 0.184710
         Iterations: 18

In [71]: plt.close('all')

In [72]:
```

close

Definition : close(*args)

Type : Function of matplotlib.pyplot module

Close a figure window.

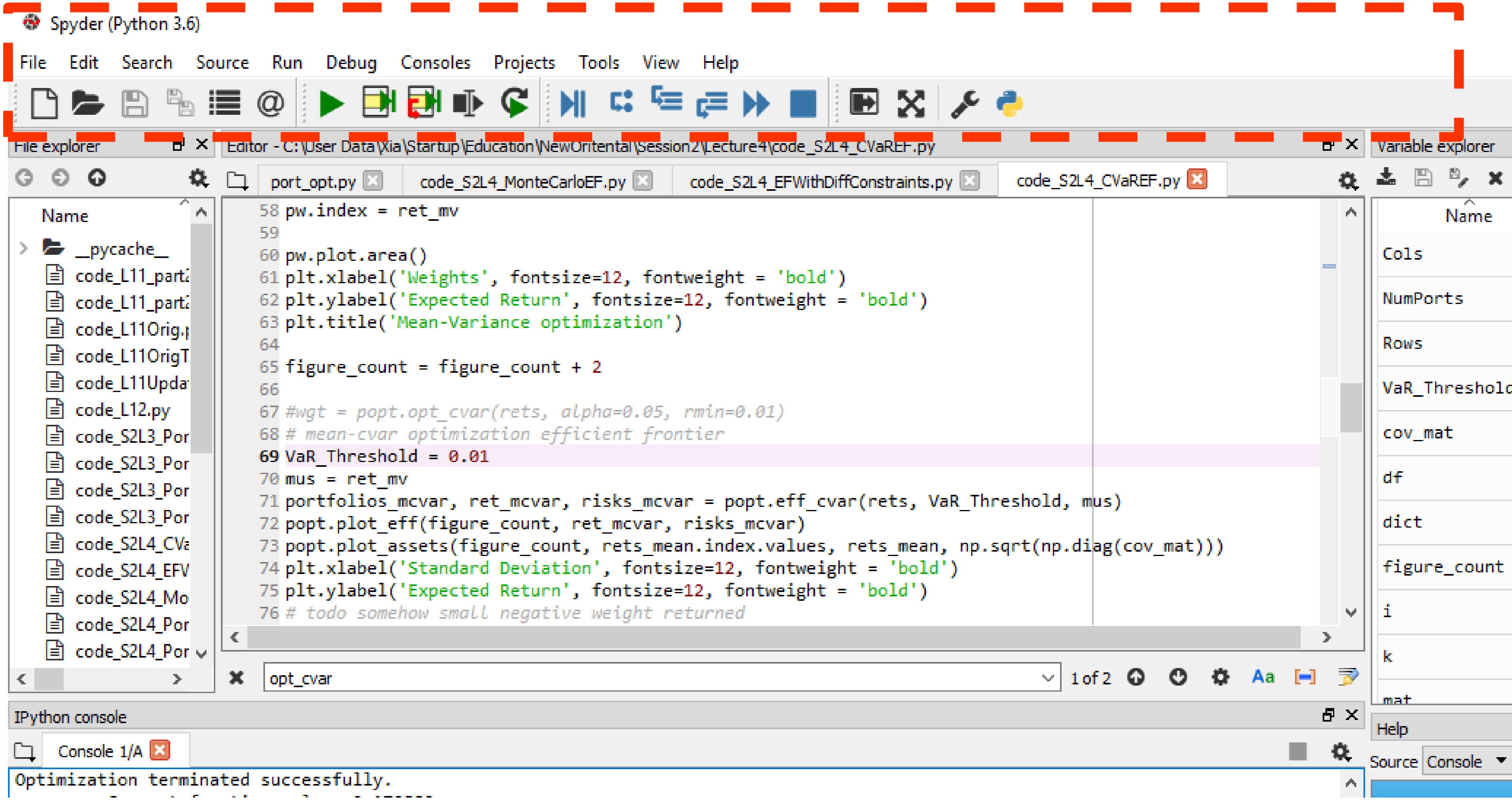
close() by itself closes the current figure

close(h) where *h* is a Figure instance, closes that figure

close(num) closes figure number *num*

Permissions: **RW** End-of-lines: **CRLF** Encoding: **UTF-8** Line: **69** Column: **21** Memory: **52 %**

User Interface – File Menu

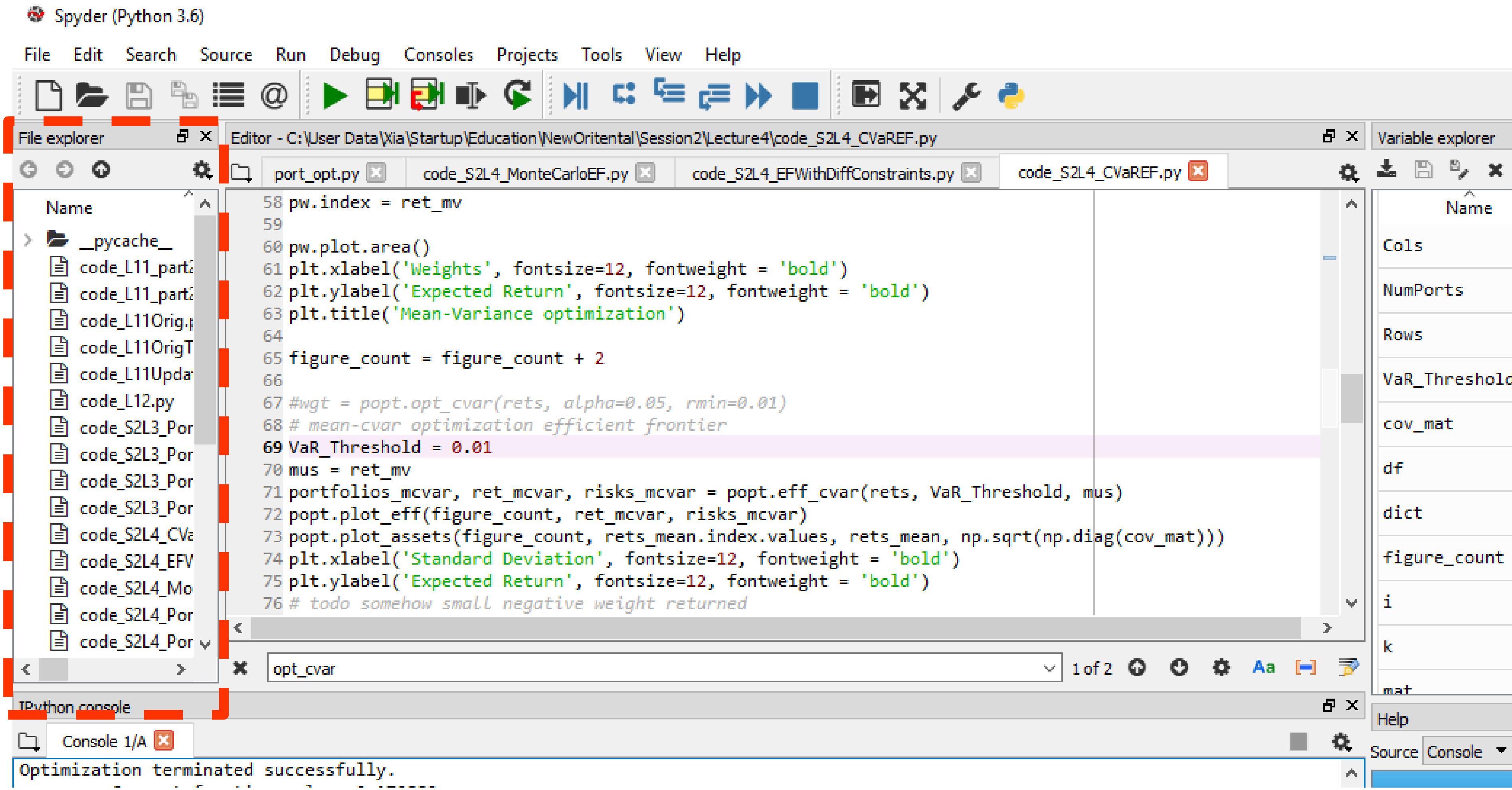


```python
pw.index = ret_mv

pw.plot.area()
plt.xlabel('Weights', fontsize=12, fontweight = 'bold')
plt.ylabel('Expected Return', fontsize=12, fontweight = 'bold')
plt.title('Mean-Variance optimization')

figure_count = figure_count + 2

#wgt = popt.opt_cvar(rets, alpha=0.05, rmin=0.01)
# mean-cvar optimization efficient frontier
VaR_Threshold = 0.01
mus = ret_mv
portfolios_mcvar, ret_mcvar, risks_mcvar = popt.eff_cvar(rets, VaR_Threshold, mus)
popt.plot_eff(figure_count, ret_mcvar, risks_mcvar)
popt.plot_assets(figure_count, rets_mean.index.values, rets_mean, np.sqrt(np.diag(cov_mat)))
plt.xlabel('Standard Deviation', fontsize=12, fontweight = 'bold')
plt.ylabel('Expected Return', fontsize=12, fontweight = 'bold')
# todo somehow small negative weight returned
```

```
Optimization terminated successfully.
```

- **Use it to open/save/create files, run/debug a script, set preference, etc.**

User Interface – File explorer


Spyder (Python 3.6)
File Edit Search Source Run Debug Consoles Projects Tools View Help
File explorer
Editor - C:\User Data\Xia\Startup\Education\NewOrientatal\Session2\Lecture4\code_S2L4_CVaREF.py
Variable explorer
port_opt.py
code_S2L4_MonteCarloEF.py
code_S2L4_EFWithDiffConstraints.py
code_S2L4_CVaREF.py
Name
IPython console
Console 1/A
Help
Source Console
1 of 2
Aa

```
Name
__pycache__
code_L11_part:
code_L11_part:
code_L11Orig.|
code_L11OrigT
code_L11Upda
code_L12.py
code_S2L3_Por
code_S2L3_Por
code_S2L3_Por
code_S2L3_Por
code_S2L4_CVa
code_S2L4_EFV
code_S2L4_Mo
code_S2L4_Por
code_S2L4_Por
```

```python
pw.index = ret_mv

pw.plot.area()
plt.xlabel('Weights', fontsize=12, fontweight = 'bold')
plt.ylabel('Expected Return', fontsize=12, fontweight = 'bold')
plt.title('Mean-Variance optimization')

figure_count = figure_count + 2

#wgt = popt.opt_cvar(rets, alpha=0.05, rmin=0.01)
# mean-cvar optimization efficient frontier
VaR_Threshold = 0.01
mus = ret_mv
portfolios_mcvar, ret_mcvar, risks_mcvar = popt.eff_cvar(rets, VaR_Threshold, mus)
popt.plot_eff(figure_count, ret_mcvar, risks_mcvar)
popt.plot_assets(figure_count, rets_mean.index.values, rets_mean, np.sqrt(np.diag(cov_mat)))
plt.xlabel('Standard Deviation', fontsize=12, fontweight = 'bold')
plt.ylabel('Expected Return', fontsize=12, fontweight = 'bold')
# todo somehow small negative weight returned
```

opt_cvar

```
Name
Cols
NumPorts
Rows
VaR_Threshold
cov_mat
df
dict
figure_count
i
k
mat
```

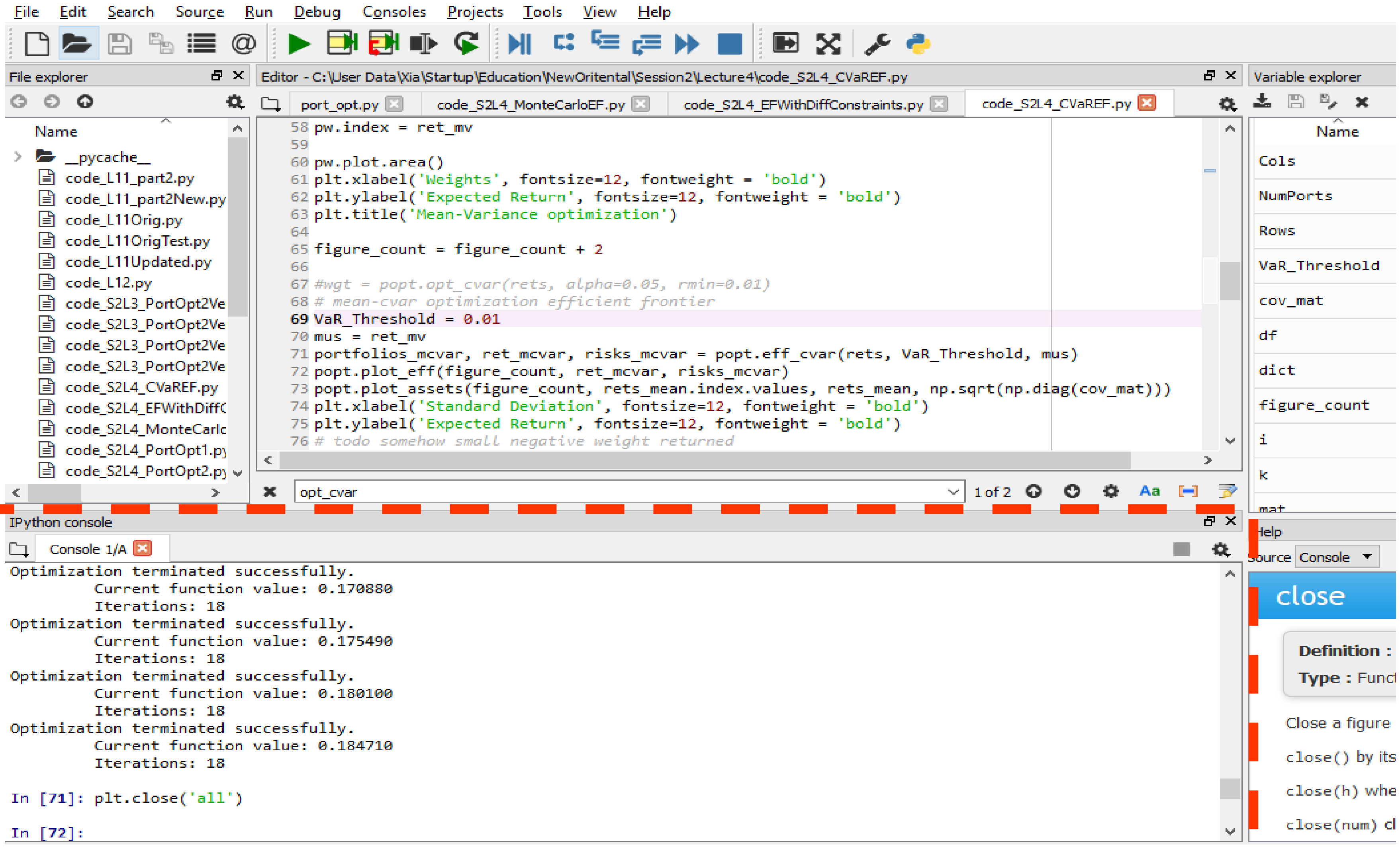
```
Optimization terminated successfully.
```

- **Shows the current folder from which Python is operating**
- **Shows the content of the current folder**

Brandeis

FIN 285A: Simulations & Risk, Fall 2019 By Steve Xia

User Interface – IPython Console

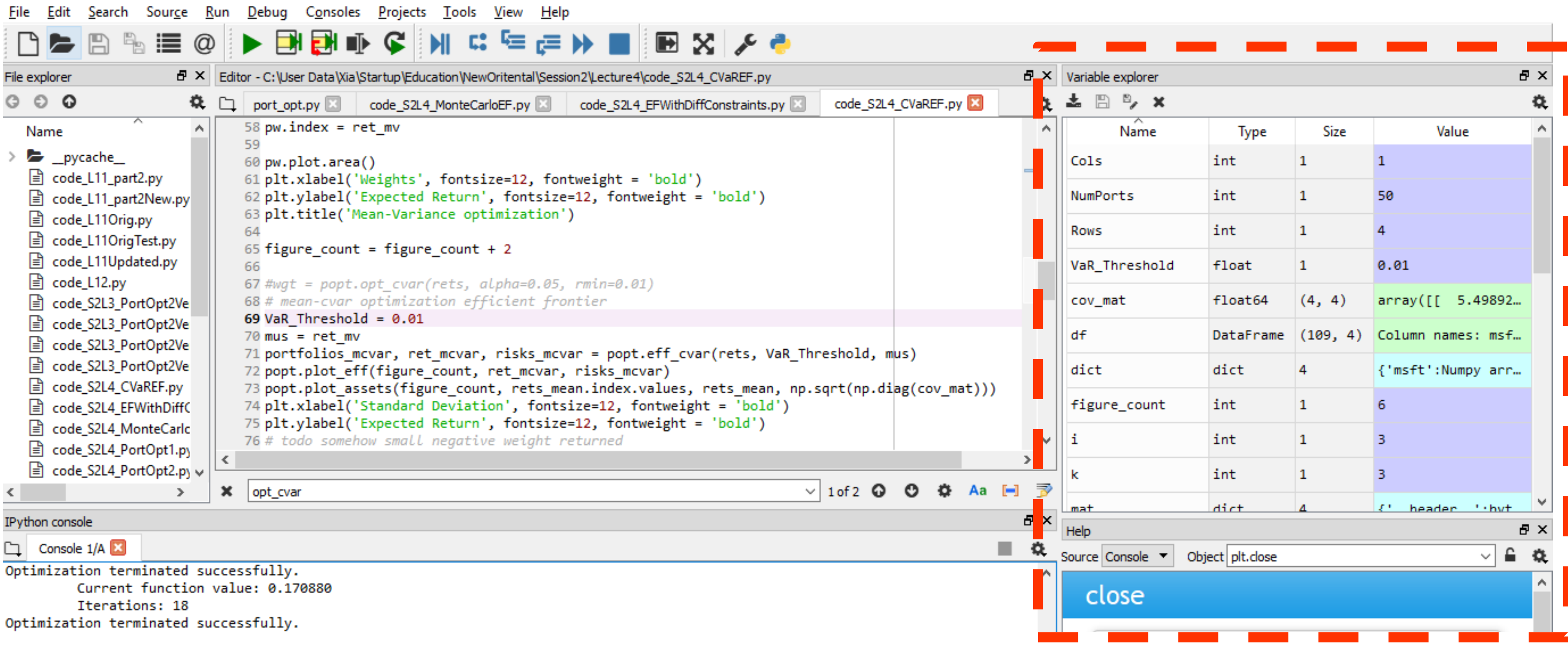


```
pw.index = ret_mv

pw.plot.area()
plt.xlabel('Weights', fontsize=12, fontweight = 'bold')
plt.ylabel('Expected Return', fontsize=12, fontweight = 'bold')
plt.title('Mean-Variance optimization')

figure_count = figure_count + 2

#wgt = popt.opt_cvar(rets, alpha=0.05, rmin=0.01)
# mean-cvar optimization efficient frontier
VaR_Threshold = 0.01
mus = ret_mv
portfolios_mcvar, ret_mcvar, risks_mcvar = popt.eff_cvar(rets, VaR_Threshold, mus)
popt.plot_eff(figure_count, ret_mcvar, risks_mcvar)
popt.plot_assets(figure_count, rets_mean.index.values, rets_mean, np.sqrt(np.diag(cov_mat)))
plt.xlabel('Standard Deviation', fontsize=12, fontweight = 'bold')
plt.ylabel('Expected Return', fontsize=12, fontweight = 'bold')
# todo somehow small negative weight returned
```

```
Optimization terminated successfully.
         Current function value: 0.170880
         Iterations: 18
Optimization terminated successfully.
         Current function value: 0.175490
         Iterations: 18
Optimization terminated successfully.
         Current function value: 0.180100
         Iterations: 18
Optimization terminated successfully.
         Current function value: 0.184710
         Iterations: 18

In [71]: plt.close('all')

In [72]:
```

- Shows all the command you typed in for Python to execute
- Shows all the output you want Python to give back to you

User Interface – Variable Explorer



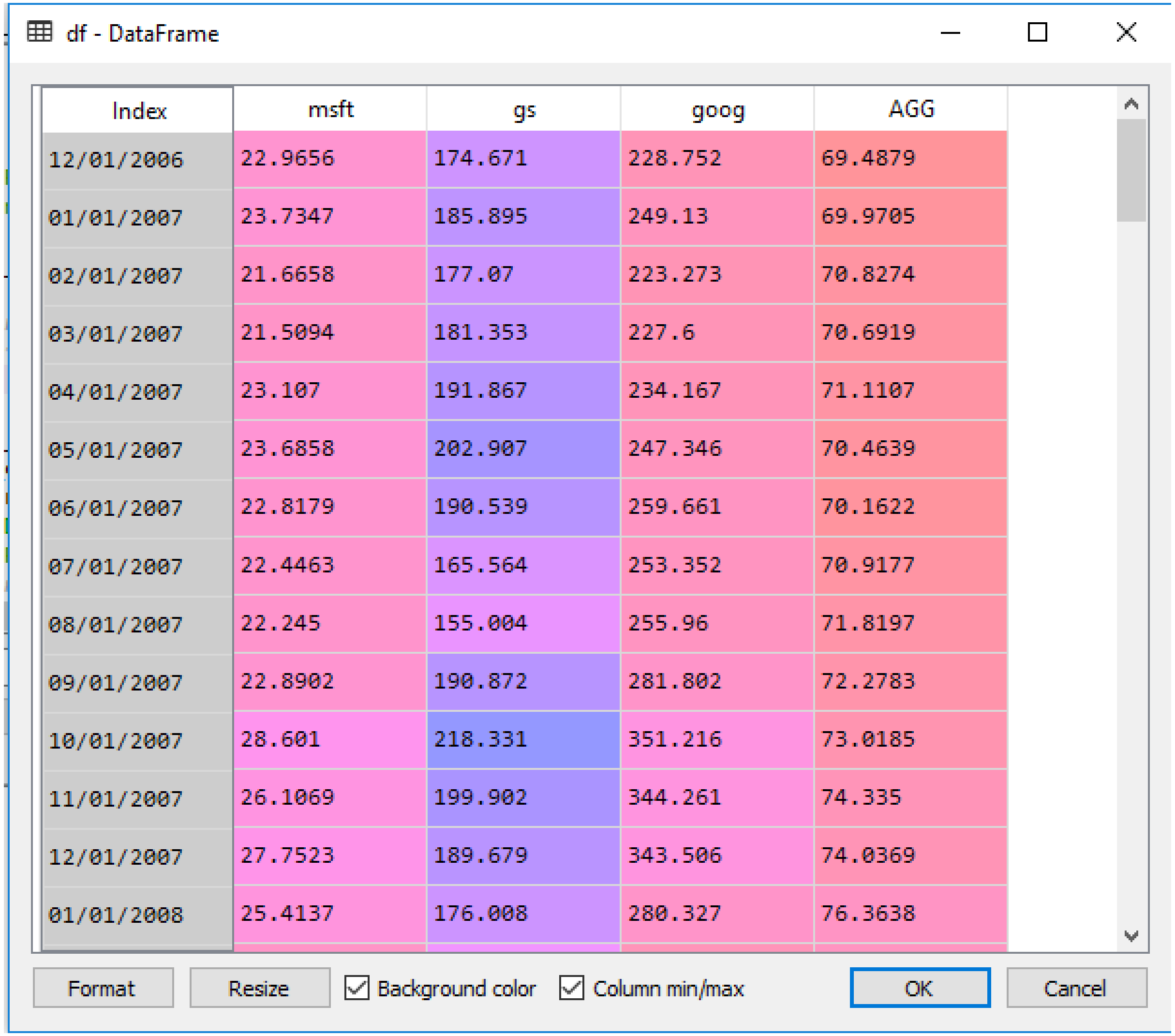
```python
pw.index = ret_mv

pw.plot.area()
plt.xlabel('Weights', fontsize=12, fontweight = 'bold')
plt.ylabel('Expected Return', fontsize=12, fontweight = 'bold')
plt.title('Mean-Variance optimization')

figure_count = figure_count + 2

#wgt = popt.opt_cvar(rets, alpha=0.05, rmin=0.01)
# mean-cvar optimization efficient frontier
VaR_Threshold = 0.01
mus = ret_mv
portfolios_mcvar, ret_mcvar, risks_mcvar = popt.eff_cvar(rets, VaR_Threshold, mus)
popt.plot_eff(figure_count, ret_mcvar, risks_mcvar)
popt.plot_assets(figure_count, rets_mean.index.values, rets_mean, np.sqrt(np.diag(cov_mat)))
plt.xlabel('Standard Deviation', fontsize=12, fontweight = 'bold')
plt.ylabel('Expected Return', fontsize=12, fontweight = 'bold')
# todo somehow small negative weight returned
```

Name	Type	Size	Value
Cols	int	1	1
NumPorts	int	1	50
Rows	int	1	4
VaR_Threshold	float	1	0.01
cov_mat	float64	(4, 4)	array([[5.49892…
df	DataFrame	(109, 4)	Column names: msf…
dict	dict	4	{'msft':Numpy arr…
figure_count	int	1	6
i	int	1	3
k	int	1	3
mat	dict	4	{'__header__':byt

```
Optimization terminated successfully.
         Current function value: 0.170880
         Iterations: 18
Optimization terminated successfully.
```

- **Shows all the Variables/Data you have created/imported**
- **Double click to open up variables/data to view them**

User Interface – Variable Viewer



Index	msft	gs	goog	AGG
12/01/2006	22.9656	174.671	228.752	69.4879
01/01/2007	23.7347	185.895	249.13	69.9705
02/01/2007	21.6658	177.07	223.273	70.8274
03/01/2007	21.5094	181.353	227.6	70.6919
04/01/2007	23.107	191.867	234.167	71.1107
05/01/2007	23.6858	202.907	247.346	70.4639
06/01/2007	22.8179	190.539	259.661	70.1622
07/01/2007	22.4463	165.564	253.352	70.9177
08/01/2007	22.245	155.004	255.96	71.8197
09/01/2007	22.8902	190.872	281.802	72.2783
10/01/2007	28.601	218.331	351.216	73.0185
11/01/2007	26.1069	199.902	344.261	74.335
12/01/2007	27.7523	189.679	343.506	74.0369
01/01/2008	25.4137	176.008	280.327	76.3638

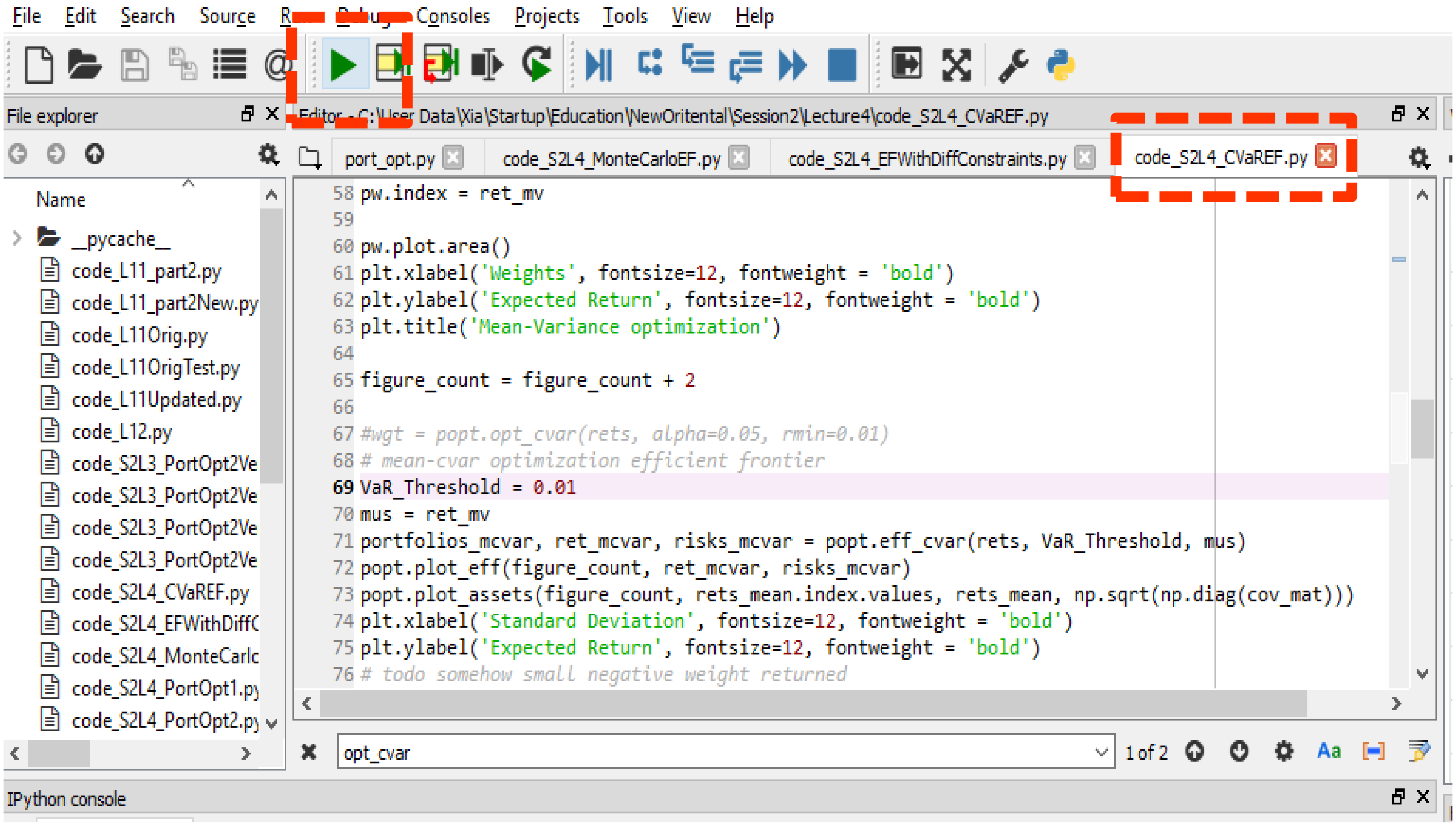
☑ Background color ☑ Column min/max

- **Shows specific variable’s content**

Brandeis

Run a Python script file

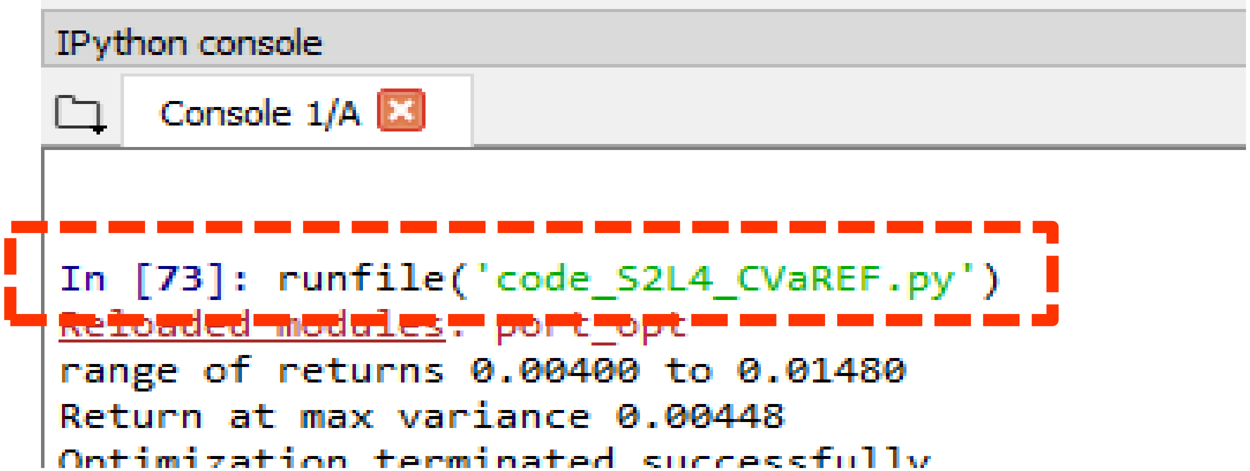
- **Save all of the commands in a .py (script) file and run all the commands in the file at once**



```
pw.index = ret_mv

pw.plot.area()
plt.xlabel('Weights', fontsize=12, fontweight = 'bold')
plt.ylabel('Expected Return', fontsize=12, fontweight = 'bold')
plt.title('Mean-Variance optimization')

figure_count = figure_count + 2

#wgt = popt.opt_cvar(rets, alpha=0.05, rmin=0.01)
# mean-cvar optimization efficient frontier
VaR_Threshold = 0.01
mus = ret_mv
portfolios_mcvar, ret_mcvar, risks_mcvar = popt.eff_cvar(rets, VaR_Threshold, mus)
popt.plot_eff(figure_count, ret_mcvar, risks_mcvar)
popt.plot_assets(figure_count, rets_mean.index.values, rets_mean, np.sqrt(np.diag(cov_mat)))
plt.xlabel('Standard Deviation', fontsize=12, fontweight = 'bold')
plt.ylabel('Expected Return', fontsize=12, fontweight = 'bold')
# todo somehow small negative weight returned
```

Run a Python script file

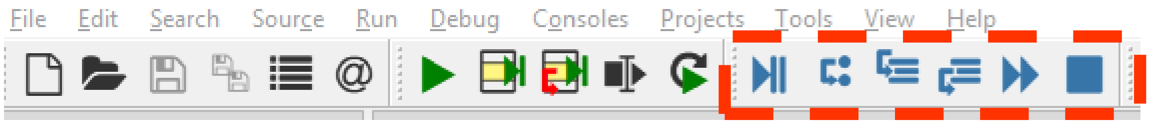
- **Through the IPython Console window by type in the commands like below**

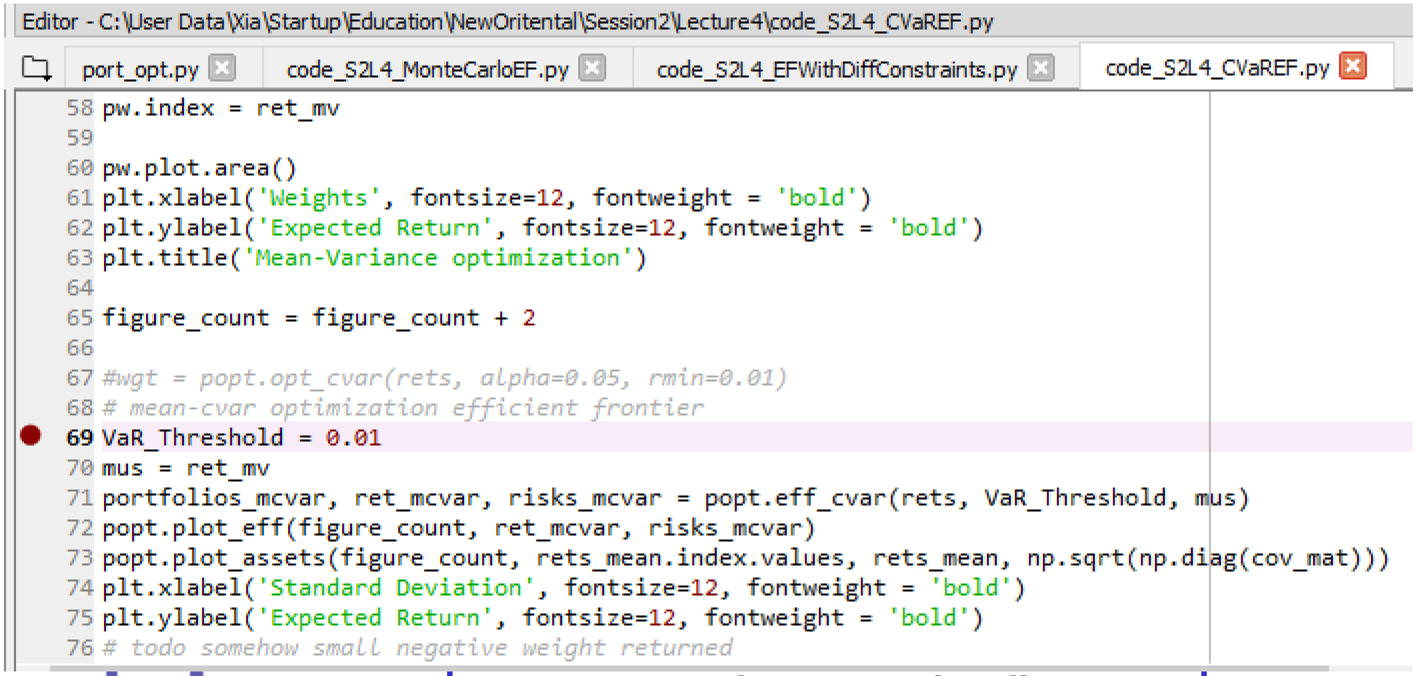


```
In [73]: runfile('code_S2L4_CVaREF.py')
Reloaded modules: port_opt
range of returns 0.00400 to 0.01480
Return at max variance 0.00448
Optimization terminated successfully
```

Debugging

- The Spyder IDE has ok debugging capability
- The key basic debugging function is done by setting break points in your code so you can step through certain parts of your code to find out what is happening




Editor - C:\User Data\Xia\Startup\Education\NewOritental\Session2\Lecture4\code_S2L4_CVaREF.py
port_opt.py
code_S2L4_MonteCarloEF.py
code_S2L4_EFWithDiffConstraints.py
code_S2L4_CVaREF.py

```
pw.index = ret_mv

pw.plot.area()
plt.xlabel('Weights', fontsize=12, fontweight = 'bold')
plt.ylabel('Expected Return', fontsize=12, fontweight = 'bold')
plt.title('Mean-Variance optimization')

figure_count = figure_count + 2

#wgt = popt.opt_cvar(rets, alpha=0.05, rmin=0.01)
# mean-cvar optimization efficient frontier
VaR_Threshold = 0.01
mus = ret_mv
portfolios_mcvar, ret_mcvar, risks_mcvar = popt.eff_cvar(rets, VaR_Threshold, mus)
popt.plot_eff(figure_count, ret_mcvar, risks_mcvar)
popt.plot_assets(figure_count, rets_mean.index.values, rets_mean, np.sqrt(np.diag(cov_mat)))
plt.xlabel('Standard Deviation', fontsize=12, fontweight = 'bold')
plt.ylabel('Expected Return', fontsize=12, fontweight = 'bold')
# todo somehow small negative weight returned
```

Extra Resources for Learning Python

- https://www.w3schools.com/python/default.asp
- https://developers.google.com/edu/python/introduction
- https://docs.python.org/3/tutorial/index.html
- https://lectures.quantecon.org/py/index_learning_python.html
- https://pandas.pydata.org/pandas-docs/stable/dsintro.html
- http://www.scipy-lectures.org/intro/numpy/numpy.html

- Book
  - Python for Data Analysis, Wes McKinney, O’Reilly Media, Inc., 2018.