**DQ Project**

DQ means Discretionary Quant. It is an experiment/system designed to combine discretionary and quantitative approaches in terms of investing.

**Building Blocks**

1. **Discretionary Leg (D)** – The opinions expressed as a probability density function (pdf) of stock prices, with conviction scores assigned. The core value is from the fundamental or macro visions, not the code itself
2. **Quant Leg (Q)** – The trading signals based on price/volume, or other available data, this part should be really “low touch or no tough”. The core value is in the code, i.e. alphas.
3. **Portfolio Construction (P)** – Use mean-var/Black-Litterman, risk parity, or others methods
4. **Back-testing (B)** – Run historical simulation the performance for quant strategies and portfolio constructions methods
5. **Risk Management (R)** – Calculate risk measures (e.g. Var) and maybe factor exposures
6. **Get Data (G)** – Get price and/or alternative data (if needed) from Quandl API
7. **Trade (T)** – IB API (will code it up later as per WQ compliance requirements)
8. **Analytics (A)** – Analyze performance and other analysis not covered by 1-7

**Timeline**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **BBlocks** | **Task Description** | **File name** | **ETA** | **Release History** |
| **G** | Use Quandl API to get daily market data, store the data in DB and query from it | GetData.py  ExecQuery.py | 20180401 | v1.0 20180401  v2.0 20180504 |
| **D** | Write the UI to interactively draw the pdf of prices, and show the performance | HelpGUI.py  RunGUI.py  Description.html | 20180505 | V1.0 20180505 |
| **P** | construction the portfolio based on the portfolio theories and from D & Q | ConstructPortfolio.py | 20180505 |  |
| R |  | RiskReport.py | 20180512 |  |
| Q |  | Strategy.py | 20180513 |  |
| B |  | BackTest.py | 20180519 |  |
| T |  | Trade.py | TBD |  |
| A |  | TBD | TBD |  |

**Work Log**

1. **20180401**
2. Reinstalled Anaconda to have Python3.6 Env
3. Researched and configed settings and data
4. Loaded the historical price/volume data into MySQL

- GetData.py

- ExecQuery.py

1. **20180402 – 20180430**
2. Uninstalled Anaconda
3. Installed Python separately and Spyder, and installed the packages needed
4. Updated the code of GetPxData
5. **20180501 – 20180506**
6. Built the first version of the GUI
7. Built the first version of the Portfolio Construction part
8. **20180901 – 20181030**
9. Added the part to load crypto daily data (10 pairs)
10. Built another GUI based on beta distribution
11. Other small modifications

**To do and Discussions**

1. **长期短期会有不同的预期**
2. **拟合用GGM**
3. **Prob vs. Return instead of price, using the closing px of the day as the benchmark price**

**Others**

1. **正式运营固定成本**

|  |  |  |  |
| --- | --- | --- | --- |
| **Product** | **Cost/Month** | **Other Costs** | **Comments** |
| **Gitlab** | **$7** | **0** |  |
| **Tableau** | **$70** | **0** | **So expensive** |
| **IB** | **$10** | **0** |  |
| 阿里云 | $83 | 0 | So expensive |
| **AlphaVantage** | **$50** | **0** |  |
| **Quandl** | **$1300/Y** | **0** | **….** |
| 共计 | 220-350/M | 0 | TBD |