***“Risk\_Analytic\_App.py*” – Documentation**

# Description:

This file contains a Risk\_Advisor class that provide risk metrics calculation. The Risk Advisor is designed to be able to work on both a single time point portfolio holding and a dynamic portfolio that evolves over time. This design ensures Risk Advisor smoothly connects to other parts in PARIS system including Back\_Testor, Live\_Trading\_Assistant and actual historical trades database.

# Dependencies:

* scipy
* numpy
* pandas
* matplotlib
* itertools
* DB\_Operator

# Class Function Descriptions:

|  |  |  |  |
| --- | --- | --- | --- |
| **Function Name** | **Inputs** | **Output** | **Description** |
| Constructor | Portfolio  (pd.df/dict),  \_Price (pd.df),  \_Volume(pd.df),  Graphic (bool),  Benchmark (list),  Factors (list),  Holding\_info (dict),  Test\_day (str),  frequency (str),  Volume\_Multiplier (dict) | None | Risk\_Advisor can be constructed in different ways by providing different arguments:   * When no argument is provided at all, it reads historical trades and price/volume info from databases and conduct historical analysis * When a single day’s holding\_info is provided, user can either specify date of holding and that day’s information will be read from database; or user can choose to input price/volume to override historical data. This design is to supports hypothetical position in live trading assistant. * When a time-series portfolio holding information is provided it will be constructed to reflect how these risk metrics evolved over time. i.e when the Risk Advisor is constructed in Back\_Testor   Graphic argument decides whether to show a graphic output or statistics in a dataframe.  Volume Multiplier is used to turn lots into units of underlying assets as streaming hear-backs from MT4 (maybe other brokers too) are in lots instead of units. |
| gross\_exp | None | Print (str)  or  line plot (plt)  or  DataFrame  (pd.df) | On single time point (i.e. some past day or real time advising): this function prints a sentence describing gross exposure.  On time series advising: this function plots how gross exposure evolved over time (gross exposure as percentage). |
| net\_exp | None | Print (str)  or  line plot (plt)  or  DataFrame  (pd.df) | On single time point (i.e. some past day or real time advising): this function prints a sentence describing net exposure.  On time series holding: this function plots how net exposure evolved over time (net exposure as percentage). |
| corr\_table | None | pd.df | This function calculates the correlation between assets invested in the portfolio and returns a dataframe of all the correlation information |
| corr\_history | window (int) | line plot (plt)  or  DataFrame (pd.df) | This function calculates the correlation between all pairs of investments held and plotted them alongside the Benchmark return, to show how they have evolved under different market conditions. |
| port\_vol\_monetary | None | Print (str)  or  line plot (plt)  or  DataFrame  (pd.df) | This function calculates the monetary volatility of the portfolio holding based on holding value and historical variance covariance matrix.  Then it returns a sentence describing this information if assessing single time point, or show how monetary vol evolved over time if assessing a time series portfolio. |
| port\_vol\_percentage | None | Print (str)  or  line plot (plt)  or  DataFrame  (pd.df) | This function calculates the percent volatility of the portfolio holding based on holding weights and historical variance covariance matrix.  Then it returns a sentence describing this information if assessing single time point or show how percentage vol evolved over time if assessing a time series portfolio. |
| Calculate\_VaR | Percentile (int) | Print (str)  or  line plot (plt)  or  DataFrame  (pd.df) | This function calculates value at risk of the portfolio at the percentile specified. Percentile has a default value of 95. Assuming return follows historical covariance matrix and normal distribution.  Then it returns a sentence describing this information if assessing single time point or show how VaR evolved over time if assessing a time series portfolio. |
| Empirical\_VaR | Percentile (int) | Print (str) | This function calculates value at risk of the portfolio using actual historical performance at the percentile. i.e. it checks the 95% actual worst day if we had held this portfolio over time. Percentile has a default value of 95.  This function only works when we deal with a single day’s holding. |
| Get\_Top\_Positions | Number (int) | Print (str)  or  line plot (plt)  or  DataFrame  (pd.df) | This function ranks the portfolio holdings’ absolute values and return top N, as specified by Number input, assets with their weights.  Then it returns a sentence describing this information if assessing single time point or line plot to show how top positions evolved over time if assessing a time series portfolio. |
| Expected\_Shortfall | Percentile (int) | Print (str)  or  line plot (plt)  or  DataFrame  (pd.df) | This function calculates Expected Shortfall of the portfolio at the percentile specified. Expected shortfall means the average loss if a tail event happens (exceeding the percentile specified). Percentile has a default value of 95. Assuming return follows historical covariance matrix and normal distribution.  Then it returns a sentence describing this information if assessing single time point or show how Expected Shortfall evolved over time if assessing a time series portfolio. |
| Liquidity | days (int) | Print (str)  or  line plot (plt)  or  DataFrame  (pd.df) | This function looks at the past n (specified by days input) bars trading volume and calculate our portfolio holding as a percentage of per bar volume.  Then it returns a sentence describing this information if assessing single time point or show how liquidity evolved over time if assessing a time series portfolio. |
| Liquidity\_Dry\_Case | Percentage (int) | Print (str) | This function calculates liquidity not using past n day average volume but rather using the x% (specified as Percentage input) worst case scenario in the history.  This function only works when we deal with a single day’s holding. |
| Risk\_Contribution | Latest (bool) | bar plot (plt)  or  pie plot (plt)  or  DataFrame  (pd.df) | This function takes a portfolio and calculates the percentage contribution of each its constituent assets towards the portfolio’s volatility.  On graphic mode, when deal with a single time point, if all the contributions are positive then it will show a pie plot of the %contribution. It’s because I find pie plots most intuitive, but if there are negative contributors to volatility (benefit from holding them) then this function will return a bar plot.  When the Latest argument is set to true, Risk Contribution will work on the streaming portfolio holding information instead of looking for the holding of any historical days  On non-graphic mode (i.e. time series portfolio hard to show), this function returns dataframe. |
| All\_Live\_Printers | None | None | This is a wrapper function that just calls above listed functions. It’s only purpose is for the Live\_Trading\_Assistant’s code to be simplified without need to call 10+ functions |