risk_parity_class

index
/home/jirong/Desktop/github/risk_parity/risk_parity_class.py

```
Created on Tue Jan  5 12:35:14 2021

@author: jirong
```

Modules

functools	pyfolio	sys
numpy	re	util
pandas	riskparityportfolio	yfinance

Functions

risk_parity_weights(cov_matrix, concatenate_weights=True)

```
Return risk parity weights.

:param cov_matrix: covariance matrix
:param contenate_weights: cocnatenate weights into string
:return: returns tuple of risk parity weights and risk contribution
```

risk_parity_weights_single_chunk(time_series_input)

```
Return risk parity based on single time slice.

:param cov_matrix: time_series_input
:return: returns concatenated risk parity weights
```

rolling_risk_parity_asset_class_returns(asset_class_dict, start_date='2015-01-01', end_date='2021-12-31', api='yfinance', freq='D', window=252, window_week=52, volatility_targeting=0.1, volatility_targeting_lookback=36, max_leverage=2.0, perc_diff_before_rebalancing=0.15, comm_fee=2, financing_fee=0.015, leverage='partial', starting_amount=1000000)

```
Return rolling risk parity weights

:param asset_class_dict: asset_class_dict containing list of tickers for in each of asset class
:param start_date: start_date
:param end_date: end_date
:param api: api used
:param freq: daily (D) or resample to weekly (W)
:param window: window length
:param window_week: window length used if sampled weekly
:param volatility_targeting: Exponential realized volatility target cap (e.g. 0.1)
:param volatility_targeting_lookback: Lookback used for risk parity (e.g. 36)
:param max_leverage: Max leverage used (e.g. 2.0)
:param perc_diff_before_rebalancing: Percentage different from optimal position before rebalancing (e.g. 0.15)
:param comm_fee: Commission fee per trade (e.g. 2)
:param financing_fee: Annual financing rate (e.g. 0.015)
:param leverage: Financing fee on entire asset or leveraged portion (e.g. 'partial' or 'full')
:param starting_amount: starting_amount of capital (e.g. 1000000)
:return: returns dataframe with full parameters and returns data
```

rolling_risk_parity_returns(tickers=['TLT', 'IEF', 'GLD', 'SPY'], start_date='2015-01-01', end_date='2021-12-31', api='yfinance', freq='D', window=252, starting_amount=1000000)

```
Return rolling risk parity weights

:param tickers: list of tickers for risk parity
:param start_date: start_date
:param end_date: end_date
:param api: api used
:param freq: daily (D) or resample to weekly (W)
:param window: window length
:param starting_amount: starting_amount of capital
:return: returns concatenated risk parity weights
```

rolling_risk_parity_weights(time_series_input, window)

```
Return rolling risk parity weights

:param cov_matrix: time_series_input
:param window: window length
:return: returns concatenated risk parity weights
```

risk_parity_sensitivity_forecasts

index
/home/jirong/Desktop/github/risk_parity/risk_parity_sensitivity_forecasts.py

```
Created on Wed Jan  6 17:36:38 2021

@author: jirong
```

Modules

matplotlib	matplotlib.pyplot	risk_parity_class	warnings
numpy	pyfolio	seaborn	yaml
pandas	re	util	

Classes

builtins.object

risk_parity_sensitivity_forecast

class **risk_parity_sensitivity_forecast**(builtins.object)

```
risk_parity_sensitivity_forecast(asset_class_dict, start_date, end_date, api, freq, window, window_week, volatility_targeting
```

Methods defined here:

__init__(self, asset_class_dict, start_date, end_date, api, freq, window, window_week, volatility_targeting, volatility_targeting_lookback, max_leverag

```
Constructor for risk_parity_sensitivity_forecast (used to create forecasts for deployment and sensitivity analysis)

:param asset_class_dict: asset_class_dict containing list of tickers for in each of asset class
:param start_date: start_date
:param end_date: end_date
:param api: api used
:param freq: daily (D) or resample to weekly (W)
:param window: window length
:param window_week: window length used if sampled weekly
:param volatility_targeting: Exponential realized volatility target cap (e.g. 0.1)
:param volatility_targeting_lookback: Lookback used for risk parity (e.g. 36)
:param max_leverage: Max leverage used (e.g. 2.0)
:param perc_diff_before_rebalancing: Percentage different from optimal position before rebalancing (e.g. 0.15)
:param comm_fee: Commission fee per trade (e.g. 2)
:param financing_fee: Annual financing rate (e.g. 0.015)
:param leverage: Financing fee on entire asset or leveraged portion (e.g. 'partial' or 'full')
:param starting_amount: starting_amount of capital (e.g. 1000000)
:param volatility_targets: list of volatility targets
:param max_leverage_targets: list of max leverage cap
:return: returns risk_parity_sensitivity_forecast object
```

get_risk_parity_df(self, vol_target, leverage_num)

```
Return rolling risk parity weights

:param vol_target: Volatility target used in iteration
:param leverage_num: Leverage ratio used in iteration
:return: returns concatenated risk parity weights
```

plot_grid(self, perf_stat_name)

```
Plot sensitivity analysis grid

:param perf_stat_name: Performance statistic name
:return: returns concatenated risk parity weights
```

rp_forecast(self, vol, lev)

```
Return risk parity forecast parameter used for deployment (not required in research jupyter notebook)

:param vol: Volatility target
:param lev: Max leverage
:return: returns full dataframe, weights to tickers and recommended leverage
```

sensitivity_analysis(self)

```
Sensitivity analysis
```

Data descriptors defined here:

__dict__

```
dictionary for instance variables (if defined)
```

__weakref__

```
list of weak references to the object (if defined)
```