

---

# **InferTrade**

***Release 0.0.1***

**InferStat**

**May 26, 2021**



**CONTENTS:**

<b>1</b>	<b>utilities</b>	<b>1</b>
1.1	utilities package . . . . .	1
<b>2</b>	<b>Indices and tables</b>	<b>5</b>
	<b>Python Module Index</b>	<b>7</b>
	<b>Index</b>	<b>9</b>



## UTILITIES

## 1.1 utilities package

### 1.1.1 Submodules

### 1.1.2 utilities.operations module

Utility code for operations such as converting positions to price predictions and vice versa.

Copyright 2021 InferStat Ltd

Licensed under the Apache License, Version 2.0 (the “License”); you may not use this file except in compliance with the License. You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

Created by: Joshua Mason Created date: 11/03/2021

**class** utilities.operations.PositionsFromPricePrediction

Bases: sklearn.base.TransformerMixin, sklearn.base.BaseEstimator

This class calculates the positions to take assuming Kelly Criterion.

**fit**(X, y=None)

**transform**(X, y=None)

**class** utilities.operations.PricePredictionFromPositions

Bases: sklearn.base.TransformerMixin, sklearn.base.BaseEstimator

This converts positions into implicit price predictions based on the Kelly Criterion and an assumed volatility.

**fit**(X, y=None)

Not used.

**transform**(X: pandas.core.frame.DataFrame, y=None)

Converts allocations into the forecast one-day price changes.

**class** utilities.operations.PricePredictionFromSignalRegression(*market\_to\_trade: Optional[str] = None*)

Bases: sklearn.base.TransformerMixin, sklearn.base.BaseEstimator

Class for creating price predictions from signal values.

**fit**(X: numpy.array, y=None)

**transform**(*X*, *y=None*)

We transform a signal input to a price prediction.

**class** utilities.operations>ReturnsFromPositions

Bases: sklearn.base.TransformerMixin, sklearn.base.BaseEstimator

This calculate returns from positions.

**fit**(*X*, *y=None*)

Not used.

**transform**(*X: pandas.core.frame.DataFrame*, *y=None*)

Converts positions into the cumulative portfolio return.

utilities.operations.diff\_log(*x: Union[numpy.ndarray, pandas.core.series.Series]*) → numpy.ndarray

Differencing and log transformation between the current and a prior element.

utilities.operations.dl\_lag(*x: Union[numpy.ndarray, pandas.core.series.Series]*, *shift: int = 1*) → numpy.ndarray

Differencing and log transformation of lagged series.

utilities.operations.lag(*x: Union[numpy.ndarray, pandas.core.series.Series]*, *shift: int = 1*) → numpy.ndarray

Lag (shift) series by desired number of periods.

utilities.operations.log\_price\_minus\_log\_research(*x: Union[numpy.ndarray, pandas.core.series.Series]*, *shift: int*) → numpy.ndarray

Difference of two lagged log series.

utilities.operations.moving\_average(*x: Union[numpy.ndarray, pandas.core.series.Series]*, *window: int*) → numpy.ndarray

Calculate moving average of series for desired number of periods (window).

utilities.operations.pct\_chg(*x: Union[numpy.ndarray, pandas.core.series.Series]*) → numpy.ndarray

Percentage change between the current and a prior element.

utilities.operations.research\_over\_price\_minus\_one(*x: Union[numpy.ndarray, pandas.core.series.Series]*, *shift: int*) → numpy.ndarray

Difference of two lagged log series.

utilities.operations.zero\_one\_dl(*x: Union[numpy.ndarray, pandas.core.series.Series]*) → numpy.ndarray

Returns ones for positive values of “diff-log” series, and zeros for negative values.

### 1.1.3 utilities.performance module

Performance calculation using the InferTrade interface.

Copyright 2021 InferStat Ltd

Licensed under the Apache License, Version 2.0 (the “License”); you may not use this file except in compliance with the License. You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

Author: Thomas Oliver Created: 11th March 2021

```
utilities.performance.calculate_allocation_from_cash(last_cash_after_trade: float,
                                                    last_securities_after_transaction: float,
                                                    spot_price: float) → float
```

Calculates the current allocation.

```
utilities.performance.calculate_portfolio_performance_python(df_with_positions:
                                                            pandas.core.frame.DataFrame,
                                                            skip_checks: bool = False,
                                                            show_absolute_bankruptcies: bool =
                                                            False, annual_strategy_fee: float =
                                                            0.0, daily_spread_percent_override:
                                                            float = 0.0, mini-
mum_allocation_change_to_adjust:
                                                            float = 0.0, detailed_output: bool =
                                                            True)
```

This is the main vanilla Python calculation of portfolio performance.

```
utilities.performance.check_if_should_skip_return_calculation(previous_portfolio_return: float,
                                                             spot_price: float, day: int,
                                                             day_of_return_to_calculate: int,
                                                             show_absolute_bankruptcies: bool,
                                                             bankrupt: bool = False) -> (<class
                                                             'bool'>, <class 'float'>)
```

This function checks if we should skip the returns calculation for the requested day.

```
utilities.performance.portfolio_index(position_on_last_good_price: float, spot_price_usd: float,
                                      last_good_price_usd: Optional[float],
                                      current_bid_offer_spread_percent: float, target_allocation_perc:
                                      float, annual_strategy_fee_perc: float, last_securities_volume:
                                      float, last_cash_after_trade_usd: float, show_working: bool =
                                      False) -> (<class 'float'>, <class 'float'>, <class 'float'>)
```

A function for calculating the cumulative return of the portfolio.

```
utilities.performance.rounded_allocation_target(unconstrained_target_position: float,
                                                minimum_allocation_change_to_adjust: float) →
                                                float
```

Determines what allocation size to take if using rounded targets.

## 1.1.4 utilities.simple\_functions module

Simple functions used across the package.

Copyright 2021 InferStat Ltd

Licensed under the Apache License, Version 2.0 (the “License”); you may not use this file except in compliance with the License. You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

Author: Thomas Oliver Created: 18th March 2021

```
utilities.simple_functions.add_package(dictionary: dict, string_label: str) → dict
```

Adds a string to every item.

### 1.1.5 Module contents

Utilities directory for functions that uses the infertrade interface.

Copyright 2021 InferStat Ltd

Licensed under the Apache License, Version 2.0 (the “License”); you may not use this file except in compliance with the License. You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

Created by: Thomas Oliver Created date: 16th March 2021



## INDICES AND TABLES

- `genindex`
- `modindex`
- `search`



## PYTHON MODULE INDEX

### U

utilities, 4  
utilities.operations, 1  
utilities.performance, 2  
utilities.simple\_functions, 3



## INDEX

### A

`add_package()` (in module *utilities.simple\_functions*), 3

### C

`calculate_allocation_from_cash()` (in module *utilities.performance*), 2

`calculate_portfolio_performance_python()` (in module *utilities.performance*), 3

`check_if_should_skip_return_calculation()` (in module *utilities.performance*), 3

### D

`diff_log()` (in module *utilities.operations*), 2

`dl_lag()` (in module *utilities.operations*), 2

### F

`fit()` (*utilities.operations.PositionsFromPricePrediction* method), 1

`fit()` (*utilities.operations.PricePredictionFromPositions* method), 1

`fit()` (*utilities.operations.PricePredictionFromSignalRegression* method), 1

`fit()` (*utilities.operations>ReturnsFromPositions* method), 2

### L

`lag()` (in module *utilities.operations*), 2

`log_price_minus_log_research()` (in module *utilities.operations*), 2

### M

module

*utilities*, 4

*utilities.operations*, 1

*utilities.performance*, 2

*utilities.simple\_functions*, 3

`moving_average()` (in module *utilities.operations*), 2

### P

`pct_chg()` (in module *utilities.operations*), 2

`portfolio_index()` (in module *utilities.performance*), 3

*PositionsFromPricePrediction* (class in *utilities.operations*), 1

*PricePredictionFromPositions* (class in *utilities.operations*), 1

*PricePredictionFromSignalRegression* (class in *utilities.operations*), 1

### R

`research_over_price_minus_one()` (in module *utilities.operations*), 2

*ReturnsFromPositions* (class in *utilities.operations*), 2

`rounded_allocation_target()` (in module *utilities.performance*), 3

### T

`transform()` (*utilities.operations.PositionsFromPricePrediction* method), 1

`transform()` (*utilities.operations.PricePredictionFromPositions* method), 1

`transform()` (*utilities.operations.PricePredictionFromSignalRegression* method), 2

`transform()` (*utilities.operations>ReturnsFromPositions* method), 2

### U

*utilities*  
    module, 4

*utilities.operations*  
    module, 1

*utilities.performance*  
    module, 2

*utilities.simple\_functions*  
    module, 3

### Z

`zero_one_dl()` (in module *utilities.operations*), 2