cognite-correlation Documentation

Release 0.1.5

Cognite

Aug 16, 2019

Contents

cognite.correlation.correlation.**columns_by_max_cross_correlation**(*df: pandas.core.frame.DataFrame, relate_to: Union[int, str], lags: pandas.core.indexes.timedeltas.Timedelta, return_cross_correlation_df: bool = False*) → Union[pandas.core.frame.DataFrame, Tuple[pandas.core.frame.DataFrame, pandas.core.frame.DataFrame]]

Find lag of highest correlation and return relevant information for all columns of the inputted DataFrame. Note that the operation requires a DataFrame with even temporal spacing.

It is recommended to either have a lot of data in the data frame, or to use a short time frame for the lags, as the results are unstable if too few data points overlap in the time shifted time series.

Note about time lag: Time lag is defined such that with perfect correlation, original time series + time lag = lagged time series This means that an effect of the relate_to time series would have a positive time lag.

Parameters

- **df** (*pandas.DataFrame*) – Time series data
- **relate_to** (*Union[int, str]*) – Name of column to compare others with
- **lags** (*pandas.TimedeltaIndex*) – Pandas sequence of timedeltas for shifting the time series
- **return_cross_correlation_df** (*bool*) – Whether or not to return the cross correlations for all columns. This is a pandas.DataFrame containing the cross correlation for all calculated lags.

Returns Pandas DataFrame containing results of calculations, and optionally a DataFrame containing the cross correlations for each column at all calculated lags.

Return type Union[pandas.DataFrame, Tuple[pandas.DataFrame, pandas.DataFrame]]

Examples

Return maximum correlations and time lags for a simple dataframe.

```
>>> df = pd.DataFrame({'datetime': pd.date_range(datetime(2017, 1, 1),␣
↪datetime(2017, 1, 3), periods=10),
>>>                    'x': np.sin(np.linspace(0, 2 * np.pi, 10)),
>>>                    'y': np.sin(np.linspace(1, 2 * np.pi + 1, 10))}).set_index(
↪'datetime')
>>> lags = pd.timedelta_range(timedelta(days=-3), timedelta(), periods=10)
>>> columns_by_cross_correlation(df, 'x', lags)
```

cognite.correlation.correlation.**cross_correlate**(*df: pandas.core.frame.DataFrame, relate_to_series: pandas.core.series.Series, lag_idx=0*)

Calculate cross correlation for a given lag.

It is recommended to either have a lot of data in the data frame, or to use a short time frame for the lags, as the results are unstable if too few data points overlap in the time shifted time series.

Parameters

- **df** (*pandas.Series*) – Time series data to correlate with some series
- **relate_to_series** (*pandas.Series*) – Pandas Series with time series data to relate df to. Must have the same temporal spacing as df.
- **lag_idx** (*int*) – How many indices to move the DataFrame in relation to the series.

Returns Pandas DataFrame containing the cross correlations of the columns.

Return type pandas.DataFrame

Examples

```
>>> df = pd.DataFrame({'x': (1, 7, 3, 5), 'y': (3, 7, 6, 4)})
>>> cross_correlate(df, df['x'], lag_idx=1)
x   -0.981981
y   -0.960769
```

cognite.correlation.plot.**plot_cross_correlations**(*cross_correlation_df: pandas.core.frame.DataFrame, cols_to_plot: Union[Iterable[str], Iterable[int]] = None, separate_plots: bool = True, mpl_args: Dict[str, Any] = None*) → None

Plot cross-correlations over time lags. The cols_to_plot parameter can either be an iterable of strings of columns to plot, or a list of integers of the indices of the sorted list of columns to display.

Parameters

- **cross_correlation_df** (*pd.DataFrame*) – The DataFrame returned from columns_by_max_correlation (with the optional parameter return_cross_correlation_df enabled).
- **cols_to_plot** (*Union[Iterable[str], Iterable[int]]*) – What columns to plot or, alternatively an iterable of indices for which of the ordered columns to plot.
- **separate_plots** (*bool*) – Whether or not to divide the plots into individual plots, or to keep them in one plot
- **mpl_args** (*Dict[str, Any]*) – Additional parameters for Matplotlib plotting function

Returns None

Examples

Plot the cross-correlation for columns numbered 1-4 of the dataframe (0 is excluded to not plot auto-correlation).

```
>>> plot_cross_correlation(cross_correlation_df, range(1,5))
```

Python Module Index

C

Index

C

P