Package 'seasonalityPlot'

July 23, 2025

Type Package

Title Seasonality Variation Plots of Stock Prices and Cryptocurrencies

Version 1.3.1

Description The price action at any given time is determined by investor sentiment and market conditions. Although there is no established principle, over a long period of time, things often move with a certain periodicity. This is sometimes referred to as anomaly.
The seasonPlot() function in this package calculates and visualizes the average value of price movements over a year for any given period.
In addition, the monthly increase or decrease in price movement is represented with a colored background.
This seasonPlot() function can use the same symbols as the 'quantmod' package (e.g. ^IXIC, ^DJI, SPY, BTC-USD, and ETH-USD etc).

Depends R (>= 4.0.0)

Imports magrittr, quantmod, dygraphs, plotrix, htmltools, grDevices, graphics, zoo, lubridate, crypto2, TTR, assertthat

Suggests testthat

License Artistic-2.0

Encoding UTF-8

URL https://github.com/kumeS/seasonalityPlot, https://kumes.github.io/seasonalityPlot/, https://skume-seasonalityplot.hf.space/__docs__/#/

BugReports https://github.com/kumeS/seasonalityPlot/issues

RoxygenNote 7.2.3

NeedsCompilation no

Author Satoshi Kume [aut, cre]

Maintainer Satoshi Kume <satoshi.kume.1984@gmail.com>

Repository CRAN

Date/Publication 2024-09-25 12:00:02 UTC

Contents

`CryptoRSIheatmap` *CryptoRSI Heatmap Function*

Description

This function generates a heatmap of RSI values for a randomly selected subset of cryptocurrencies. The coins are chosen based on their market cap ranking, and the function provides insights into market sentiment using RSI. It allows for visualizing potential overbought or oversold conditions.

Usage

```
CryptoRSIheatmap(
  coin_num = 200,
  useRank = 1000,
  n = 21,
  minDataPoints = NULL,
  useRankPlot = TRUE,
  OutputData = FALSE
)
```

Arguments

`coin_num`	An integer specifying the number of coins to display in the heatmap. Must be less than the value of 'useRank'.
`useRank`	An integer defining the range within which coins are randomly selected based on their market cap ranking. Defaults to 1000.
`n`	An integer indicating the number of periods for calculating moving averages in the RSI computation. Defaults to 21.
`minDataPoints`	An integer specifying the minimum number of data points required for each coin. Defaults to 'n + 5'.
`useRankPlot`	A boolean that determines if the x-axis should plot ranks instead of sequential numbers. Defaults to TRUE.
`OutputData`	A boolean that decides if the function should return the final plot data table or only display the heatmap plot. Defaults to FALSE.

Details

CryptoRSI Heatmap Function

Generates a heatmap of the Relative Strength Index (RSI) for a randomly selected subset of cryptocurrencies. This function uses the 'crypto2' and 'TTR' packages to fetch cryptocurrency data and calculate RSI values, respectively. The heatmap visualizes RSI values to identify potential overbought or oversold conditions in the crypto market.

Value

If 'OutputData' is TRUE, returns a data frame with symbols, ranks (or sequential numbers), RSI values, and colors for plotting. Otherwise, displays a heatmap plot.

Author(s)

Satoshi Kume

Examples

```
## Not run:

# A heatmap of 200 coins using 21 days RSI
CryptoRSIheatmap(coin_num = 200, n = 21)

# A heatmap of 300 coins using 90 days RSI
CryptoRSIheatmap(coin_num = 300, n = 90)

## End(Not run)
```

---

seasonPlot *Plot Seasonality Patterns of Stock Prices or Cryptocurrencies*

Description

This function retrieves price data for a specified symbol, calculates the percentage change from the beginning of each year, and visualizes seasonality patterns by averaging over multiple years. The plot highlights average monthly changes and can optionally color months with positive or negative growth. The function automatically excludes years or days with excessive missing data to improve accuracy. Customization options for line colors, background modes, fonts, and more are available.

Usage

```
seasonPlot(
  Symbols,
  StartYear = lubridate::year(Sys.Date()) - 11,
  EndYear = lubridate::year(Sys.Date()) - 1,
  useAdjusted = FALSE,
  LineColor = 1,
```

```
  xlab = "Month",
  BackgroundMode = TRUE,
  alpha = 0.05,
  OutputData = FALSE,
  Save = FALSE,
  output_width = 1000,
  output_height = 700,
  family = "Helvetica",
  PlotAll = FALSE,
  YearMissingThreshold = 366 * 0.5,
  DayMissingThreshold = NULL
)
```

Arguments

Symbols	A character string representing the symbol for which to retrieve data. Examples include '^IXIC' (NASDAQ Composite), '^DJI' (Dow Jones Industrial Average), 'SPY' (SPDR S&P500 ETF), 'BTC-USD' (Bitcoin), 'ETH-USD' (Ethereum), and 'XRP-USD' (Ripple).
StartYear	A numeric value specifying the starting year (Gregorian calendar) for data aggregation. Defaults to 11 years before the current year.
EndYear	A numeric value specifying the ending year (Gregorian calendar) for data aggregation. Defaults to the previous year.
useAdjusted	Logical; if 'TRUE', the adjusted closing price (adjusted for dividends and splits) is used. If 'FALSE', the regular closing price is used. For cryptocurrencies, both options yield the same results.
LineColor	A numeric value (1 to 4) specifying the line color: '1' for red, '2' for blue, '3' for green, and '4' for black. Ignored when 'BackgroundMode' is 'TRUE'.
xlab	A character string for the X-axis label. Default is '"Month"'.
BackgroundMode	Logical; if 'TRUE', the background is colored based on whether the average monthly change is positive (green) or negative (red).
alpha	A numeric value (0.0 to 1.0) specifying the transparency level for the background color.
OutputData	Logical; if 'TRUE', returns the data used for plotting as a 'data.frame'.
Save	Logical; if 'TRUE', saves the plot as a PNG image.
output_width	Width of the saved PNG image in pixels. Default is 1000.
output_height	Height of the saved PNG image in pixels. Default is 700.
family	A character string specifying the font family for plot text. Default is '"Helvetica"'.
PlotAll	Logical; if 'TRUE', displays the entire time series data using the 'dygraph' package before creating the seasonality plot.
YearMissingThreshold	A numeric threshold specifying the maximum allowable number of missing years.

DayMissingThreshold
: A numeric threshold specifying the maximum allowable number of missing days per year.

Value

A plot of the seasonality patterns for the specified symbol. If ‘OutputData‘ is ‘TRUE‘, returns a list containing the symbol and the data used for the plot.

Author(s)

Satoshi Kume

Examples

```
## Not run:
## Plot seasonality of NASDAQ Composite Index (^IXIC)
seasonPlot(Symbols = "^IXIC", useAdjusted = TRUE)

## Plot seasonality of Bitcoin (BTC-USD)
seasonPlot(Symbols = "BTC-USD", StartYear = 2015, EndYear = 2020)

## Customize missing value tolerances
seasonPlot(Symbols = "^IXIC", YearMissingThreshold = 200, DayMissingThreshold = 5)

## End(Not run)
```

Index