RchangActuarialF User Manual in R

Manual del Usuario RchangActuarialF en R

Dr. Roberto Chang López

About me

I am a professional-researcher with over 20 years of experience in the fields of economics, public policies, education, health, security, social security, climate change, and monetary and fiscal policy, both within national companies and international organizations. My expertise includes robust experience in time series analysis, programming, Big Data handling, Data Science and the development and evaluation of economic and financial models.

I currently serve on the Scientific Committee of the World Health Organization (WHO) Journal, as well as FLACSO Ecuador and other regional journals. Additionally, I am a professor at a national and international level, teaching Advanced Statistics for Scientific Research in R and Python, and have authored books available on Amazon and European libraries. Moreover, I have a substantial publication record of over a dozen scientific articles across various knowledge domains, both in English and Spanish

Acerca de mi

Soy un profesional-investigador con más de 20 años de experiencia en el área de economía, políticas públicas, educación, salud, seguridad, previsión social, cambio climático y política monetaria y fiscal, tanto en empresas nacionales como en organismos internacionales. Mi trayectoria incluye una sólida experiencia en el análisis de series de tiempo, programación, manejo de bases de datos (Big Data) y Data Science, así como la formulación y evaluación de modelos económicos y financieros.

Actualmente, formo parte del Comité Científico de la Revista de la World Health Organization (OMS), así como de FLACSO Ecuador y otras revistas regionales. También, ejerzo como profesor de doctorado a nivel nacional e internacional, impartiendo conocimientos en Estadística Avanzada para la Investigación Científica en R y Python, y he publicado libros en Amazon y librerías europeas. Además, cuento con una extensa publicación de más de una docena de artículos científicos en distintas áreas del conocimiento, en inglés y castellano.

rchangunah.edu.hn / rchan@gunitec.edu

Professor of Doctoral and Master's Programs in Systems, Economics, and Statistics for Scientific Research

Profesor de Doctorado y Maestría en Sistemas, Economía y Estadística para la Investigación Científica

Installation

Installation in R Studio

1. Install and load the devtools library, then run:

```{r}

2. devtools::install_github("rchang-1/RchangActuarialF")

```

Enter one or more numbers, or an empty line to skip updates

You can press enter, Skip updates

Installation in Google Colab

To work in Google Colab, it is suggested to run the following lines before executing the previous line:

install.packages("plotly")

install.packages("htmltools")

install.packages("devtools")

install.packages("TTR")

install.packages("textshaping")

install.packages("quantmod")

install.packages("xts")

Then load the libraries

library("plotly")

library("TTR")

library("textshaping")

library("quantmod")

library("xts")

library("htmltools")

library("devtools")

Finally, execute:

devtools::install_github("rchang-1/RchangActuarialF")

Instalación

Instalación en R Studio

1. Instalar y cargar la librería devtools, luego ejecutar:

```{r}

2. devtools::install_github("rchang-1/RchangActuarialF")

```

#Enter one or more numbers, or an empty line to skip updates

#Puedes Darle enter, Skip updates (Salta los updates)

Instalación en Google Colab

Para trabajar en google colab to.R se sugiere antes de correr la línea anterior, ejecutar las siguientes:

install.packages("plotly")

install.packages("htmltools")

install.packages("devtools")

install.packages("TTR")

install.packages("textshaping")

install.packages("quantmod")

install.packages("xts")

Luego cargar las librerias

library("plotly")

library("TTR")

library("textshaping")

library("quantmod")

library("xts")

library("htmltools")

library("devtools")

Por último ejecutar:

devtools::install_github("rchang-1/RchangActuarialF")

Contents

amortization_schedule

#' title Amortization Schedule Function

#' description Creates an amortization schedule given a principal amount, interest rate, and number of periods.

#' param principal The principal amount (initial loan amount).

#' param rate The interest rate per period.

#' param nper The number of periods (loan term).

#' return A data frame with columns: period, principal_paid, interest_paid, and balance.

#' examples

#' # Example 1: Amortization schedule for a loan of $100,000 with a 5% annual interest rate over 20 periods

#' amortization_schedule(100000, 0.05, 20)

#' # Example 2: Amortization schedule for a loan of $50,000 with a 3% annual interest rate over 15 periods

 # amortization_schedule(50000, 0.03, 15)

period = 1:nper,

principal_paid = numeric(nper),

interest_paid = numeric(nper),

balance = numeric(nper)

 # Explicación en español:

Esta función `amortization_schedule` crea un cronograma de amortización para un préstamo, dado un monto principal,

una tasa de interés y un número de periodos. La función calcula el pago mensual (pmt), el interés pagado y el principal

pagado en cada periodo, y luego actualiza el saldo pendiente.

 # Parámetros:

- `principal`: El monto principal del préstamo.

- `rate`: La tasa de interés por periodo.

- `nper`: El número de periodos del préstamo.

 # Salida:

- Un marco de datos con las columnas: `period`, `principal_paid`, `interest_paid` y `balance`.

 # Ejemplos Aplicados:

1. Cronograma de amortización para un préstamo de $100,000 con una tasa de interés anual del 5% durante 20 periodos.

amortization_schedule(100000, 0.05, 20)

2. Cronograma de amortización para un préstamo de $50,000 con una tasa de interés anual del 3% durante 15 periodos.

amortization_schedule(50000, 0.03, 15)

annuity_fv

#' title Future Value (FV) of an Annuity Function

#' description Calculates the Future Value (FV) of an annuity.

#' param rate The discount rate.

#' param nper The number of periods.

#' param pmt The payment amount per period.

#' return The Future Value.

#' examples

#' # Example 1: Future Value of an annuity with a 5% discount rate over 10 periods and $100 payment per period

#' annuity_fv(0.05, 10, 100)

#' # Example 2: Future Value of an annuity with a 3% discount rate over 15 periods and $200 payment per period

annuity_fv(0.03, 15, 200)

 # Explicación en español:

Esta función `annuity_fv` calcula el Valor Futuro (FV) de una anualidad, dado un tipo de descuento,

el número de periodos y el monto de pago por periodo. La función utiliza la fórmula del valor futuro de una anualidad.

 # Parámetros:

- `rate`: La tasa de descuento.

- `nper`: El número de periodos.

- `pmt`: El monto de pago por periodo.

Salida:

- El Valor Futuro de la anualidad.

 # Ejemplos Aplicados:

1. Valor Futuro de una anualidad con una tasa de descuento del 5% durante 10 periodos y un pago de $100 por periodo.

annuity_fv(0.05, 10, 100)

2. Valor Futuro de una anualidad con una tasa de descuento del 3% durante 15 periodos y un pago de $200 por periodo.

annuity_fv(0.03, 15, 200)

annuity_pv

title Present Value (PV) of an Annuity Function

#' description Calculates the Present Value (PV) of an annuity.

#' param rate The discount rate.

#' param nper The number of periods.

#' param pmt The payment amount per period.

#' return The Present Value.

#' examples

#' # Example 1: Present Value of an annuity with a 5% discount rate over 10 periods and $100 payment per period

annuity_pv(0.05, 10, 100)

#' # Example 2: Present Value of an annuity with a 3% discount rate over 15 periods and $200 payment per period

annuity_pv(0.03, 15, 200)

 # Explicación en español:

Esta función `annuity_pv` calcula el Valor Presente (PV) de una anualidad, dado un tipo de descuento,

el número de periodos y el monto de pago por periodo. La función utiliza la fórmula del valor presente de una anualidad.

 # Parámetros:

- `rate`: La tasa de descuento.

- `nper`: El número de periodos.

- `pmt`: El monto de pago por periodo.

 # Salida:

- El Valor Presente de la anualidad.

 # Ejemplos Aplicados:

1. Valor Presente de una anualidad con una tasa de descuento del 5% durante 10 periodos y un pago de $100 por periodo.

annuity_pv(0.05, 10, 100)

2. Valor Presente de una anualidad con una tasa de descuento del 3% durante 15 periodos y un pago de $200 por periodo.

annuity_pv(0.03, 15, 200)

bdy

#' title Computing Bank Discount Yield (BDY)

#' description Computes the bank discount yield (BDY) for a T-bill.

#' param d The dollar discount, which is equal to the difference between the face value of the bill and the purchase price.

#' param f The face value (par value) of the bill.

#' param t Number of days remaining until maturity.

#' return The bank discount yield as a decimal.

#' # Example usage:

bdy(d = 1500, f = 100000, t = 120)

 # Explicación en español:

Esta función `bdy` calcula el rendimiento de descuento bancario (BDY) para un T-bill, dado el descuento en dólares,

el valor nominal del T-bill y el número de días hasta el vencimiento.

 # Parámetros:

- `d`: El descuento en dólares, que es igual a la diferencia entre el valor nominal del T-bill y el precio de compra.

- `f`: El valor nominal (valor a la par) del T-bill.

- `t`: Número de días restantes hasta el vencimiento.

 # Ejemplos Aplicados:

1. Calcular el BDY con un descuento de $1500, un valor nominal de $100,000 y 120 días hasta el vencimiento.

bdy(d = 1500, f = 100000, t = 120)

bdy2mmy

#' title Computing Money Market Yield (MMY)

#' description Computes the money market yield (MMY) for a T-bill.

#' param bdy The bank discount yield.

#' param t Number of days remaining until maturity.

#' return The money market yield as a decimal.

#' examples

#' # Example usage:

bdy2mmy(bdy = 0.045, t = 120)

 # Explicación en español:

Esta función `bdy2mmy` calcula el rendimiento del mercado monetario (MMY) para un T-bill, dado el rendimiento de descuento bancario

y el número de días hasta el vencimiento.

 # Parámetros:

- `bdy`: El rendimiento de descuento bancario.

- `t`: Número de días restantes hasta el vencimiento.

 # Ejemplos Aplicados:

1. Calcular el MMY con un rendimiento de descuento bancario de 0.045 y 120 días hasta el vencimiento.

bdy2mmy(bdy = 0.045, t = 120)

bond_price

#' title Bond Price Function

#' description Calculates the price of a bond given its face value, coupon rate, market rate, and number of periods.

#' param face_value The face value (par value) of the bond.

#' param coupon_rate The annual coupon rate of the bond.

#' param market_rate The current market interest rate.

#' param nper The number of periods (years) until bond maturity.

#' return The price of the bond.

#' examples

#' # Example 1: Calculate the price of a bond with a face value of $1000, 5% coupon rate, 3% market rate, and 10 years until maturity.

bond_price(1000, 0.05, 0.03, 10)

 # Explicación en español:

Esta función `bond_price` calcula el precio de un bono dado su valor nominal, tasa de cupón, tasa de interés de mercado

y número de períodos hasta el vencimiento del bono. Utiliza la fórmula del valor presente para calcular el valor actual

de los pagos de cupones y el valor nominal del bono.

 # Parámetros:

- `face_value`: El valor nominal (valor par) del bono.

- `coupon_rate`: La tasa de cupón anual del bono.

- `market_rate`: La tasa de interés de mercado actual.

- `nper`: El número de períodos (años) hasta el vencimiento del bono.

 # Salida:

- El precio del bono.

 # Ejemplos Aplicados:

1. Calcular el precio de un bono con un valor nominal de $1000, tasa de cupón del 5%, tasa de interés de mercado del 3% y

10 años hasta el vencimiento.

bond_price(1000, 0.05, 0.03, 10)

bond_yield_plotly

#' title Bond Yield Function Plot using Plotly

#' description This function plots the bond yield function given the bond parameters.

#' param face_value The face value (par value) of the bond.

#' param price The current market price of the bond.

#' param coupon_rate The annual coupon rate of the bond.

#' param nper The number of periods (years) until bond maturity.

#' return A plotly object representing the bond yield function.

#' examples

#' # Example 1: Plot the bond yield function for a bond with a face value of $1000, current price $950, 5% coupon rate, and 10 years until maturity.

bond_yield_plotly(1000, 950, 0.05, 10)

#' # Example 2: Plot the bond yield function for a bond with a face value of $500, current price $480,

#' # 4.5% coupon rate, and 5 years until maturity.

bond_yield_plotly(500, 480, 0.045, 5)

bond_yield_plotly <- function(face_value, price, coupon_rate, nper) {

 # Explicación en español:

Esta función `bond_yield_plotly` grafica la función de rendimiento de un bono dado los parámetros del bono.

 # Parámetros:

- `face_value`: El valor nominal (valor par) del bono.

- `price`: El precio de mercado actual del bono.

- `coupon_rate`: La tasa de cupón anual del bono.

- `nper`: El número de períodos (años) hasta el vencimiento del bono.

 # Salida:

- Un objeto plotly que representa la función de rendimiento del bono.

 # Ejemplos Aplicados:

1. Graficar la función de rendimiento de un bono con valor nominal de $1000, precio actual $950,

tasa de cupón del 5% y 10 años hasta el vencimiento.

bond_yield_plotly(1000, 950, 0.05, 10)

2. Graficar la función de rendimiento de un bono con valor nominal de $500, precio actual $480,

tasa de cupón del 4.5% y 5 años hasta el vencimiento.

bond_yield_plotly(500, 480, 0.045, 5)

bond_yield

#' title Bond Yield Function

#' description Calculates the yield (rate of return) of a bond given its price, face value, coupon rate, and number of periods.

#' param face_value The face value (par value) of the bond.

#' param price The current market price of the bond.

#' param coupon_rate The annual coupon rate of the bond.

#' param nper The number of periods (years) until bond maturity.

#' return The bond yield.

#' examples

#' # Example usage:

bond_yield(1000, 950, 0.05, 10)

 # Explicación en español:

Esta función `bond_yield` calcula el rendimiento (tasa de retorno) de un bono dado su precio, valor nominal, tasa de cupón y número de períodos.

 # Parámetros:

- `face_value`: El valor nominal (valor par) del bono.

- `price`: El precio de mercado actual del bono.

- `coupon_rate`: La tasa de cupón anual del bono.

- `nper`: El número de períodos (años) hasta el vencimiento del bono.

 # Salida:

- El rendimiento del bono.

calculate_irr

#' Calculate Internal Rate of Return (IRR)

#' This function calculates the internal rate of return of a series of cash flows given an initial investment.

#' param initial_investment A numeric value representing the initial investment.

#' param cash_flows A numeric vector of cash flows.

#' return A numeric value representing the internal rate of return.

#' examples

initial_investment <- -1000

cash_flows <- c(100, 200, 300)

calculate_irr(initial_investment, cash_flows)

 # Explicación en español:

Esta función `calculate_irr` calcula la tasa interna de retorno (TIR) de una serie de flujos de caja dado una inversión inicial.

 # Parámetros:

- `initial_investment`: Un valor numérico que representa la inversión inicial.

- `cash_flows`: Un vector numérico de flujos de caja.

 # Salida:

- Un valor numérico que representa la tasa interna de retorno.

 # Ejemplo Aplicado:

initial_investment <- -1000

cash_flows <- c(100, 200, 300)

calculate_irr(initial_investment, cash_flows)

calculate_mirr

#' Calculate Modified Internal Rate of Return (MIRR)

#' This function calculates the modified internal rate of return (MIRR) of a series of cash flows given an initial investment, a finance rate, and a reinvestment rate.

#' param initial_investment A numeric value representing the initial investment.

#' param cash_flows A numeric vector of cash flows.

#' param finance_rate A numeric value representing the finance rate.

#' param reinvestment_rate A numeric value representing the reinvestment rate.

#' return A numeric value representing the modified internal rate of return.

#' examples

initial_investment <- -1000

cash_flows <- c(100, 200, 300, 400, 500)

finance_rate <- 0.05

reinvestment_rate <- 0.08

calculate_mirr(initial_investment, cash_flows, finance_rate, reinvestment_rate)

 # Explicación en español:

Esta función `calculate_mirr` calcula la Tasa Interna de Retorno Modificada (MIRR) de una serie de flujos de caja dados una inversión inicial, una tasa de financiamiento y una tasa de reinversión.

 # Parámetros:

- `initial_investment`: Valor numérico que representa la inversión inicial.

- `cash_flows`: Vector numérico de flujos de caja que representan los ingresos y egresos a lo largo del tiempo.

- `finance_rate`: Tasa numérica que representa la tasa de financiamiento.

- `reinvestment_rate`: Tasa numérica que representa la tasa de reinversión.

 # Salida:

- Valor numérico que representa la Tasa Interna de Retorno Modificada (MIRR).

 # Ejemplos Aplicados:

1. Calcula la MIRR para una inversión inicial de -1000 unidades monetarias con flujos de caja de 100, 200, 300, 400 y 500 unidades monetarias en periodos sucesivos, utilizando una tasa de financiamiento del 5% y una tasa de reinversión del 8%.

calculate_mirr(initial_investment = -1000, cash_flows = c(100, 200, 300, 400, 500), finance_rate = 0.05, reinvestment_rate = 0.08)

calculate_npv

#' Calculate Net Present Value (NPV)

#' This function calculates the net present value of a series of cash flows given an initial investment and a discount rate.

#' param initial_investment A numeric value representing the initial investment.

#' param cash_flows A numeric vector of cash flows.

#' param rate A numeric value representing the discount rate.

#' return A numeric value representing the net present value of the cash flows.

#' examples

initial_investment <- -1000

cash_flows <- c(100, 200, 300)

rate <- 0.05

calculate_npv(initial_investment, cash_flows, rate)

 # Explicación en español:

Esta función `calculate_npv` calcula el Valor Presente Neto (NPV) de una serie de flujos de caja dados una inversión inicial y una tasa de descuento.

 # Parámetros:

- `initial_investment`: Valor numérico que representa la inversión inicial.

- `cash_flows`: Vector numérico de flujos de caja que representan los ingresos y egresos a lo largo del tiempo.

- `rate`: Tasa numérica que representa la tasa de descuento.

Salida:

- Valor numérico que representa el Valor Presente Neto (NPV) de los flujos de caja.

 # Ejemplos Aplicados:

1. Calcula el NPV para una inversión inicial de -1000 unidades monetarias con flujos de caja de 100, 200 y 300 unidades monetarias en periodos sucesivos, utilizando una tasa de descuento del 5%.

calculate_npv(initial_investment = -1000, cash_flows = c(100, 200, 300), rate = 0.05)

calculate_pv

#' Calculate Present Value (PV)

#' This function calculates the present value of a series of cash flows given a discount rate.

#' param cash_flows A numeric vector of cash flows.

#' param rate A numeric value representing the discount rate.

#' return A numeric value representing the present value of the cash flows.

 # Explicación en español:

Esta función `calculate_pv` calcula el Valor Presente (PV) de una serie de flujos de caja dado una tasa de descuento.

 # Parámetros:

- `cash_flows`: Vector numérico de flujos de caja que representan los ingresos y egresos a lo largo del tiempo.

- `rate`: Tasa numérica que representa la tasa de descuento.

 # Salida:

- Valor numérico que representa el Valor Presente (PV) de los flujos de caja.

 # Ejemplos Aplicados:

1. Calcula el PV para flujos de caja de 100, 200 y 300 unidades monetarias en periodos sucesivos, utilizando una tasa de descuento del 5%.

calculate_pv(cash_flows = c(100, 200, 300), rate = 0.05)

candlestickChart

#' title Candlestick Chart

#' description Shows prices for each period as a continuous line. The box is clear if the closing price is higher than the opening price, or filled red if the closing is lower than the opening price.

#' param ohlc Data frame with columns Date, Open, High, Low, Close.

#' param start Start date to plot, if not specified, all dates in ohlc will be included.

#' param end End date to plot.

#' param main An overall title for the plot.

#' param ... Additional arguments to be passed to ggplot.

#' return A ggplot object showing the candlestick chart.

#' # Example usage:

data <- data.frame(

Date = as.Date('2020-01-01') + 0:9,

Open = c(100, 102, 104, 103, 105, 107, 106, 108, 110, 111),

High = c(102, 104, 106, 105, 107, 109, 108, 110, 112, 113),

Low = c(98, 100, 102, 101, 103, 105, 104, 106, 108, 109),

Close = c(101, 103, 105, 104, 106, 108, 107, 109, 111, 112))

#' candlestickChart(data, main = "Candlestick Chart Example")

 # Explicación en español:

Esta función `candlestickChart` muestra los precios de cada periodo como una línea continua. La caja está vacía si el precio de cierre

es mayor que el precio de apertura, o llena de rojo si el precio de cierre es menor que el precio de apertura.

 # Parámetros:

- `ohlc`: Data frame con columnas Date, Open, High, Low, Close.

- `start`: Fecha de inicio para el gráfico, si no se especifica, se incluirán todas las fechas en `ohlc`.

- `end`: Fecha de fin para el gráfico.

- `main`: Título general para el gráfico.

 # Ejemplos Aplicados:

1. Crear un gráfico de velas con datos ficticios.

data <- data.frame(

Date = as.Date('2020-01-01') + 0:9,

Open = c(100, 102, 104, 103, 105, 107, 106, 108, 110, 111),

High = c(102, 104, 106, 105, 107, 109, 108, 110, 112, 113),

Low = c(98, 100, 102, 101, 103, 105, 104, 106, 108, 109),

Close = c(101, 103, 105, 104, 106, 108, 107, 109, 111, 112))

candlestickChart(data, main = "Candlestick Chart Example")

cash_flow_diagram_ggplot2

#' Plot Cash Flow Diagram using ggplot2

#' This function plots an interactive cash flow diagram given a series of cash flows and periods.

#' param cash_flows A numeric vector of cash flows.

#' param periods A numeric vector of periods corresponding to the cash flows.

#' return A ggplot object representing the cash flow diagram.

 # Explicación en español:

Esta función `cash_flow_diagram_ggplot2` crea un diagrama de flujo de efectivo interactivo utilizando ggplot2,

dado un vector de flujos de efectivo y periodos correspondientes.

 # Parámetros:

- `cash_flows`: Vector numérico de flujos de efectivo.

- `periods`: Vector numérico de periodos correspondientes a cada flujo de efectivo.

 # Salida:

- Un objeto ggplot que representa el diagrama de flujo de efectivo.

 # Ejemplos Aplicados:

1. Crear un diagrama de flujo de efectivo para flujos de efectivo de -1000, 200, 300, 400 y 800 en periodos 0 a 4.

cash_flows <- c(-1000, 200, 300, 400, 800)

periods <- 0:4

cash_flow_diagram_ggplot2(cash_flows, periods)

cash.ratio

#' title Cash Ratio

#' description Computes the cash ratio – Liquidity ratios measure the firm's ability to satisfy its short-term obligations as they come due.

#' param cash Cash.

#' param ms Marketable securities.

#' param cl Current liabilities.

#' return The cash ratio as a decimal.

#' # Example usage:

#' cash.ratio(cash = 3000, ms = 2000, cl = 2000)

 # Explicación en español:

Esta función `cash.ratio` calcula el ratio de efectivo, que mide la capacidad de la empresa para satisfacer sus obligaciones a corto plazo

a medida que vencen.

 # Parámetros:

- `cash`: Efectivo.

- `ms`: Valores negociables.

- `cl`: Pasivos corrientes.

 # Ejemplos Aplicados:

1. Calcular el ratio de efectivo con $3000 en efectivo, $2000 en valores negociables y $2000 en pasivos corrientes.

cash.ratio(cash = 3000, ms = 2000, cl = 2000)

cca

#' title Comparable Companies Analysis (CCA) Function

#' description Estimates the fair value of a stock using Comparable Companies Analysis.

#' param comparable_pe_ratio The average Price-to-Earnings (P/E) Ratio of comparable companies.

#' param earnings_per_share The current earnings per share.

#' return The estimated fair value of the stock.

#' examples

#' # Example usage:

comparable_pe_ratio <- 20

earnings_per_share <- 4

cca(comparable_pe_ratio, earnings_per_share)

 # Explicación en español:

Esta función `cca` calcula el valor justo de una acción utilizando el análisis de compañías comparables (CCA),

dados el promedio del Precio/Ganancias (P/E) de compañías comparables y las ganancias por acción actuales.

 # Parámetros:

- `comparable_pe_ratio`: El Precio/Ganancias (P/E) promedio de las compañías comparables.

- `earnings_per_share`: Las ganancias por acción actuales.

 # Salida:

- El valor estimado justo de la acción.

 # Ejemplos Aplicados:

1. Calcular el valor justo de una acción con un P/E promedio de 20 y ganancias por acción de 4.

comparable_pe_ratio <- 20

earnings_per_share <- 4

cca(comparable_pe_ratio, earnings_per_share)

clean_price

#' title Clean Price of a Bond

#' description Calculates the clean price of a bond given its dirty price, coupon rate, days since last coupon, days in coupon period, and face value.

#' param dirty_price The dirty price (full price) of the bond.

#' param coupon_rate The annual coupon rate of the bond.

#' param days_since_last_coupon The number of days since the last coupon payment.

#' param days_in_coupon_period The total number of days in the coupon period.

#' param face_value The face value (par value) of the bond.

#' param coupons_per_year The number of coupon payments per year.

#' return The clean price of the bond.

#' # Example usage:

clean_price(960, 0.05, 60, 180, 1000, 2)

Parámetros

#' param dirty_price El precio sucio (precio completo) del bono.

#' param coupon_rate La tasa de cupón anual del bono.

#' param days_since_last_coupon El número de días desde el último pago de cupón.

#' param days_in_coupon_period El número total de días en el período de cupón.

#' param face_value El valor nominal (valor par) del bono.

#' param coupons_per_year El número de pagos de cupón por año.

#' return El precio limpio del bono.

#' # Ejemplo de uso:

clean_price(960, 0.05, 60, 180, 1000, 2)

 # Explicación y Uso:

Precio Sucio (dirty_price): Esta función toma el precio limpio del bono, la tasa de cupón, los días transcurridos desde el último pago de cupón, los días en el período del cupón, el valor nominal del bono y la cantidad de pagos de cupón por año para calcular el precio sucio completo del bono.

Precio Limpio (clean_price): Esta función toma el precio sucio del bono, la tasa de cupón, los días transcurridos desde el último pago de cupón, los días en el período del cupón, el valor nominal del bono y la cantidad de pagos de cupón por año para calcular el precio limpio del bono.

Estas funciones son útiles para realizar cálculos financieros detallados relacionados con bonos, teniendo en cuenta los días transcurridos desde el último pago de cupón. Esto es esencial para determinar precios precisos tanto limpios como sucios en mercados financieros.

coefficient.variation

#' title Computing Coefficient of Variation

#' description Computes the coefficient of variation.

#' param sd Standard deviation.

#' param avg Average value.

#' return The coefficient of variation as a decimal.

#' # Example usage:

coefficient.variation(sd = 0.15, avg = 0.39)

 # Explicación en español:

Esta función `coefficient.variation` calcula el coeficiente de variación, que es una medida de dispersión relativa.

 # Parámetros:

- `sd`: Desviación estándar.

- `avg`: Valor promedio.

 # Ejemplos Aplicados:

1. Calcular el coeficiente de variación con una desviación estándar de 0.15 y un valor promedio de 0.39.

coefficient.variation(sd = 0.15, avg = 0.39)

cogs

#' title Cost of Goods Sold and Ending Inventory

#' description Computes the cost of goods sold and ending inventory under three methods: FIFO, LIFO, Weighted average.

#' param uinv Units of beginning inventory.

#' param pinv Price of beginning inventory.

#' param units nx1 vector of inventory units purchased ordered by time (from first to last).

#' param price nx1 vector of inventory price. Same order as units.

#' param sinv Units of sold inventory.

#' param method Inventory methods: FIFO (first in first out), LIFO (last in first out), WAC (weighted average cost).

#' return A list containing the cost of goods sold and the ending inventory value.

#' examples

#' # Example usage:

cogs(uinv = 2, pinv = 2, units = c(3, 5), price = c(3, 5), sinv = 7, method = "FIFO")

cogs(uinv = 2, pinv = 2, units = c(3, 5), price = c(3, 5), sinv = 7, method = "LIFO")

cogs(uinv = 2, pinv = 2, units = c(3, 5), price = c(3, 5), sinv = 7, method = "WAC")

 # Explicación en español:

Esta función `cogs` calcula el costo de los bienes vendidos y el valor del inventario final bajo tres métodos: FIFO, LIFO y promedio ponderado.

 # Parámetros:

- `uinv`: Unidades de inventario inicial.

- `pinv`: Precio del inventario inicial.

- `units`: Vector nx1 de unidades de inventario compradas ordenadas por tiempo (de primero a último).

- `price`: Vector nx1 de precios de inventario. Mismo orden que `units`.

- `sinv`: Unidades de inventario vendidas.

- `method`: Métodos de inventario: FIFO (primero en entrar, primero en salir), LIFO (último en entrar, primero en salir), WAC (costo promedio ponderado).

 # Ejemplos Aplicados:

1. Calcular el COGS y el inventario final usando FIFO.

cogs(uinv = 2, pinv = 2, units = c(3, 5), price = c(3, 5), sinv = 7, method = "FIFO")

2. Calcular el COGS y el inventario final usando LIFO.

cogs(uinv = 2, pinv = 2, units = c(3, 5), price = c(3, 5), sinv = 7, method = "LIFO")

3. Calcular el COGS y el inventario final usando promedio ponderado.

cogs(uinv = 2, pinv = 2, units = c(3, 5), price = c(3, 5), sinv = 7, method = "WAC")

current.ratio

#' title Current Ratio

#' description Computes the current ratio – Liquidity ratios measure the firm's ability to satisfy its short-term obligations as they come due.

#' param ca Current assets.

#' param cl Current liabilities.

#' return The current ratio as a decimal.

#' examples

#' # Example usage:

#' current.ratio(ca = 8000, cl = 2000)

 # Explicación en español:

Esta función `current.ratio` calcula el ratio corriente, que mide la capacidad de la empresa para satisfacer sus obligaciones a corto plazo

a medida que vencen.

 # Parámetros:

- `ca`: Activos corrientes.

- `cl`: Pasivos corrientes.

 # Ejemplos Aplicados:

1. Calcular el ratio corriente con $8000 en activos corrientes y $2000 en pasivos corrientes.

current.ratio(ca = 8000, cl = 2000)

dcf

#' title Discounted Cash Flow (DCF) Function

#' description Calculates the present value of future cash flows using the Discounted Cash Flow method.

#' param cash_flows A numeric vector of expected future cash flows.

#' param discount_rate The discount rate (required rate of return).

#' return The estimated fair value of the stock.

#' # Example usage:

cash_flows <- c(10, 12, 15, 18, 20)

discount_rate <- 0.08

dcf(cash_flows, discount_rate)

 # Explicación en español:

Esta función `dcf` calcula el valor presente de los flujos de efectivo futuros utilizando el método de Flujo de Caja Descontado (DCF),

dados los flujos de efectivo futuros esperados y la tasa de descuento (tasa de retorno requerida).

 # Parámetros:

- `cash_flows`: Un vector numérico de los flujos de efectivo futuros esperados.

- `discount_rate`: La tasa de descuento (tasa de retorno requerida).

 # Salida:

- El valor estimado justo de la acción.

 # Ejemplos Aplicados:

1. Calcular el valor presente de flujos de efectivo futuros con flujos de efectivo de [10, 12, 15, 18, 20] y una tasa de descuento del 8%.

cash_flows <- c(10, 12, 15, 18, 20)

discount_rate <- 0.08

dcf(cash_flows, discount_rate)

ddb

#' title Depreciation Expense Recognition – Double-Declining Balance

#' description Computes depreciation expense recognition using the double-declining balance (DDB) method, which applies two times the straight-line rate to the declining balance.

#' param cost Cost of long-lived assets.

#' param rv Residual value of the long-lived assets at the end of its useful life. DDB does not explicitly use the asset's residual value in the calculations, but depreciation ends once the estimated residual value has been reached. If the asset is expected to have no residual value, the DB method will never fully depreciate it, so the DB method is typically changed to straight-line at some point in the asset's life.

#' param t Length of the useful life.

#' return A data frame showing the depreciation expense for each year.

 # # Example usage:

ddb(cost = 1200, rv = 200, t = 5)

 # Explicación en español:

Esta función `ddb` calcula el reconocimiento de los gastos de depreciación utilizando el método del balance decreciente doble (DDB), que aplica dos veces la tasa lineal al balance decreciente.

 # Parámetros:

- `cost`: Costo de los activos de larga duración.

- `rv`: Valor residual de los activos de larga duración al final de su vida útil.

- `t`: Duración de la vida útil.

 # Ejemplos Aplicados:

1. Calcular la depreciación con un costo de $1200, un valor residual de $200 y una vida útil de 5 años.

ddb(cost = 1200, rv = 200, t = 5)

ddgm

#' title Discounted Dividend Growth Model (DDGM) Function

#' description Calculates the fair value of a stock using the Discounted Dividend Growth Model.

#' param dividend The current dividend per share.

#' param growth_rate The expected constant growth rate of dividends.

#' param discount_rate The discount rate (required rate of return).

#' return The estimated fair value of the stock.

#' # Example usage:

dividend <- 2.5

growth_rate <- 0.04

discount_rate <- 0.1

ddgm(dividend, growth_rate, discount_rate)

 # Explicación en español:

Esta función `ddgm` calcula el valor justo de una acción utilizando el Modelo de Crecimiento de Dividendos Descontados (DDGM),

dados el dividendo actual por acción, la tasa de crecimiento esperada constante de los dividendos y la tasa de descuento

(tasa de retorno requerida).

 # Parámetros:

- `dividend`: El dividendo actual por acción.

- `growth_rate`: La tasa de crecimiento esperada constante de los dividendos.

- `discount_rate`: La tasa de descuento (tasa de retorno requerida).

 # Salida:

- El valor estimado justo de la acción.

 # Ejemplos Aplicados:

1. Calcular el valor justo de una acción con un dividendo actual de $2.5, una tasa de crecimiento de dividendos del 4%,

y una tasa de descuento del 10%.

dividend <- 2.5

growth_rate <- 0.04

discount_rate <- 0.1

ddgm(dividend, growth_rate, discount_rate)

ddm

#' title Dividend Discount Model (DDM) Function

#' description Calculates the present value of future dividends using the Dividend Discount Model.

#' param dividends A numeric vector of expected future dividends.

#' param discount_rate The discount rate (required rate of return).

#' return The estimated fair value of the stock.

#' # Example usage:

dividends <- c(2.5, 2.7, 3.0, 3.2, 3.5)

discount_rate <- 0.1

ddm(dividends, discount_rate)

 # Explicación en español:

Esta función `ddm` calcula el valor presente de futuros dividendos utilizando el Modelo de Descuento de Dividendos (DDM),

dados los dividendos esperados futuros y la tasa de descuento (tasa de retorno requerida).

 # Parámetros:

- `dividends`: Un vector numérico de los dividendos futuros esperados.

- `discount_rate`: La tasa de descuento (tasa de retorno requerida).

 # Salida:

- El valor estimado justo de la acción.

 # Ejemplos Aplicados:

1. Calcular el valor presente de futuros dividendos con dividendos esperados de c(2.5, 2.7, 3.0, 3.2, 3.5) y una tasa de descuento del 10%.

dividends <- c(2.5, 2.7, 3.0, 3.2, 3.5)

discount_rate <- 0.1

ddm(dividends, discount_rate)

debt.ratio

#' title Debt Ratio

#' description Computes the debt ratio – Solvency ratios measure the firm's ability to satisfy its long-term obligations.

#' param td Total debt.

#' param ta Total assets.

#' return The debt ratio as a decimal.

#' # Example usage:

debt.ratio(td = 6000, ta = 20000)

 # Explicación en español:

Esta función `debt.ratio` calcula el ratio de deuda, que mide la capacidad de la empresa para satisfacer sus obligaciones a largo plazo.

 # Parámetros:

- `td`: Deuda total.

- `ta`: Activos totales.

 # Ejemplos Aplicados:

1. Calcular el ratio de deuda con una deuda total de $6000 y activos totales de $20000.

debt.ratio(td = 6000, ta = 20000)

declining_balance_depreciation

#' Declining Balance Depreciation

#' This function calculates the declining balance depreciation of an asset.

#' param cost Initial cost of the asset.

#' param rate Depreciation rate.

#' param life Useful life of the asset in years.

#' return A data frame with columns Year, Depreciation, and Book Value.

#' # Example usage:

declining_balance_depreciation(10000, 0.2, 5)

#' Esta función calcula la depreciación por saldo decreciente de un activo.

#' param cost Costo inicial del activo.

#' param rate Tasa de depreciación.

#' param life Vida útil del activo en años.

#' return Un data frame con las columnas Year (Año), Depreciation (Depreciación) y Book Value (Valor Contable).

#' # Ejemplo de uso:

#' declining_balance_depreciation(10000, 0.2, 5)

diluted.EPS

#' title Diluted Earnings Per Share

#' description Computes diluted earnings per share.

#' param ni Net income.

#' param pd Preferred dividends.

#' param cpd Dividends on convertible preferred stock.

#' param cdi Interest on convertible debt.

#' param tax Tax rate.

#' param w Weighted average number of common shares outstanding.

#' param cps Shares from conversion of convertible preferred stock.

#' param cds Shares from conversion of convertible debt.

#' param iss Shares issuable from stock options.

#' return Diluted earnings per share.

#' # Example usage:

#' diluted.EPS(ni = 115600, pd = 10000, cdi = 42000, tax = 0.4, w = 200000, cds = 60000)

#' diluted.EPS(ni = 115600, pd = 10000, cpd = 10000, w = 200000, cps = 40000)

#' diluted.EPS(ni = 115600, pd = 10000, w = 200000, iss = 2500)

diluted.EPS(ni = 115600, pd = 10000, cpd = 10000, cdi = 42000, tax = 0.4, w = 200000, cps = 40000, cds = 60000, iss = 2500)

 # Explicación en español:

Esta función `diluted.EPS` calcula las ganancias diluidas por acción.

 # Parámetros:

- `ni`: Ingreso neto.

- `pd`: Dividendos preferentes.

- `cpd`: Dividendos de acciones preferentes convertibles.

- `cdi`: Intereses de deuda convertible.

- `tax`: Tasa de impuestos.

- `w`: Número promedio ponderado de acciones comunes en circulación.

- `cps`: Acciones de la conversión de acciones preferentes convertibles.

- `cds`: Acciones de la conversión de deuda convertible.

- `iss`: Acciones emitibles de opciones sobre acciones.

 # Ejemplos Aplicados:

1. Calcular las ganancias diluidas por acción con un ingreso neto de $115600, dividendos preferentes de $10000, intereses de deuda convertible de $42000, tasa de impuestos de 0.4, número promedio ponderado de acciones comunes de 200000, y 60000 acciones de la conversión de deuda convertible.

diluted.EPS(ni = 115600, pd = 10000, cdi = 42000, tax = 0.4, w = 200000, cds = 60000)

2. Calcular las ganancias diluidas por acción con un ingreso neto de $115600, dividendos preferentes de $10000, dividendos de acciones preferentes convertibles de $10000, número promedio ponderado de acciones comunes de 200000, y 40000 acciones de la conversión de acciones preferentes convertibles.

diluted.EPS(ni = 115600, pd = 10000, cpd = 10000, w = 200000, cps = 40000)

3. Calcular las ganancias diluidas por acción con un ingreso neto de $115600, dividendos preferentes de $10000, número promedio ponderado de acciones comunes de 200000, y 2500 acciones emitibles de opciones sobre acciones.

diluted.EPS(ni = 115600, pd = 10000, w = 200000, iss = 2500)

4. Calcular las ganancias diluidas por acción con un ingreso neto de $115600, dividendos preferentes de $10000, dividendos de acciones preferentes convertibles de $10000, intereses de deuda convertible de $42000, tasa de impuestos de 0.4, número promedio ponderado de acciones

comunes de 200000, 40000 acciones de la conversión de acciones preferentes convertibles, 60000 acciones de la conversión de deuda convertible, y 2500 acciones emitibles de opciones sobre acciones.

diluted.EPS(ni = 115600, pd = 10000, cpd = 10000, cdi = 42000, tax = 0.4, w = 200000, cps = 40000, cds = 60000, iss = 2500)

dirty_price

##' title Calculate Dirty Price

#' description Calculates the dirty price of a bond given the clean price and the accrued interest.

#' param clean_price The clean price of the bond.

#' param accrued_interest The accrued interest of the bond.

#' return The dirty price of the bond.

#' # Example usage:

#' dirty_price(100, 2)

#' title Calcular Precio Sucio

#' description Calcula el precio sucio de un bono dado el precio limpio y el interés acumulado.

#' param clean_price El precio limpio del bono.

#' param accrued_interest El interés acumulado del bono.

#' return El precio sucio del bono.

#' # Ejemplo de uso:

#' dirty_price(100, 2)

discount.rate

' title Discount Rate Calculation

#' description Computes the rate of return for each period based on given parameters.

#' param n Number of periods.

#' param pv Present value.

#' param fv Future value.

#' param pmt Payment per period.

#' param type 0 for payments at the end of each period; 1 for payments at the beginning of each period.

#' return Rate of return for each period.

#' # Example usage:

#discount.rate(n = 5, pv = 10, fv = 600, pmt = -100, type = 0)

 # Explicación en español:

Esta función `discount.rate` calcula la tasa de retorno para cada período con base en los parámetros proporcionados.

 # Parámetros:

- `n`: Número de períodos.

- `pv`: Valor presente.

- `fv`: Valor futuro.

- `pmt`: Pago por período.

- `type`: 0 para pagos al final de cada período; 1 para pagos al inicio de cada período.

 # Retorno:

La función retorna la tasa de retorno para cada período calculada según los parámetros dados.

 # Ejemplos:

1. Calcular la tasa de retorno con 5 períodos, valor presente 0, valor futuro 600, pago por período -100, y pagos al final de cada período:

discount.rate(n = 5, pv = 10, fv = 600, pmt = -100, type = 0)

ear.continuous

#' title Continuous Effective Annual Rate (EAR)

#' description Converts a stated annual rate to the effective annual rate with continuous compounding.

#' param r Stated annual rate.

#' return Effective annual rate with continuous compounding.

#' # Example usage:

#' ear.continuous(r = 0.1)

ear.continuous(0.03)

 # Explicación en español:

Esta función `ear.continuous` convierte una tasa anual declarada a la tasa anual efectiva (EAR) con capitalización continua.

 # Parámetros:

- `r`: Tasa anual declarada.

 # Ejemplos Aplicados:

1. Convertir una tasa anual declarada del 10% a la tasa anual efectiva (EAR) con capitalización continua.

ear.continuous(r = 0.1)

2. Convertir una tasa anual del 3% a la tasa anual efectiva (EAR) con capitalización continua.

ear.continuous(0.03)

ear

#' title Effective Annual Rate (EAR)

#' description Converts a stated annual rate to the effective annual rate.

#' param r Stated annual rate.

#' param m Number of compounding periods per year.

#' return Effective annual rate.

#' # Example usage:

#' ear(r = 0.12, m = 12)

ear(0.04, 365)

 # Explicación en español:

Esta función `ear` convierte una tasa anual declarada a la tasa anual efectiva (EAR).

 # Parámetros:

- `r`: Tasa anual declarada.

- `m`: Número de períodos de capitalización por año.

 # Ejemplos Aplicados:

1. Convertir una tasa anual declarada del 12% a la tasa anual efectiva (EAR) con 12 períodos de capitalización por año.

ear(r = 0.12, m = 12)

2. Convertir una tasa anual del 4% a la tasa anual efectiva (EAR) con 365 períodos de capitalización por año.

ear(0.04, 365)

ear2bey

#' title Bond-Equivalent Yield (BEY)

#' description Converts the effective annual rate (EAR) to the bond-equivalent yield (BEY), which is 2 times the semiannual discount rate.

#' param ear Effective annual rate (EAR).

#' return Bond-equivalent yield (BEY).

#' # Example usage:

ear2bey(ear = 0.06)

 # Explicación en español:

Esta función `ear2bey` convierte la tasa anual efectiva (EAR) a la tasa de rendimiento equivalente de bonos (BEY), que es el doble de la tasa de descuento semestral.

 # Parámetros:

- `ear`: Tasa anual efectiva (EAR).

 # Ejemplos Aplicados:

1. Convertir una tasa anual efectiva (EAR) del 6% a la tasa de rendimiento equivalente de bonos (BEY).

ear2bey(ear = 0.06)

ear2hpr

#' title Computing HPR, the holding period return

#' description Computes the holding period return (HPR) based on the effective annual rate (EAR) and the number of days remaining until maturity.

#' param ear Effective annual rate.

#' param t Number of days remaining until maturity.

#' return Holding period return (HPR).

#' # Example usage:

ear2hpr(ear=0.05039, t=150)

 # Explicación en español:

Esta función `ear2hpr` calcula el retorno del período de tenencia (HPR) basado en la tasa anual efectiva (EAR) y el número de días restantes hasta el vencimiento.

 # Parámetros:

- `ear`: Tasa anual efectiva.

- `t`: Número de días restantes hasta el vencimiento.

 # Retorno:

Retorno del período de tenencia (HPR).

 # Ejemplos:

1. Calcular el HPR con una tasa anual efectiva de 0.05039 y 150 días restantes:

ear2hpr(ear=0.05039, t=150)

efficient_frontier

#' Efficient Frontier Calculation

#' This function calculates the Efficient Frontier for a portfolio given returns and covariance matrix.

#' param returns A matrix or data frame where each column represents the returns of an asset.

#' param cov_matrix A covariance matrix of asset returns.

#' param risk_free_rate The risk-free rate of return (default is 0.02).

#' return A data frame with columns: Return (Expected Return), Risk (Standard Deviation), and Sharpe Ratio.

#' # Example

data(EuStockMarkets)

Select columns CAC, DAX, FTSE, SMI

returns <- as.matrix(scale(EuStockMarkets[, c("CAC", "DAX", "FTSE", "SMI")]))

cov_matrix <- cov(returns)

efficient_frontier(returns, cov_matrix)

#' param returns Una matriz o data frame donde cada columna representa los retornos de un activo.

#' param cov_matrix Una matriz de covarianza de los retornos de los activos.

#' param risk_free_rate La tasa de interés libre de riesgo (por defecto es 0.02).

#' return Un data frame con las columnas: Return (Retorno Esperado), Risk (Desviación Estándar), y Sharpe Ratio.

#' # Ejemplo con el conjunto de datos EuStockMarkets

#' data(EuStockMarkets)

#' # Seleccionar las columnas CAC, DAX, FTSE, SMI

#' returns <- as.matrix(scale(EuStockMarkets[, c("CAC", "DAX", "FTSE", "SMI")]))

#' cov_matrix <- cov(returns)

#' efficient_frontier(returns, cov_matrix)

EIR

#' title Equivalent/proportional Interest Rates

#' description An interest rate to be applied n times per year can be converted to an equivalent rate to be applied p times per year.

#' param r Interest rate to be applied n times per year (r is annual rate!).

#' param n Times that the interest rate r is compounded per year.

#' param p Times that the equivalent rate is compounded per year.

#' param type Equivalent interest rates ('e', default) or proportional interest rates ('p').

#' return Equivalent or proportional interest rate.

#' # Example usage:

#' EIR(r=0.05, n=1, p=12)

#' EIR(r=0.05, n=2, p=12)

#' EIR(r=0.05, n=4, p=12)

#' EIR(r=0.05, n=12, p=1)

#' EIR(r=0.05, n=1, p=4)

EIR(r=0.05, n=12, p=4)

EIR(r=0.05, p=12, type='p')

 # Explicación en español:

Esta función `EIR` convierte una tasa de interés aplicada n veces por año a una tasa equivalente aplicada p veces por año.

 # Parámetros:

- `r`: Tasa de interés aplicada n veces por año (r es tasa anual).

- `n`: Veces que la tasa de interés r se capitaliza por año.

- `p`: Veces que la tasa equivalente se capitaliza por año.

- `type`: Tasas de interés equivalentes ('e', predeterminado) o tasas de interés proporcionales ('p').

 # Retorno:

Tasa de interés equivalente o proporcional.

 # Ejemplos:

1. Tasa de interés mensual equivalente a 5% capitalizado anualmente:

EIR(r=0.05, n=1, p=12)

2. Tasa de interés mensual equivalente a 5% capitalizado semestralmente:

EIR(r=0.05, n=2, p=12)

3. Tasa de interés mensual equivalente a 5% capitalizado trimestralmente:

EIR(r=0.05, n=4, p=12)

4. Tasa de interés anual equivalente a 5% capitalizado mensualmente:

EIR(r=0.05, n=12, p=1)

5. Tasa de interés trimestral equivalente a 5% capitalizado anualmente:

EIR(r=0.05, n=1, p=4)

6. Tasa de interés trimestral equivalente a 5% capitalizado mensualmente:

EIR(r=0.05, n=12, p=4)

7. Tasa de interés mensual proporcional equivalente a una tasa anual simple de 5%:

EIR(r=0.05, p=12, type='p')

EPS

#' title Basic Earnings Per Share

#' description Computes the basic earnings per share (EPS) based on net income, preferred dividends, and the weighted average number of common shares outstanding.

#' param ni Net income.

#' param pd Preferred dividends.

#' param w Weighted average number of common shares outstanding.

#' return Basic earnings per share (EPS).

 # Explicación en español:

Esta función `EPS` calcula la ganancia básica por acción (EPS) basada en el ingreso neto, los dividendos preferentes y el promedio ponderado del número de acciones comunes en circulación.

 # Parámetros:

- `ni`: Ingreso neto.

- `pd`: Dividendos preferentes.

- `w`: Promedio ponderado del número de acciones comunes en circulación.

 # Retorno:

Ganancia básica por acción (EPS).

 # Ejemplos:

1. Calcular la EPS con un ingreso neto de 10,000, dividendos preferentes de 1,000 y 11,000 acciones comunes en circulación:

EPS(ni=10000, pd=1000, w=11000)

eva

#' title Economic Value Added (EVA) Model Function

#' description Calculates the Economic Value Added (EVA) for a stock.

#' param net_operating_profit_after_tax Net Operating Profit After Tax (NOPAT).

#' param weighted_average_cost_of_capital Weighted Average Cost of Capital (WACC).

#' return The estimated fair value of the stock.

#' # Example usage:

#' net_operating_profit_after_tax <- 1000000

#' weighted_average_cost_of_capital <- 0.08

#' eva(net_operating_profit_after_tax, weighted_average_cost_of_capital)

#' Esta función calcula el Valor Económico Agregado (EVA) para una acción.

#' param net_operating_profit_after_tax Ganancia Operativa Neta después de Impuestos (NOPAT).

#' param weighted_average_cost_of_capital Costo Promedio Ponderado de Capital (WACC).

#' return El valor estimado justo de la acción.

#' # Ejemplo de uso:

net_operating_profit_after_tax <- 1000000

weighted_average_cost_of_capital <- 0.08

eva(net_operating_profit_after_tax, weighted_average_cost_of_capital)

financial.leverage

#' title Financial Leverage

#' description Computes the financial leverage, which is a solvency ratio that measures the firm's ability to satisfy its long-term obligations.

#' param te Total equity.

#' param ta Total assets.

#' return Financial leverage ratio.

#' # Example usage:

financial.leverage(te=16000, ta=20000)

 # Explicación en español:

Esta función `financial.leverage` calcula el apalancamiento financiero, que es un ratio de solvencia que mide la capacidad de la empresa para satisfacer sus obligaciones a largo plazo.

 # Parámetros:

- `te`: Capital total.

- `ta`: Activos totales.

 # Retorno:

Ratio de apalancamiento financiero.

 # Ejemplos:

1. Calcular el apalancamiento financiero con un capital total de 16,000 y activos totales de 20,000:

financial.leverage(te=16000, ta=20000)

fv.annuity

#' title Estimate Future Value of an Annuity

#' description Estimates the future value of an annuity based on the discount rate, number of periods, payment per period, and payment type.

#' param r Discount rate, or the interest rate at which the amount will be compounded each period.

#' param n Number of periods.

#' param pmt Payment per period.

#' param type Payments occur at the end of each period (type=0); payments occur at the beginning of each period (type=1).

#' return Future value of the annuity.

#' # Example usage:

#' fv.annuity(r=0.03, n=12, pmt=-1000)

fv.annuity(r=0.03, n=12, pmt=-1000, type=1)

 # Explicación en español:

Esta función `fv.annuity` estima el valor futuro de una anualidad basada en la tasa de descuento, número de períodos, pago por período, y el tipo de pago.

 # Parámetros:

- `r`: Tasa de descuento, o la tasa de interés a la cual se capitalizará el monto en cada período.

- `n`: Número de períodos.

- `pmt`: Pago por período.

- `type`: Los pagos ocurren al final de cada período (type=0); los pagos ocurren al inicio de cada período (type=1).

 # Retorno:

Valor futuro de la anualidad.

 # Ejemplos:

1. Estimar el valor futuro de una anualidad con una tasa de descuento de 0.03, 12 períodos, y pago por período de -1000:

fv.annuity(r=0.03, n=12, pmt=-1000)

2. Estimar el valor futuro de una anualidad con una tasa de descuento de 0.03, 12 períodos, pago por período de -1000, y pagos al inicio de cada período:

fv.annuity(r=0.03, n=12, pmt=-1000, type=1)

fv

#' title Estimate Future Value (FV)

#' description Estimates the future value (FV) based on the discount rate, number of periods, present value, payment per period, and payment type.

#' param r Discount rate, or the interest rate at which the amount will be compounded each period.

#' param n Number of periods.

#' param pv Present value.

#' param pmt Payment per period.

#' param type Payments occur at the end of each period (type=0); payments occur at the beginning of each period (type=1).

#' return Future value (FV).

#' # Example usage:

fv(r=0.07, n=10, pv=1000, pmt=10)

 # Explicación en español:

Esta función `fv` estima el valor futuro (FV) basado en la tasa de descuento, número de períodos, valor presente, pago por período, y el tipo de pago.

 # Parámetros:

- `r`: Tasa de descuento, o la tasa de interés a la cual se capitalizará el monto en cada período.

- `n`: Número de períodos.

- `pv`: Valor presente.

- `pmt`: Pago por período.

- `type`: Los pagos ocurren al final de cada período (type=0); los pagos ocurren al inicio de cada período (type=1).

 # Retorno:

Valor futuro (FV).

 # Ejemplos:

1. Estimar el valor futuro con una tasa de descuento de 0.07, 10 períodos, valor presente de 1000, y pago por período de 10:

fv(r=0.07, n=10, pv=1000, pmt=10)

fv.simple

#' title Estimate Future Value of a Single Sum

#' description Estimates the future value (FV) of a single sum based on the discount rate, number of periods, and present value.

#' param r Discount rate, or the interest rate at which the amount will be compounded each period.

#' param n Number of periods.

#' param pv Present value.

#' return Future value (FV) of the single sum.

#' # Example usage:

#' fv.simple(0.08, 10, -300)

fv.simple(r=0.04, n=20, pv=-50000)

 # Explicación en español:

Esta función `fv.simple` estima el valor futuro (FV) de una suma única basada en la tasa de descuento, número de períodos, y el valor presente.

 # Parámetros:

- `r`: Tasa de descuento, o la tasa de interés a la cual se capitalizará el monto en cada período.

- `n`: Número de períodos.

- `pv`: Valor presente.

 # Retorno:

Valor futuro (FV) de la suma única.

 # Ejemplos:

1. Estimar el valor futuro de una suma única con una tasa de descuento de 0.08, 10 períodos, y valor presente de -300:

fv.simple(0.08, 10, -300)

2. Estimar el valor futuro de una suma única con una tasa de descuento de 0.04, 20 períodos, y valor presente de -50000:

fv.simple(r=0.04, n=20, pv=-50000)

fv.uneven

#' title Computing the Future Value of an Uneven Cash Flow Series

#' description Computes the future value of an uneven cash flow series based on the stated annual rate and the cash flow series.

#' param r Stated annual rate.

#' param cf Uneven cash flow series.

#' return Future value of the uneven cash flow series.

#' # Example usage:

fv.uneven(r=0.1, cf=c(-1000, -500, 0, 4000, 3500, 2000))

 # Explicación en español:

Esta función `fv.uneven` calcula el valor futuro de una serie de flujos de caja desiguales basada en la tasa anual declarada y la serie de flujos de caja.

 # Parámetros:

- `r`: Tasa anual declarada.

- `cf`: Serie de flujos de caja desiguales.

 # Retorno:

Valor futuro de la serie de flujos de caja desiguales.

 # Ejemplos:

1. Calcular el valor futuro de una serie de flujos de caja desiguales con una tasa anual de 0.1 y flujos de caja de c(-1000, -500, 0, 4000, 3500, 2000):

fv.uneven(r=0.1, cf=c(-1000, -500, 0, 4000, 3500, 2000))

geometric.mean

#' title Geometric Mean Return

#' description Computes the geometric mean return based on returns over multiple periods.

#' param r Returns over multiple periods.

#' return Geometric mean return.

#' # Example usage:

geometric.mean(r=c(-0.0934, 0.2345, 0.0892))

 # Explicación en español:

Esta función `geometric.mean` calcula el retorno medio geométrico basado en los retornos durante múltiples períodos.

 # Parámetros:

- `r`: Retornos durante múltiples períodos.

 # Retorno:

Retorno medio geométrico.

 # Ejemplos:

1. Calcular el retorno medio geométrico con los retornos de c(-0.0934, 0.2345, 0.0892):

geometric.mean(r=c(-0.0934, 0.2345, 0.0892))

get.ohlc.yahoo

#' title Download Stock Prices from Yahoo Finance

#' description Downloads stock prices from Yahoo Finance (open, high, low, close, volume, adjusted).

#' param symbol Symbol of stock, e.g., AAPL, GOOG, SPX.

#' param start Start date, e.g., 2013-01-01.

#' param end End date, e.g., 2013-08-06.

#' param freq Time interval, e.g., d: daily, w: weekly, m: monthly.

#' return Data frame with columns: Date, Open, High, Low, Close, Volume, Adjusted.

#' # Example usage:

#' # get.ohlc.yahoo(symbol="AAPL")

#' # get.ohlc.yahoo(symbol="AAPL", start="2013-08-01", freq="d")

get.ohlc.yahoo(symbol="AAPL", start="2013-07-01", end="2013-08-01", freq="w")

 # Explicación en español:

Esta función `get.ohlc.yahoo` descarga los precios de las acciones desde Yahoo Finance (apertura, máximo, mínimo, cierre, volumen, ajustado).

 # Parámetros:

- `symbol`: Símbolo de la acción, por ejemplo, AAPL, GOOG, SPX.

- `start`: Fecha de inicio, por ejemplo, 2013-01-01.

- `end`: Fecha de fin, por ejemplo, 2013-08-06.

- `freq`: Intervalo de tiempo, por ejemplo, d: diario, w: semanal, m: mensual.

 # Retorno:

Un data frame con columnas: Date, Open, High, Low, Close, Volume, Adjusted.

 # Ejemplos:

1. Descargar precios de acciones para AAPL desde Yahoo Finance:

get.ohlc.yahoo(symbol="AAPL")

2. Descargar precios de acciones para AAPL desde el 2013-08-01 con datos diarios:

get.ohlc.yahoo(symbol="AAPL", start="2013-08-01", freq="d")

3. Descargar precios de acciones para AAPL desde el 2013-07-01 hasta el 2013-08-01 con datos semanales:

get.ohlc.yahoo(symbol="AAPL", start="2013-07-01", end="2013-08-01", freq="w")

ggm

#' Calculate Fair Value using Gordon Growth Model (GGM)

#' This function calculates the fair value of a stock using the Gordon Growth Model (GGM).

#' param dividend A numeric value representing the current dividend per share.

#' param growth_rate A numeric value representing the expected constant growth rate of dividends.

#' param discount_rate A numeric value representing the discount rate (required rate of return).

#' return A numeric value representing the estimated fair value of the stock.

#' # Example usage:

ggm(dividend = 2, growth_rate = 0.05, discount_rate = 0.1)

 # Explicación en español:

Esta función `ggm` calcula el valor justo de una acción utilizando el Modelo de Crecimiento de Gordon (GGM).

 # Parámetros:

- `dividend`: Valor numérico que representa el dividendo actual por acción.

- `growth_rate`: Valor numérico que representa la tasa de crecimiento constante esperada de los dividendos.

- `discount_rate`: Valor numérico que representa la tasa de descuento (tasa de retorno requerida).

 # Salida:

- Valor numérico que representa el valor justo estimado de la acción.

 # Ejemplos Aplicados:

1. Calcula el valor justo para un dividendo actual de 2 unidades monetarias, una tasa de crecimiento esperada de 5% y una tasa de descuento del 10%.

ggm(dividend = 2, growth_rate = 0.05, discount_rate = 0.1)

gpm

#' title Gross Profit Margin

#' description Computes the gross profit margin, evaluating a company's financial performance.

#' param gp Gross profit, equal to revenue minus cost of goods sold (cogs).

#' param rv Revenue (sales).

#' return Gross profit margin.

 # Explicación en español:

Esta función `gpm` calcula el margen de utilidad bruta, evaluando el rendimiento financiero de una empresa.

 # Parámetros:

- `gp`: Utilidad bruta, igual a ingresos menos costo de ventas (cogs).

- `rv`: Ingresos (ventas).

 # Retorno:

Margen de utilidad bruta.

 # Ejemplos:

1. Calcular el margen de utilidad bruta con gp = 1000 y rv = 20000:

gpm(gp = 1000, rv = 20000)

harmonic.mean

#' title Harmonic Mean

#' description Computes the harmonic mean, average price over multiple periods.

#' param p Price over multiple periods.

#' return Harmonic mean.

#' # Example usage:

harmonic.mean(p = c(8, 9, 10))

 # Explicación en español:

Esta función `harmonic.mean` calcula la media armónica, que es el promedio de precios sobre múltiples períodos.

 # Parámetros:

- `p`: Precio sobre múltiples períodos.

 # Retorno:

Media armónica.

 # Ejemplos:

1. Calcular la media armónica con p = c(8, 9, 10):

harmonic.mean(p = c(8, 9, 10))

hpr

#' title Holding Period Return

#' description Computes the holding period return (HPR) based on beginning and ending values and cash flows received.

#' param ev Ending value.

#' param bv Beginning value.

#' param cfr Cash flow received.

#' return Holding period return.

#' # Example usage:

hpr(ev = 33, bv = 30, cfr = 0.5)

 # Explicación en español:

Esta función `hpr` calcula el rendimiento del período de tenencia (HPR) basado en los valores inicial y final y los flujos de efectivo recibidos.

 # Parámetros:

- `ev`: Valor final.

- `bv`: Valor inicial.

- `cfr`: Flujo de efectivo recibido.

 # Retorno:

Rendimiento del período de tenencia.

 # Ejemplos:

1. Calcular el HPR con ev = 33, bv = 30, cfr = 0.5:

hpr(ev = 33, bv = 30, cfr = 0.5)

hpr2bey

#' title Bond-Equivalent Yield (BEY)

#' description Computes the bond-equivalent yield (BEY), which is 2 times the semiannual discount rate.

#' param hpr Holding period return.

#' param t Number of months remaining until maturity.

#' return Bond-equivalent yield (BEY).

#' # Example usage:

hpr2bey(hpr = 0.02, t = 3)

 # Explicación en español:

Esta función `hpr2bey` calcula el rendimiento equivalente al bono (BEY), que es 2 veces la tasa de descuento semestral.

 # Parámetros:

- `hpr`: Rendimiento del período de tenencia.

- `t`: Número de meses restantes hasta el vencimiento.

 # Retorno:

Rendimiento equivalente al bono (BEY).

Ejemplos:

1. Calcular el BEY con hpr = 0.02 y t = 3:

hpr2bey(hpr = 0.02, t = 3)

hpr2ear

#' title Effective Annual Rate (EAR)

#' description Converts the holding period return (HPR) to the effective annual rate (EAR).

#' param hpr Holding period return.

#' param t Number of days remaining until maturity.

#' return Effective annual rate (EAR).

 # Explicación en español:

Esta función `hpr2ear` convierte el rendimiento del período de tenencia (HPR) a la tasa anual efectiva (EAR).

 # Parámetros:

- `hpr`: Rendimiento del período de tenencia.

- `t`: Número de días restantes hasta el vencimiento.

 # Retorno:

Tasa anual efectiva (EAR).

 # Ejemplos:

1. Convertir HPR a EAR con hpr = 0.015228 y t = 120:

hpr2ear(hpr = 0.015228, t = 120)

interest_rate

#' title Interest Rate Calculation Function

#' description Calculates the interest rate given the present value, future value, and the period.

#' param pv The present value.

#' param fv The future value.

#' param period The period over which the interest rate is calculated.

#' return The interest rate.

#' examples

#' interest_rate(100, 200, 5)

#' param pv El valor presente.

#' param fv El valor futuro.

#' param period El periodo sobre el cual se calcula la tasa de interés.

#' return La tasa de interés.

#' # Ejemplo de uso:

interest_rate(100, 200, 5)

irr_plot

Function to plot IRR against initial guesses using plotly

#' title Internal Rate of Return (IRR) Plot Function

#' description Plots the Internal Rate of Return (IRR) against a range of initial guesses using plotly.

#' param cash_flows A numeric vector of cash flows.

#' param guesses A numeric vector of initial guesses to plot. Default is seq(0.01, 0.5, by = 0.01).

#' return An interactive plot of IRR against initial guesses.

#' # Example usage:

irr_plot(c(-100, 50, 75, 200))

#' @title Función para Graficar la Tasa Interna de Retorno (TIR)

#' @description Grafica la Tasa Interna de Retorno (TIR) en función de un rango de estimaciones iniciales usando plotly.

#' @param flujos_caja Un vector numérico de flujos de caja.

#' @param estimaciones Un vector numérico de estimaciones iniciales para graficar. El valor por defecto es seq(0.01, 0.5, by = 0.01).

#' @return Un gráfico interactivo de la TIR en función de las estimaciones iniciales.

#' @examples

#' irr_plot(c(-100, 50, 75, 200))

Irr

#' title Internal Rate of Return (IRR) Function

#' description Calculates the Internal Rate of Return (IRR) given a series of cash flows.

#' param cash_flows A numeric vector of cash flows.

#' param guess An initial guess for the IRR (default is 0.1).

#' return The Internal Rate of Return (IRR).

#' examples

#' irr(c(-100, 50, 75, 200))

#' Calcula la Tasa Interna de Retorno (TIR) dado una serie de flujos de efectivo.

#' param cash_flows Un vector numérico de flujos de efectivo.

#' param guess Una estimación inicial para la TIR (por defecto es 0.1).

#' return La Tasa Interna de Retorno (TIR).

#' # Ejemplo de uso:

irr(c(-100, 50, 75, 200))

irr2

#' title Internal Rate of Return (IRR)

#' description This function calculates the internal rate of return (IRR) for cash flows, including negative values.

#' param cf Cash flows. The first cash flow is the initial outlay (negative value).

#' param cutoff Threshold to consider NPV as zero.

#' param from Smallest IRR to try.

#' param to Largest IRR to try.

#' param step Increment of the IRR.

#' return Internal rate of return (IRR).

#' # Example usage:

#' irr2(cf = c(-5, 1.6, 2.4, 2.8))

irr2(cf = c(-200, 50, 60, -70, 30, 20))

 # Explicación en español:

Esta función `irr2` calcula la tasa interna de retorno (IRR) para flujos de efectivo, incluyendo valores negativos.

 # Parámetros:

- `cf`: Flujos de efectivo. El primer flujo de efectivo es la inversión inicial (valor negativo).

- `cutoff`: Umbral para considerar el valor presente neto (NPV) como cero.

- `from`: Menor IRR a probar.

- `to`: Mayor IRR a probar.

- `step`: Incremento del IRR.

 # Retorno:

Tasa interna de retorno (IRR).

 # Ejemplos:

1. Calcular el IRR con cf = c(-5, 1.6, 2.4, 2.8):

irr2(cf = c(-5, 1.6, 2.4, 2.8))

2. Calcular el IRR con cf = c(-200, 50, 60, -70, 30, 20):

irr2(cf = c(-200, 50, 60, -70, 30, 20))

mirr

#' title Modified Internal Rate of Return (MIRR) Function

#' description Calculates the Modified Internal Rate of Return (MIRR) given cash flows, finance rate, and reinvestment rate.

#' param cash_flows A numeric vector of cash flows.

#' param finance_rate The finance rate for discounting cash outflows.

#' param reinvest_rate The reinvestment rate for compounding cash inflows.

#' return The Modified Internal Rate of Return.

#' # Example usage:

mirr(c(-100, 50, 75, 200), 0.05, 0.08)

#' description Calcula la Tasa Interna de Retorno Modificada (MIRR) dados los flujos de caja, la tasa de financiamiento y la tasa de reinversión.

#' param flujos_caja Un vector numérico de flujos de caja.

#' param tasa_financiamiento La tasa de financiamiento para descontar los flujos de caja negativos.

#' param tasa_reinversion La tasa de reinversión para componer los flujos de caja positivos.

#' return La Tasa Interna de Retorno Modificada.

#' examples

#' mirr(c(-100, 50, 75, 200), 0.05, 0.08)

npv_plot

#' description Plots the NPV against a range of discount rates.

#' This function plots the Net Present Value (NPV) against a range of discount rates using Plotly.

#' param cash_flows A numeric vector of cash flows.

#' param rates A numeric vector of discount rates to plot. Default is seq(0, 0.2, by = 0.01).

#' return A plot of NPV against discount rates.

#' # Example usage:

#' # Define cash flows for the example

#' # cash_flows <- c(-1000, 200, 300, 400)

#' npv_plot(cash_flows)

#' Esta función gráfica el Valor Actual Neto (VAN) contra un rango de tasas de descuento utilizando Plotly.

#' param cash_flows Un vector numérico de flujos de efectivo.

#' param rates Un vector numérico de tasas de descuento para graficar. Por defecto es seq(0, 0.2, by = 0.01).

#' return Un gráfico del VAN contra las tasas de descuento.

#' # Ejemplo de uso:

#' # Definir flujos de efectivo para el ejemplo

cash_flows <- c(-1000, 200, 300, 800)

#' npv_plot(cash_flows)

npv

#' title Net Present Value (NPV) Function

#' description Calculates the Net Present Value (NPV) given a discount rate and cash flows.

#' This function calculates the Net Present Value (NPV) given a discount rate and a vector of cash flows.

#' param rate The discount rate.

#' param cash_flows A numeric vector of cash flows.

#' return The Net Present Value.

#' # Example usage:

#' npv(0.1, c(-100, 50, 75, 200))

#' Calcula el Valor Presente Neto (VPN) dado una tasa de descuento y flujos de efectivo.

#' param rate La tasa de descuento.

#' param cash_flows Un vector numérico de flujos de efectivo.

#' return El Valor Presente Neto (VPN).

#' # Ejemplo de uso:

npv(0.1, c(-100, 50, 75, 200))

perpetuity_pv

title Present Value (PV) of a Perpetuity Function

#' description Calculates the Present Value (PV) of a perpetuity.

#' This function calculates the Present Value (PV) of a perpetuity given a discount rate and payment amount per period.

#' param rate The discount rate.

#' param pmt The payment amount per period.

#' return The Present Value.

#' # Example usage:

#' perpetuity_pv(0.05, 100)

#' Función para calcular el Valor Presente (PV) de una perpetuidad

#'#' Calcula el Valor Presente (PV) de una perpetuidad dado una tasa de descuento y el monto del pago por periodo.

#' param rate La tasa de descuento.

#' param pmt El monto del pago por periodo.

#' return El Valor Presente.

#' # Ejemplo de uso:

perpetuity_pv(0.05, 100)

plot_efficient_frontier_A

#' Plot Efficient Frontier

#' This function plots the Efficient Frontier given a data frame containing Return, Risk, and Sharpe Ratio.

#' param frontier_data A data frame with columns Return, Risk, and Sharpe_Ratio.

#' param max_sharpe_data A data frame with columns Max_Sharpe_Risk and Max_Sharpe_Return.

#' return A Plotly object representing the Efficient Frontier.

#' # Example usage with efficient_frontier function

#' returns <- matrix(rnorm(100 * 4, mean = 0.01, sd = 0.02), nrow = 100, ncol = 4)

#' colnames(returns) <- c("AMZN", "MSFT", "TSLA", "AAPL")

#' cov_matrix <- cov(returns)

#' frontier <- efficient_frontier(returns, cov_matrix)

#' plot_efficient_frontier_A(frontier)

 # Ejemplo de uso

returns <- matrix(rnorm(100 * 4, mean = 0.01, sd = 0.02), nrow = 100, ncol = 4)

colnames(returns) <- c("AMZN", "MSFT", "TSLA", "AAPL")

cov_matrix <- cov(returns)

frontier <- efficient_frontier(returns, cov_matrix)

 # # Definir el punto del Máximo Sharpe Ratio (ejemplo)

max_sharpe_data <- data.frame(Max_Sharpe_Risk = 0.04, Max_Sharpe_Return = 0.15)

 # # Graficar la Frontera Eficiente con el punto del Máximo Sharpe Ratio

plot_efficient_frontier_A(frontier, max_sharpe_data)

plot_efficient_frontier_B

#' Plot Efficient Frontier using ggplot2

#' This function plots the Efficient Frontier given a data frame containing Return, Risk, and Sharpe Ratio using ggplot2.

#' param frontier_data A data frame with columns Return, Risk, and Sharpe_Ratio.

#' Un marco de datos con las columnas Return, Risk y Sharpe_Ratio.

#' return A ggplot2 object representing the Efficient Frontier.

#' Un objeto ggplot2 que representa la Frontera Eficiente.

#' # Example usage with efficient_frontier function

#' returns <- matrix(rnorm(100 * 4, mean = 0.01, sd = 0.02), nrow = 100, ncol = 4)

#' colnames(returns) <- c("AMZN", "MSFT", "TSLA", "AAPL")

#' cov_matrix <- cov(returns)

#' frontier <- efficient_frontier(returns, cov_matrix)

#' plot_efficient_frontier_B(frontier)

 # Ejemplo de uso

returns <- matrix(rnorm(100 * 4, mean = 0.01, sd = 0.02), nrow = 100, ncol = 4)

colnames(returns) <- c("AMZN", "MSFT", "TSLA", "AAPL")

cov_matrix <- cov(returns)

frontier <- efficient_frontier(returns, cov_matrix)

 # # Graficar la Frontera Eficiente con el punto de máximo Sharpe Ratio etiquetado

plot_efficient_frontier_B(frontier)

plot_efficient_frontier

#' This function plots the Efficient Frontier given a data frame containing Return, Risk, and Sharpe Ratio.

#' param frontier_data A data frame with columns Return, Risk, and Sharpe_Ratio.

#' return A ggplot2 object representing the Efficient Frontier.

#' # Example usage with efficient_frontier function

#' returns <- as.matrix(scale(EuStockMarkets[, c("CAC", "DAX", "FTSE", "SMI")]))

#' cov_matrix <- cov(returns)

#' frontier <- efficient_frontier(returns, cov_matrix)

#' plot_efficient_frontier(frontier)

#' #' # Example usage with efficient_frontier function

data(EuStockMarkets)

Seleccionar las columnas CAC, DAX, FTSE, SMI

returns <- as.matrix(scale(EuStockMarkets[, c("CAC", "DAX", "FTSE", "SMI")]))

cov_matrix <- cov(returns)

frontier <- efficient_frontier(returns, cov_matrix)

plot_efficient_frontier(frontier)

#' Graficar Frontera Eficiente

#' Esta función grafica la Frontera Eficiente dados un marco de datos que contiene Return (Retorno), Risk (Riesgo), y Sharpe Ratio.

#' param frontier_data Un marco de datos con las columnas Return, Risk, y Sharpe_Ratio.

#' return Un objeto ggplot2 que representa la Frontera Eficiente.

#' # Ejemplo de uso con la función efficient_frontier

returns <- as.matrix(scale(EuStockMarkets[, c("CAC", "DAX", "FTSE", "SMI")]))

cov_matrix <- cov(returns)

frontier <- efficient_frontier(returns, cov_matrix)

plot_efficient_frontier(frontier)

plot_portfolio_optimization

#' This function plots the optimal portfolio weights given asset returns and covariance matrix.

#'First you need to calculate portfolio_optimization

#' param returns A matrix or data frame of asset returns.

#' param covariance_matrix A covariance matrix of asset returns.

#' return A Plotly object representing the optimal portfolio weights.

#' # Example usage:

returns <- matrix(c(0.1, 0.2, 0.15, 0.05, 0.1, 0.12), ncol = 2)

covariance_matrix <- matrix(c(0.005, -0.010, -0.010, 0.040), ncol = 2)

plot_portfolio_optimization(returns, covariance_matrix)

##Para utilizar esta función debes calcular primero portfolio_optimization

#' Optimización de Portafolio

#' Esta función grafica los pesos óptimos del portafolio dados los rendimientos de activos y la matriz de covarianza.

#' Primero es necesario calcular la optimización del portafolio.

#' param returns Una matriz o marco de datos de los rendimientos de activos.

#' param covariance_matrix Una matriz de covarianza de los rendimientos de activos.

#' return Un objeto Plotly que representa los pesos óptimos del portafolio.

#' # Ejemplo de uso:

#' returns <- matrix(c(0.1, 0.2, 0.15, 0.05, 0.1, 0.12), ncol = 2)

#' covariance_matrix <- matrix(c(0.005, -0.010, -0.010, 0.040), ncol = 2)

#' plot_portfolio_optimization(returns, covariance_matrix)

plot_portfolio_return

#' description This function plots the portfolio return given different asset weights and returns.

#' Esta función grafica el rendimiento de un portafolio dado diferentes pesos de activos y sus rendimientos.

#' param weights A numeric vector of portfolio weights (proportions).

#' Vector numérico de los pesos del portafolio (proporciones).

#' param returns A numeric vector or matrix of asset returns.

#' Vector numérico o matriz de rendimientos de activos.

#' return A Plotly object representing the portfolio return.

#' Objeto Plotly que representa el rendimiento del portafolio.

#' # Example usage:

#' # Ejemplo de uso:

#' # Generate example data

#' # Generar datos de ejemplo

set.seed(123)

weights <- c(0.5, 0.3, 0.2)

asset_returns <- matrix(rnorm(100 * length(weights), mean = 0.01, sd = 0.02), nrow = 100, ncol = length(weights))

#' plot_portfolio_return(weights, asset_returns)

pmt

#' title Period Payment (PMT)

#' description Estimates the payment per period for a loan or investment based on given parameters.

#' param r Discount rate, or the interest rate at which the amount will be compounded each period.

#' param n Number of periods.

#' param pv Present value.

#' param fv Future value.

#' param type Payments occur at the end of each period (type = 0); payments occur at the beginning of each period (type = 1).

#' return Payment per period.

#' # Example usage:

#' pmt(r = 0.08, n = 10, pv = -1000, fv = 10)

#' pmt(r = 0.08, n = 10, pv = -1000, fv = 0)

#' pmt(r = 0.08, n = 10, pv = -1000, fv = 10, type = 1)

 # Explicación en español:

Esta función `pmt` estima el pago por período para un préstamo o inversión con base en los parámetros proporcionados.

 # Parámetros:

- `r`: Tasa de descuento o tasa de interés a la cual se compone el monto por período.

- `n`: Número de períodos.

- `pv`: Valor presente.

- `fv`: Valor futuro.

- `type`: Pagos al final de cada período (type = 0); pagos al inicio de cada período (type = 1).

 # Retorno:

Pago por período.

 # Ejemplos:

1. Estimar el pago por período con una tasa de descuento del 8%, 10 períodos, un valor presente de -1000 y un valor futuro de 10:

pmt(r = 0.08, n = 10, pv = -1000, fv = 10)

portfolio_composition

#' Plot Portfolio Composition

#' This function plots the composition of a portfolio given asset weights and names.

#' param weights A numeric vector of asset weights.

#' param asset_names A character vector of asset names.

#' return A ggplot2 object representing the portfolio composition.

#' # Example usage:

#' weights <- c(0.3, 0.5, 0.2)

#' asset_names <- c("Stock A", "Stock B", "Stock C")

#' portfolio_composition(weights, asset_names)

#' Graficar la Composición del Portafolio

#' Esta función grafica la composición de un portafolio dados los pesos y nombres de los activos.

#' param pesos Un vector numérico con los pesos de los activos.

#' param nombres_activos Un vector de caracteres con los nombres de los activos.

#' return Un objeto ggplot2 que representa la composición del portafolio.

#' pesos <- c(0.3, 0.5, 0.2)

#' nombres_activos <- c("Acción A", "Acción B", "Acción C")

#' portfolio_composition (pesos, nombres_activos)

portfolio_optimization

#' title Portfolio Optimization Function

#' description Optimizes portfolio weights to maximize return or minimize risk.

#' param returns A matrix or data frame of asset returns.

#' param covariance_matrix A covariance matrix of asset returns.

#' return A numeric vector of optimal portfolio weights.

#' # Example usage:

returns <- matrix(c(0.1, 0.2, 0.15, 0.05, 0.1, 0.12), ncol = 2)

covariance_matrix <- matrix(c(0.005, -0.010, -0.010, 0.040), ncol = 2)

portfolio_optimization(returns, covariance_matrix)

 # The `portfolio_optimization` function uses the quadratic programming algorithm

to find the optimal weights of an investment portfolio, minimizing risk for a given

level of return. The function returns a numeric vector representing the optimal

weights assigned to each asset in the portfolio.

In this example, the `optimal_weights` vector contains the optimal weights for

the two assets in the portfolio, assigning approximately 66.67% to the first

asset and 33.33% to the second asset.

#' description Optimiza los pesos del portafolio para maximizar el retorno o minimizar el riesgo.

#' param retornos Una matriz o un data frame de retornos de los activos.

#' param matriz_covarianza Una matriz de covarianza de los retornos de los activos.

#' return Un vector numérico con los pesos óptimos del portafolio.

#' examples

#' retornos <- matrix(c(0.1, 0.2, 0.15, 0.05, 0.1, 0.12), ncol = 2)

#' matriz_covarianza <- matrix(c(0.005, -0.010, -0.010, 0.040), ncol = 2)

#' portfolio_optimization (retornos, matriz_covarianza)

#' # La función `optimizacion_portafolio` utiliza el algoritmo de programación cuadrática

#' # para encontrar los pesos óptimos de un portafolio de inversión, minimizando el riesgo

#' # para un nivel dado de retorno. La función devuelve un vector numérico que representa

#' # los pesos óptimos asignados a cada activo en el portafolio.

#' # En este ejemplo, el vector `pesos_optimos` contiene los pesos óptimos para

#' # los dos activos en el portafolio, asignando aproximadamente 66.67% al primer

#' # activo y 33.33% al segundo activo.

portfolio_return

#' title Portfolio Return Function

#' description Calculates the return of a portfolio given asset weights and returns.

#' param weights A numeric vector of portfolio weights (proportions).

#' param returns A numeric vector or matrix of asset returns.

#' return The portfolio return.

#' # Example usage:

weights <- c(0.5, 0.3, 0.2)

asset_returns <- c(0.1, 0.05, 0.08)

portfolio_return(weights, asset_returns)

#' title Función de Retorno del Portafolio

#' description Calcula el retorno de un portafolio dados los pesos y los retornos de los activos.

#' param pesos Un vector numérico con los pesos del portafolio (proporciones).

#' param retornos Un vector numérico o una matriz de retornos de los activos.

#' return El retorno del portafolio.

weights <- c(0.5, 0.3, 0.2)

asset_returns <- c(0.1, 0.05, 0.08)

portfolio_return(weights, asset_returns)

portfolio_variance

#' title Portfolio Variance Function

#' description Calculates the variance of a portfolio given asset weights and a covariance matrix.

#' Esta función calcula la varianza de un portafolio dado los pesos de los activos y una matriz de covarianza.

#' param weights A numeric vector of portfolio weights (proportions).

#' Vector numérico de los pesos del portafolio (proporciones).

#' param covariance_matrix A covariance matrix of asset returns.

#' Matriz de covarianza de los rendimientos de los activos.

#' return The portfolio variance.

#' La varianza del portafolio.

#' # Example usage:

#' # Ejemplo de uso:

weights <- c(0.5, 0.3, 0.2)

cov_matrix <- matrix(c(0.1, 0.03, 0.02, 0.03, 0.12, 0.05, 0.02, 0.05, 0.15), nrow = 3, ncol = 3)

portfolio_variance(weights, cov_matrix)

pre_ratio

#' title Price-to-Earnings (P/E) Ratio Model Function

#' description Estimates the fair value of a stock using the Price-to-Earnings (P/E) Ratio.

#' Esta función estima el valor justo de una acción utilizando el Price-to-Earnings (P/E) Ratio.

#' param earnings_per_share The current earnings per share.

#' Ganancias por acción actuales.

#' param pe_ratio The desired Price-to-Earnings (P/E) Ratio.

#' Ratio precio-ganancias (P/E) deseado.

#' return The estimated fair value of the stock.

#' El valor estimado justo de la acción.

#' # Example usage:

#' # Ejemplo de uso:

earnings_per_share <- 3.5

pe_ratio <- 15

pre_ratio(earnings_per_share, pe_ratio)

pv.annuity

#' title Present Value of an Annuity (PV.Annuity)

#' description Estimates the present value (PV) of an annuity based on given parameters.

#' param r Discount rate, or the interest rate at which the amount will be compounded each period.

#' param n Number of periods.

#' param pmt Payment per period.

#' param type Payments occur at the end of each period (type = 0); payments occur at the beginning of each period (type = 1).

#' return Present value (PV) of an annuity.

#' # Example usage:

#' pv.annuity(r = 0.03, n = 12, pmt = 1000)

#' pv.annuity(r = 0.0425, n = 3, pmt = 30000)

 # Explicación en español:

Esta función `pv.annuity` estima el valor presente (PV) de una anualidad con base en los parámetros proporcionados.

 # Parámetros:

- `r`: Tasa de descuento o tasa de interés a la cual se compone el monto por período.

- `n`: Número de períodos.

- `pmt`: Pago por período.

- `type`: Payments occur at the end of each period (type = 0); payments occur at the beginning of each period (type = 1).

pv

#' title Present Value (PV) Function

#' description Calculates the Present Value (PV) of a single cash flow.

#' param rate The discount rate.

#' param cash_flow The cash flow amount.

#' param period The period in which the cash flow occurs.

#' return The Present Value.

#' # Example usage:

pv(0.1, 100, 1)

#' description Calcula el Valor Presente (VP) de un único flujo de efectivo.

#' param tasa La tasa de descuento.

#' param flujo_efectivo El monto del flujo de efectivo.

#' param periodo El período en el cual ocurre el flujo de efectivo.

#' return El Valor Presente.

#' # Ejemplo de uso:

#' pv(0.1, 100, 1)

pv.uneven

#' title Present Value of Uneven Cash Flows

#' description Calculates the present value of a series of uneven cash flows.

#' param r The discount rate.

#' param cf A vector of cash flows.

#' return The present value of the cash flows.

#' # Example usage:

pv.uneven(r=0.1, cf=c(-1000, -500, 0, 4000, 3500, 2000))

 # Explicación en español:

Esta función `pv.uneven` calcula el valor presente de una serie de flujos de caja desiguales,

dado una tasa de descuento y un vector de flujos de caja.

 # Parámetros:

- `r`: La tasa de descuento.

- `cf`: Un vector de flujos de caja.

 # Salida:

- El valor presente de los flujos de caja.

 # Ejemplos Aplicados:

1. Calcular el valor presente de una serie de flujos de caja desiguales con una tasa de descuento del 10%.

pv.uneven(r=0.1, cf=c(-1000, -500, 0, 4000, 3500, 2000))

quick.ratio

#' title Quick Ratio

#' description Liquidity ratios measure the firm's ability to satisfy its short-term obligations as they come due.

#' param cash The amount of cash.

#' param ms Marketable securities.

#' param rc Receivables.

#' param cl Current liabilities.

#' return The quick ratio.

#' # Example usage:

quick.ratio(cash=3000, ms=2000, rc=1000, cl=2000)

 # Explicación en español:

Esta función `quick.ratio` calcula el ratio rápido, un indicador de liquidez que mide la capacidad de la empresa

para satisfacer sus obligaciones a corto plazo a medida que vencen.

 # Parámetros:

- `cash`: La cantidad de efectivo.

- `ms`: Valores negociables.

- `rc`: Cuentas por cobrar.

- `cl`: Pasivos corrientes.

 # Salida:

- El ratio rápido.

 # Ejemplos Aplicados:

1. Calcular el ratio rápido con $3000 en efectivo, $2000 en valores negociables, $1000 en cuentas por cobrar,

y $2000 en pasivos corrientes.

quick.ratio(cash=3000, ms=2000, rc=1000, cl=2000)

r.continuous

#' title Convert Nominal Rate to Continuous Compounded Rate

#' description Converts a given nominal rate to a continuously compounded rate.

#' param r Nominal rate.

#' param m Number of times compounded each year.

#' return The continuously compounded rate.

#' # Example usage:

r.continuous(0.03, 4)

Esta función `r.continuous` convierte una tasa nominal dada en una tasa compuesta continuamente.

 # Parámetros:

- `r`: Tasa nominal.

- `m`: Número de veces que se compone al año.

 # Salida:

- La tasa compuesta continuamente.

 # Ejemplos Aplicados:

1. Convertir una tasa nominal del 3% compuesta 4 veces al año a una tasa compuesta continuamente.

r.continuous(0.03, 4)

r.norminal

#' title Convert Continuous Compounded Rate to Nominal Rate

#' description Converts a given continuous compounded rate to a nominal rate.

#' param rc Continuous compounded rate.

#' param m Number of desired times compounded each year.

#' return The nominal rate.

#' # Example usage:

r.norminal(0.03, 1)

r.norminal(0.03, 4)

 # Explicación en español:

Esta función `r.norminal` convierte una tasa compuesta continuamente en una tasa nominal.

 # Parámetros:

- `rc`: Tasa compuesta continuamente.

- `m`: Número de veces deseado para componer al año.

 # Salida:

- La tasa nominal.

 # Ejemplos Aplicados:

1. Convertir una tasa compuesta continuamente del 3% a una tasa nominal compuesta 1 vez al año.

r.norminal(0.03, 1)

2. Convertir una tasa compuesta continuamente del 3% a una tasa nominal compuesta 4 veces al año.

r.norminal(0.03, 4)

r.perpetuity

#' title Rate of Return for a Perpetuity

#' description Calculates the rate of return for a perpetuity.

#' param pmt Payment per period.

#' param pv Present value.

#' return The rate of return.

#' # Example usage:

r.perpetuity(pmt=4.5, pv=-75)

Explicación en español:

Esta función `r.perpetuity` calcula la tasa de retorno para una perpetuidad.

 # Parámetros:

- `pmt`: Pago por periodo.

- `pv`: Valor presente.

 # Salida:

- La tasa de retorno.

 # Ejemplos Aplicados:

1. Calcular la tasa de retorno para una perpetuidad con un pago de $4.5 por periodo y un valor presente de -$75.

r.perpetuity(pmt=4.5, pv=-75)

rcf

#' title Residual Cash Flow (RCF) Model Function

#' description Calculates the present value of future residual cash flows using the Residual Cash Flow Model.

#' Esta función calcula el valor presente de los flujos de efectivo residual futuros utilizando el modelo de Residual Cash Flow.

#' param residual_cash_flows A numeric vector of expected future residual cash flows.

#' Vector numérico de los flujos de efectivo residual esperados futuros.

#' param discount_rate The discount rate (required rate of return).

#' Tasa de descuento (tasa requerida de retorno).

#' return The estimated fair value of the stock.

#' El valor estimado justo de la acción.

#' # Example usage:

#' # Ejemplo de uso:

residual_cash_flows <- c(5, 5.5, 6, 6.5, 7)

discount_rate <- 0.1

rcf(residual_cash_flows, discount_rate)

rim

#' title Residual Income Model (RIM) Function

#' description Calculates the present value of future residual incomes using the Residual Income Model.

#' Esta función calcula el valor presente de los ingresos residuales futuros utilizando el modelo de Residual Income.

#' param residual_incomes A numeric vector of expected future residual incomes.

#' Vector numérico de los ingresos residuales futuros esperados.

#' param discount_rate The discount rate (required rate of return).

#' Tasa de descuento (tasa requerida de retorno).

#' return The estimated fair value of the stock.

#' El valor estimado justo de la acción.

#' # Example usage:

#' # Ejemplo de uso:

residual_incomes <- c(5, 5.5, 6, 6.5, 7)

discount_rate <- 0.1

rim(residual_incomes, discount_rate)

sampling.error

#' title Computing Sampling Error

#' description Computes the sampling error.

#' param sm Sample mean.

#' param mu Population mean.

#' return The sampling error.

#' # Example usage:

sampling.error(sm=0.45, mu=0.5)

 # Explicación en español:

Esta función `sampling.error` calcula el error de muestreo.

 # Parámetros:

- `sm`: Media muestral.

- `mu`: Media poblacional.

 # Salida:

- El error de muestreo.

 # Ejemplos Aplicados:

1. Calcular el error de muestreo con una media muestral de 0.45 y una media poblacional de 0.5.

sampling.error(sm=0.45, mu=0.5)

SFRatio

#' title Computing Roy's Safety-First Ratio

#' description Computes Roy's safety-first ratio.

#' param rp Portfolio return.

#' param rl Threshold level return.

#' param sd Standard deviation of portfolio returns.

#' return Roy's safety-first ratio.

#' # Example usage:

SFRatio(rp=0.09, rl=0.03, sd=0.12)

 # Explicación en español:

Esta función `SFRatio` calcula el ratio de seguridad de Roy.

 # Parámetros:

- `rp`: Retorno del portafolio.

- `rl`: Retorno del nivel de umbral.

- `sd`: Desviación estándar de los retornos del portafolio.

 # Salida:

- El ratio de seguridad de Roy.

 # Ejemplos Aplicados:

1. Calcular el ratio de seguridad de Roy con un retorno del portafolio del 9%, un retorno del nivel de umbral del 3%,

y una desviación estándar de los retornos del portafolio del 12%.

SFRatio(rp=0.09, rl=0.03, sd=0.12)

Sharpe.ratio

#' title Computing Sharpe Ratio

#' description Computes the Sharpe Ratio.

#' param rp Portfolio return.

#' param rf Risk-free return.

#' param sd Standard deviation of portfolio returns.

#' return The Sharpe Ratio.

#' # Example usage:

Sharpe.ratio(rp=0.038, rf=0.015, sd=0.07)

 # Explicación en español:

Esta función `Sharpe.ratio` calcula el Ratio de Sharpe.

 # Parámetros:

- `rp`: Retorno del portafolio.

- `rf`: Retorno libre de riesgo.

- `sd`: Desviación estándar de los retornos del portafolio.

 # Salida:

- El Ratio de Sharpe.

 # Ejemplos Aplicados:

1. Calcular el Ratio de Sharpe con un retorno del portafolio del 3.8%, un retorno libre de riesgo del 1.5%, y una desviación estándar de los retornos del portafolio del 7%.

Sharpe.ratio(rp=0.038, rf=0.015, sd=0.07)

slde

#' title Depreciation Expense Recognition – Straight-line Depreciation (SL)

#' description Allocates an equal amount of depreciation each year over the asset's useful life.

#' param cost Cost of long-lived assets.

#' param rv Residual value of the long-lived assets at the end of its useful life.

#' param t Length of the useful life.

#' return The annual depreciation expense.

#' # Example usage:

slde(cost=1200, rv=200, t=5)

 # Explicación en español:

Esta función `slde` asigna una cantidad igual de depreciación cada año durante la vida útil del activo.

 # Parámetros:

- `cost`: Costo de los activos de larga duración.

- `rv`: Valor residual de los activos de larga duración al final de su vida útil.

- `t`: Duración de la vida útil.

 # Salida:

- El gasto anual de depreciación.

 # Ejemplos Aplicados:

1. Calcular el gasto anual de depreciación para un activo con un costo de $1200, un valor residual de $200, y una vida útil de 5 años.

slde(cost=1200, rv=200, t=5)

straight_line_depreciation

#' Straight-Line Depreciation

#' This function calculates the straight-line depreciation of an asset.

#' param cost Initial cost of the asset.

#' param salvage Salvage value of the asset at the end of its useful life.

#' param life Useful life of the asset in years.

#' return A data frame with columns Year, Depreciation, and Book Value.

#' # Example usage:

#' straight_line_depreciation(10000, 2000, 5)

 # # Example usage:

straight_line_depreciation(10000, 2000, 5)

#' Esta función calcula la depreciación lineal de un activo.

#' param costo Costo inicial del activo.

#' param valor_residual Valor residual del activo al final de su vida útil.

#' param vida Vida útil del activo en años.

#' return Un data frame con las columnas Año, Depreciación y Valor en Libros.

#' # Ejemplo de uso:

#' straight_line_depreciation (10000, 2000, 5)

sum_of_years_digits_depreciation

#' Sum-of-the-Years-Digits Depreciation

#' This function calculates the sum-of-the-years-digits depreciation of an asset.

#' param cost Initial cost of the asset.

#' param salvage Salvage value of the asset at the end of its useful life.

#' param life Useful life of the asset in years.

#' return A data frame with columns Year, Depreciation, and Book Value.

#' # Example usage:

#' sum_of_years_digits_depreciation(10000, 2000, 5)

```
 schedule <- data.frame(Year = 1:life, Depreciation = depreciation, Book_Value = book_value)
 return(schedule)
```

#' Sum-of-the-Years-Digits Depreciation

#' This function calculates the sum-of-the-years-digits depreciation of an asset.

#' param cost Initial cost of the asset.

#' param salvage Salvage value of the asset at the end of its useful life.

#' param life Useful life of the asset in years.

#' return A data frame with columns Year, Depreciation, and Book Value.

#' # Example usage:

#' sum_of_years_digits_depreciation(10000, 2000, 5)

#' Esta función calcula la depreciación de la suma de los dígitos de los años de un activo.

#' param costo Costo inicial del activo.

#' param valor_residual Valor residual del activo al final de su vida útil.

#' param vida Vida útil del activo en años.

#' return Un data frame con las columnas Año, Depreciación y Valor en Libros.

#' # Ejemplo de uso:

#' sum_of_years_digits_depreciation (10000, 2000, 5)

tasa_indiferencia_lempiras

#' title Indifference Rate Function (Lempiras)

#' description Calculates the indifference rate in Lempiras given a dollar interest rate, annual devaluation rate of Lempira.

#' param tasa_en_dolares The dollar interest rate.

#' param devaluacion_anual The annual devaluation rate of Lempira.

#' return The indifference rate in Lempiras.

#' # Example usage:

#' tasa_en_dolares <- 0.06 # 6% dollar interest rate

#' devaluacion_anual <- 0.05 # 5% annual Lempira devaluation rate

#' tasa_indiferencia_lempiras(tasa_en_dolares, devaluacion_anual)

#' title Función de Tasa de Indiferencia (Lempiras)

#' description Calcula la tasa de indiferencia en Lempiras dada una tasa de interés en dólares, la tasa de devaluación anual del Lempiras.

#' param tasa_en_dolares La tasa de interés en dólares.

#' param devaluacion_anual La tasa de devaluación anual del Lempira.

#' return La tasa de indiferencia en Lempiras.

#' # Ejemplo de uso:

tasa_en_dolares <- 0.06 # Tasa de interés en dólares del 6%

devaluacion_anual <- 0.05 # Tasa de devaluación anual del Lempira del 5%

tasa_indiferencia_lempiras(tasa_en_dolares, devaluacion_anual)

total.d2e

#' title Total Debt-to-Equity Ratio

#' description Solvency ratios measure the firm's ability to satisfy its long-term obligations.

#' param td Total debt.

#' param te Total equity.

#' return The total debt-to-equity ratio.

#' # Example usage:

total.d2e(td=6000, te=20000)

 # Explicación en español:

Esta función `total.d2e` calcula el ratio de deuda total a patrimonio, un indicador de solvencia que mide la capacidad de la empresa para satisfacer sus obligaciones a largo plazo.

 # Parámetros:

- `td`: Deuda total.

- `te`: Patrimonio total.

 # Salida:

- El ratio de deuda total a patrimonio.

 # Ejemplos Aplicados:

1. Calcular el ratio de deuda total a patrimonio con una deuda total de $6000 y un patrimonio total de $20000.

total.d2e(td=6000, te=20000)

twrr

#' title Computing TWRR

#' description Computes the time-weighted rate of return (TWRR).

#' param ev Ordered ending value list.

#' param bv Ordered beginning value list.

#' param cfr Ordered cash flow received list.

#' return The time-weighted rate of return.

#' # Example usage:

twrr(ev=c(120, 260), bv=c(100, 240), cfr=c(2, 4))

 # Explicación en español:

Esta función `twrr` calcula la tasa de retorno ponderada por tiempo (TWRR).

 # Parámetros:

- `ev`: Lista ordenada de valores finales.

- `bv`: Lista ordenada de valores iniciales.

- `cfr`: Lista ordenada de flujos de caja recibidos.

 # Salida:

- La tasa de retorno ponderada por tiempo.

 # Ejemplos Aplicados:

1. Calcular la tasa de retorno ponderada por tiempo con valores finales de 120 y 260, valores iniciales de 100 y 240,

y flujos de caja recibidos de 2 y 4.

twrr(ev=c(120, 260), bv=c(100, 240), cfr=c(2, 4))

units_of_production_depreciation

#' param cost Initial cost of the asset.

#' param salvage Salvage value of the asset at the end of its useful life.

#' param total_units Expected total units of production over the asset's useful life.

#' param units_produced A numeric vector representing the units produced each year.

#' return A data frame with columns Year, Depreciation, and Book Value.

#' # Example usage:

#' units_of_production_depreciation(10000, 2000, 50000, c(10000, 12000, 8000, 15000, 5000))

#' Depreciación por Unidades de Producción

#' Esta función calcula la depreciación por unidades de producción de un activo.

#' param cost Costo inicial del activo.

#' param salvage Valor de rescate del activo al final de su vida útil.

#' param total_units Unidades totales esperadas de producción durante la vida útil del activo.

#' param units_produced Un vector numérico que representa las unidades producidas cada año.

#' return Un data frame con las columnas Año, Depreciación, y Valor en Libros.

 # Example usage:

units_of_production_depreciation(10000, 2000, 50000, c(10000, 12000, 8000, 15000, 5000))

#' # Ejemplo de uso:

units_of_production_depreciation(10000, 2000, 50000, c(10000, 12000, 8000, 15000, 5000))

volumeChart

#' title Volume Chart

#' description Shows each period's volume as a vertical line.

#' param ohlc Output from get.ohlc.yahoo or get.ohlc.google.

#' param main An overall title for the plot.

#' pagoogle <- get.ohlc.yahoo("GOOG");

data <- data.frame(

Date = as.Date('2020-01-01') + 0:9,

Volume = c(100, 200, 150, 300, 250, 400, 350, 450, 500, 550)

volumeChart(data, main = "Volume Chart Example")

#' return A ggplot object showing the volume chart.

#' # Example usage:

 # Explicación en español:

Esta función `volumeChart` muestra el volumen de cada periodo como una línea vertical.

was

#' description Calculate weighted average shares – weighted average number of common shares.

#' param ns n x 1 vector of number of shares.

#' param nm n x 1 vector of number of months related to ns.

#' return The weighted average number of common shares.

#' # Example usage:

#' s <- c(10000, 2000)

#' m <- c(12, 6)

#' was(ns = s, nm = m)

s <- c(11000, 4400, -3000)

m <- c(12, 9, 4)

was(ns = s, nm = m)

 # Explicación en español:

Esta función `was` calcula el promedio ponderado de acciones comunes, dado un vector de número de acciones y un vector de número de meses relacionados con cada número de acciones.

 # Parámetros:

- `ns`: Vector de n x 1 de número de acciones.

- `nm`: Vector de n x 1 de número de meses relacionados con `ns`.

 # Ejemplos Aplicados:

1. Calcular el promedio ponderado de acciones comunes con 10000 acciones durante 12 meses y 2000 acciones durante 6 meses.

s <- c(10000, 2000)

m <- c(12, 6)

was(ns = s, nm = m)

2. Calcular el promedio ponderado de acciones comunes con 11000 acciones durante 12 meses, 4400 acciones durante 9 meses

y -3000 acciones durante 4 meses.

s <- c(11000, 4400, -3000)

m <- c(12, 9, 4)

was(ns = s, nm = m)

wpr

#' title Weighted Mean as a Portfolio Return

#' description Calculate the weighted mean as a portfolio return.

#' param r Returns of the individual assets in the portfolio.

#' param w Corresponding weights associated with each of the individual assets.

#' return The weighted mean portfolio return.

#' # Example usage:

wpr(r = c(0.12, 0.07, 0.03), w = c(0.5, 0.4, 0.1))

 # Explicación en español:

Esta función `wpr` calcula la media ponderada como el retorno de una cartera, dado un vector de retornos de los activos individuales

en la cartera y un vector de pesos correspondientes asociados a cada uno de los activos individuales.

 # Parámetros:

- `r`: Retornos de los activos individuales en la cartera.

- `w`: Pesos correspondientes asociados a cada uno de los activos individuales.

 # Ejemplos Aplicados:

1. Calcular la media ponderada del retorno de una cartera con retornos de 0.12, 0.07 y 0.03, y pesos de 0.5, 0.4 y 0.1 respectivamente.

wpr(r = c(0.12, 0.07, 0.03), w = c(0.5, 0.4, 0.1))