# Finanzas en R

**Notebook Ejercicios 02**

Sebastián Egaña Santibáñez     Nicolás Leiva Díaz

---

## Enlaces del profesor

https://segana.netlify.app

https://github.com/sebaegana

https://www.linkedin.com/in/sebastian-egana-santibanez/

---

### Métricas básicas

1. Calculamos la mediana de un vector:

### Solución:

```
# Sample data
data <- c(10, 15, 20, 25, 30)

# Calculating median
median_value <- median(data)
print(median_value)
```

```
[1] 20
```

Sin usar funciones propias de R:

```r
# Sample data
data <- c(10, 15, 20, 25, 30)

# Sorting the data
sorted_data <- sort(data)

# Calculating the median
n <- length(sorted_data)
if (n %% 2 == 0) {
  median_value <- (sorted_data[n/2] + sorted_data[n/2 + 1]) / 2
} else {
  median_value <- sorted_data[(n + 1) / 2]
}

print(median_value)
```

```
[1] 20
```

2. Calculamos la media de un vector

```r
# Sample data
data <- c(10, 15, 20, 25, 30)

# Calculating average (mean)
average_value <- mean(data)
print(average_value)
```

```
[1] 20
```

Sin usar funciones propias de R:

```r
# Sample data
data <- c(10, 15, 20, 25, 30)

# Calculating the sum
sum_value <- sum(data)
```

```r
# Calculating the number of elements
n <- length(data)

# Calculating the average
average_value <- sum_value / n

print(average_value)
```

```
[1] 20
```

3. Calculamos la desviación estándar

```r
# Sample data
data <- c(10, 15, 20, 25, 30)

# Calculating standard deviation
std_dev_value <- sd(data)
print(std_dev_value)
```

```
[1] 7.905694
```

Sin usar funciones propias de R:

```r
# Sample data
data <- c(10, 15, 20, 25, 30)

# Calculating the average (mean)
mean_value <- sum(data) / length(data)

# Calculating the squared differences from the mean
squared_diff <- (data - mean_value)^2

# Calculating the variance
variance <- sum(squared_diff) / length(data)

# Calculating the standard deviation
std_dev_value <- sqrt(variance)

print(std_dev_value)
```

```
[1] 7.071068
```

### Desempeño de un activo

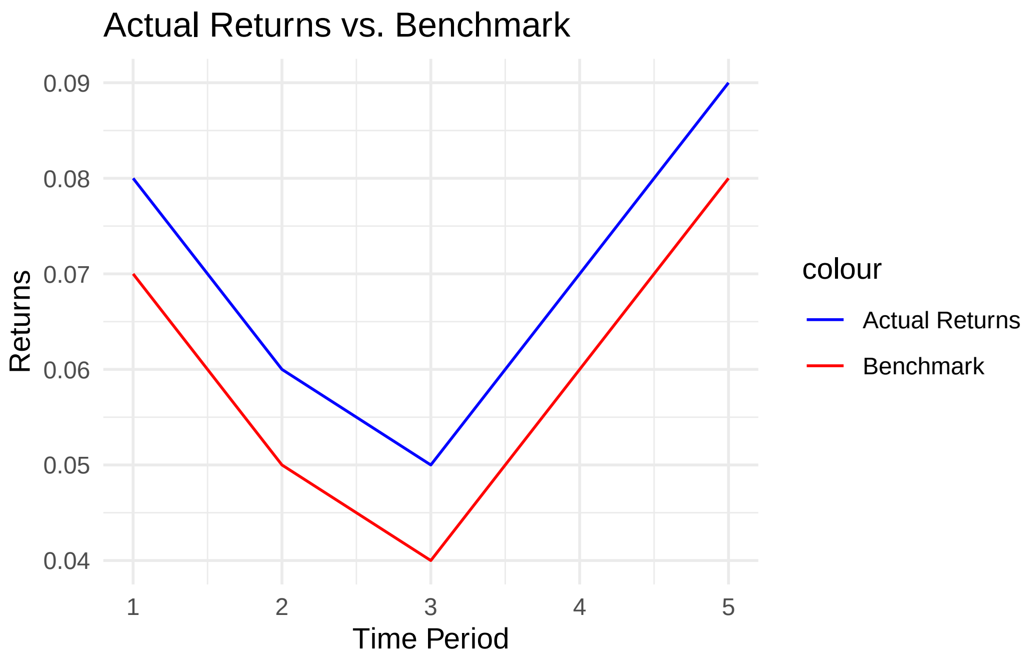
Veamos el desempeño de un activo financiero versus su benchmark conocido:

```r
# Load necessary library
library(ggplot2)

# Sample data
actual_returns <- c(0.08, 0.06, 0.05, 0.07, 0.09)  # Actual returns of the investment
benchmark_returns <- c(0.07, 0.05, 0.04, 0.06, 0.08)  # Returns of the benchmark
dates <- 1:5  # Assuming 5 time periods

# Create a data frame
data <- data.frame(dates, actual_returns, benchmark_returns)

# Plot using ggplot
ggplot(data, aes(x = dates)) +
  geom_line(aes(y = actual_returns, color = "Actual Returns")) +
  geom_line(aes(y = benchmark_returns, color = "Benchmark")) +
  labs(x = "Time Period", y = "Returns", title = "Actual Returns vs. Benchmark") +
  scale_color_manual(values = c("Actual Returns" = "blue", "Benchmark" = "red")) +
  theme_minimal()
```



Calculemos algunas métricas de comparación:

1. Ex-post Alpha:

$$\text{Ex-post Alpha} = \frac{1}{n}\sum_{i=1}^{n}(R_i - R_{\text{benchmark},i})$$

```r
# Sample data
actual_returns <- c(0.08, 0.06, 0.05, 0.07, 0.09)  # Actual returns of the investment
benchmark_returns <- c(0.07, 0.05, 0.04, 0.06, 0.08)  # Returns of the benchmark

# Calculating ex-post alpha
ex_post_alpha <- mean(actual_returns - benchmark_returns)
print(ex_post_alpha)
```

```
[1] 0.01
```

Consideramos que un Alpha positivo corresponde un desempeño por encima del benchmark.

2. Treynor ratio:

$$\text{Treynor Ratio} = \frac{\text{Average Excess Return}}{\beta}$$

```r
# Sample data
excess_returns <- actual_returns - 0.03  # Assuming risk-free rate of 3%
beta <- 1.2  # Example beta

# Calculating Treynor ratio
treynor_ratio <- mean(excess_returns) / beta
print(treynor_ratio)
```

```
[1] 0.03333333
```

En el caso del benchmark:

```r
# Sample data
excess_returns_01 <- benchmark_returns - 0.03  # Assuming risk-free rate of 3%
beta <- 1.2  # Example beta
```

```r
# Calculating Treynor ratio
treynor_ratio_01 <- mean(excess_returns_01) / beta
print(treynor_ratio_01)
```

```
[1] 0.025
```

La comparación implica un menor retorno en relación a cada unidad de riesgo, siendo el riesgo medido en base al beta.

3. Sharpe ratio:

$$\text{Sharpe Ratio} = \frac{\text{Average Excess Return}}{\text{Standard Deviation of Returns}}$$

```r
# Sample data
risk_free_rate <- 0.03

# Calculating excess returns
excess_returns <- actual_returns - risk_free_rate

# Calculating Sharpe ratio
sharpe_ratio <- mean(excess_returns) / sd(actual_returns)
print(sharpe_ratio)
```

```
[1] 2.529822
```

Para el benchmark:

```r
# Sample data
risk_free_rate <- 0.03

# Calculating excess returns
excess_returns_01 <- benchmark_returns - risk_free_rate

# Calculating Sharpe ratio
sharpe_ratio_01 <- mean(excess_returns_01) / sd(benchmark_returns)
print(sharpe_ratio_01)
```

```
[1] 1.897367
```

Reflexión parecida a la del Treynor ratio.

4. Information ratio:

$$\text{Information Ratio} = \frac{\text{Average Active Return}}{\text{Tracking Error}}$$

```r
# Sample data
tracking_error <- sd(actual_returns - benchmark_returns)

# Calculating information ratio
information_ratio <- mean(actual_returns - benchmark_returns) / tracking_error
print(information_ratio)
```

```
[1] 1.598814e+15
```

Esto corresponde una metríca distinta de evaluación.

5. Expected shortfall

$$\text{Expected Shortfall (ES)} = \frac{1}{\alpha}\sum_{i=1}^{\alpha n} R_{(i)}$$

```r
# Sample data
alpha <- 0.05  # Confidence level (e.g., 95%)

# Calculating expected shortfall
sorted_returns <- sort(actual_returns)
n <- length(sorted_returns)
es_index <- ceiling(alpha * n)
expected_shortfall <- mean(sorted_returns[1:es_index])
print(expected_shortfall)
```

```
[1] 0.05
```