Script-Empirical-Finance.R

```r
library(dplyr)

library(moments)
library(readxl)

returns <- read_excel("C:/Users/acer/Downloads/returns.xlsx")

#changing name of returns_France equity to returns_France_equity
names(returns)[names(returns) == "returns_France equity"] <-
"returns_France_equity"
View(returns)

# Cleaning data (removing the first NA row)
returns_clean <- returns %>%

filter(!is.na(returns_MSCI),!is.na(`returns_France_equity`),!is.na(returns_Go
ld))

#Exercise 2

# Descriptive statistics function
desc_stats <- function(x) { x <- x[is.finite(x)]
  c(Mean = mean(x),
    Median = median(x),
    Min = min(x),
    Max = max(x),
    SD = sd(x),
    Skewness = skewness(x),
    ExcessKurtosis = kurtosis(x) - 3,
    P5 = as.numeric(quantile(x, 0.05)),
    P95 = as.numeric(quantile(x, 0.95)) ) }

# table (rows = ETFs)
stats_table <- rbind(
  MSCI_World    = desc_stats(returns_clean$returns_MSCI),
  France_Equity = desc_stats(returns_clean$`returns_France_equity`),
  Gold          = desc_stats(returns_clean$returns_Gold))

# Round for presentation
stats_table <- round(stats_table, 6)
stats_table

##                   Mean   Median       Min      Max       SD  Skewness
## MSCI_World    0.001912 0.002947 -0.089325 0.049227 0.019459 -0.635521
## France_Equity 0.001362 0.002176 -0.101225 0.058158 0.022539 -0.706354
## Gold          0.006136 0.001500 -0.097652 0.160322 0.048904  0.333747
```

```
##                  ExcessKurtosis        P5       P95
## MSCI_World             2.101434 -0.031000 0.032449
## France_Equity          2.026537 -0.035814 0.036369
## Gold                  -0.057786 -0.069690 0.099373
```

```r
#Testing normality (Jarque Bera test)
library(tseries)

jb_MSCI   <- jarque.bera.test(returns_clean$returns_MSCI)
jb_France <- jarque.bera.test(returns_clean$returns_France_equity)
jb_Gold   <- jarque.bera.test(returns_clean$returns_Gold)

jb_MSCI
```

```
##
##  Jarque Bera Test
##
## data:  returns_clean$returns_MSCI
## X-squared = 53.279, df = 2, p-value = 2.695e-12
```

```r
jb_France
```

```
##
##  Jarque Bera Test
##
## data:  returns_clean$returns_France_equity
## X-squared = 53.906, df = 2, p-value = 1.97e-12
```

```r
jb_Gold
```

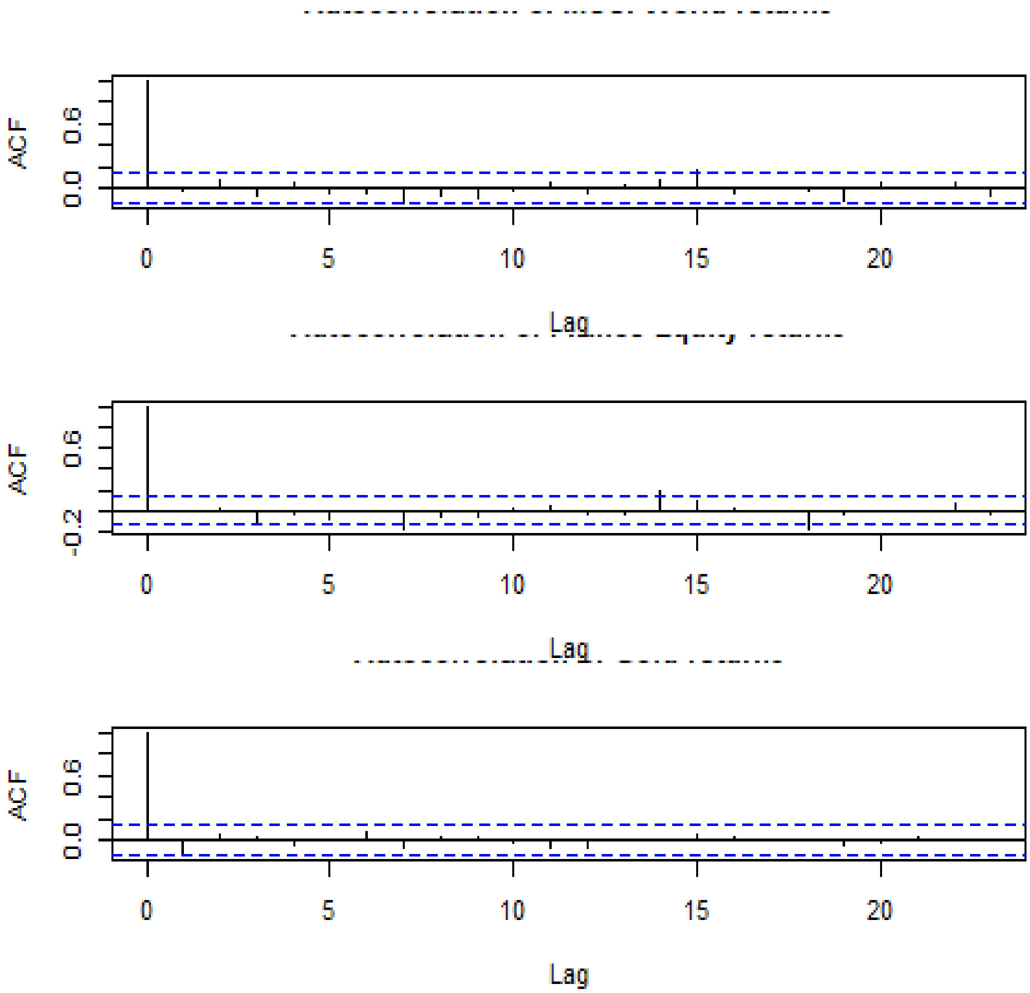
```
##
##  Jarque Bera Test
##
## data:  returns_clean$returns_Gold
## X-squared = 3.9652, df = 2, p-value = 0.1377
```

```r
# Correlation matrix of returns
cor_matrix <- cor(
  returns[, c("returns_MSCI","returns_France_equity","returns_Gold")], use =
"complete.obs")

round(cor_matrix, 3)
```

```
##                         returns_MSCI returns_France_equity returns_Gold
## returns_MSCI                   1.000                 0.653        0.223
## returns_France_equity          0.653                 1.000        0.188
## returns_Gold                   0.223                 0.188        1.000
```

```r
# Calculating autocorrelations up to lag 20 (returns)
acf_MSCI   <- acf(returns_clean$returns_MSCI, plot = FALSE, lag.max = 20)
acf_France <- acf(returns_clean$returns_France_equity, plot = FALSE, lag.max
= 20)
acf_Gold   <- acf(returns_clean$returns_Gold, plot = FALSE, lag.max = 20)

# Extracting the values (removing lag 0)
lags <- 1:20
ac_table <- data.frame(
  Lag = lags,
  ACF_MSCI   = as.numeric(acf_MSCI$acf)[-1],
  ACF_France = as.numeric(acf_France$acf)[-1],
  ACF_Gold   = as.numeric(acf_Gold$acf)[-1])

round(ac_table, 3)

##    Lag ACF_MSCI ACF_France ACF_Gold
## 1    1   -0.011     -0.012   -0.115
## 2    2    0.079      0.018    0.045
## 3    3   -0.081     -0.126    0.023
## 4    4    0.055     -0.035   -0.038
## 5    5   -0.039     -0.078    0.005
## 6    6   -0.042      0.003    0.064
## 7    7   -0.138     -0.167   -0.078
## 8    8   -0.070     -0.043    0.038
## 9    9   -0.090     -0.040    0.031
## 10  10   -0.031      0.030   -0.019
## 11  11    0.043      0.056   -0.064
## 12  12   -0.043     -0.031   -0.076
## 13  13    0.029     -0.017    0.015
## 14  14    0.073      0.197   -0.001
## 15  15    0.166      0.099    0.046
## 16  16   -0.040      0.022    0.028
## 17  17    0.000     -0.010   -0.005
## 18  18   -0.034     -0.170   -0.002
## 19  19   -0.117     -0.027   -0.040
## 20  20    0.046      0.004   -0.017



# Autocorrelation plots (ACF)
par(mfrow = c(3, 1), mar = c(4, 4, 2, 1))  # 3 plots stacked + margins

acf(returns_clean$returns_MSCI,
    main = "Autocorrelation of MSCI World returns")

acf(returns_clean$returns_France_equity,
    main = "Autocorrelation of France Equity returns")
```

```r
acf(returns_clean$returns_Gold,
    main = "Autocorrelation of Gold returns")
```



```r
library(readxl)

library(dplyr)

returns <- read_excel("C:/Users/user/OneDrive/Bureau/returns.xlsx")
View(returns)

# Exercise 3
# A. Estimate the Jensen’s equations of your “narrow” ETF and compare them to
the market (large ETF). Test the reliability of the coefficients, explain you
r methods, and the results.

rf <- 0

df <- returns_clean %>%
  dplyr::mutate(
    M_ex  = returns_MSCI - rf,
    F_ex  = returns_France_equity - rf,
    G_ex  = returns_Gold - rf)

# France vs Market
m_fr <- lm(F_ex ~ M_ex, data = df)
summary(m_fr)

##
## Call:
## lm(formula = F_ex ~ M_ex, data = df)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.099598 -0.008384 -0.000988  0.010376  0.048414
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.0000832  0.0011813   -0.07    0.944
## M_ex         0.7559236  0.0605574   12.48   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.01712 on 210 degrees of freedom
## Multiple R-squared:  0.4259, Adjusted R-squared:  0.4232
## F-statistic: 155.8 on 1 and 210 DF,  p-value: < 2.2e-16

# Gold vs Market
m_G <- lm(G_ex ~ M_ex, data = df)
summary(m_G)

##
## Call:
## lm(formula = G_ex ~ M_ex, data = df)
##
```

```
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.116702 -0.032807 -0.005731  0.031993  0.163212
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.005066   0.003298   1.536   0.1260
## M_ex        0.559420   0.169071   3.309   0.0011 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.04779 on 210 degrees of freedom
## Multiple R-squared:  0.04955,    Adjusted R-squared:  0.04502
## F-statistic: 10.95 on 1 and 210 DF,  p-value: 0.001103

# B. Test the stability of the coefficient, explain the method you use, and t
he results.

library(sandwich)

library(lmtest)

library(strucchange)

# Define breakpoint (mid-sample)
break_point <- floor(nrow(df) / 2)

# For France narrow ETF
chow_fr <- sctest(F_ex ~ M_ex, type = "Chow", point = break_point, data = df)
chow_fr

##
##  Chow test
##
## data:  F_ex ~ M_ex
## F = 1.9637, p-value = 0.1429

# For Gold narrow ETF
chow_G <- sctest(G_ex ~ M_ex, type = "Chow", point = break_point, data = df)
chow_G

##
##  Chow test
##
## data:  G_ex ~ M_ex
## F = 1.6249, p-value = 0.1994

# Exercise 4 (6 points)
# A. Using alternative model-based measure(s) of performance, estimate the pe
rformance of your two “narrow” ETFs. Explain the method(s) you use and commen
t on the results.
```

```r
# We compute mean returns and volatility
# Mean weekly returns
mean_fr <- mean(df$F_ex)
mean_G <- mean(df$G_ex)

# Volatility (standard deviation)
sd_fr <- sd(df$F_ex)
sd_G <- sd(df$G_ex)

#Sharpe ratios
sharpe_fr <- mean_fr / sd_fr
sharpe_G <- mean_G / sd_G

#Treynor Ratios (using estimated betas)
beta_fr <- coef(m_fr)["M_ex"]
beta_G <- coef(m_G)["M_ex"]

treynor_fr <- mean_fr / beta_fr
treynor_G <- mean_G / beta_G



#Summary table

performance <- data.frame(
  ETF = c("France Equity", "Gold"),
  Sharpe = c(sharpe_fr, sharpe_G),
  Treynor = c(treynor_fr, treynor_G))

performance[, -1] <- round(performance[, -1], 4)
performance

##             ETF Sharpe Treynor
## 1 France Equity 0.0604  0.0018
## 2          Gold 0.1255  0.0110

# B. Use measure(s) of performance adjusted to the risk to assess the perform
ance of your “narrow” ETFs. Compare them to the market. Explain the methods y
ou use and comment on the results.

df$M_ex <- returns_clean$returns_MSCI

mean_m <- mean(df$M_ex)
sd_m   <- sd(df$M_ex)

sharpe_m <- mean_m / sd_m

treynor_m <- mean_m
```

```r
comparison <- data.frame(
  ETF = c("Market (MSCI World)", "France Equity", "Gold"),
  Sharpe = round(c(sharpe_m, sharpe_fr, sharpe_G), 4),
  Treynor = round(c(treynor_m, treynor_fr, treynor_G), 4))

comparison
```

```
##                   ETF Sharpe Treynor
## 1 Market (MSCI World) 0.0982  0.0019
## 2       France Equity 0.0604  0.0018
## 3                Gold 0.1255  0.0110
```