Istanbul Stock Exchange Forecast

Faith Lucy Kirabo

2026-01-17

```
# install.packages('tseries')
install.packages("~/Downloads/DataCombine_0.2.21.tar", repos = NULL, type = "source")
```

```
# Load the training data
data <- read.csv("~/Desktop/MSADS Courses/Time Series/Datasets/data_akbilgic - train.
csv")

# Preview Data Structure
str(data)
```

```
## 'data.frame':    526 obs. of  7 variables:
##  $ date   : chr  "5-Jan-09" "6-Jan-09" "7-Jan-09" "8-Jan-09" ...
##  $ ISE    : num  0.03838 0.03181 -0.02635 -0.08472 0.00966 ...
##  $ SP     : num  -0.00468 0.00779 -0.03047 0.00339 -0.02153 ...
##  $ DAX    : num  0.00219 0.00846 -0.01783 -0.01173 -0.01987 ...
##  $ FTSE   : num  0.003894 0.012866 -0.028735 -0.000466 -0.01271 ...
##  $ NIKKEI : num  0 0.00416 0.01729 -0.04006 -0.00447 ...
##  $ BOVESPA: num  0.03119 0.01892 -0.0359 0.02828 -0.00976 ...
```

```
head(data)
```

```
##        date          ISE           SP          DAX         FTSE       NIKKEI
## 1  5-Jan-09  0.038376187 -0.004679315  0.002193419  0.003894376  0.000000000
## 2  6-Jan-09  0.031812743  0.007786738  0.008455341  0.012865611  0.004162452
## 3  7-Jan-09 -0.026352966 -0.030469134 -0.017833062 -0.028734593  0.017292932
## 4  8-Jan-09 -0.084715902  0.003391364 -0.011726277 -0.000465999 -0.040061309
## 5  9-Jan-09  0.009658112 -0.021533208 -0.019872754 -0.012709717 -0.004473502
## 6 12-Jan-09 -0.042361155 -0.022822626 -0.013525735 -0.005025533 -0.049038532
##       BOVESPA
## 1  0.03119023
## 2  0.01891958
## 3 -0.03589858
## 4  0.02828315
## 5 -0.00976388
## 6 -0.05384947
```

1. Determine if all the TS are stationary - what do you observe and why?

- **Qualitatively:** Since all variables are daily returns (not index levels), we expect them to fluctuate around a constant mean with relatively constant variance hence likely stationary.
- **Quantitatively** (using ADF and KPSS from package tseries)

```
# Create a list of column names for looping
returns <- data[, c("ISE", "SP", "DAX", "FTSE", "NIKKEI", "BOVESPA")]
ts_names <-colnames(returns)

# Initialize storage
adf_results <- list()
kpss_results <- list()

# Loop through columns to perform adf and kpss tests
for (name in ts_names){

  series <- na.omit(returns[[name]])

  # ADF Test (null hypothesis: has unit root i.e. non-stationary)
  adf_results[[name]] <- adf.test(series)

  # KPSS test (null hypothesis: stationary)
  kpss_results[[name]] <- kpss.test(series)
}

adf_results
```

```
## $ISE
##
##  Augmented Dickey-Fuller Test
##
## data:  series
## Dickey-Fuller = -7.8918, Lag order = 8, p-value = 0.01
## alternative hypothesis: stationary
##
##
## $SP
##
##  Augmented Dickey-Fuller Test
##
## data:  series
## Dickey-Fuller = -8.0713, Lag order = 8, p-value = 0.01
## alternative hypothesis: stationary
##
##
## $DAX
##
##  Augmented Dickey-Fuller Test
##
## data:  series
```

```
## Dickey-Fuller = -8.0694, Lag order = 8, p-value = 0.01
## alternative hypothesis: stationary
##
##
## $FTSE
##
##  Augmented Dickey-Fuller Test
##
## data:  series
## Dickey-Fuller = -7.7724, Lag order = 8, p-value = 0.01
## alternative hypothesis: stationary
##
##
## $NIKKEI
##
##  Augmented Dickey-Fuller Test
##
## data:  series
## Dickey-Fuller = -7.698, Lag order = 8, p-value = 0.01
## alternative hypothesis: stationary
##
##
## $BOVESPA
##
##  Augmented Dickey-Fuller Test
##
## data:  series
## Dickey-Fuller = -7.6107, Lag order = 8, p-value = 0.01
## alternative hypothesis: stationary
```

```
kpss_results
```

```
## $ISE
##
##  KPSS Test for Level Stationarity
##
## data:  series
## KPSS Level = 0.17143, Truncation lag parameter = 6, p-value = 0.1
##
##
## $SP
##
##  KPSS Test for Level Stationarity
##
## data:  series
## KPSS Level = 0.087735, Truncation lag parameter = 6, p-value = 0.1
##
##
## $DAX
##
##  KPSS Test for Level Stationarity
##
## data:  series
## KPSS Level = 0.085319, Truncation lag parameter = 6, p-value = 0.1
##
##
## $FTSE
##
##  KPSS Test for Level Stationarity
##
## data:  series
## KPSS Level = 0.08608, Truncation lag parameter = 6, p-value = 0.1
##
##
## $NIKKEI
##
##  KPSS Test for Level Stationarity
##
## data:  series
## KPSS Level = 0.062996, Truncation lag parameter = 6, p-value = 0.1
##
##
## $BOVESPA
##
##  KPSS Test for Level Stationarity
##
## data:  series
## KPSS Level = 0.27703, Truncation lag parameter = 6, p-value = 0.1
```

Quantitatively, the ADF test rejects the null hypothesis of a unit root (p-values < 0.01) for all series, while the KPSS test fails to reject the null of stationarity (p-values > 0.10). These results strongly suggest stationarity for all time series.

2. Linearly regress ISE against the remaining 5 stock index returns - determine which coefficients are equal or better than 0.02 (*) level of significance?

```
lr_model <- lm(
  ISE ~ SP + DAX + FTSE + NIKKEI + BOVESPA,
  data = data
)

summary(lr_model)
```

```
##
## Call:
## lm(formula = ISE ~ SP + DAX + FTSE + NIKKEI + BOVESPA, data = data)
##
## Residuals:
##       Min         1Q     Median         3Q        Max
## -0.071180 -0.009248  0.000083  0.009304  0.051863
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.0008833  0.0006640   1.330 0.183979
## SP          -0.0607521  0.0770823  -0.788 0.430970
## DAX          0.3417440  0.0961243   3.555 0.000412 ***
## FTSE         0.6033493  0.1077621   5.599 3.50e-08 ***
## NIKKEI       0.3266529  0.0462163   7.068 5.09e-12 ***
## BOVESPA      0.1117630  0.0626647   1.784 0.075087 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0152 on 520 degrees of freedom
## Multiple R-squared:  0.493,  Adjusted R-squared:  0.4881
## F-statistic: 101.1 on 5 and 520 DF,  p-value: < 2.2e-16
```

Significant Coefficients (p-value ≤ 0.02): At the 2% significance level, three predictors show statistically significant relationships with ISE returns:

- DAX (p = 0.000412): Coefficient = 0.342, indicating that a 1% increase in DAX returns is associated with a 0.342% increase in ISE returns
- FTSE (p = 3.50e-08): Coefficient = 0.603, the strongest predictor, showing a 1% increase in FTSE returns corresponds to a 0.603% increase in ISE returns
- NIKKEI (p = 5.09e-12): Coefficient = 0.327, demonstrating that a 1% increase in NIKKEI returns is associated with a 0.327% increase in ISE returns

Non-Significant Coefficients (p-value > 0.02):

- SP (p = 0.431): Not significant; no reliable linear relationship detected
- BOVESPA (p = 0.075): Not significant at the 2% level (though marginally significant at 10%)

Model Performance: The model explains approximately **49.3%** of the variance in ISE returns ($R^2 = 0.493$), with an overall F-statistic that is highly significant ($p < 2.2e-16$), confirming that the model as a whole provides meaningful explanatory power. The significant coefficients for European and Asian indices *(DAX, FTSE, NIKKEI)* suggest that the Istanbul Stock Exchange is more strongly influenced by these markets than by the U.S. *(SP)* or South American *(BOVESPA)* markets during the same trading day.

3. For the non-significant coefficients, continue to lag by 1 day until all coefficients are better than 0.02 (*) level of significance.

Use slide() function from package DataCombine. Remember you will need to lag, so you slideBy = -1 each step. How many lags are needed for each independent variable?

```
# Ensure data is in time order before lagging
data$date <- as.Date(data$date, format = "%d-%b-%y")
data <- data[order(data$date), ]
row.names(data) <- NULL

# Identify variables with p-value>0.02, not significant at 2%
lag_vars <- c("BOVESPA","SP")

# Store number of lags needed per variable
lag_count <- list()

# Start with original dataset
data_lag <- data

# Store the optimal lag for each variable first
# THEN build final model with ALL optimal lags together
# Approach: Lag variables sequentially, updating the dataset each time

# Loop through lag columns
for (var in lag_vars){

  lag <- 0   # counts how many days we lag the var
  p_val <- 1 # initial placeholder
  current_var <- var # track which variable we're currently using

  while (p_val > 0.02){
    lag <- lag + 1
    lag_name <- paste0(var, "_lag", lag)

    #Create lagged variable
    data_lag <- slide(
```

```
      data_lag,         # dataset
      Var = current_var,# use previously lagged variable
      slideBy = -1,     # lag back one day
      NewVar = lag_name # name of the new lagged column
    )

    current_var <- lag_name # Update current_var to point to the newly created lagged
variable
    temp_data <-na.omit(data_lag) # remove NA rows from lagging

    # Rebuild regression formula dynamically
    # Start with variables that never need lagging
    predictors <- c("DAX","FTSE","NIKKEI")

    for (v in lag_vars) {
      if (v == var) {
        # Current variable being tested - use its current lag
        predictors <- c(predictors, lag_name)
      } else if (!is.null(lag_count[[v]])) {
        # Variable already processed - use its optimal lag
        predictors <- c(predictors, paste0(v, "_lag", lag_count[[v]]))
      } else {
        # Use original if not yet processed
        predictors <- c(predictors, v)
      }
    }

    formula_text <- paste("ISE ~", paste(predictors, collapse = " + "))
    model_temp <- lm(as.formula(formula_text), data = temp_data)

    # Extract new p-value of lagged coefficient
    p_val <- summary(model_temp)$coefficients[lag_name, 4]

    # Stop conditions
    if (!is.na(p_val) && p_val <= 0.02) {
      cat(sprintf("%s requires %d lag(s), p-value = %.4f\n", var, lag, p_val))
      cat(sprintf("  Tested in model: %s\n", formula_text))
      break
    }
    if (lag > 10){
      cat(sprintf("Warning: %s exceeded 10 lags, stopping at lag %d\n", var, lag))
      break
      } # Safety stop
     }

  lag_count[[var]] <- lag # store lag result
}
```

```
##
## Remember to put data_lag in time order before running.
```

```
##
## Lagging BOVESPA by 1 time units.
```

```
## BOVESPA requires 1 lag(s), p-value = 0.0000
##   Tested in model: ISE ~ DAX + FTSE + NIKKEI + BOVESPA_lag1 + SP
```

```
##
## Remember to put data_lag in time order before running.
```

```
##
## Lagging SP by 1 time units.
```

```
##
## Remember to put data_lag in time order before running.
```

```
##
## Lagging SP_lag1 by 1 time units.
```

```
## SP requires 2 lag(s), p-value = 0.0179
##   Tested in model: ISE ~ DAX + FTSE + NIKKEI + BOVESPA_lag1 + SP_lag2
```

```
print(lag_count) # print final output
```

```
## $BOVESPA
## [1] 1
##
## $SP
## [1] 2
```

```
# Show final model summary with optimal lags
final_predictors <- c("DAX", "FTSE", "NIKKEI")
for (var in lag_vars) {
  final_predictors <- c(final_predictors, paste0(var, "_lag", lag_count[[var]]))
}

formula_final <- paste("ISE ~", paste(final_predictors, collapse = " + "))
model_final <- lm(as.formula(formula_final), data = na.omit(data_lag))
summary(model_final)
```

```
##
## Call:
## lm(formula = as.formula(formula_final), data = na.omit(data_lag))
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.063412 -0.009491  0.000468  0.008739  0.050599
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.0007513  0.0006491   1.157 0.247635
## DAX           0.3355329  0.0890788   3.767 0.000184 ***
## FTSE          0.6368064  0.1024472   6.216 1.05e-09 ***
## NIKKEI        0.2395311  0.0489366   4.895 1.32e-06 ***
## BOVESPA_lag1  0.2057244  0.0452856   4.543 6.91e-06 ***
## SP_lag2      -0.1082670  0.0455658  -2.376 0.017861 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.01481 on 518 degrees of freedom
## Multiple R-squared:  0.516,  Adjusted R-squared:  0.5113
## F-statistic: 110.4 on 5 and 518 DF,  p-value: < 2.2e-16
```

Lagged regression results: Non-significant predictors were iteratively lagged using slide() with slideBy = -1 (one day back) until coefficients were significant at the 2% level. The lagging was performed sequentially:

BOVESPA was lagged first and achieved significance with a 1-day lag ($p < 0.0001$) when tested alongside the original SP. SP was then lagged with BOVESPA_lag1 already in the model. SP required a 2-day lag to achieve significance ($p = 0.0179$).

Final Model Results:

- All predictors are now significant at the 2% level
- DAX, FTSE, NIKKEI: Contemporaneous (same-day) effects remain significant
- BOVESPA_lag1: 1-day lag, coefficient = 0.206 ($p < 0.0001$)
- SP_lag2: 2-day lag, coefficient = -0.108 ($p = 0.0179$)

Interpretation: The differential lag structure reflects the timing of information transmission from international markets to the Istanbul Stock Exchange. BOVESPA's 1-day lag suggests relatively direct transmission of emerging market sentiment, while SP's 2-day lag (with a negative coefficient) indicates a more delayed and complex relationship, possibly reflecting time zone differences, indirect transmission channels, or contrarian market dynamics. The negative coefficient for SP_lag2 is particularly noteworthy, suggesting that after controlling for other markets, strong US market performance two days prior is associated with slightly lower ISE returns, which may reflect profit-taking or regional capital flow patterns. The model's R^2 increased from 0.493 to 0.516, indicating that the lagged structure provides improved explanatory power for ISE returns.

4. Find correlations between ISE and each independent variable. Sum the square of the correlations. How does it compare to R-squared from #2? Why?

```
# Using Pearson's correlation
vars <- c("SP", "DAX", "FTSE", "NIKKEI", "BOVESPA")

# Compute correlation with ISE
cors<- sapply(vars, function(v) cor(data$ISE,data[[v]]))

# square the correlation
cors_sq <- cors^2

# sum the squares
cor_sum <- sum(cors_sq)

cat("\nIndividual squared correlations:\n")
```

```
##
## Individual squared correlations:
```

```
print(cors_sq)
```

```
##        SP       DAX      FTSE    NIKKEI   BOVESPA
## 0.2019759 0.3982557 0.4205850 0.1537869 0.2000873
```

```
cat("\nSum of squared correlations:", cor_sum, "\n")
```

```
##
## Sum of squared correlations: 1.374691
```

```
cat("R-squared from multiple regression:", summary(lr_model)$r.squared, "\n")
```

```
## R-squared from multiple regression: 0.4930134
```

The sum of squared correlations **(1.37)** exceeds the R-squared **(0.493)** because:

1. Each squared correlation (r^2) represents the proportion of ISE variance explained by that predictor ALONE in a simple regression.
2. Since the global stock indices are correlated with each other (multicollinearity), they share overlapping explanatory power.
3. When we sum the individual r^2 values, we double-count this overlap.

4. The multiple regression R^2 accounts for this overlap and reports the TOTAL unique variance explained (bounded between 0 and 1).
5. The sum of squared correlations can exceed 1.0 (as it does here) precisely because of this multicollinearity-induced redundancy.

5. Use the predict() function using the lm regression object from #2 to forecast the ISE index for the next 10 days.

```
# Refit and store the same linear regression model from #2
model_ise <- lm(ISE ~ SP + DAX + FTSE + NIKKEI + BOVESPA, data = data)

# Last observed predictor values
last_obs <- tail(data,1)

# Create dataset with constant predictors (naive assumption)
future_data <- last_obs[rep(1,10),
                        c("SP", "DAX", "FTSE", "NIKKEI", "BOVESPA")]

# Generate forecasts with PREDICTION intervals (not confidence intervals)
# Prediction intervals are wider and more appropriate for forecasting
ise_forecast_pred <- predict(model_ise,
                             newdata = future_data,
                             interval = "prediction",
                             level = 0.95
                             )
# Add day labels
forecast_df <- data.frame(
  Day = 1:10,
  Forecast = ise_forecast_pred[, "fit"],
  Lower_95 = ise_forecast_pred[, "lwr"],
  Upper_95 = ise_forecast_pred[, "upr"],
  row.names = NULL
)

print(forecast_df)
```

```
##    Day    Forecast   Lower_95   Upper_95
## 1    1 0.008538015 -0.0213554 0.03843143
## 2    2 0.008538015 -0.0213554 0.03843143
## 3    3 0.008538015 -0.0213554 0.03843143
## 4    4 0.008538015 -0.0213554 0.03843143
## 5    5 0.008538015 -0.0213554 0.03843143
## 6    6 0.008538015 -0.0213554 0.03843143
## 7    7 0.008538015 -0.0213554 0.03843143
## 8    8 0.008538015 -0.0213554 0.03843143
## 9    9 0.008538015 -0.0213554 0.03843143
## 10  10 0.008538015 -0.0213554 0.03843143
```

All forecasts are identical (0.008538) because the predictor values are held constant. The prediction intervals show the uncertainty around these forecasts, accounting for both parameter estimation error AND future random variation.

Recommendation: For realistic stock market forecasting, consider:

- Vector Autoregression (VAR) models
- ARIMA models for ISE returns directly
- Dynamic regression with ARIMA errors

Limitations: This forecast assumes all predictor variables remain constant at their last observed values for 10 days, which is unrealistic for volatile stock market returns. All 10 forecasts are therefore identical. In practice, a more sophisticated approach such as Vector Autoregression (VAR) would be needed to jointly forecast ISE and the predictor indices, or alternatively, a univariate ARIMA model could be applied directly to ISE returns.