Análisis del tipo de cambio USD/PEN alrededor de elecciones peruanas

N. Leon

15.01.2026

1. Introducción

La hipótesis de esta investigación (breve y exploratoria) es:

> El tipo de cambio USD/PEN se aprecia en los N días previos a las elecciones y se deprecia en los N días posteriores a las elecciones.

siendo $N = 5, 10, 15, 20, 30, 60, 90, 120$ días.

1.1. Contexto y Motivación

Las elecciones presidenciales peruanas se caracterizan por un sistema de dos vueltas, donde la primera ronda selecciona a los dos candidatos más votados y la segunda ronda define al ganador. Este proceso genera distintos períodos de incertidumbre política que pueden afectar diferencialmente el comportamiento del tipo de cambio:

1. **Primera vuelta:** Alta incertidumbre sobre el abanico de posibles ganadores
2. **Segunda vuelta:** Incertidumbre reducida a dos opciones, pero potencialmente mayor polarización

1.2. Metodología General

El análisis sigue un enfoque por etapas:

1. **Preprocesamiento de datos:** Limpieza, transformación y segmentación de datos (creando tres conjuntos: primera vuelta, segunda vuelta y total)
2. **Análisis descriptivo:** Visualización de patrones temporales alrededor de eventos electorales
3. **Modelado estadístico:** Pruebas de hipótesis adaptadas al tamaño muestral limitado
4. **Recomendaciones especulativas:** Estrategias de trading exploratorias basadas en hallazgos

2. Configuración

```r
# Load libraries
library(tidyverse)
library(summariser)
library(dplyr)
library(lubridate)
library(quantmod)
library(ggplot2)
library(knitr)
library(scales)
library(ggrepel)
library(patchwork)
library(purrr)

set.seed(42) # Set random seed for reproducibility
theme_set(theme_minimal(base_size = 12)) # Set plotting theme
```

3. Preprocesamiento de datos

Primero se preprocesan los cierres del tipo de cambio desde 2004 hasta 2026. Esta inforamción es extraída del repositorio del BCRP.

```r
df_ex_ra <- read.csv("./data/exchange_rate.csv")
df_ex_ra$value <- as.double(df_ex_ra$value)
df_ex_ra$date <- as.Date(df_ex_ra$date, "%d.%m.%y")
df_ex_ra <- df_ex_ra %>% arrange(date)
summary(df_ex_ra)
```

```
##       date                value
##  Min.   :2004-01-19   Min.   :2.537
##  1st Qu.:2009-07-16   1st Qu.:2.906
##  Median :2015-01-14   Median :3.265
##  Mean   :2015-01-14   Mean   :3.252
##  3rd Qu.:2020-07-13   3rd Qu.:3.488
##  Max.   :2026-01-09   Max.   :4.137
##                       NA's   :263
```

En el caso de la información respecto a las elecciones, se creó un archivo de forma manual extrayendo información de Wikipedia. La información divide entre rondas electorales (primera y segunda).

```r
df_elect <- read.csv("./data/elections.csv", header = TRUE, sep = ",", dec = ".",)
df_elect$date <- as.Date(df_elect$date, "%d.%m.%y")
df_elect$round <- as.factor(df_elect$round)
df_elect$winner <- as.factor(df_elect$winner)
summary(df_elect)
```

```
##       date               round     winner
##  Min.   :2006-04-09   first :4   left :2
##  1st Qu.:2010-01-22   second:4   none :4
##  Median :2013-11-06              right:2
##  Mean   :2013-11-06
```

```
##  3rd Qu.:2017-08-21
##  Max.   :2021-06-06
```

Siguiendo la metodología de la Subsección 1.2., se generan tres datasets separados para el análisis comparativo.

```
df_total <- df_elect # 1. TODAS las elecciones (primera + segunda vuelta)
df_first<- df_elect %>% filter(round == "first") # 2. Solo PRIMERA vuelta
df_second <- df_elect %>% filter(round == "second") # 3. Solo SEGUNDA vuelta
```

En general, se trata de 4 elecciones generales, cada una con una primera y segunda vuelta desde el 2004 hasta 2025

4. Función principal

La función principal calcula los retornos previos y posterios según el número de días elegido.

```
# Function to calculate pre and post returns for a given window
calculate_window_returns <- function(df_exchange, df_elections, window_days) {
  results <- list()

  election_dates <- df_elections$date

  for(i in seq_along(election_dates)) {
```

```r
  election_date <- election_dates[i]

  # Find the closest trading day to the election date
  date_diffs <- abs(df_exchange$date - election_date)
  election_idx <- which.min(date_diffs)
  closest_date <- df_exchange$date[election_idx]

  # Check if the closest date is within a reasonable range
  if(as.numeric(difftime(closest_date, election_date, units = "days")) > 2) {
    warning(paste("No exchange rate data found within 2 days of election date:",
                  election_date))
    next
  }

  # Find pre-election window start (N trading days before)
  pre_window_start_date <- election_date - window_days

  # Find the closest trading day to pre_window_start_date
  pre_diffs <- abs(df_exchange$date - pre_window_start_date)
  pre_start_idx <- which.min(pre_diffs)
  pre_start_date <- df_exchange$date[pre_start_idx]

  # Find pre-election window end (1 trading day before election)
  dates_before <- df_exchange$date[df_exchange$date < election_date]
  if(length(dates_before) == 0) {
    warning(paste("No exchange rate data before election date:", election_date))
    next
  }
  pre_end_date <- max(dates_before)
  pre_end_idx <- which(df_exchange$date == pre_end_date)

  # Find post-election window start (1 trading day after election)
  dates_after <- df_exchange$date[df_exchange$date > election_date]
  if(length(dates_after) == 0) {
    warning(paste("No exchange rate data after election date:", election_date))
    next
  }
  post_start_date <- min(dates_after)
  post_start_idx <- which(df_exchange$date == post_start_date)

  post_window_end_date <- election_date + window_days

  # Find the closest trading day to post_window_end_date
  post_diffs <- abs(df_exchange$date - post_window_end_date)
  post_end_idx <- which.min(post_diffs)
  post_end_date <- df_exchange$date[post_end_idx]

  # Ensure indices are within bounds
  if(pre_start_idx >= 1 && pre_start_idx <= nrow(df_exchange) &&
     pre_end_idx >= 1 && pre_end_idx <= nrow(df_exchange) &&
     post_start_idx >= 1 && post_start_idx <= nrow(df_exchange) &&
     post_end_idx >= 1 && post_end_idx <= nrow(df_exchange)) {
```

```
      pre_start_price <- df_exchange$value[pre_start_idx]
      pre_end_price <- df_exchange$value[pre_end_idx]

      post_start_price <- df_exchange$value[post_start_idx]
      post_end_price <- df_exchange$value[post_end_idx]

      # Calculate percentage returns
      pre_return <- (pre_end_price - pre_start_price) / pre_start_price * 100
      post_return <- (post_end_price - post_start_price) / post_start_price * 100

      # Calculate actual number of calendar days in each window
      pre_calendar_days <- as.numeric(difftime(pre_end_date, pre_start_date, units = "days"))
      post_calendar_days <- as.numeric(difftime(post_end_date, post_start_date, units = "days"))

      results[[i]] <- tibble(
        election_id = i,
        election_date = election_date,
        round = ifelse("round" %in% names(df_elections),
                       df_elections$round[i], "none"),
        winner = ifelse("winner" %in% names(df_elections),
                       df_elections$winner[i], "none"), #TODO: FIX
        window_days = window_days,
        pre_start_date = pre_start_date,
        pre_end_date = pre_end_date,
        post_start_date = post_start_date,
        post_end_date = post_end_date,
        pre_calendar_days = pre_calendar_days,
        post_calendar_days = post_calendar_days,
        pre_start_price = pre_start_price,
        pre_end_price = pre_end_price,
        post_start_price = post_start_price,
        post_end_price = post_end_price,
        pre_return_pct = pre_return,
        post_return_pct = post_return,
        net_effect = pre_return - post_return,
        # Annualized returns for comparison across different window lengths
        pre_annualized = ifelse(pre_calendar_days > 0,
                                (1 + pre_return/100)^(365/pre_calendar_days) - 1,
                                NA),
        post_annualized = ifelse(post_calendar_days > 0,
                                 (1 + post_return/100)^(365/post_calendar_days) - 1,
                                 NA)
      )
    }
  }

  # Combine all elections for this window
  if(length(results) > 0) {
    bind_rows(results)
  } else {
    tibble()  # Return empty tibble if no results
  }
}
```

4.1. Funciones secundarias

Las funciones secundarias automatizan el cálculo de la función principal.

```
# Wrapper function to calculate for multiple windows
calculate_all_windows <- function(df_exchange, df_elections, windows = c(5, 10)) {
  all_results <- map_df(windows, ~calculate_window_returns(df_exchange, df_elections, .x))

  all_results <- all_results %>%
    group_by(window_days) %>%
    mutate(
      pre_rank = rank(-pre_return_pct),  # Rank by pre-return (highest = 1)
      post_rank = rank(post_return_pct)   # Rank by post-return (lowest = 1)
    ) %>%
    ungroup()

  return(all_results)
}

# Wrapper function to create summary statistics
create_summary_table <- function(all_results) {
  summary_table <- all_results %>%
    group_by(window_days) %>%
    summarise(
      n_elections = n(),
      avg_pre_return = mean(pre_return_pct, na.rm = TRUE),
      median_pre_return = median(pre_return_pct, na.rm = TRUE),
      sd_pre_return = sd(pre_return_pct, na.rm = TRUE),
      avg_post_return = mean(post_return_pct, na.rm = TRUE),
      median_post_return = median(post_return_pct, na.rm = TRUE),
      sd_post_return = sd(post_return_pct, na.rm = TRUE),
      pre_positive_pct = mean(pre_return_pct > 0, na.rm = TRUE) * 100,
      post_negative_pct = mean(post_return_pct < 0, na.rm = TRUE) * 100,
      avg_net_effect = mean(net_effect, na.rm = TRUE)
    ) %>%
    mutate(across(where(is.numeric), ~round(., 3)))
  return(summary_table)
}
```

Considerando los datasets de la Sección 3 y la metodología de la Sección 1, se obtienen los siguientes resultados:

```
windows <- c(5, 10, 15, 20, 30, 45, 60, 75, 90, 105, 120)

total_results <- calculate_all_windows(df_ex_ra, df_elect, windows)
total_table <- create_summary_table(total_results)

first_results <- calculate_all_windows(df_ex_ra, df_first, windows)
first_table <- create_summary_table(first_results)

second_results <- calculate_all_windows(df_ex_ra, df_second, windows)
second_table <- create_summary_table(second_results)
```

5. Resultados

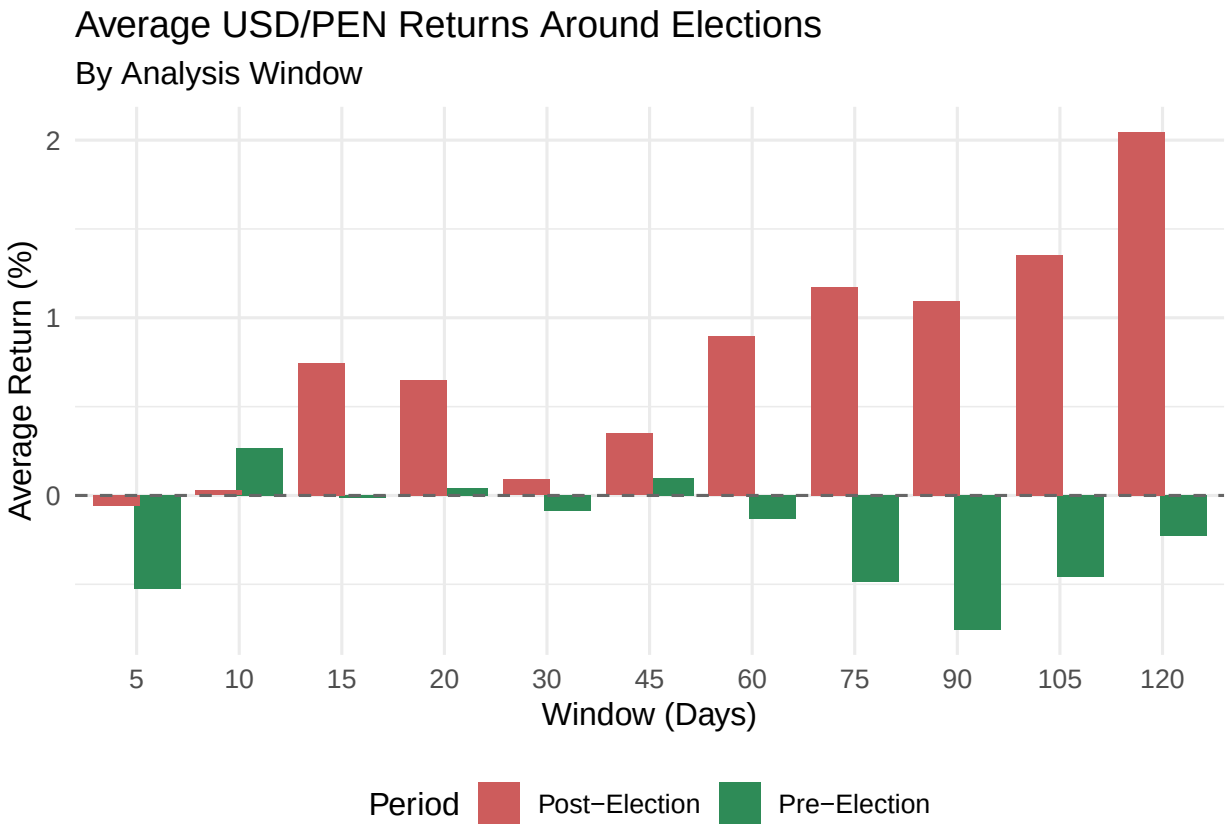
5.1. Resultados generales

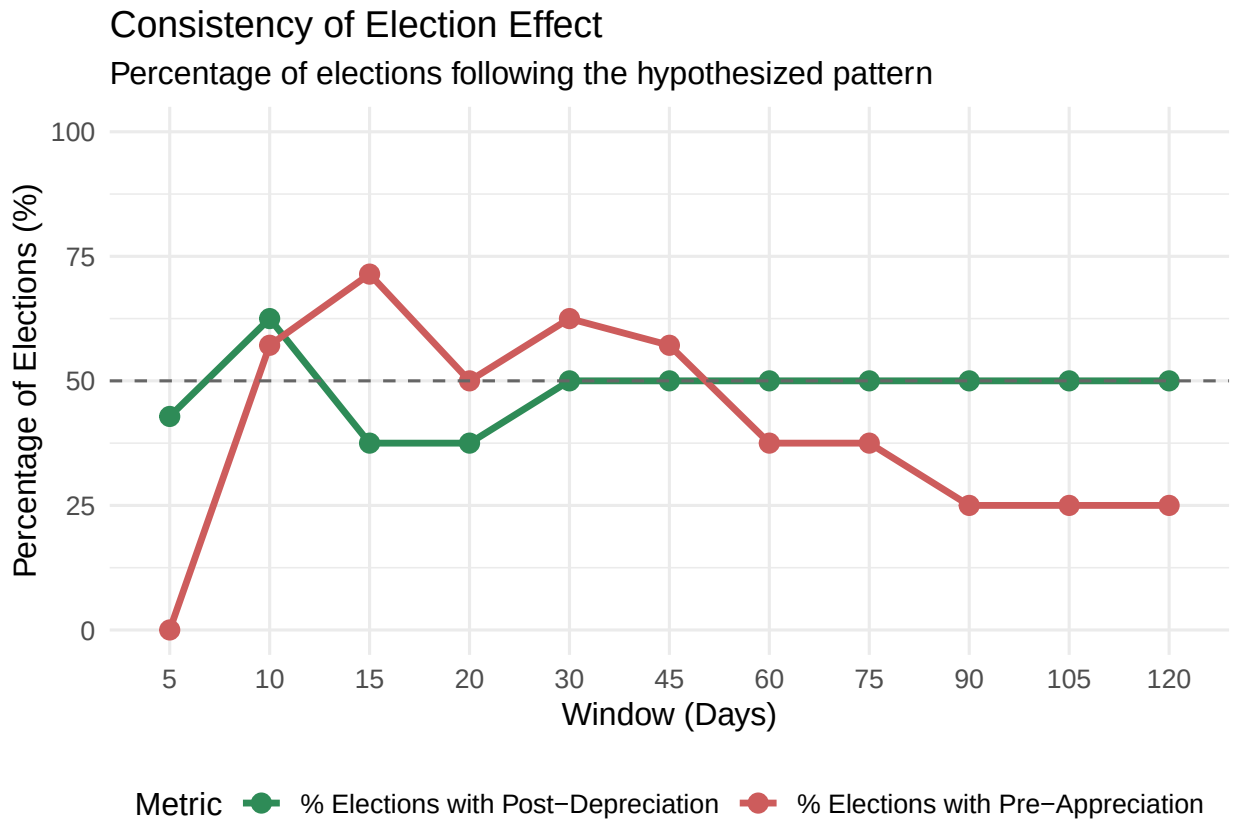
```
kable(total_table[c(1,3,5,6,8)])
```

window_days	avg_pre_return	sd_pre_return	avg_post_return	sd_post_return
5	-0.525	0.377	-0.059	0.783
10	0.267	0.758	0.028	1.207
15	-0.015	1.570	0.747	2.262
20	0.041	1.669	0.649	1.748
30	-0.088	1.273	0.090	1.380
45	0.096	2.496	0.349	2.813
60	-0.134	3.327	0.895	3.239
75	-0.486	2.015	1.174	4.026
90	-0.757	2.181	1.096	4.151
105	-0.457	2.834	1.352	3.772
120	-0.229	2.759	2.047	5.246

```
kable(total_table[c(9:11)])
```

pre_positive_pct	post_negative_pct	avg_net_effect
0.000	42.857	-0.495
57.143	62.500	0.599
71.429	37.500	-0.830
50.000	37.500	-0.608
62.500	50.000	-0.178
57.143	50.000	-0.574
37.500	50.000	-1.029
37.500	50.000	-1.660
25.000	50.000	-1.853
25.000	50.000	-1.809
25.000	50.000	-2.276


Average USD/PEN Returns Around Elections
By Analysis Window
Average Return (%)
2
1
0
5
10
15
20
30
45
60
75
90
105
120
Window (Days)
Period
Post−Election
Pre−Election



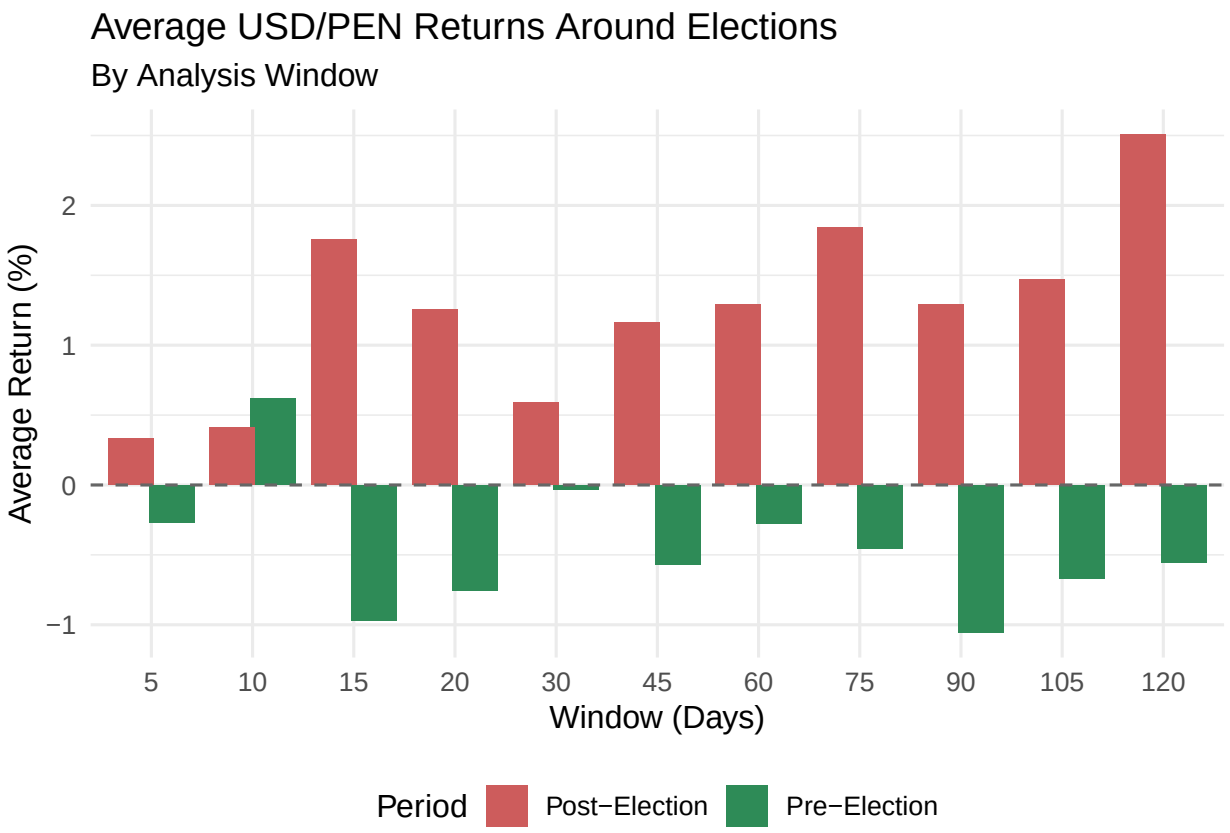
5.1. Resultados primera vuelta

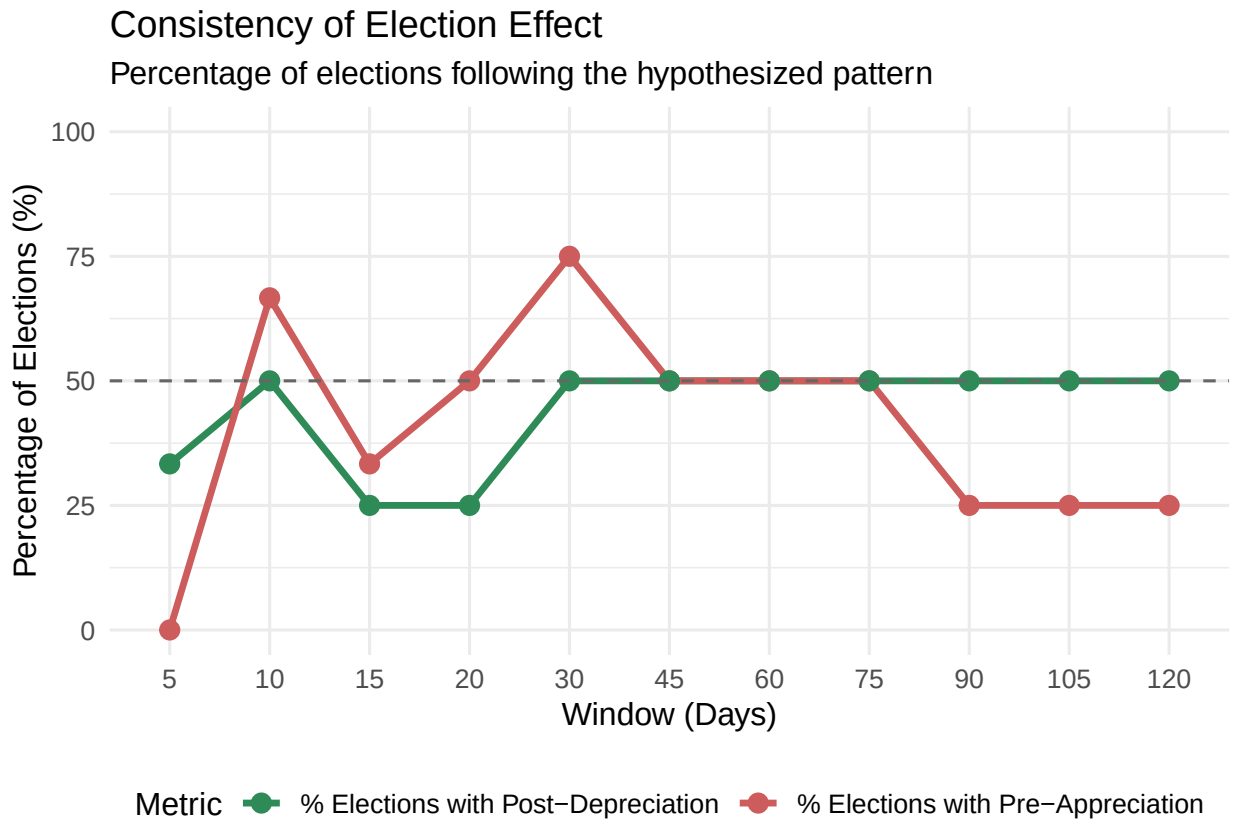
```
kable(first_table[c(1,3,5,6,8)])
```

window_days	avg_pre_return	sd_pre_return	avg_post_return	sd_post_return
5	-0.269	0.181	0.335	0.443
10	0.619	1.122	0.415	1.592
15	-0.972	1.779	1.755	2.929
20	-0.756	1.349	1.256	2.325
30	-0.031	1.606	0.591	1.725
45	-0.571	2.872	1.162	3.967
60	-0.275	2.660	1.289	4.394
75	-0.453	1.637	1.843	5.356
90	-1.056	1.255	1.289	5.721
105	-0.668	0.784	1.469	5.023
120	-0.555	1.059	2.508	7.570

```
kable(first_table[c(9:11)])
```

pre_positive_pct	post_negative_pct	avg_net_effect
0.000	33.333	-0.584
66.667	50.000	0.914
33.333	25.000	-3.220
50.000	25.000	-2.012
75.000	50.000	-0.622
50.000	50.000	-1.733
50.000	50.000	-1.565
50.000	50.000	-2.296
25.000	50.000	-2.346
25.000	50.000	-2.137
25.000	50.000	-3.063





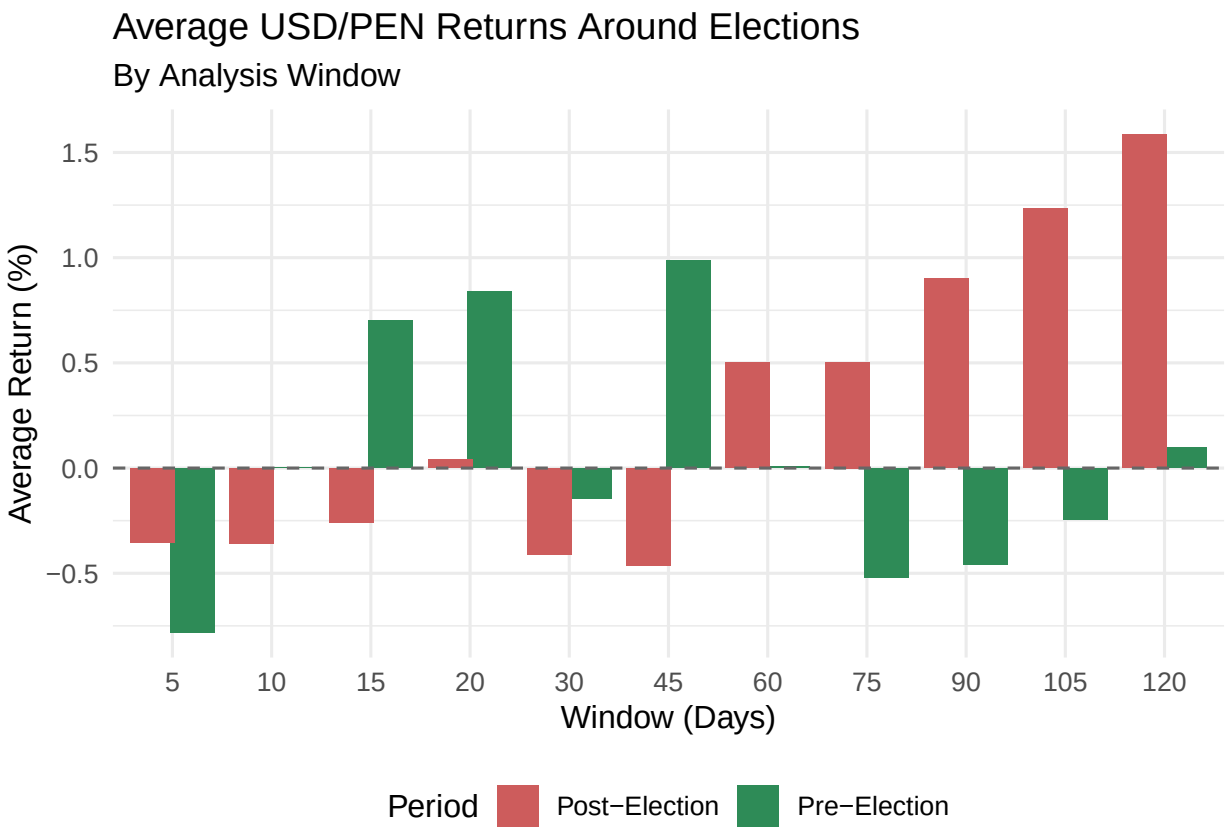
5.2. Resultados segunda vuelta

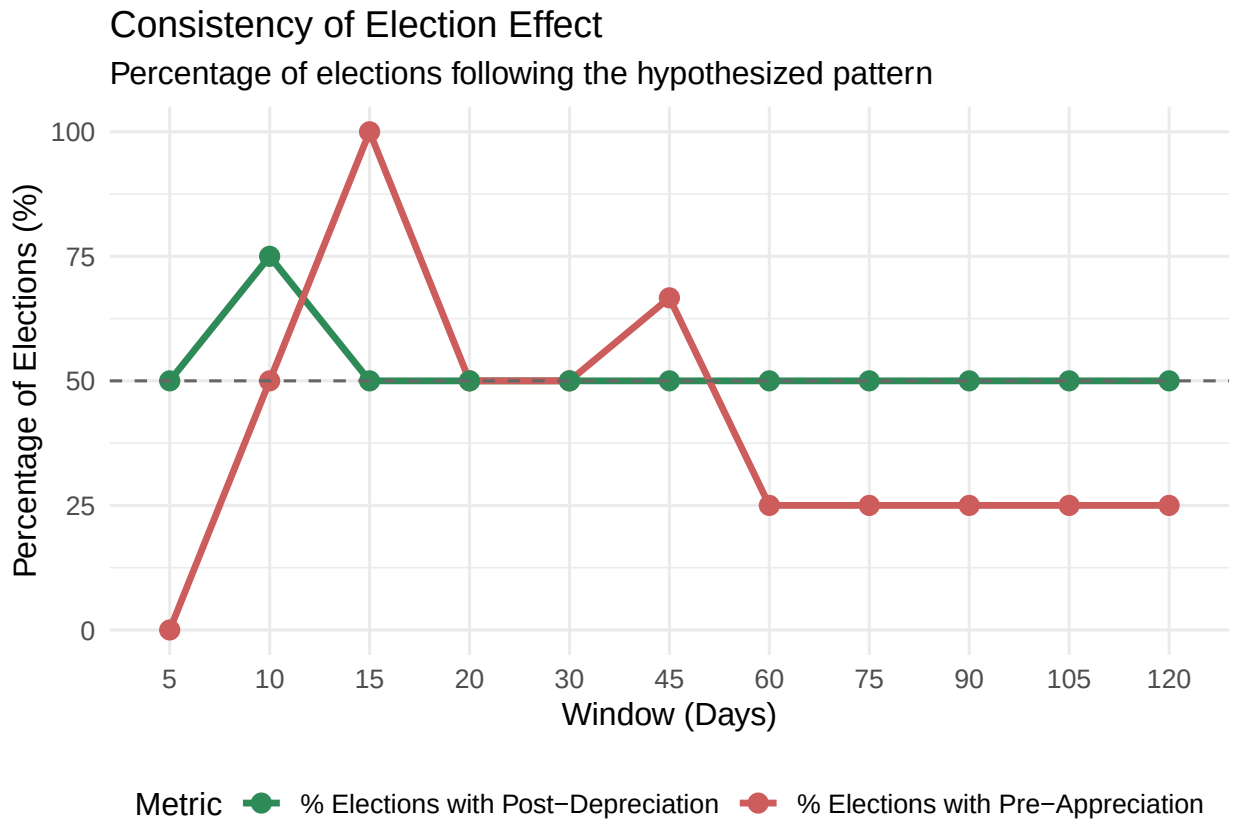
```
kable(second_table[c(1,3,5,6,8)])
```

window_days	avg_pre_return	sd_pre_return	avg_post_return	sd_post_return
5	-0.782	0.351	-0.354	0.907
10	0.004	0.307	-0.359	0.685
15	0.702	1.103	-0.260	0.808
20	0.838	1.727	0.042	0.862
30	-0.145	1.093	-0.411	0.893
45	0.986	2.058	-0.464	0.984
60	0.007	4.324	0.500	2.180
75	-0.519	2.606	0.504	2.815
90	-0.458	3.047	0.902	2.716
105	-0.246	4.243	1.235	2.817
120	0.096	4.044	1.586	2.519

```
kable(second_table[c(9:11)])
```

pre_positive_pct	post_negative_pct	avg_net_effect
0.000	50	-0.427
50.000	75	0.363
100.000	50	0.962
50.000	50	0.795
50.000	50	0.265
66.667	50	0.972
25.000	50	-0.493
25.000	50	-1.024
25.000	50	-1.360
25.000	50	-1.481
25.000	50	-1.490





6. Análisis de resultados

En base a los resultados y las visualizaciones, podemos adapatar las hipótesis:

a) El tipo de cambio USD/PEN se aprecia en los 10 (+/- 5) días previos a la primer vuelta.
b) El tipo de cambio USD/PEN se aprecia en los 30 (+/- 15) días previos a la primer vuelta.
c) El tipo de cambio USD/PEN se aprecia en los 15 (+/- 5) días previos a la segunda vuelta.
d) El tipo de cambio USD/PEN se aprecia en los 45 (+/- 15) días previos a la segunda vuelta.
e) El tipo de cambio USD/PEN se deprecia en los 10 (+/- 5) días posteriores a la segunda vuelta.

Para confirmar estas hipótesis se realizarán modelos estadísticos.

6.1. Hipótesis a - b

Para confirmar/negar estas hipótesis se construye un modelo **Event study + bootstrap (permutation test)**, donde se evalúa si el retorno alrededor del evento electoral es estadísticamente distinto de cero **en promedio.**

```r
# Preprocesar a retornos logarítmicos (más sensibles a cambios mínimos)
df_ex_ra_test <- df_ex_ra %>%
  arrange(date) %>%
  mutate(
    ret = log(value) - log(lag(value))
  ) %>%
  filter(!is.na(ret))

# Extraer ventana de eventos alrededor del evento
get_event_window <- function(df, event_date, pre_days, post_days) {

  idx_event <- which.min(abs(df$date - event_date))

  start_idx <- idx_event - pre_days
  end_idx   <- idx_event + post_days

  if (start_idx < 1 || end_idx > nrow(df)) {
    return(NULL)
  }

  tibble(
    day = -pre_days:post_days,
    ret = df$ret[start_idx:end_idx]
  )
}

# Calcula Retornos acumulados (CAR)
compute_car <- function(event_window, from, to) {
  event_window %>%
    filter(day >= from, day <= to) %>%
    summarise(car = sum(ret)) %>%
    pull(car)
}

# Hipótesis:
# H0: mediana(CAR) = 0
# H1: mediana(CAR) > 0 (pre)

run_sign_test <- function(df_ex, df_events, pre_days, post_days,
                          car_from, car_to) {

  cars <- map_dbl(df_events$date, function(d) {
    ew <- get_event_window(df_ex, d, pre_days, post_days)
    if (is.null(ew)) return(NA_real_)
    compute_car(ew, car_from, car_to)
  })

  cars <- na.omit(cars)

  test <- binom.test(
    x = sum(cars > 0),
    n = length(cars),
    p = 0.5,
```

```
    alternative = ifelse(car_to < 0, "less", "greater")
  )

  list(
    cars = cars,
    test = test
  )
}

run_bootstrap_test <- function(df_ex, df_events,
                               pre_days, post_days,
                               car_from, car_to,
                               n_boot = 5000,
                               min_gap = 30) {

  # CAR observado
  observed_cars <- map_dbl(df_events$date, function(d) {
    ew <- get_event_window(df_ex, d, pre_days, post_days)
    if (is.null(ew)) return(NA_real_)
    compute_car(ew, car_from, car_to)
  })

  observed_mean <- mean(observed_cars, na.rm = TRUE)

  # Fechas elegibles
  valid_dates <- df_ex$date[
    (df_ex$date > min(df_ex$date) + pre_days + min_gap) &
    (df_ex$date < max(df_ex$date) - post_days - min_gap)
  ]

  # Bootstrap
  boot_means <- replicate(n_boot, {

    fake_dates <- sample(valid_dates, nrow(df_events))

    fake_cars <- map_dbl(fake_dates, function(d) {
      ew <- get_event_window(df_ex, d, pre_days, post_days)
      if (is.null(ew)) return(NA_real_)
      compute_car(ew, car_from, car_to)
    })

    mean(fake_cars, na.rm = TRUE)
  })

  p_value <- mean(
    if (observed_mean > 0)
      boot_means >= observed_mean
    else
      boot_means <= observed_mean
  )

  list(
    observed_mean = observed_mean,
```

```
    boot_distribution = boot_means,
    p_value = p_value
  )
}
```

a. El tipo de cambio USD/PEN se aprecia en los 10 (+/- 5) días previos a la primer vuelta.

```
sign_test_a <- run_sign_test(
  df_ex = df_ex_ra_test,
  df_events = df_first,
  pre_days = 15,
  post_days = 1,
  car_from = -15,
  car_to = -5
)

sign_test_a$test
```

```
##
##  Exact binomial test
##
## data:  sum(cars > 0) and length(cars)
## number of successes = 4, number of trials = 4, p-value = 1
## alternative hypothesis: true probability of success is less than 0.5
## 95 percent confidence interval:
##  0 1
## sample estimates:
## probability of success
##                      1
```

```
boot_test_a <- run_bootstrap_test(
  df_ex = df_ex_ra_test,
  df_events = df_first,
  pre_days = 15,
  post_days = 1,
  car_from = -15,
  car_to = -5,
  n_boot = 10000
)

boot_test_a$observed_mean
```

```
## [1] 0.008723597
```

```
boot_test_a$p_value
```

```
## [1] 0.0717
```

b. El tipo de cambio USD/PEN se aprecia en los 30 (+/- 15) días previos a la primer vuelta.

```r
sign_test_b <- run_sign_test(
  df_ex = df_ex_ra_test,
  df_events = df_first,
  pre_days = 45,
  post_days = 15,
  car_from = -45,
  car_to = -15
)

sign_test_b$test
```

```
##
##  Exact binomial test
##
## data:  sum(cars > 0) and length(cars)
## number of successes = 3, number of trials = 4, p-value = 0.9375
## alternative hypothesis: true probability of success is less than 0.5
## 95 percent confidence interval:
##  0.0000000 0.9872585
## sample estimates:
## probability of success
##                   0.75
```

```r
boot_test_b <- run_bootstrap_test(
  df_ex = df_ex_ra_test,
  df_events = df_first,
  pre_days = 45,
  post_days = 15,
  car_from = -45,
  car_to = -15,
  n_boot = 10000
)

boot_test_b$observed_mean
```

```
## [1] 0.002269936
```

```r
boot_test_b$p_value
```

```
## [1] 0.3863
```

6.2. Hipótesis c - d

c. El tipo de cambio USD/PEN se aprecia en los 15 (+/- 5) días previos a la segunda vuelta.

```r
sign_test_c <- run_sign_test(
  df_ex = df_ex_ra_test,
  df_events = df_second,
  pre_days = 45,
  post_days = 15,
```

```
  car_from = -45,
  car_to = -15
)

sign_test_c$test
```

```
##
##  Exact binomial test
##
## data:  sum(cars > 0) and length(cars)
## number of successes = 0, number of trials = 4, p-value = 0.0625
## alternative hypothesis: true probability of success is less than 0.5
## 95 percent confidence interval:
##  0.0000000 0.5271292
## sample estimates:
## probability of success
##                      0
```

```
boot_test_c <- run_bootstrap_test(
  df_ex = df_ex_ra_test,
  df_events = df_second,
  pre_days = 45,
  post_days = 15,
  car_from = -45,
  car_to = -15,
  n_boot = 10000
)

boot_test_c$observed_mean
```

```
## [1] -0.01057952
```

```
boot_test_c$p_value
```

```
## [1] 0.1386
```

d. El tipo de cambio USD/PEN se aprecia en los 45 (+/- 15) días previos a la segunda vuelta.

```
sign_test_d <- run_sign_test(
  df_ex = df_ex_ra_test,
  df_events = df_second,
  pre_days = 60,
  post_days = 30,
  car_from = -60,
  car_to = -30
)

sign_test_d$test
```

```
##
```

```
##  Exact binomial test
##
## data:  sum(cars > 0) and length(cars)
## number of successes = 3, number of trials = 4, p-value = 0.9375
## alternative hypothesis: true probability of success is less than 0.5
## 95 percent confidence interval:
##  0.0000000 0.9872585
## sample estimates:
## probability of success
##                   0.75
```

```
boot_test_d <- run_bootstrap_test(
  df_ex = df_ex_ra_test,
  df_events = df_second,
  pre_days = 60,
  post_days = 30,
  car_from = -60,
  car_to = -30,
  n_boot = 10000
)

boot_test_d$observed_mean
```

```
## [1] 0.005908383
```

```
boot_test_d$p_value
```

```
## [1] 0.261
```

6.3. Hipótesis e

e. El tipo de cambio USD/PEN se deprecia en los 10 (+/- 5) días posteriores a la segunda vuelta.

```
sign_test_e <- run_sign_test(
  df_ex = df_ex_ra_test,
  df_events = df_second,
  pre_days = 5,
  post_days = 15,
  car_from = 5,
  car_to = 15
)

sign_test_e$test
```

```
##
##  Exact binomial test
##
## data:  sum(cars > 0) and length(cars)
## number of successes = 1, number of trials = 4, p-value = 0.9375
## alternative hypothesis: true probability of success is greater than 0.5
## 95 percent confidence interval:
```

```
##  0.01274146 1.00000000
## sample estimates:
## probability of success
##                   0.25
```

```
boot_test_e <- run_bootstrap_test(
  df_ex = df_ex_ra_test,
  df_events = df_second,
  pre_days = 5,
  post_days = 15,
  car_from = 5,
  car_to = 15,
  n_boot = 10000
)

boot_test_e$observed_mean
```

```
## [1] -0.0008805774
```

```
boot_test_e$p_value
```

```
## [1] 0.4544
```

7. Discusión

Se analizan resultados finales.

8. Cálculo de retorno

Se seleccionan las hipótesis más válidas y se calcula el retorno

8. Recomendaciones

Se recomiendan estrategias de trading