Final

Javier Chiquín

2025-05-24

```
# Cargar paquetes
library(haven)
```

```
## Warning: package 'haven' was built under R version 4.4.3
```

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.4.3
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.4     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(haven)
library(dplyr)
library(ggplot2)
library(skimr) # Para resumen detallado
library(corrplot) # Para matriz de correlación
```

```
## corrplot 0.95 loaded
```

```
library(summarytools)
```

```
## Warning: package 'summarytools' was built under R version 4.4.3
```

```
##
## Adjuntando el paquete: 'summarytools'
##
## The following object is masked from 'package:tibble':
##
##     view
```

```r
# Definir la ruta a los archivos SPSS
ruta <- "C:\\Users\\javie\\Downloads\\Divorcios\\"

# Leer archivos SPSS (2014-2024)
años_spss <- 2014:2024
datos_spss <- list()
for (año in años_spss) {
  archivo <- paste0(ruta, "div_", año, ".sav")
  datos_spss[[as.character(año)]] <- read_sav(archivo)
}

# Seleccionar columnas comunes y convertir CIUOHOM/CIUOMUJ a character
columnas_comunes <- c("AÑOREG", "EDADHOM", "EDADMUJ", "CIUOHOM", "CIUOMUJ")
for (año in names(datos_spss)) {
  datos_spss[[año]] <- datos_spss[[año]] %>%
    select(all_of(columnas_comunes)) %>%
    mutate(
      CIUOHOM = as.character(CIUOHOM),
      CIUOMUJ = as.character(CIUOMUJ)
    )
}

# Unificar datasets SPSS
datos_unificados <- bind_rows(datos_spss, .id = "año")

# Limpiar datos: eliminar NA en columnas clave y filtrar edades inválidas
datos_limpios <- datos_unificados %>%
  drop_na(AÑOREG, EDADHOM, EDADMUJ) %>%
  filter(EDADHOM != 999 & EDADMUJ != 999)

# Agregar datos por año para crear la variable respuesta
datos_agregados <- datos_limpios %>%
  group_by(AÑOREG) %>%
  summarise(
    divorcios = n(),
    edadhom_promedio = mean(EDADHOM, na.rm = TRUE),
    edadmuj_promedio = mean(EDADMUJ, na.rm = TRUE)
  ) %>%
  mutate(
    pandemia = ifelse(AÑOREG %in% 2020:2022, 1, 0)
  )

# Guardar el dataset limpio y agregado
write.csv(datos_limpios, "divorcios_limpios_2014_2024.csv", row.names = FALSE)
write.csv(datos_agregados, "divorcios_agregados_2014_2024.csv", row.names = FALSE)

# Verificar los datasets
cat("Dataset limpio (individual):\n")
```

```
## Dataset limpio (individual):
```

```r
summary(datos_limpios)
```

```
##      año                AÑOREG        EDADHOM         EDADMUJ
##  Length:39993       Min.   :2014   Min.   :17.00   Min.   :16.00
##  Class :character   1st Qu.:2017   1st Qu.:29.00   1st Qu.:26.00
##  Mode  :character   Median :2020   Median :34.00   Median :31.00
##                     Mean   :2020   Mean   :35.85   Mean   :32.73
##                     3rd Qu.:2022   3rd Qu.:41.00   3rd Qu.:37.00
##                     Max.   :2024   Max.   :98.00   Max.   :81.00
##    CIUOHOM            CIUOMUJ
##  Length:39993       Length:39993
##  Class :character   Class :character
##  Mode  :character   Mode  :character
##
##
##
```

```
dim(datos_limpios)
```

```
## [1] 39993     6
```

```
names(datos_limpios)
```

```
## [1] "año"     "AÑOREG"  "EDADHOM" "EDADMUJ" "CIUOHOM" "CIUOMUJ"
```

```
cat("\nDataset agregado (anual):\n")
```

```
##
## Dataset agregado (anual):
```

```
print(datos_agregados)
```

```
## # A tibble: 11 x 5
##    AÑOREG divorcios edadhom_promedio edadmuj_promedio pandemia
##     <dbl>     <int>            <dbl>            <dbl>    <dbl>
##  1   2014      3124             34.5             31.2        0
##  2   2015      2650             34.3             30.7        0
##  3   2016      2490             34.2             30.9        0
##  4   2017      2580             34.7             31.4        0
##  5   2018      2952             34.9             31.7        0
##  6   2019      4352             35.4             32.4        0
##  7   2020      2295             36.0             32.7        1
##  8   2021      5501             36.2             33.2        1
##  9   2022      6403             36.9             33.9        1
## 10   2023      6632             37.3             34.4        0
## 11   2024      1014             37.7             34.8        0
```

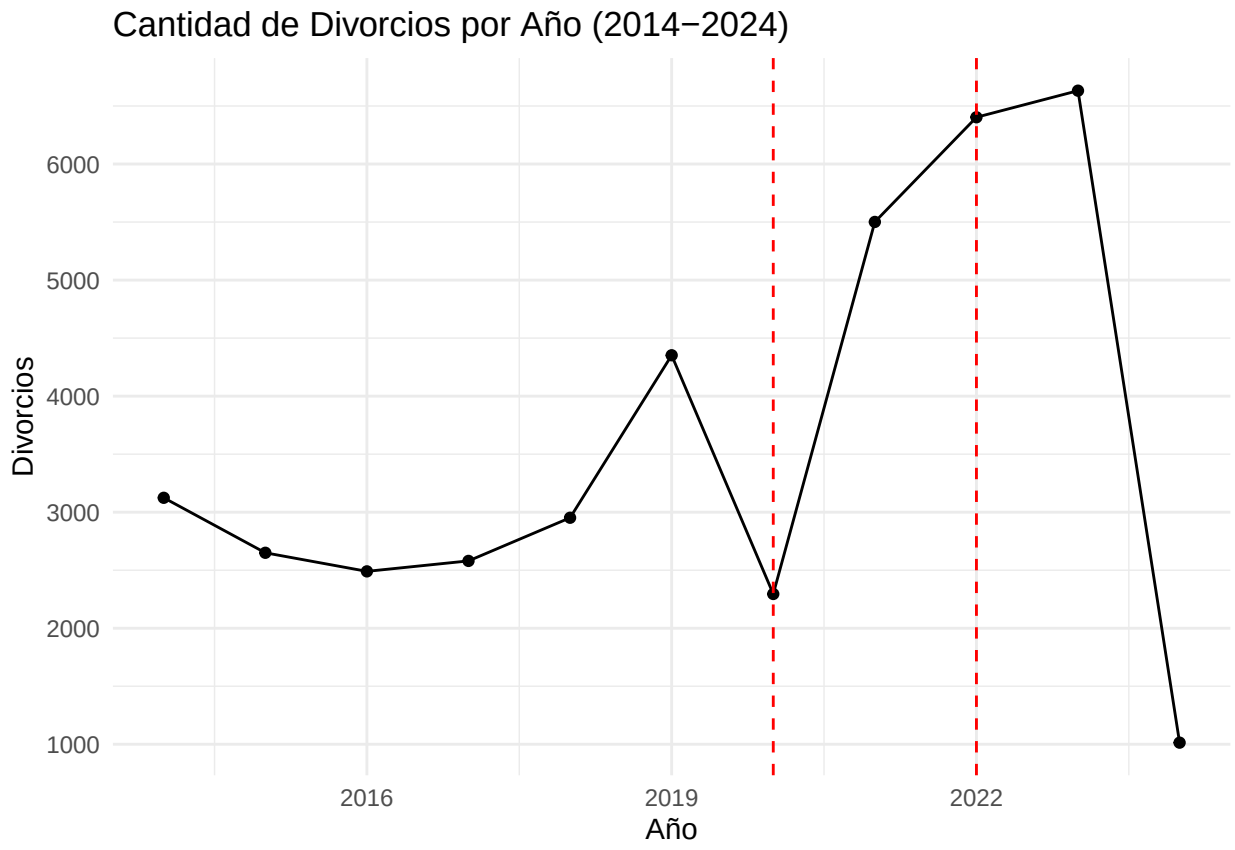
```
dim(datos_agregados)
```

```
## [1] 11  5
```

```r
names(datos_agregados)
```

```
## [1] "AÑOREG"           "divorcios"        "edadhom_promedio" "edadmuj_promedio"
## [5] "pandemia"
```

```r
# Gráfico exploratorio: Divorcios por año
ggplot(datos_agregados, aes(x = AÑOREG, y = divorcios)) +
  geom_line() +
  geom_point() +
  theme_minimal() +
  labs(title = "Cantidad de Divorcios por Año (2014-2024)",
       x = "Año", y = "Divorcios") +
  geom_vline(xintercept = 2020, linetype = "dashed", color = "red") +
  geom_vline(xintercept = 2022, linetype = "dashed", color = "red")
```




```r
# Guardar el gráfico
ggsave("divorcios_por_año_2014_2024.png")
```

```
## Saving 6.5 x 4.5 in image
```

```r
# Mostrar avisos si los hay
warnings()
```

```r
# Verificar el dataset limpio
summary(datos_limpios)
```

```
##      año                AÑOREG        EDADHOM         EDADMUJ
##  Length:39993       Min.   :2014   Min.   :17.00   Min.   :16.00
##  Class :character   1st Qu.:2017   1st Qu.:29.00   1st Qu.:26.00
##  Mode  :character   Median :2020   Median :34.00   Median :31.00
##                     Mean   :2020   Mean   :35.85   Mean   :32.73
##                     3rd Qu.:2022   3rd Qu.:41.00   3rd Qu.:37.00
##                     Max.   :2024   Max.   :98.00   Max.   :81.00
##    CIUOHOM            CIUOMUJ
##  Length:39993       Length:39993
##  Class :character   Class :character
##  Mode  :character   Mode  :character
##
##
##
```

```r
dim(datos_limpios)
```

```
## [1] 39993     6
```

```r
names(datos_limpios)
```

```
## [1] "año"     "AÑOREG"  "EDADHOM" "EDADMUJ" "CIUOHOM" "CIUOMUJ"
```

Empezamos el análisis exploratorio

```r
# 1. Estadísticas descriptivas para edad_hombre y edad_mujer
print("Estadísticas descriptivas para edad_hombre:")
```

```
## [1] "Estadísticas descriptivas para edad_hombre:"
```

```r
summary(datos_limpios$EDADHOM)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   17.00   29.00   34.00   35.85   41.00   98.00
```

```r
print("Estadísticas descriptivas para edad_mujer:")
```

```
## [1] "Estadísticas descriptivas para edad_mujer:"
```

```r
summary(datos_limpios$EDADMUJ)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   16.00   26.00   31.00   32.73   37.00   81.00
```

```r
# 2. Análisis temporal: Número de divorcios por año
divorcios_por_año <- datos_limpios %>%
  group_by(año) %>%
  summarise(cantidad = n())
print("Número de divorcios por año:")
```

```
## [1] "Número de divorcios por año:"
```

```r
print(divorcios_por_año)
```

```
## # A tibble: 11 x 2
##    año   cantidad
##    <chr>    <int>
##  1 2014      2019
##  2 2015      2330
##  3 2016      2474
##  4 2017      2670
##  5 2018      3085
##  6 2019      4212
##  7 2020      2371
##  8 2021      5835
##  9 2022      6239
## 10 2023      6739
## 11 2024      2019
```

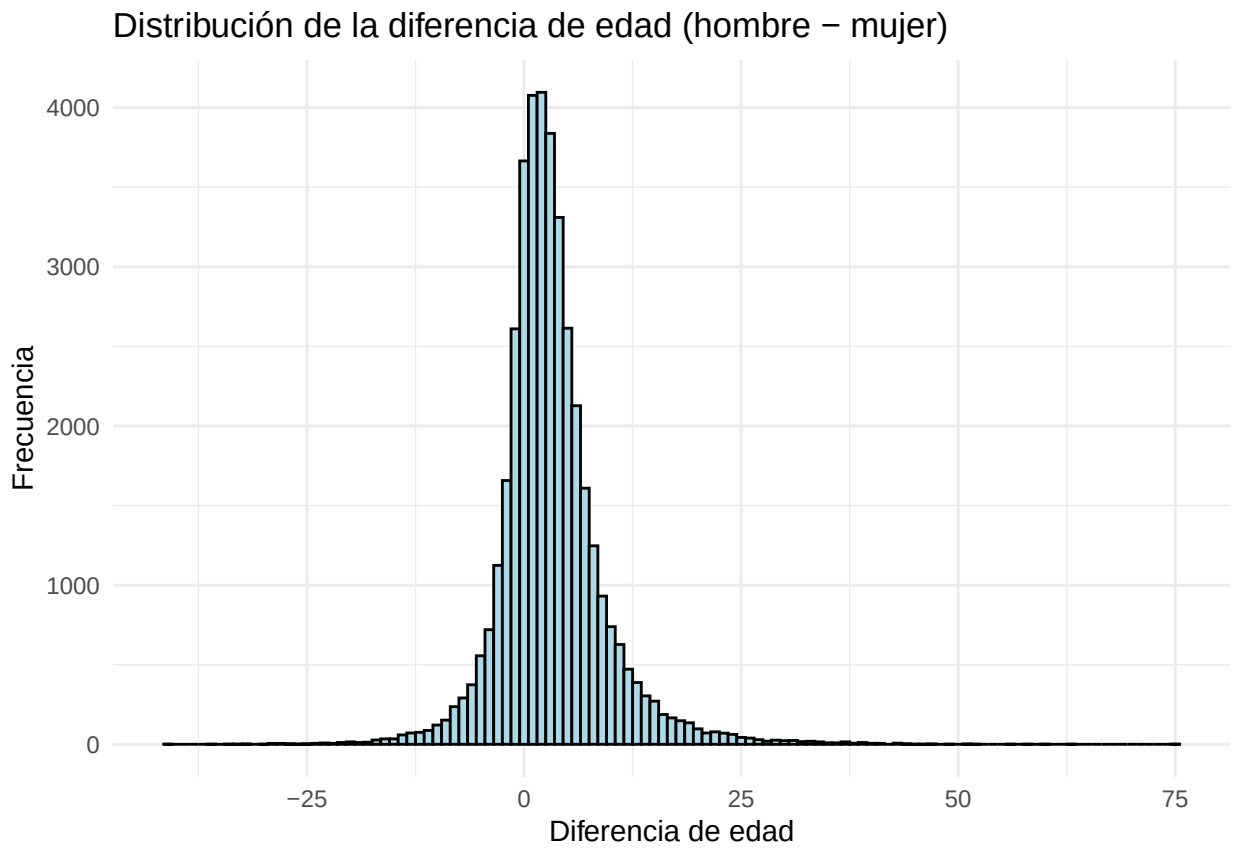
```r
# 3. Comparación por grupos: Diferencia de edad promedio entre hombres y mujeres
datos_limpios <- datos_limpios %>%
  mutate(dif_edad = EDADHOM - EDADMUJ)

print("Estadísticas de la diferencia de edad (hombre - mujer):")
```

```
## [1] "Estadísticas de la diferencia de edad (hombre - mujer):"
```

```r
summary(datos_limpios$dif_edad)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -41.000   0.000   2.000   3.123   6.000  75.000
```

```r
# Histograma de la diferencia de edad
ggplot(datos_limpios, aes(x = dif_edad)) +
  geom_histogram(binwidth = 1, fill = "lightblue", color = "black") +
  labs(title = "Distribución de la diferencia de edad (hombre - mujer)",
       x = "Diferencia de edad", y = "Frecuencia") +
  theme_minimal()
```




```
# 4. Distribución por ocupación: Divorcios por ocupación_hombre y ocupación_mujer
divorcios_ocupacion_hombre <- datos_limpios %>%
  group_by(CIUOHOM) %>%
  summarise(cantidad = n()) %>%
  arrange(desc(cantidad))

divorcios_ocupacion_mujer <- datos_limpios %>%
  group_by(CIUOMUJ) %>%
  summarise(cantidad = n()) %>%
  arrange(desc(cantidad))

print("Divorcios por ocupación (hombre):")
```

```
## [1] "Divorcios por ocupación (hombre):"
```

```
print(divorcios_ocupacion_hombre)
```

```
## # A tibble: 48 x 2
##    CIUOHOM cantidad
##    <chr>      <int>
##  1 97          8743
##  2 92          5124
##  3 52          3697
##  4 43          1936
##  5 98          1514
```

```
##  6 23          1485
##  7 99          1446
##  8 83          1282
##  9 72          1267
## 10 71          1187
## # i 38 more rows
```

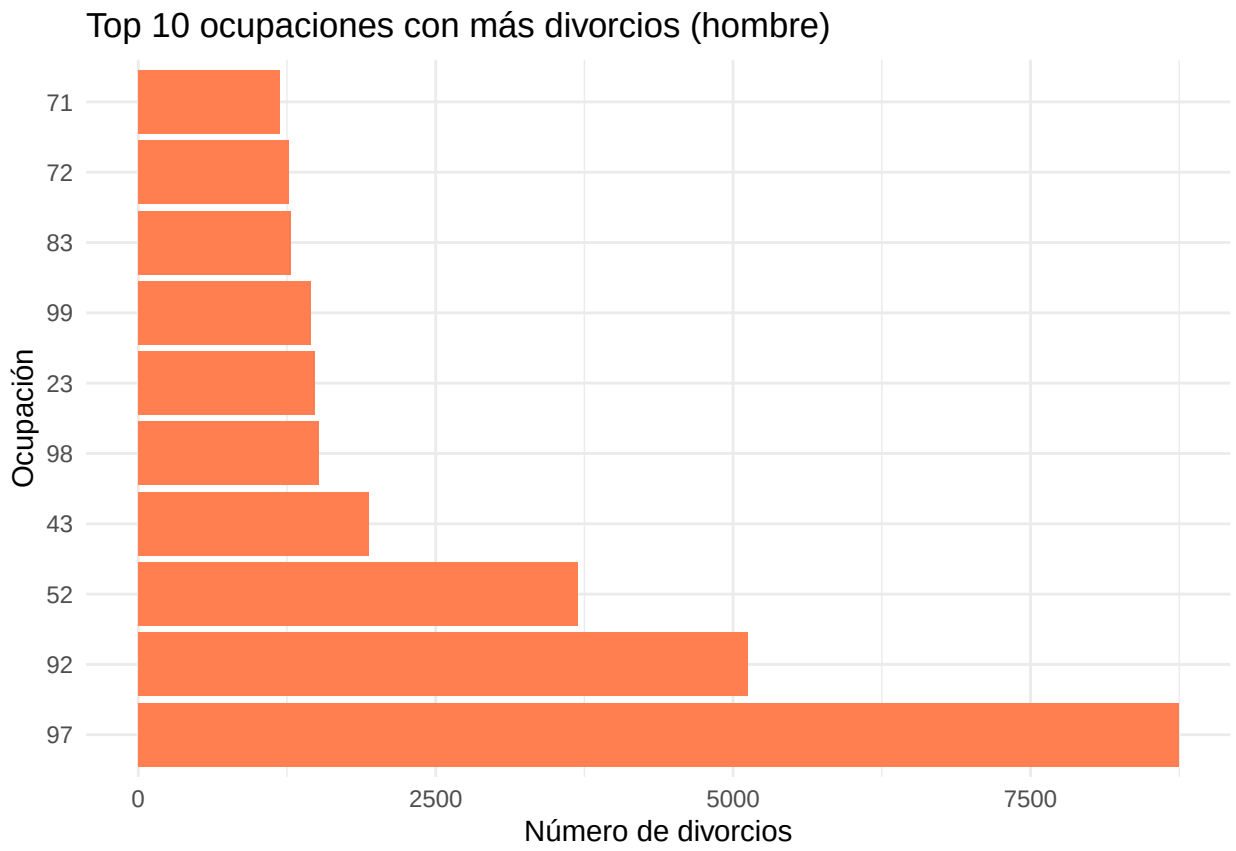
```r
print("Divorcios por ocupación (mujer):")
```

```
## [1] "Divorcios por ocupación (mujer):"
```

```r
print(divorcios_ocupacion_mujer)
```

```
## # A tibble: 45 x 2
##    CIUOMUJ cantidad
##    <chr>      <int>
##  1 97         18941
##  2 23          3329
##  3 98          2938
##  4 NEOG        2412
##  5 41          2013
##  6 99          1359
##  7 52          1308
##  8 43          1282
##  9 26           870
## 10 34           716
## # i 35 more rows
```

```r
# Gráfico de barras para ocupación_hombre (top 10 ocupaciones)
ggplot(head(divorcios_ocupacion_hombre, 10), aes(x = reorder(CIUOHOM, -cantidad), y = cantidad)) +
  geom_bar(stat = "identity", fill = "coral") +
  labs(title = "Top 10 ocupaciones con más divorcios (hombre)",
       x = "Ocupación", y = "Número de divorcios") +
  theme_minimal() +
  coord_flip()
```



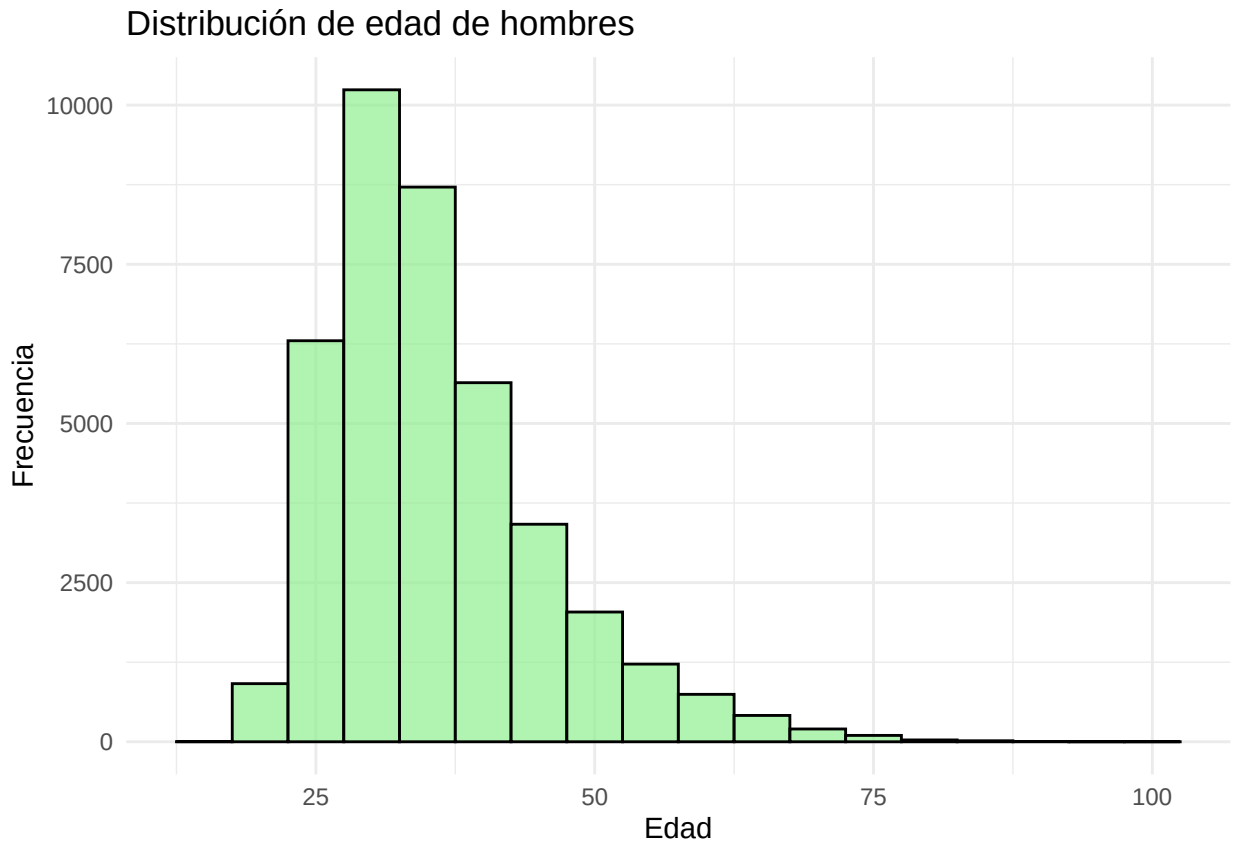

```
# 5. Edad promedio por ocupación (hombre y mujer)
edad_promedio_ocupacion <- datos_limpios %>%
  group_by(CIUOHOM) %>%
  summarise(edad_prom_hombre = mean(EDADHOM, na.rm = TRUE)) %>%
  arrange(desc(edad_prom_hombre))

print("Edad promedio de hombres por ocupación:")
```

```
## [1] "Edad promedio de hombres por ocupación:"
```

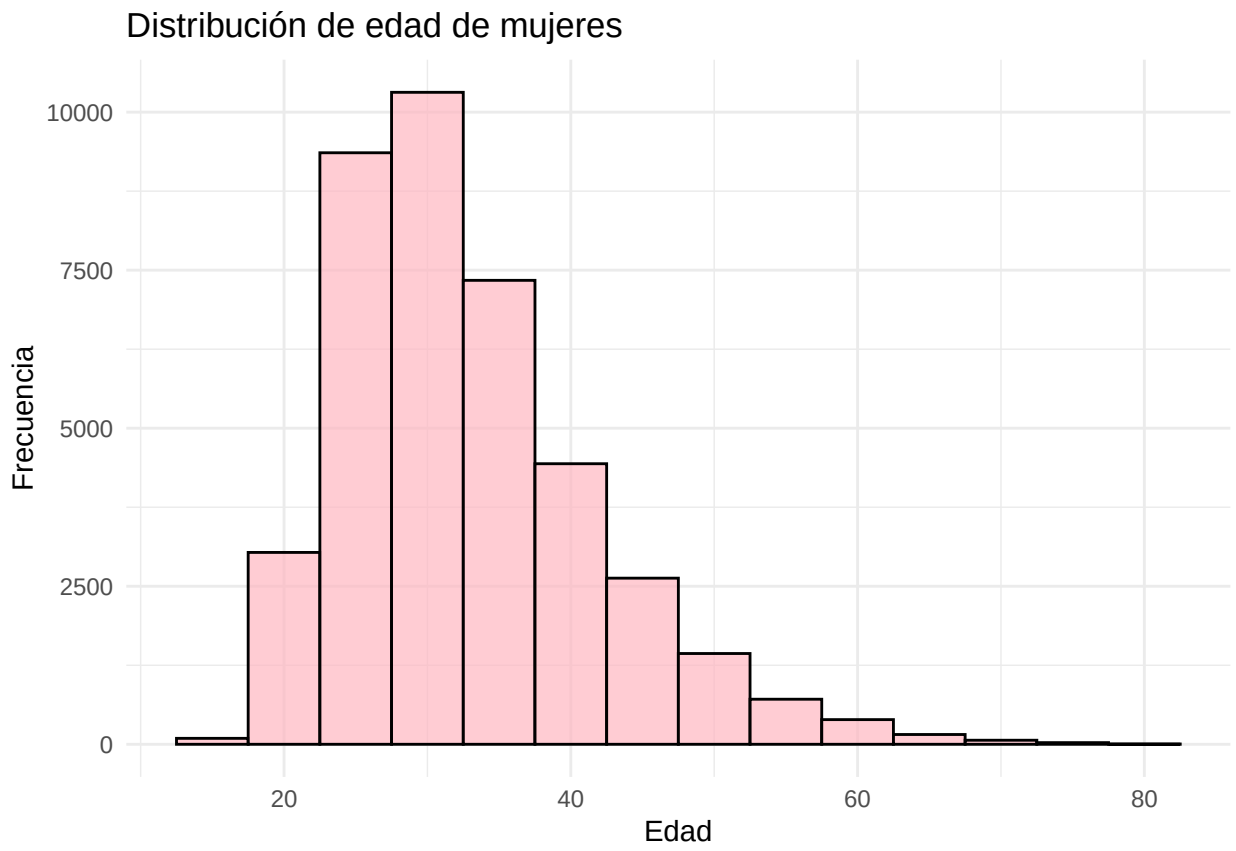
```
print(head(edad_promedio_ocupacion, 10))
```

```
## # A tibble: 10 x 2
##    CIUOHOM edad_prom_hombre
##    <chr>              <dbl>
##  1 95                  42.2
##  2 26                  41.3
##  3 22                  41.2
##  4 24                  40.4
##  5 99                  40.2
##  6 21                  40.1
##  7 12                  39.5
##  8 91                  39.2
##  9 83                  39.2
## 10 41                  38.9
```

```r
# Histogramas de edades (hombre y mujer) - Corregido
ggplot(datos_limpios, aes(x = EDADHOM)) +
  geom_histogram(binwidth = 5, fill = "lightgreen", color = "black", alpha = 0.7) +
  labs(title = "Distribución de edad de hombres",
       x = "Edad", y = "Frecuencia") +
  theme_minimal()
```



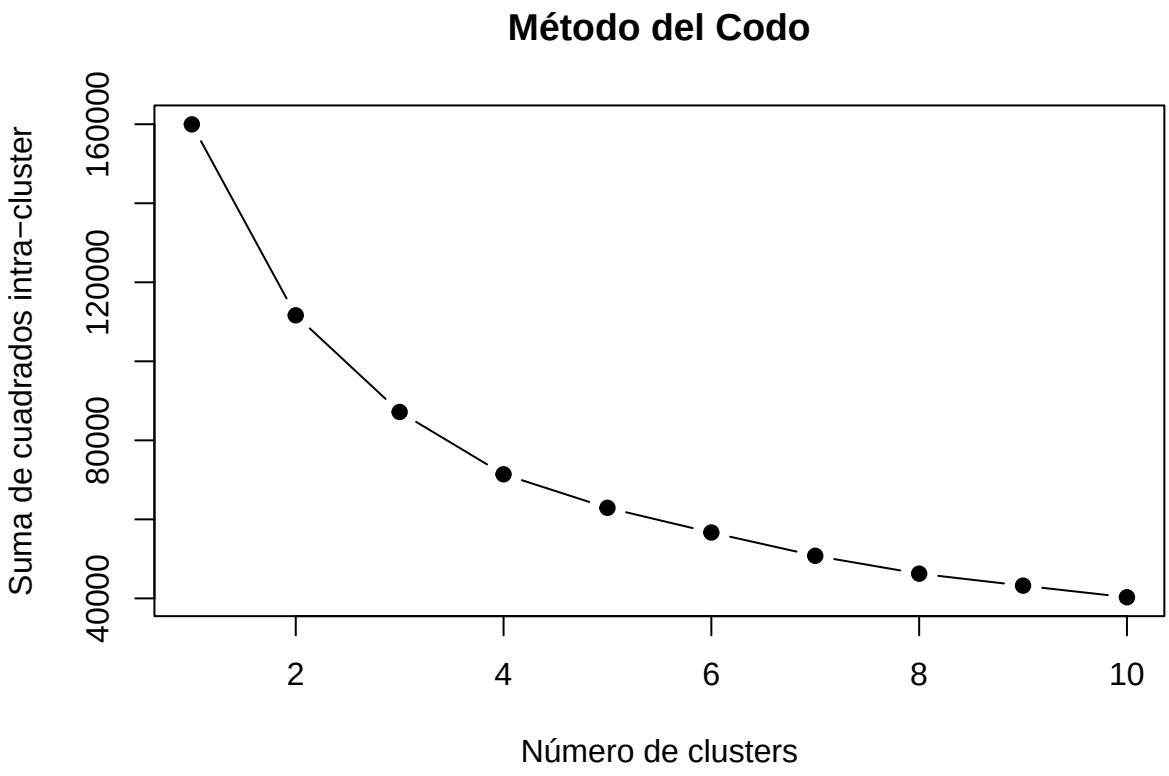

```r
ggplot(datos_limpios, aes(x = EDADMUJ)) +
  geom_histogram(binwidth = 5, fill = "lightpink", color = "black", alpha = 0.7) +
  labs(title = "Distribución de edad de mujeres",
       x = "Edad", y = "Frecuencia") +
  theme_minimal()
```



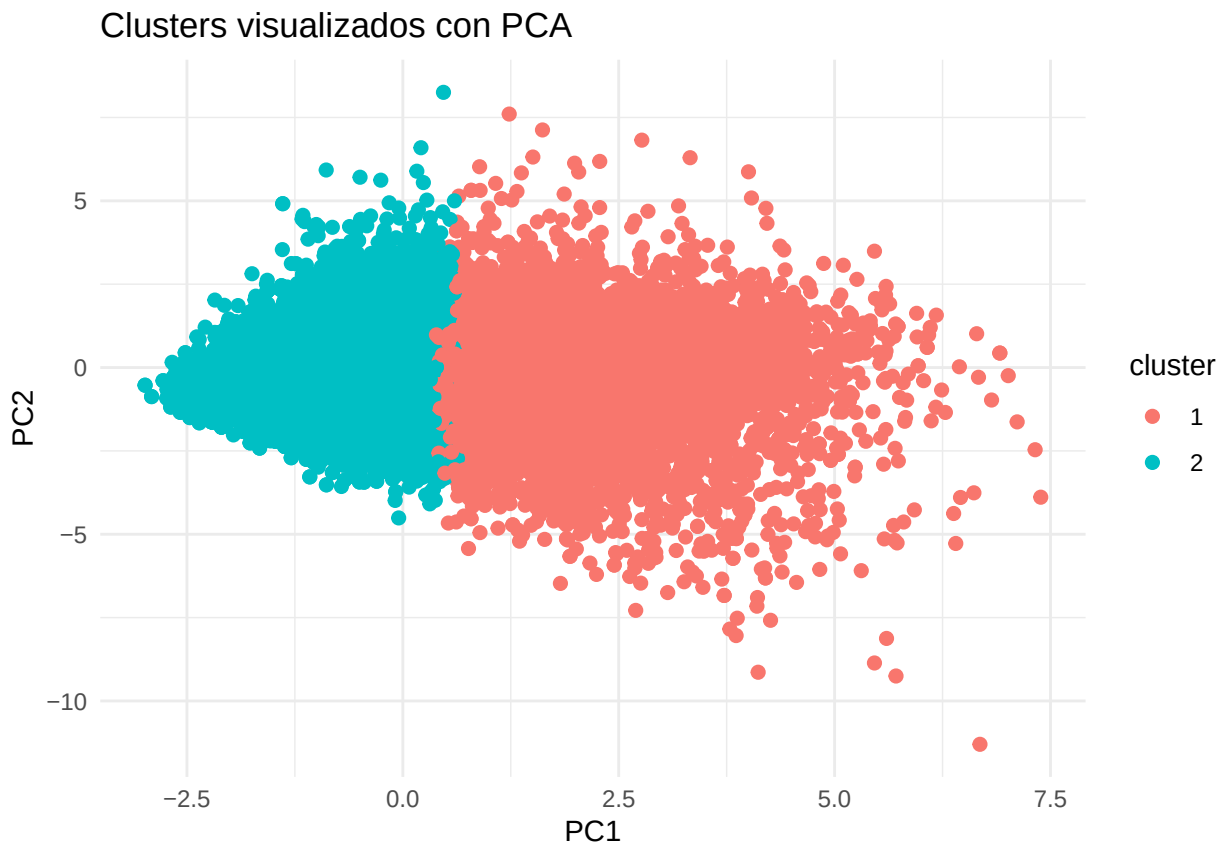

```
#PARTE DEL CLUSTERING
#Preambulo
# Seleccionar solo columnas numéricas
datos_numericos <- datos_limpios %>% select(where(is.numeric))

# Eliminar columnas con NA si las hubiera
datos_numericos <- na.omit(datos_numericos)

# Escalar los datos (muy importante en clustering)
datos_escalados <- scale(datos_numericos)

#a. Determinar el número óptimo de clusters
# Calcular total within-cluster sum of squares para k = 1 a 10
wss <- sapply(1:10, function(k) {
  kmeans(datos_escalados, centers = k, nstart = 10)$tot.withinss
})
```

```
## Warning: did not converge in 10 iterations
```

```
# Graficar el codo
plot(1:10, wss, type = "b", pch = 19,
     xlab = "Número de clusters",
     ylab = "Suma de cuadrados intra-cluster",
     main = "Método del Codo")
```




```r
#b. Ahora aplicar K-means cluster
# Elegir k según el gráfico anterior (ejemplo: 3)
k <- 2

set.seed(123)
kmeans_resultado <- kmeans(datos_escalados, centers = k, nstart = 25)

# Añadir el cluster al dataframe original
datos_limpios$cluster <- factor(kmeans_resultado$cluster)

#c. Visualización de los clusters
# PCA para reducir dimensiones
pca <- prcomp(datos_escalados)
pca_df <- data.frame(pca$x[, 1:2], cluster = datos_limpios$cluster)

# Visualización
ggplot(pca_df, aes(x = PC1, y = PC2, color = cluster)) +
  geom_point(size = 2) +
  labs(title = "Clusters visualizados con PCA") +
  theme_minimal()
```




```
# Ver que caracteriza cada cluster
aggregate(datos_numericos, by = list(Cluster = datos_limpios$cluster), FUN = mean)
```

```
##   Cluster   AÑOREG  EDADHOM  EDADMUJ dif_edad
## 1       1 2020.172 48.24173 42.80737 5.434359
## 2       2 2019.319 31.04731 28.82135 2.225955
```

Pasamos a la fase de modelación:

```
library(tidyverse)
library(caret)
```

```
## Cargando paquete requerido: lattice
```

```
##
## Adjuntando el paquete: 'caret'
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 4.4.3
```

```
## randomForest 4.7-1.2

## Type rfNews() to see new features/changes/bug fixes.

##
## Adjuntando el paquete: 'randomForest'

## The following object is masked from 'package:dplyr':
##
##     combine

## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
library(e1071)
library(glmnet)
```

```
## Warning: package 'glmnet' was built under R version 4.4.3

## Cargando paquete requerido: Matrix

##
## Adjuntando el paquete: 'Matrix'

## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack

## Loaded glmnet 4.1-8
```

```
library(ggplot2)
```

```
# Cargar dataset agregado
datos_agregados <- read.csv("divorcios_agregados_2014_2024.csv")

# Diagnóstico inicial: Verificar NA en datos_agregados
cat("Resumen de datos_agregados:\n")
```

```
## Resumen de datos_agregados:
```

```
summary(datos_agregados)
```

```
##      AÑOREG       divorcios    edadhom_promedio edadmuj_promedio
##  Min.   :2014   Min.   :1014   Min.   :34.24    Min.   :30.71
##  1st Qu.:2016   1st Qu.:2535   1st Qu.:34.62    1st Qu.:31.29
##  Median :2019   Median :2952   Median :35.36    Median :32.35
##  Mean   :2019   Mean   :3636   Mean   :35.64    Mean   :32.47
##  3rd Qu.:2022   3rd Qu.:4926   3rd Qu.:36.53    3rd Qu.:33.53
```

```
##  Max.   :2024   Max.   :6632   Max.   :37.70    Max.   :34.79
##     pandemia
##  Min.   :0.0000
##  1st Qu.:0.0000
##  Median :0.0000
##  Mean   :0.2727
##  3rd Qu.:0.5000
##  Max.   :1.0000
```

```
cat("\nValores NA en datos_agregados:\n")
```

```
##
## Valores NA en datos_agregados:
```

```
print(colSums(is.na(datos_agregados)))
```

```
##           AÑOREG        divorcios edadhom_promedio edadmuj_promedio
##                0                0                0                0
##         pandemia
##                0
```

```
cat("\nDatos agregados completos:\n")
```

```
##
## Datos agregados completos:
```

```
print(datos_agregados)
```

```
##    AÑOREG divorcios edadhom_promedio edadmuj_promedio pandemia
## 1    2014      3124         34.53585         31.21831        0
## 2    2015      2650         34.25660         30.70906        0
## 3    2016      2490         34.23534         30.88313        0
## 4    2017      2580         34.70969         31.36512        0
## 5    2018      2952         34.93767         31.66192        0
## 6    2019      4352         35.35639         32.35064        0
## 7    2020      2295         35.97996         32.73856        1
## 8    2021      5501         36.15979         33.18033        1
## 9    2022      6403         36.89208         33.88833        1
## 10   2023      6632         37.32343         34.40003        0
## 11   2024      1014         37.70316         34.78698        0
```

```
# Imputar NA en edadhom_promedio y edadmuj_promedio (si los hay)
datos_agregados <- datos_agregados %>%
  mutate(
    edadhom_promedio = ifelse(is.na(edadhom_promedio),
                              mean(edadhom_promedio, na.rm = TRUE),
                              edadhom_promedio),
    edadmuj_promedio = ifelse(is.na(edadmuj_promedio),
                              mean(edadmuj_promedio, na.rm = TRUE),
                              edadmuj_promedio)
```

```
  )

# Verificar después de imputación
cat("\nValores NA en datos_agregados después de imputación:\n")
```

```
##
## Valores NA en datos_agregados después de imputación:
```

```
print(colSums(is.na(datos_agregados)))
```

```
##           AÑOREG        divorcios edadhom_promedio edadmuj_promedio
##                0                0                0                0
##         pandemia
##                0
```

```
# Dividir datos (entrenamiento: 2014-2019, prueba: 2020-2024)
train_data <- datos_agregados %>% filter(AÑOREG <= 2019)
test_data <- datos_agregados %>% filter(AÑOREG >= 2020)

# Escalar predictores (excluir pandemia por varianza cero)
predictors <- c("AÑOREG", "edadhom_promedio", "edadmuj_promedio")
train_scaled <- train_data
test_scaled <- test_data
train_scaled[predictors] <- scale(train_data[predictors])
test_scaled[predictors] <- scale(test_data[predictors],
                                 center = attr(scale(train_data[predictors]), "scaled:center"),
                                 scale = attr(scale(train_data[predictors]), "scaled:scale"))

# Diagnóstico: Verificar NA y varianza
cat("Resumen de train_data:\n")
```

```
## Resumen de train_data:
```

```
summary(train_data)
```

```
##      AÑOREG       divorcios    edadhom_promedio edadmuj_promedio    pandemia
##  Min.   :2014   Min.   :2490   Min.   :34.24    Min.   :30.71    Min.   :0
##  1st Qu.:2015   1st Qu.:2598   1st Qu.:34.33    1st Qu.:30.97    1st Qu.:0
##  Median :2016   Median :2801   Median :34.62    Median :31.29    Median :0
##  Mean   :2016   Mean   :3025   Mean   :34.67    Mean   :31.36    Mean   :0
##  3rd Qu.:2018   3rd Qu.:3081   3rd Qu.:34.88    3rd Qu.:31.59    3rd Qu.:0
##  Max.   :2019   Max.   :4352   Max.   :35.36    Max.   :32.35    Max.   :0
```

```
cat("\nValores NA en train_data:\n")
```

```
##
## Valores NA en train_data:
```

```r
print(colSums(is.na(train_data)))
```

```
##           AÑOREG        divorcios edadhom_promedio edadmuj_promedio
##                0                0                0                0
##         pandemia
##                0
```

```r
cat("\nResumen de train_scaled:\n")
```

```
##
## Resumen de train_scaled:
```

```r
summary(train_scaled)
```

```
##      AÑOREG          divorcios    edadhom_promedio  edadmuj_promedio
##  Min.   :-1.3363   Min.   :2490   Min.   :-1.0166   Min.   :-1.1103
##  1st Qu.:-0.6682   1st Qu.:2598   1st Qu.:-0.8045   1st Qu.:-0.6736
##  Median : 0.0000   Median :2801   Median :-0.1145   Median :-0.1236
##  Mean   : 0.0000   Mean   :3025   Mean   : 0.0000   Mean   : 0.0000
##  3rd Qu.: 0.6682   3rd Qu.:3081   3rd Qu.: 0.4861   3rd Qu.: 0.3777
##  Max.   : 1.3363   Max.   :4352   Max.   : 1.5938   Max.   : 1.6696
##     pandemia
##  Min.   :0
##  1st Qu.:0
##  Median :0
##  Mean   :0
##  3rd Qu.:0
##  Max.   :0
```

```r
cat("\nValores NA en train_scaled:\n")
```

```
##
## Valores NA en train_scaled:
```

```r
print(colSums(is.na(train_scaled)))
```

```
##           AÑOREG        divorcios edadhom_promedio edadmuj_promedio
##                0                0                0                0
##         pandemia
##                0
```

```r
cat("\nVarianza de predictores en train_data:\n")
```

```
##
## Varianza de predictores en train_data:
```

```r
print(sapply(train_data[predictors], var, na.rm = TRUE))
```

```
##           AÑOREG edadhom_promedio edadmuj_promedio
##        3.5000000        0.1844297        0.3487265
```

```r
# Configurar validación cruzada para series temporales
ctrl <- trainControl(method = "timeslice",
                     initialWindow = 4,
                     horizon = 1,
                     fixedWindow = TRUE,
                     skip = 0,
                     summaryFunction = defaultSummary)
```

```r
# Modelo 1: Lasso
lasso_model <- train(
  divorcios ~ .,
  data = train_data,
  method = "glmnet",
  trControl = ctrl,
  tuneGrid = expand.grid(alpha = 1, lambda = c(0.1, 0.5, 1, 2)),
  metric = "RMSE"
)
```

```
## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info = trainInfo,
## : There were missing values in resampled performance measures.
```

```r
# Modelo 2: Random Forest
rf_model <- train(
  divorcios ~ .,
  data = train_data,
  method = "rf",
  trControl = ctrl,
  tuneGrid = expand.grid(mtry = c(1, 2, 3)),
  metric = "RMSE"
)
```

```
## Warning in randomForest.default(x, y, mtry = param$mtry, ...): The response has
## five or fewer unique values.  Are you sure you want to do regression?
```

```
## Warning in randomForest.default(x, y, mtry = param$mtry, ...): The response has
## five or fewer unique values.  Are you sure you want to do regression?
## Warning in randomForest.default(x, y, mtry = param$mtry, ...): The response has
## five or fewer unique values.  Are you sure you want to do regression?
## Warning in randomForest.default(x, y, mtry = param$mtry, ...): The response has
## five or fewer unique values.  Are you sure you want to do regression?
## Warning in randomForest.default(x, y, mtry = param$mtry, ...): The response has
## five or fewer unique values.  Are you sure you want to do regression?
## Warning in randomForest.default(x, y, mtry = param$mtry, ...): The response has
## five or fewer unique values.  Are you sure you want to do regression?
```

```
## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info = trainInfo,
## : There were missing values in resampled performance measures.
```

```r
# Modelo 3: SVR
svr_model <- train(
  divorcios ~ .,
```

```r
  data = train_scaled,
  method = "svmRadial",
  trControl = ctrl,
  tuneGrid = expand.grid(sigma = c(0.1, 1), C = c(0.1, 1, 10)),
  metric = "RMSE"
)
```

```
## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.

## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.
## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.
## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.
## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.
## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.
## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.
## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.
## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.
## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.
## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.
## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.

## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info = trainInfo,
## : There were missing values in resampled performance measures.

## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.
```

Resultados de los modelos:

```r
# Evaluar modelos en el conjunto de prueba
lasso_pred <- predict(lasso_model, test_data)
rf_pred <- predict(rf_model, test_data)
svr_pred <- predict(svr_model, test_scaled)

lasso_metrics <- postResample(lasso_pred, test_data$divorcios)
rf_metrics <- postResample(rf_pred, test_data$divorcios)
svr_metrics <- postResample(svr_pred, test_data$divorcios)

# Mostrar resultados
cat("Lasso Metrics:\n"); print(lasso_metrics)
```

```
## Lasso Metrics:

##         RMSE     Rsquared          MAE
## 2.602916e+03 2.766903e-04 1.901340e+03
```

```r
cat("Random Forest Metrics:\n"); print(rf_metrics)
```

```
## Random Forest Metrics:

##     RMSE Rsquared      MAE
## 2344.306       NA 2276.733
```

```r
cat("SVR Metrics:\n"); print(svr_metrics)
```

```
## SVR Metrics:

##         RMSE     Rsquared          MAE
## 2768.5354737    0.2031767 2484.3917162
```

```r
# Mejores parámetros
cat("\nMejores parámetros:\n")
```

```
##
## Mejores parámetros:
```

```r
cat("Lasso:", toString(lasso_model$bestTune), "\n")
```

```
## Lasso: 1, 2
```

```r
cat("Random Forest:", toString(rf_model$bestTune), "\n")
```

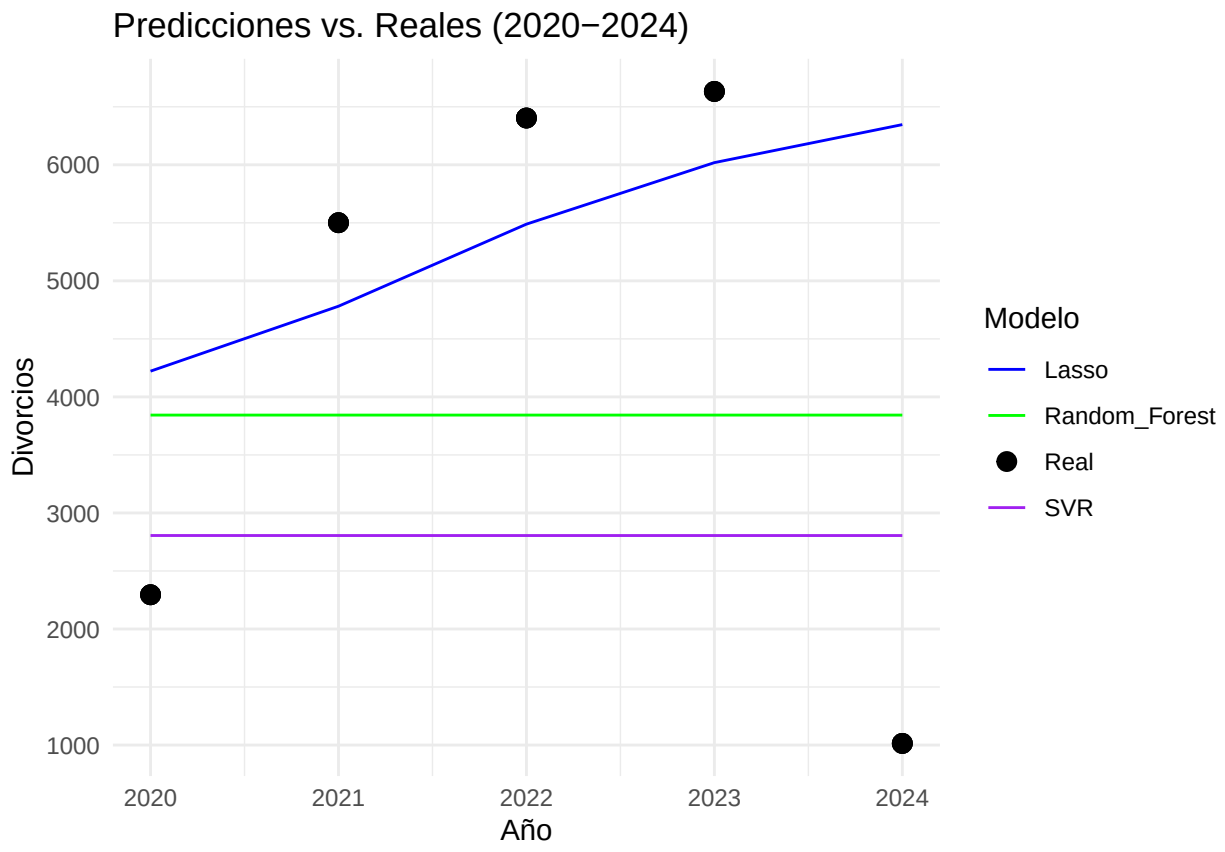
```
## Random Forest: 3
```

```r
cat("SVR:", toString(svr_model$bestTune), "\n")
```

```
## SVR: 1, 10
```

```r
# Guardar resultados en un CSV
resultados <- data.frame(
  Modelo = c("Lasso", "Random Forest", "SVR"),
  RMSE = c(lasso_metrics["RMSE"], rf_metrics["RMSE"], svr_metrics["RMSE"]),
  MAE = c(lasso_metrics["MAE"], rf_metrics["MAE"], svr_metrics["MAE"])
)
write.csv(resultados, "resultados_modelos_2014_2024.csv", row.names = FALSE)

# Graficar predicciones vs. reales (todos los modelos)
predicciones <- data.frame(
  Año = test_data$AÑOREG,
  Real = test_data$divorcios,
  Lasso = lasso_pred,
  Random_Forest = rf_pred,
  SVR = svr_pred
)
predicciones_long <- predicciones %>%
  pivot_longer(cols = c(Lasso, Random_Forest, SVR),
               names_to = "Modelo",
               values_to = "Predicho")

ggplot(predicciones_long, aes(x = Año, y = Predicho, color = Modelo)) +
  geom_line() +
  geom_point(aes(y = Real, color = "Real"), size = 3) +
```

```
  theme_minimal() +
  labs(title = "Predicciones vs. Reales (2020-2024)",
       x = "Año", y = "Divorcios") +
  scale_color_manual(values = c("Lasso" = "blue", "Random_Forest" = "green",
                                "SVR" = "purple", "Real" = "black"))
```

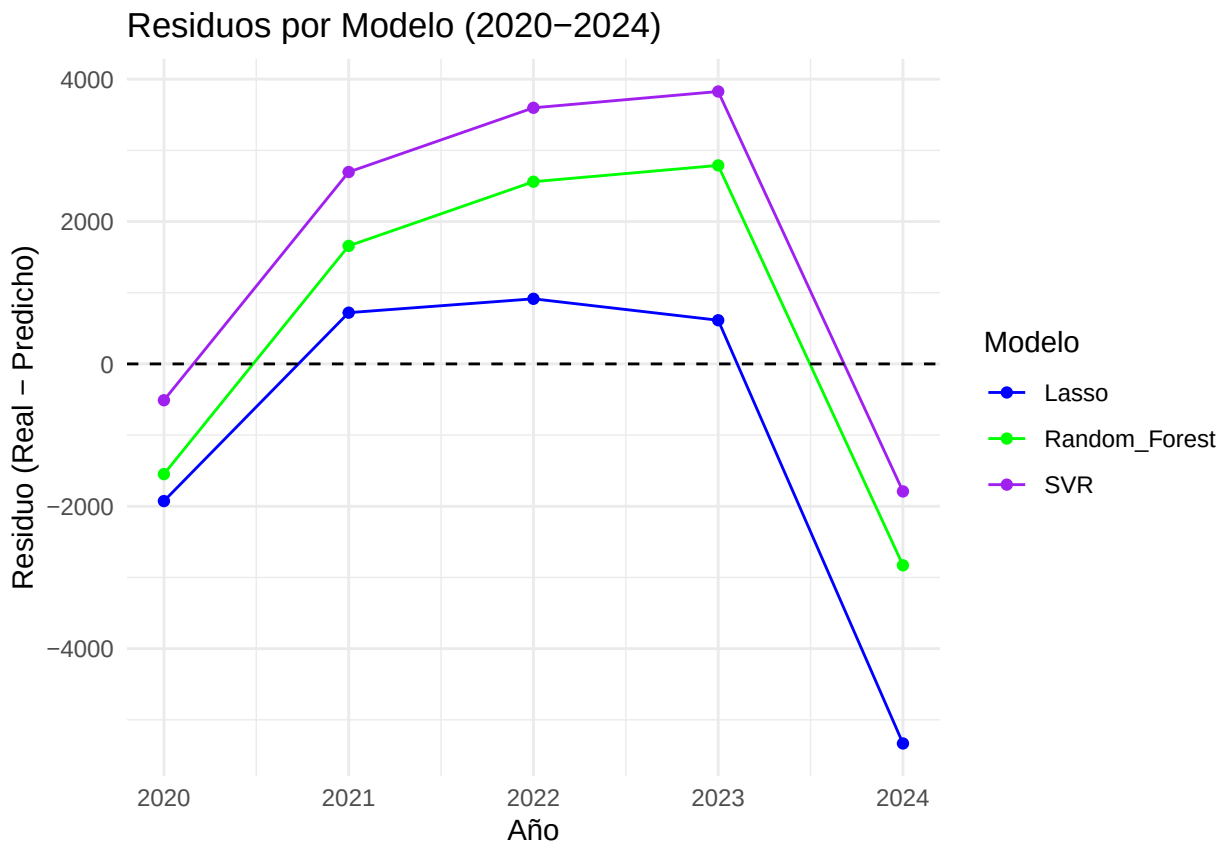



```
ggsave("predicciones_vs_reales_2014_2024.png")
```

```
## Saving 6.5 x 4.5 in image
```

```
# Graficar residuos
residuos <- data.frame(
  Año = test_data$AÑOREG,
  Lasso = test_data$divorcios - lasso_pred,
  Random_Forest = test_data$divorcios - rf_pred,
  SVR = test_data$divorcios - svr_pred
)
residuos_long <- residuos %>%
  pivot_longer(cols = c(Lasso, Random_Forest, SVR),
               names_to = "Modelo",
               values_to = "Residuo")

ggplot(residuos_long, aes(x = Año, y = Residuo, color = Modelo)) +
  geom_line() +
```

```
  geom_point() +
  geom_hline(yintercept = 0, linetype = "dashed") +
  theme_minimal() +
  labs(title = "Residuos por Modelo (2020-2024)",
       x = "Año", y = "Residuo (Real - Predicho)") +
  scale_color_manual(values = c("Lasso" = "blue", "Random_Forest" = "green", "SVR" = "purple"))
```




```
ggsave("residuos_modelos_2014_2024.png")
```

```
## Saving 6.5 x 4.5 in image
```

```
# Guardar modelos
saveRDS(lasso_model, "lasso_model_2014_2024.rds")
saveRDS(rf_model, "rf_model_2014_2024.rds")
saveRDS(svr_model, "svr_model_2014_2024.rds")
```