Tarea 05

```
## Librerías
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------------------------------
## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.3     v dplyr   1.0.1
## v tidyr   1.1.1     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0
## -- Conflicts ------------------------------------------------------------------------------------
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(patchwork)
```

```
# Funciones relevantes


## Función para simular varias tomas de muestra de datos y calcular el valor de interés
muest_simul_val <- function(datos, muestra, simuls){
  # Función para simular varias tomas de muestra de la tabla de crímenes y calcular el valor de interés
  #   args:
  #     datos (dataframe): datos de crímenes
  #     muestra (int): tamaño de la muestra que se tomará
  #     simuls (int): número de simulaciones
  #   returns:
  #     simul_res (dataframe): tabla con los resultados de todas las simulaciones

  ## Lista con número de simulaciones
  simuls <- seq(1:simuls)

  ## Creación de listas vacías donde se almacenarán los resultados
  simuls_lst <- c()
  res_lst <- c()

  ## Loop para llevar a cabo cada una de las simulaciones
  for (simul in simuls) {

    smpl <- sample_n(datos, muestra)
    val_calc <- calc_muestra(smpl) %>%
      pull(val)

    simuls_lst <- append(simuls_lst, simul)
    res_lst <- append(res_lst, val_calc)

  }
```

```
  res_tbl <- tibble(
    simulacion = simuls_lst,
    resultado = res_lst
  )

  res_tbl

}
```

1. **Proporciones.** Usaremos datos de reincidencia en conducta criminal del estado de Iowa, este estado sigue a los delincuentes por un periodo de 3 años y registra el número de días hasta reincidencia para aquellos que son readmitidos en prisión. El deprtamento de correcciones utiliza los datos de reincidencia para evaluar sus programas de prevención de recaída en conducta criminal.

Los datos Recidivism contienen información de todos los delincuentes condenados por dos tipos de delito durante 2010 (*Recid* indica si recayeron en conducta criminal).

- De éstos 31.6% reincidieron y volvieron a prisión. Utiliza simulación para aproximar la simulación muestral de $\hat{p}$, la proporción de delincuentes que reincidieron para muestras de tamaño 25.

```
recidivism <- read_csv("Recidivism.csv")
```

```
## Parsed with column specification:
## cols(
##   Gender = col_character(),
##   Age = col_character(),
##   Age25 = col_character(),
##   Race = col_character(),
##   Offense = col_character(),
##   Recid = col_character(),
##   Type = col_character(),
##   Days = col_double()
## )
```

```
## Validación de cifra estadística de reincidencias.
vals <- recidivism %>%
  count(Recid) %>%
  mutate(prop = 100*n/sum(n)) %>%
  filter(Recid == "Yes") %>%
  select(-n)
vals
```

```
## # A tibble: 1 x 2
##   Recid  prop
##   <chr> <dbl>
## 1 Yes    31.6
```

```
## Muestra para aproximar valor de interés
muestra_dat <- sample_n(recidivism, 25) %>%
  count(Recid) %>%
  mutate(prop = 100*n/sum(n))
muestra_dat
```

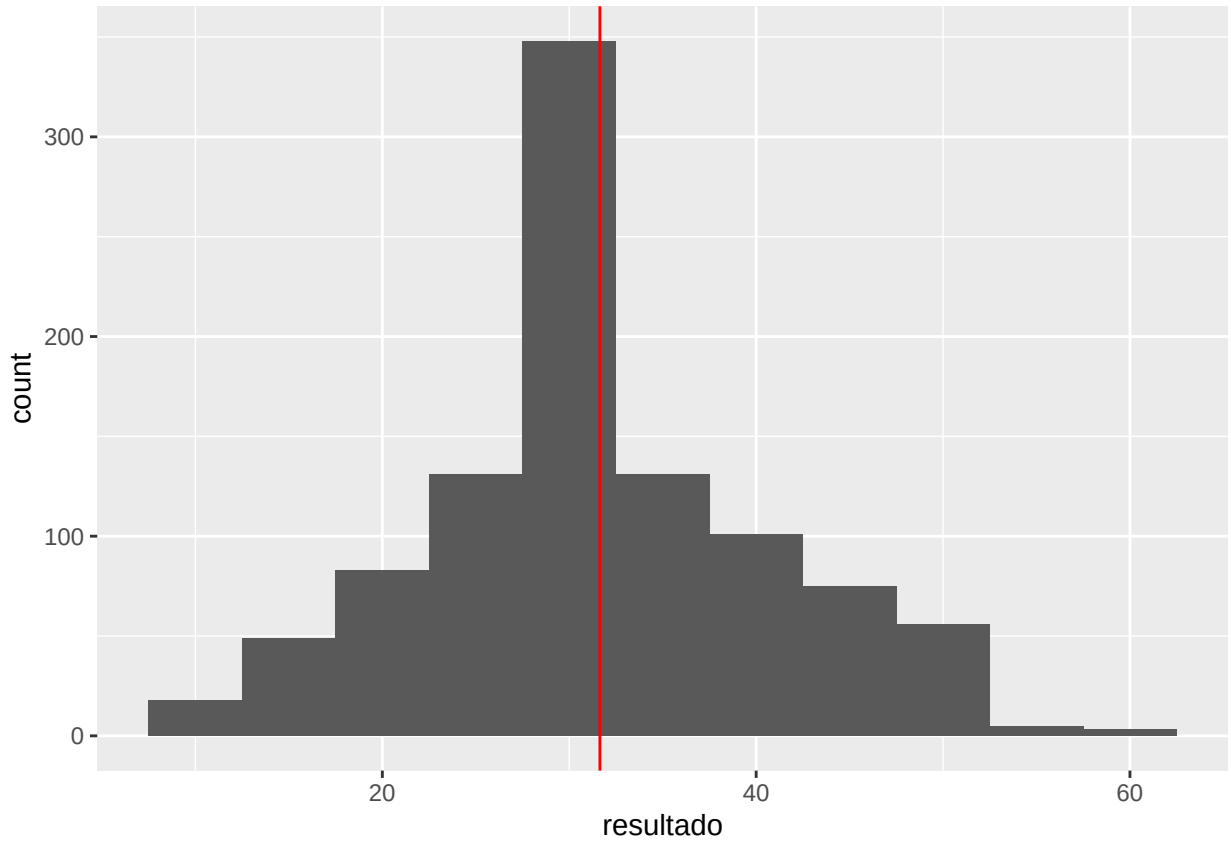
```
## # A tibble: 2 x 3
##   Recid     n  prop
##   <chr> <int> <dbl>
## 1 No       21    84
```

```
## 2 Yes       4    16
```

```
## Función para obtener proporción de individuos que incurrieron en incidencia
calc_muestra <- function(datos){
  # Función para obtener proporción de individuos que incurrieron en incidencia
  #   args:
  #     datos (dataframe): tabla con datos de interés
  #     col_reinc (string): columna que contiene la categorización de reincidencia
  #   returns:
  #     res (dataframe): resultado de agrupación

  res <- datos %>%
    count(Recid) %>%
    mutate(val = 100*n/sum(n)) %>%
    filter(Recid == "Yes") %>%
    select(-n)
  res
}


## Ejecución de la función creada
muest_simul_res <- muest_simul_val(recidivism, 25, 1000)
muest_simul_res
```

```
## # A tibble: 1,000 x 2
##    simulacion resultado
##         <int>     <dbl>
##  1          1        24
##  2          2        36
##  3          3        28
##  4          4        36
##  5          5        32
##  6          6        24
##  7          7        24
##  8          8        36
##  9          9        32
## 10         10        32
## # ... with 990 more rows
```

```
## Distribución de los resultados obtenidos
graf <- ggplot(
  muest_simul_res,
  aes(x = resultado)
) +
  geom_histogram(binwidth = 5) +
  geom_vline(xintercept = vals$prop, colour = "red")
graf
```



- Calcula el error estándar de $\hat{p}$, y compáralo con el teórico $\sqrt{p(1-p)/n}$.

```
## Cálculo del error estándar
err_std <- sd(muest_simul_res$resultado)
err_std
```

```
## [1] 9.185303
```

```
## Error téorico
err_teo <- sqrt((vals$prop/100)*(1 - vals$prop/100)/nrow(recidivism))
err_teo
```

```
## [1] 0.003564669
```

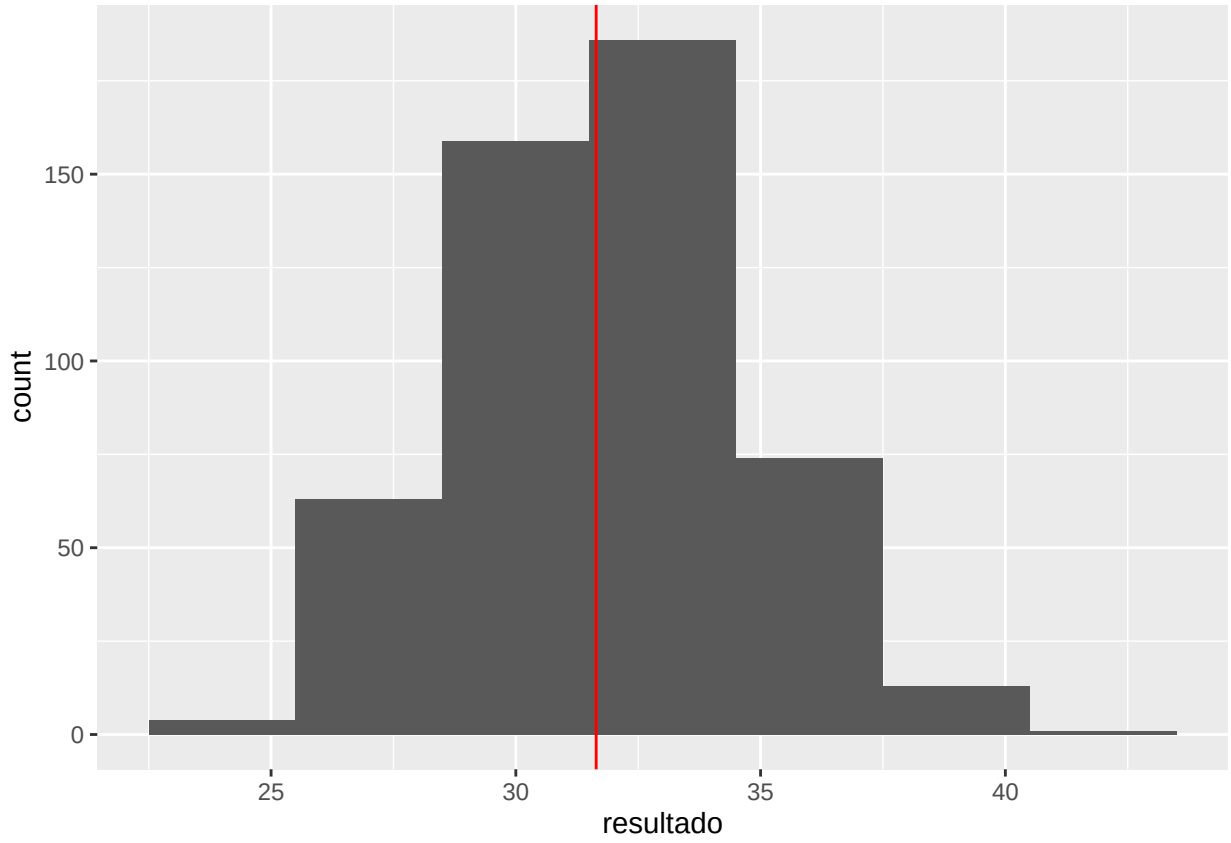
- Repite para muestras de tamaño 250 y compara.

```
## Ejecución de la función creada
muest_simul_res <- muest_simul_val(recidivism, 250, 500)
muest_simul_res
```

```
## # A tibble: 500 x 2
##    simulacion resultado
##         <int>     <dbl>
##  1          1      31.2
##  2          2      29.6
##  3          3      31.2
##  4          4      33.2
##  5          5      34.8
##  6          6      31.6
##  7          7      32.8
```

```
##  8          8      31.6
##  9          9      31.6
## 10         10      30
## # ... with 490 more rows
```

```
## Distribución de los resultados obtenidos
graf <- ggplot(
  muest_simul_res,
  aes(x = resultado)
) +
  geom_histogram(binwidth = 3) +
  geom_vline(xintercept = vals$prop, colour = "red")
graf
```




```
## Cálculo del error estándar
err_std <- sd(muest_simul_res$resultado)
err_std
```

```
## [1] 2.848312
```

2. **El error estándar de una media.** Supongamos que x es una variable aleatoria que toma valores en los reales con distribución de probabilidad F. Denotamos por μ y σ^2 la media y varianza de F,

$$\mu = E(x),$$

$$\sigma^2 = var(x) = E[(x - \mu)^2]$$

Ahora, sea $(X_1, ..., X_n)$ una muestra aleatoria de F, de tamaño n, la media de la muestra $\bar{X} = \sum_{i=1}^{n} X_i/n$ tiene:

- esperanza μ,

- varianza σ^2/n.

En palabras: la esperanza de $\bar{X}$ es la misma que la esperanza de x, pero la varianza de $\bar{X}$ es $1/n$ veces la varianza de x, así que entre mayor es la n tenemos una mejor estimación de μ.

En el caso del estimador de la media $\bar{X}$, el error estándar quedaría

$$ee(\bar{X}) = [var(\bar{X})]^{1/2} = \sigma/\sqrt{n}.$$

Entonces,

- Consideramos los datos de ENLACE edo. de México (ENLACE era una prueba estandarizada que se aplicaba a todos los alumnos de primaria en México), y la columna de calificaciones de español 3º de primaria (`esp_3`).

```
enlace <- read_csv("enlace_15.csv")
```

```
## Parsed with column specification:
## cols(
##   id = col_double(),
##   cve_ent = col_double(),
##   turno = col_character(),
##   tipo = col_character(),
##   esp_3 = col_double(),
##   esp_6 = col_double(),
##   n_eval_3 = col_double(),
##   n_eval_6 = col_double()
## )
```

- Genera un histograma de las calificaciones de 3º de primaria. Calcula la media y la desviación estándar.

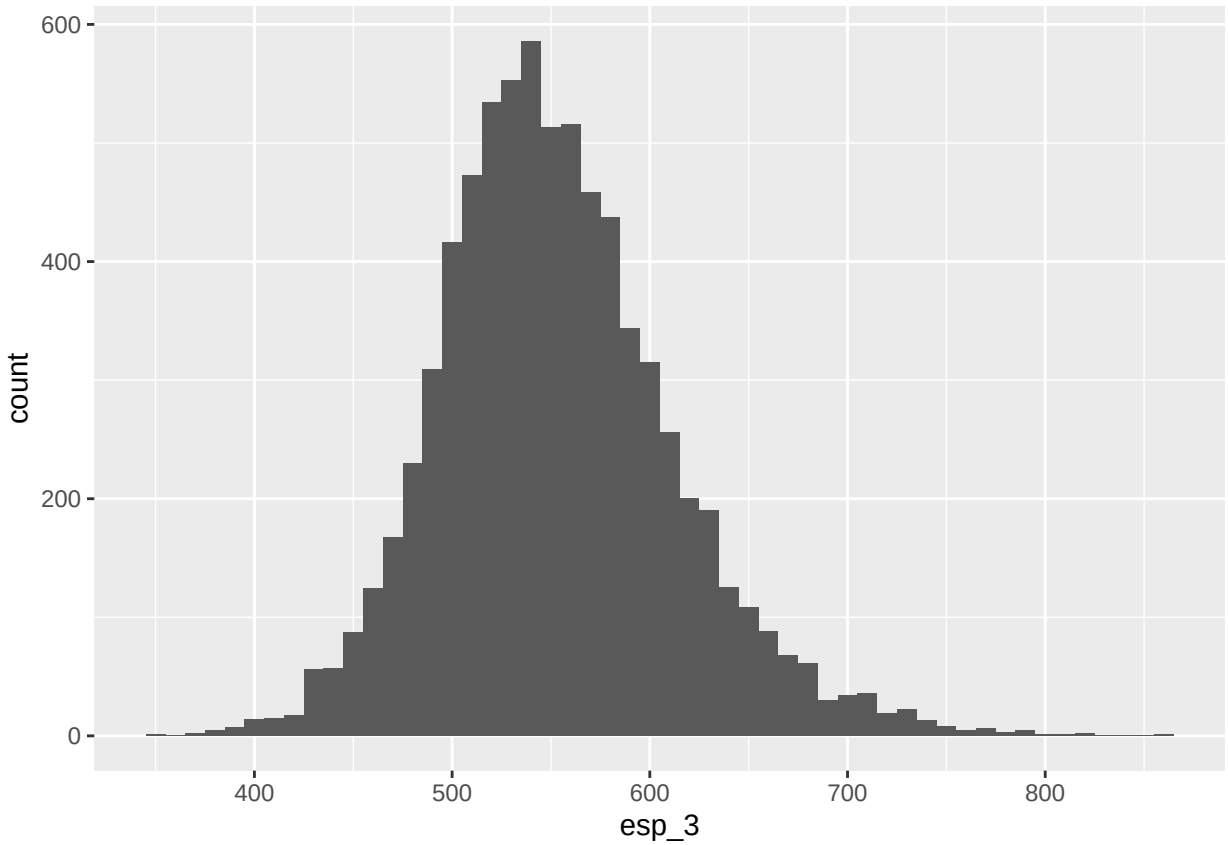
```
## Cálculo de media y desviación estándar
sprintf("La media de las calificaciones de los alumnos de 3ro de primaria es: %0.2f", mean(enlace$esp_3
```

```
## [1] "La media de las calificaciones de los alumnos de 3ro de primaria es: 552.99"
```

```
sprintf("La desviación estándar de las calificaciones de los alumnos de 3ro de primaria es: %0.2f", sd(
```

```
## [1] "La desviación estándar de las calificaciones de los alumnos de 3ro de primaria es: 59.26"
```

```
## Histograma con calificaciones
g_1 <- ggplot(
  enlace,
  aes(x = esp_3)
) +
  geom_histogram(binwidth = 10)
g_1
```

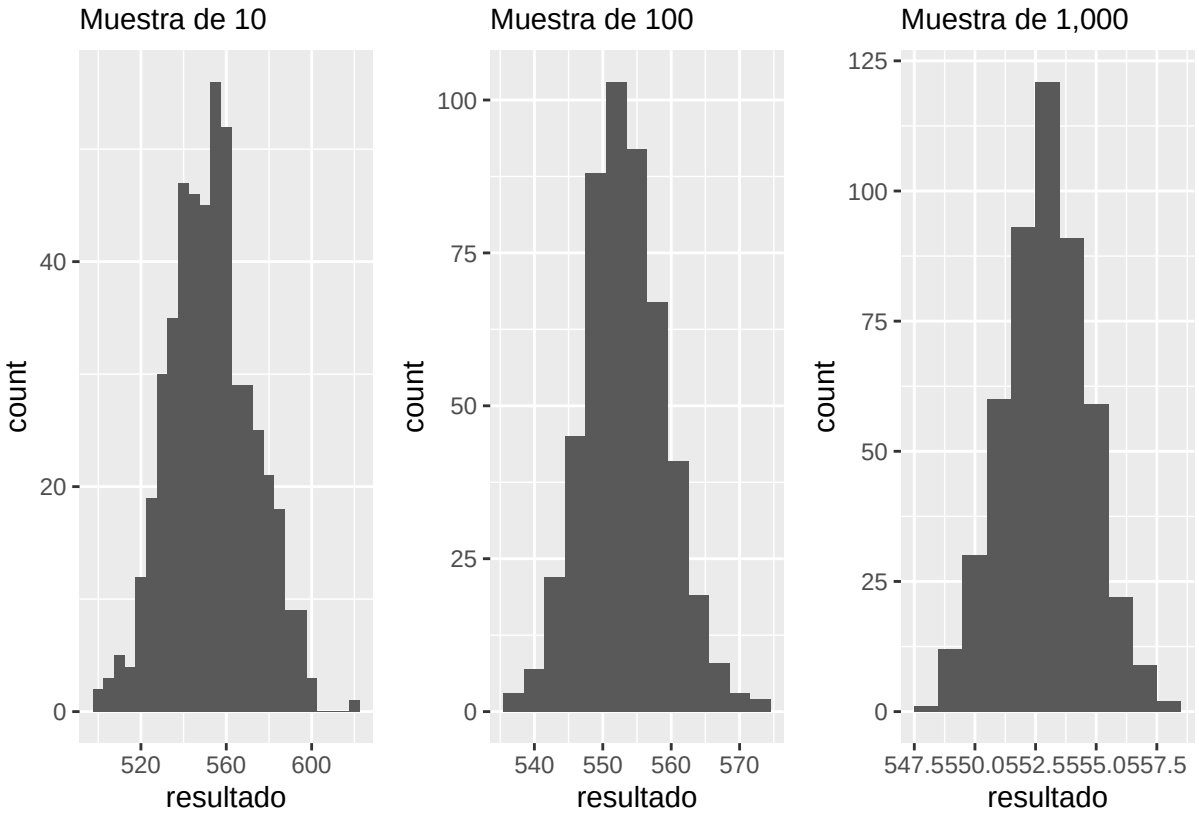


- Para tamaños de muestra $n = 10, 100, 1000$:
- Aproximareos la distribución muestral:

i) Simula 5,000 muestras aleatorias,
ii) Calcula la media en cada muestra,
iii) Realiza un histograma de la distribución muestral de las medias (las medias del paso anterior)
iv) Aproxima el error estándar calculando la desviación estándar de las medias del paso ii.

```
## Función para calcular la media de una muestra
calc_muestra <- function(data){
  res <- data %>%
    summarise(val = mean(esp_3))
  res
}


## Ejecución de la función creada
muest_simul_res_10 <- muest_simul_val(enlace, 10, 500)
muest_simul_res_100 <- muest_simul_val(enlace, 100, 500)
muest_simul_res_1000 <- muest_simul_val(enlace, 1000, 500)


## Distribución de los resultados obtenidos
graf_10 <- ggplot(
  muest_simul_res_10,
  aes(x = resultado)
) +
```

```
  geom_histogram(binwidth = 5) +
  labs(subtitle = "Muestra de 10")

graf_100 <- ggplot(
  muest_simul_res_100,
  aes(x = resultado)
) +
  geom_histogram(binwidth = 3) +
  labs(subtitle = "Muestra de 100")

graf_1000 <- ggplot(
  muest_simul_res_1000,
  aes(x = resultado)
) +
  geom_histogram(binwidth = 1) +
  labs(subtitle = "Muestra de 1,000")

graf_10 + graf_100 + graf_1000
```



- Calcula el error estándar de la media para cada tamaño de muestra usando la fórmula derivada arriba y compara con tus simulaciones.

```
## Cálculo de errores estándar para cada una de las simulaciones
sprintf("Tamaño de muestra de 10 -> %0.2f", sd(muest_simul_res_10$resultado)/sqrt(nrow(muest_simul_res_
```

```
## [1] "Tamaño de muestra de 10 -> 0.88"
```

```
sprintf("Tamaño de muestra de 100 -> %0.2f", sd(muest_simul_res_100$resultado)/sqrt(nrow(muest_simul_res
```

```
## [1] "Tamaño de muestra de 100 -> 0.27"
```

```
sprintf("Tamaño de muestra de 1,000 -> %0.2f", sd(muest_simul_res_1000$resultado)/sqrt(nrow(muest_simul
```

```
## [1] "Tamaño de muestra de 1,000 -> 0.08"
```

- ¿Cómo se comparan los errores estándar correspondientes a los distintos tamaños de muestra?