Examen Parcial 1759004 - Jorge III Altamirano Astorga

Examen Parcial

1759004 Jorge III Altamirano Astorga

Doctor Berdichevsky, presento este examen parcial para la Materia de Estadística Computacional (Otoño 2017). Agradeciendo de antemano su consideración y atenciones.

1. Manipulación y Visualización de Datos

a) Sí, es una de las bases de datos limpios debido a que cumple algunos de estos requisitos:

1. Cada observación corresponde a un renglón
2. Cada variable tiene su columna
3. Cada tipo de unidad observacional forma una tabla.

Adicionalmente yo agregaría que todas no existen datos NA.

```
summary(iris)
```

```
##   Sepal.Length    Sepal.Width     Petal.Length    Petal.Width
##  Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100
##  1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300
##  Median :5.800   Median :3.000   Median :4.350   Median :1.300
##  Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199
##  3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800
##  Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
##        Species
##  setosa    :50
##  versicolor:50
##  virginica :50
##
##
##
```

b) Impresión de las primeras y últimas 6 líneas

```
head(iris, n = 6)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
## 4          4.6         3.1          1.5         0.2  setosa
## 5          5.0         3.6          1.4         0.2  setosa
## 6          5.4         3.9          1.7         0.4  setosa
```

```
tail(iris, n = 6)
```

```
##     Sepal.Length Sepal.Width Petal.Length Petal.Width   Species
## 145          6.7         3.3          5.7         2.5 virginica
## 146          6.7         3.0          5.2         2.3 virginica
## 147          6.3         2.5          5.0         1.9 virginica
## 148          6.5         3.0          5.2         2.0 virginica
## 149          6.2         3.4          5.4         2.3 virginica
```

```
## 150          5.9         3.0          5.1         1.8 virginica
```

c) Tenemos 150 observaciones y 5 variables

```
dim(iris)
```

```
## [1] 150   5
```

```
str(iris)
```

```
## 'data.frame':    150 obs. of  5 variables:
##  $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
##  $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
##  $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
##  $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
##  $ Species     : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 ...
```

d) La clase atómica de las variables en base al comando del inciso anterior son:

- iris es un `data.frame`
- Sepal.Length es un `num` ó numérica
- Sepal.Width es un `num` ó numérica
- Petal.Length es un `num` ó numérica
- Petal.Width es un `num` ó numérica
- Species es un `Factor` con 3 niveles: setosa, versicolor, virginica

e) Salida de las primeras 6 líneas sólo de la especie *Setosa*

```
iris %>% filter(Species == 'setosa') %>% head(n = 6)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
## 4          4.6         3.1          1.5         0.2  setosa
## 5          5.0         3.6          1.4         0.2  setosa
## 6          5.4         3.9          1.7         0.4  setosa
```

f) Salida de las primeras 6 observaciones en orden descendiente de *Petal.Length*

```
iris %>% arrange(desc(Petal.Length)) %>% head(n=6)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width   Species
## 1          7.7         2.6          6.9         2.3 virginica
## 2          7.7         3.8          6.7         2.2 virginica
## 3          7.7         2.8          6.7         2.0 virginica
## 4          7.6         3.0          6.6         2.1 virginica
## 5          7.9         3.8          6.4         2.0 virginica
## 6          7.3         2.9          6.3         1.8 virginica
```

g) Se crea la nueva variable de en mm de *Sepal.Length*. Mostrando las primeras 6 observaciones

```
sepalLength <- iris$Sepal.Length*10
head(sepalLength, n=6)
```

```
## [1] 51 49 47 46 50 54
```

Sin embargo en caso de que se quiera una columna aparte, es necesario usar `mutate`. Ahí no me quedó muy clara la pregunta. Por lo que anoto ambas respuestas aquí crée la columna *Sepal.Length.mm*.

```
iris <- iris %>% mutate(Sepal.Length.mm = Sepal.Length*10)
head(iris, n=6)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
## 4          4.6         3.1          1.5         0.2  setosa
## 5          5.0         3.6          1.4         0.2  setosa
## 6          5.4         3.9          1.7         0.4  setosa
##   Sepal.Length.mm
## 1              51
## 2              49
## 3              47
## 4              46
## 5              50
## 6              54
```

h) Eliminando las observaciones con valores faltantes en *Sepal.Width* e indicando el número de observaciones. *No existen observaciones no numéricas o valores negativos*, por lo que el número de observaciones se mantiene en 150.

```
iris %>% filter(is.na(Sepal.Width) || Sepal.Width <= 0)
```

```
## [1] Sepal.Length    Sepal.Width     Petal.Length    Petal.Width
## [5] Species         Sepal.Length.mm
## <0 rows> (or 0-length row.names)
```

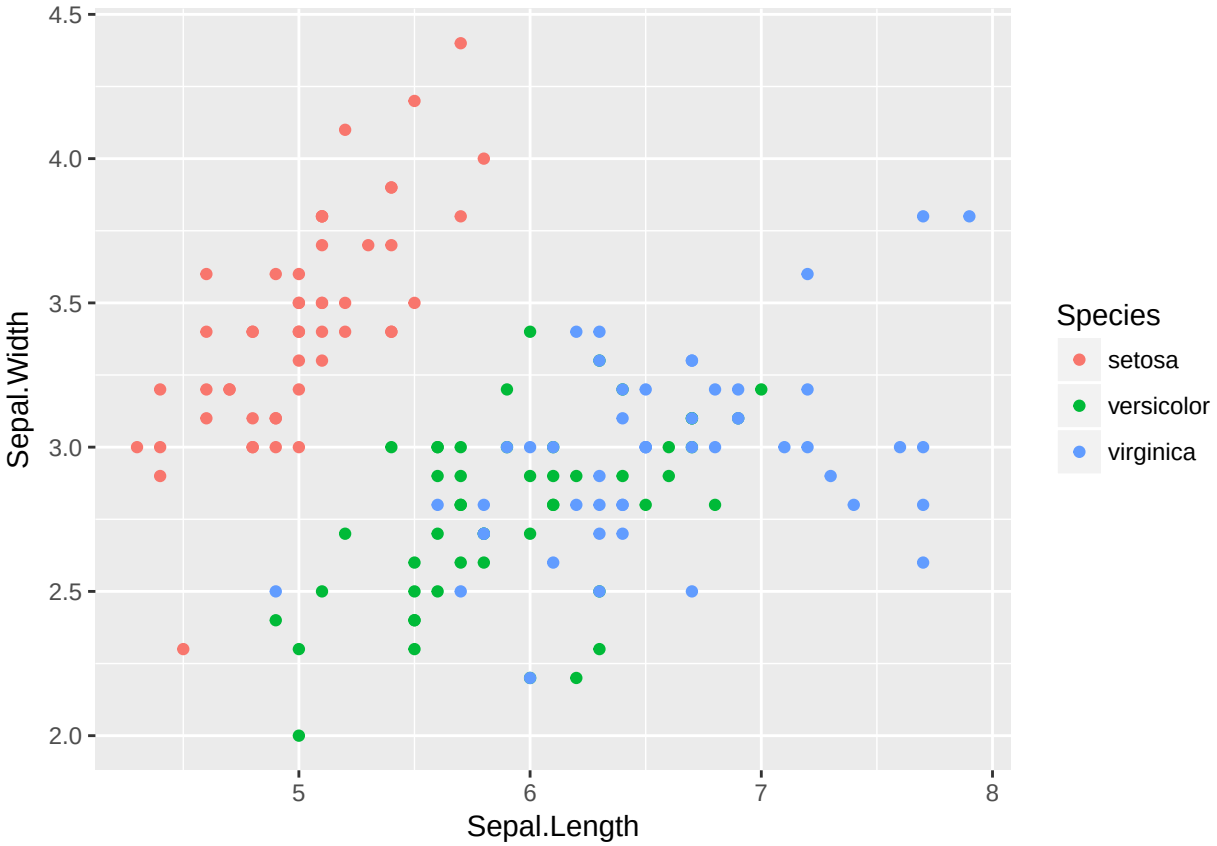
i) La media de *Petal.Width* para cada una de las especies

```
iris %>% select(Species, Petal.Width) %>% group_by(Species) %>% summarise(media = mean(Petal.Width))
```

```
## # A tibble: 3 x 2
##      Species media
##       <fctr> <dbl>
## 1     setosa 0.246
## 2 versicolor 1.326
## 3  virginica 2.026
```

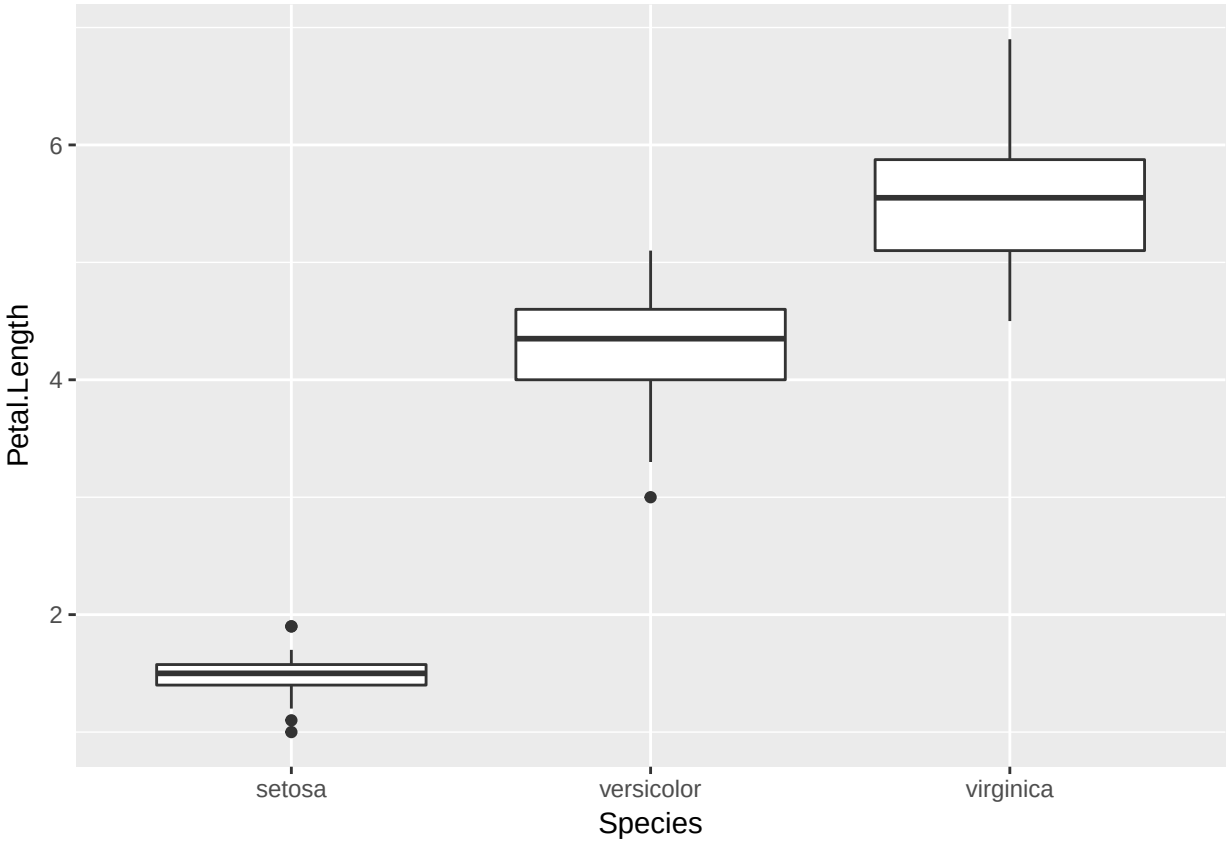
j) Gráfica de dispersión donde $x = Sepal.Length$ y $y = Sepal.Width$; donde el color representa la especie.

```
ggplot(data = iris) +
  geom_point(aes(x = Sepal.Length, y = Sepal.Width, color = Species))
```



k) Gráfica de cajas de *Petal.Length* donde se distingan las distintas especies.

```
ggplot(iris, aes(y = Petal.Length, x = Species)) +
  geom_boxplot()
```



2. Espacio de probabilidad y Variables Aleatorias

Considere un experimento que consiste en una carrera de caballos con tres caballos numerados del 1 al 3. Si no está permitido que dos o más caballos lleguen a la meta en la misma posición:

a. ¿Cuál es el espacio de resultados Ω del experimento?

$$\Omega = 123, 231, 312, 321, 213, 132$$

b. ¿Cuál es esta probabilidad $P(\omega)$?

$$P(\omega) = \frac{1}{\omega} = \frac{1}{6} = 1.\overline{6}$$

```
1/6
```

```
## [1] 0.1666667
```

c. ¿Cuáles son los elementos de los eventos A y B, respectivamente?

$$A = \{123, 132, 213\}$$
$$B = \{132, 231\}$$

d. ¿Cuáles son los elementos del evento $A \cap B$?

$$A \cap B = 132$$

e. ¿Cuáles son los elementos del evento $A \cup B$?

$$A \cup B = \{123, 132, 313, 231\}$$

f. ¿Cuál es la probabilidad $P(B)$ de que ocurra B? *33%*

```
2*(1/6)
```

```
## [1] 0.3333333
```

g. Liste los valores $X(\omega)$ que toma la variable X para cada uno de los elementos $\omega \in \Omega$

$$X(\omega) = \{1, 2, 3\}$$

h. ¿Cuál es la probabilidad de $P(X = 1)$?

```
2*(1/6)
```

```
## [1] 0.3333333
```

3. Probabilidad condicional

a. ¿Son X y Y variables independientes? *No son independientes, dado que se afectan mutuamente y a sus probabilidades.*

b. Calcule las probabilidad condicionales $P(Y|X = x)$ para $x = 1, 2, 3, 4$

```
print('P(X=1) para Y={0,1,2}')
```

```
## [1] "P(X=1) para Y={0,1,2}"
```

```
0.06/0.2
```

```
## [1] 0.3
```

```
0.12/0.2
```

```
## [1] 0.6
```

```
0.02/0.2
```

```
## [1] 0.1
```

```
print('P(x=2) para Y={0,1,2}')
```

```
## [1] "P(x=2) para Y={0,1,2}"
```

```
0.03/0.42
```

```
## [1] 0.07142857
```

```
0.18/0.42
```

```
## [1] 0.4285714
```

```
0.21/0.42
```

```
## [1] 0.5
```

```
print('P(x=3) para Y={0,1,2,3}')
```

```
## [1] "P(x=3) para Y={0,1,2,3}"
```

```
0.01/0.28
```

```
## [1] 0.03571429
```

```
0.09/0.28
```

```
## [1] 0.3214286
```

```
0.11/0.28
```

```
## [1] 0.3928571
```

```
0.07/0.28
```

```
## [1] 0.25
```

```
print('P(x=4) para Y={1,2,3,4}')
```

```
## [1] "P(x=4) para Y={1,2,3,4}"
```

```
0.02/0.1
```

```
## [1] 0.2
```

```
0.05/0.1
```

```
## [1] 0.5
```

```
0.02/0.1
```

```
## [1] 0.2
```

```
0.01/0.1
```

```
## [1] 0.1
```

c. Verifique que $P(Y|X)$ satisface la segunda regla de probabilidad $\sum_{y=o}^{4} P(Y = y|X = x) = 1$ para $x = 1, 2, 3, 4$

```
print('P(X=1) para Y={0,1,2}')
```

```
## [1] "P(X=1) para Y={0,1,2}"
```

```
0.06/0.2 + 0.12/0.2 + 0.02/0.2
```

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

```
print('P(x=2) para Y={0,1,2}')
```

```
## [1] "P(x=2) para Y={0,1,2}"
```

```
0.03/0.42 + 0.18/0.42 + 0.21/0.42
```

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

```
print('P(x=3) para Y={0,1,2,3}')
```

```
## [1] "P(x=3) para Y={0,1,2,3}"
```

```
0.01/0.28 + 0.09/0.28 + 0.11/0.28 + 0.07/0.28
```

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

```
print('P(x=4) para Y={1,2,3,4}')
```

```
## [1] "P(x=4) para Y={1,2,3,4}"
```

```
0.02/0.1 + 0.05/0.1 + 0.02/0.1 + 0.01/0.1
```

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

d. Calcule los valores esperados condicionales $E[Y|X = x]$ para $x = 1, 2, 3, 4$.

```
print('Para X=1, Y={1,2,3,4}')
```

```
## [1] "Para X=1, Y={1,2,3,4}"
```

```
E_X1 <- sum(c(1*.12, 2*.18, 3*.09, 4*.02)); E_X1
```

```
## [1] 0.83
```

```
print('Para X=2, Y={1,2,3,4}')
```

```
## [1] "Para X=2, Y={1,2,3,4}"
```

```
E_X2 <- sum(c(1*.02, 2*.21, 3*.09, 4*.05)); E_X2
```

```
## [1] 0.91
```

```
print('Para X=3, Y={1,2,3,4}')
```

```
## [1] "Para X=3, Y={1,2,3,4}"
```

```
E_X3 <- sum(c(0, 0, 3*.07, 4*.02)); E_X3
```

```
## [1] 0.29
```

```
print('Para X=4, Y={1,2,3,4}')
```

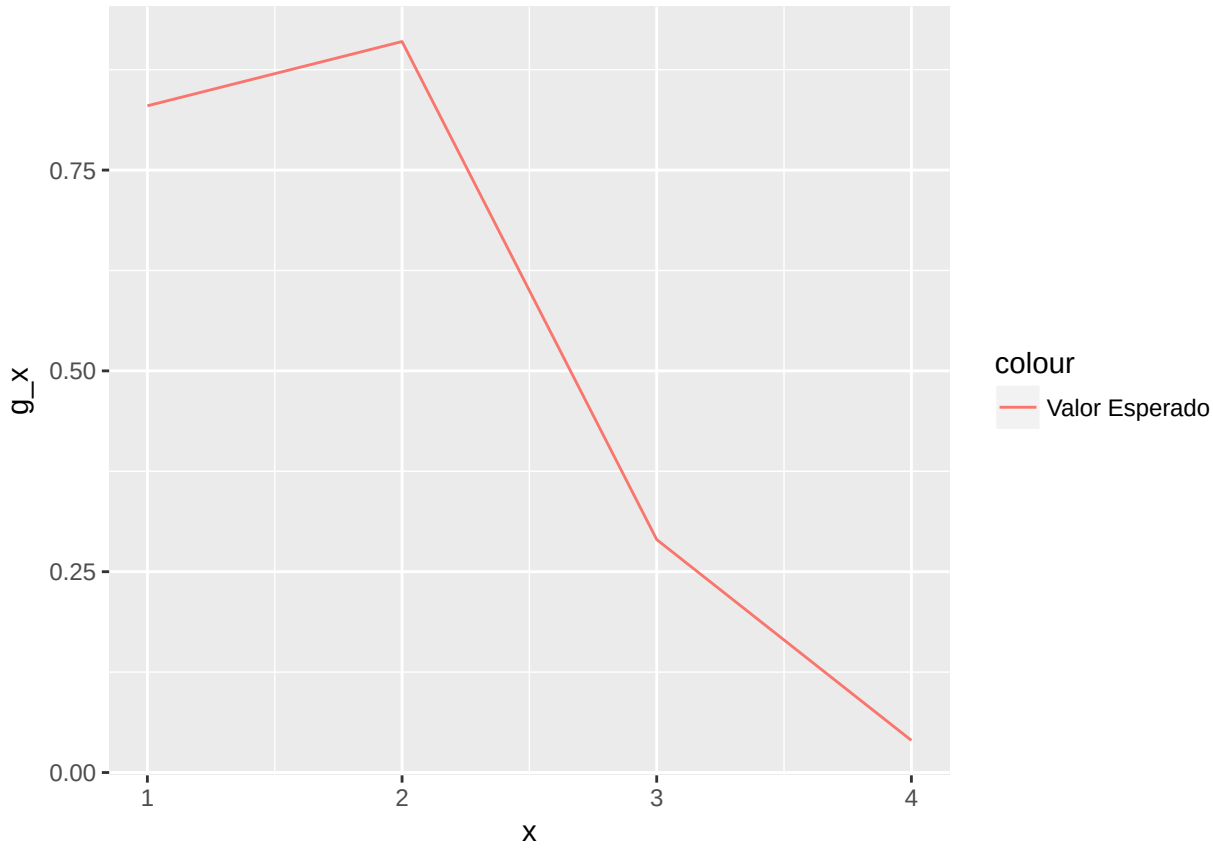
```
## [1] "Para X=4, Y={1,2,3,4}"
```

```
E_X4 <- sum(c(0,0,0,4*.01)); E_X4
```

```
## [1] 0.04
```

e. Grafique $g(x) = E[Y|X = x]$ para $x = 1, 2, 3, 4$.

```
df <- data.frame(x = c(1,2,3,4), g_x = c(E_X1, E_X2, E_X3, E_X4))
ggplot(df, aes(x=x,y=g_x,col="Valor Esperado")) + geom_line()
```



4. Bootstrap

I.

```
set.seed(261285)
bootstr <- rexp(n=20,rate = 1)
bootstr
```

```
##  [1] 1.26562612 2.00905427 1.53277040 3.16665193 2.91957971 1.39760569
##  [7] 0.69141493 1.28580721 0.02734916 0.23568531 2.94539575 0.15912479
## [13] 0.47047100 1.99842617 4.75564875 0.27125596 0.21680047 0.23216209
## [19] 0.40212096 2.07900431
```

```
paste("Std Dev", sd(bootstr))
```

```
## [1] "Std Dev 1.28752666879493"
```

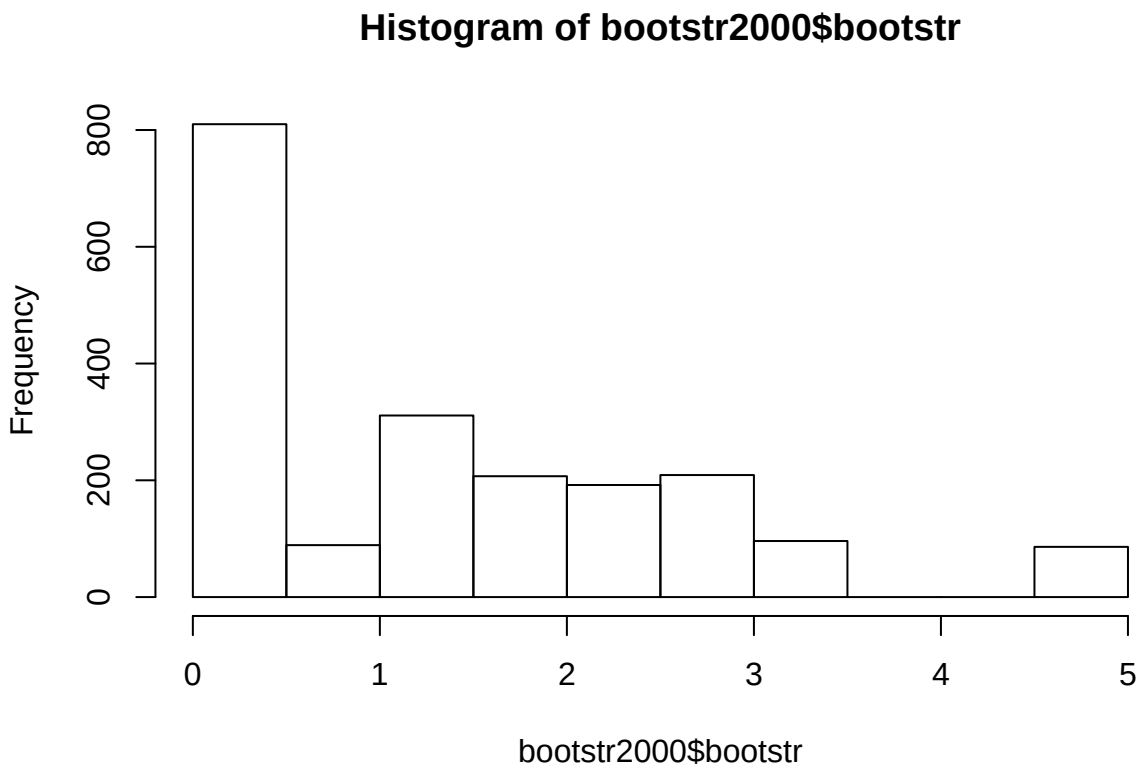
```
G <- cov(data.frame(bootstr)*20/19)
eigen_G <- eigen(G)
theta_hat <- eigen_G$values[1] / sum(eigen_G$values)
paste("El valor de theta_hat sí es 1: ", theta_hat)
```

```
## [1] "El valor de theta_hat sí es 1:  1"
```

II.

```
set.seed(261285)
bootstr2000 <- sample_n(data.frame(bootstr), size = 2000, replace = T)
```

```r
hist(bootstr2000$bootstr)
```




```r
sd(bootstr2000$bootstr)
```

```
## [1] 1.226482
```

III.

```r
pc_boot <- function(){
    muestra_boot <- sample_n(data.frame(bootstr), size = 2000, replace = T)
    G <- cov(muestra_boot) * 2000/1999
    eigen_G <- eigen(G)
    theta_hat <- eigen_G$values[1] / sum(eigen_G$values)
}
thetas_boot <- rerun(5000, pc_boot()) %>% flatten_dbl()
theta_se <- sd(thetas_boot)
theta_se
```

```
## [1] 0
```

```r
theta_hat - theta_se
```

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

a. La siguiente tabla al repetirla 500 veces

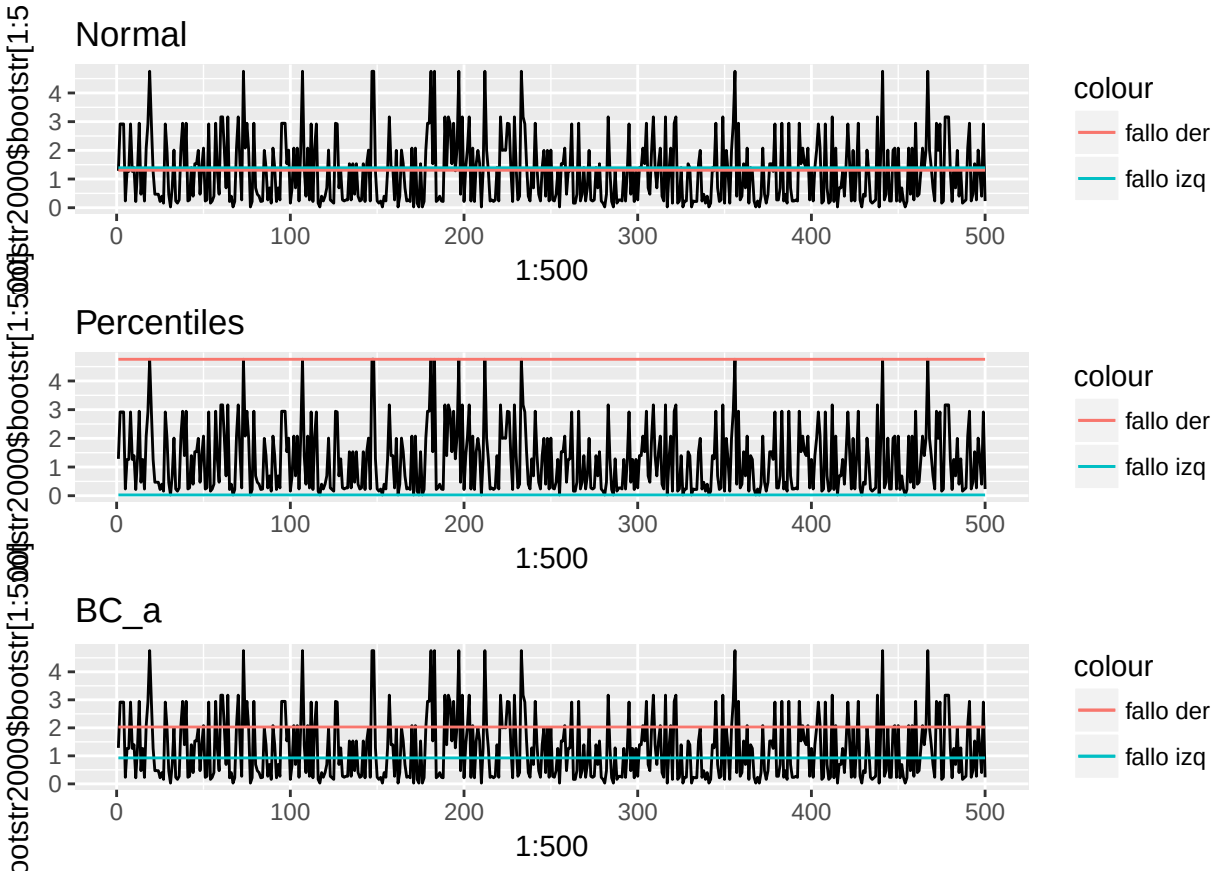
```r
metodo <- c("Normal", "Percentiles", "BC_a")
bca <- bcanon(x = bootstr, nboot = 2000, theta = mean)
fallo_izq <- c(-(1-1.95*sd(bootstr2000$bootstr)),
```

```
              as.double(quantile(bootstr2000$bootstr,prob=0.025)),
              bca$confpoints[1,2])
fallo_der <- c((1+.25*sd(bootstr2000$bootstr)),
              as.double(quantile(bootstr2000$bootstr,prob=0.975)),
              bca$confpoints[8,2])
cobertura <- c(100-fallo_izq[1]-fallo_der[1],
              100-fallo_izq[2]-fallo_der[2],
              100-fallo_izq[3]-fallo_der[3])
bootstrap <- data.frame(metodo, fallo_izq, fallo_der, cobertura)
bootstrap
```

```
##        metodo  fallo_izq fallo_der cobertura
## 1      Normal 1.39163925  1.306620  97.30174
## 2 Percentiles 0.02734916  4.755649  95.21700
## 3        BC_a 0.92661393  2.029203  97.04418
```

b. Realización de una gráfica de páneles mostrando los métodos normal, percentil y BC_a, en el eje horizontal graficando el número de intervalos de confianza $(1, 2, ..., 500)$ y en el eje vertical graficando los límites de los intervalos, utilizando la función `geom_line` para las 2 líneas, correspondientes a los límites de los intervalos inferiores y superiores.

```
plot_normal <- ggplot() +
  geom_line(aes(x=1:500,y=bootstr2000$bootstr[1:500])) +
  geom_line(aes(x=1:500,y=fallo_izq[1], col = "fallo izq")) +
  geom_line(aes(x=1:500,y=fallo_der[1], col = "fallo der")) +
  ggtitle("Normal")
plot_percentiles <- ggplot() +
  geom_line(aes(x=1:500,y=bootstr2000$bootstr[1:500])) +
  geom_line(aes(x=1:500,y=fallo_izq[2], col = "fallo izq")) +
  geom_line(aes(x=1:500,y=fallo_der[2], col = "fallo der")) +
  ggtitle("Percentiles")
plot_bca <- ggplot() +
  geom_line(aes(x=1:500,y=bootstr2000$bootstr[1:500])) +
  geom_line(aes(x=1:500,y=fallo_izq[3], col = "fallo izq")) +
  geom_line(aes(x=1:500,y=fallo_der[3], col = "fallo der")) +
  ggtitle("BC_a")
gridExtra::grid.arrange(plot_normal, plot_percentiles, plot_bca)
```



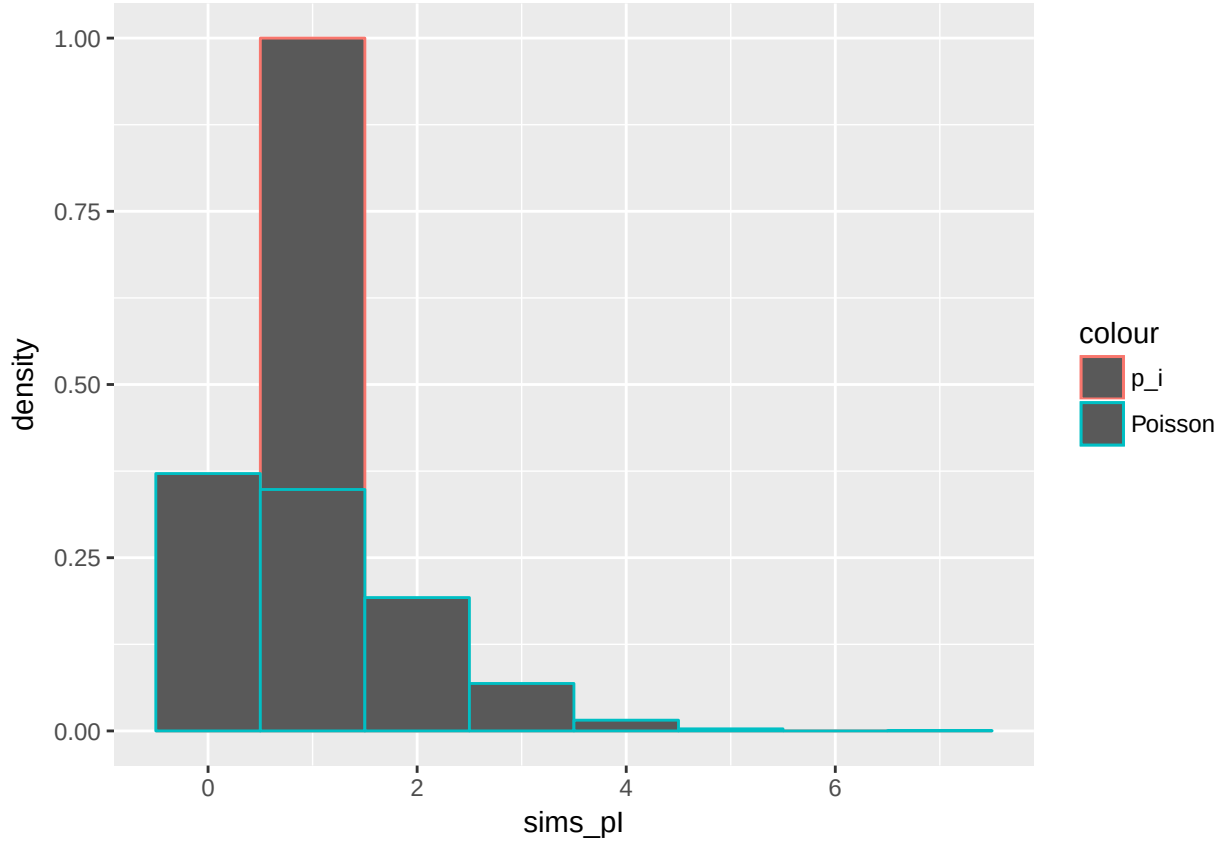
5. Simulación de Variables Aleatorias

```
rpoisI <- function(lambda = 1){
  U <- runif(1)
  i <- 0
  p <- exp(-lambda)
  P <- p
  while(U >= P){
    p <- lambda * p / (i + 1)
    P <- P + p
    i <- i + 1
  }
  i
}
rI <- function(n = 1){
  U <- runif(1)
  i <- 0
  p <- 0
  # p <- exp(-lambda)
  P <- p
  while(U >= P){
    p <- (choose(n,i)*p^i)*(1-p)^(n-i)
    P <- P + p
    i <- i + 1
```

```
  }
  i
}
sims_pois <- rerun(2000, rpoisI()) %>% flatten_dbl()
```

a. Encuentre la relación de recurrencia entre p_{i+1} y p_i para $i \geq 0$

```
sims_pI <- rerun(2000, rI()) %>% flatten_dbl()
ggplot() +
  geom_histogram(aes(x = sims_pI, y = ..density.., col = "p_i"), binwidth = 1) +
  geom_histogram(aes(x = sims_pois, y = ..density.., col="Poisson"), binwidth = 1)
```



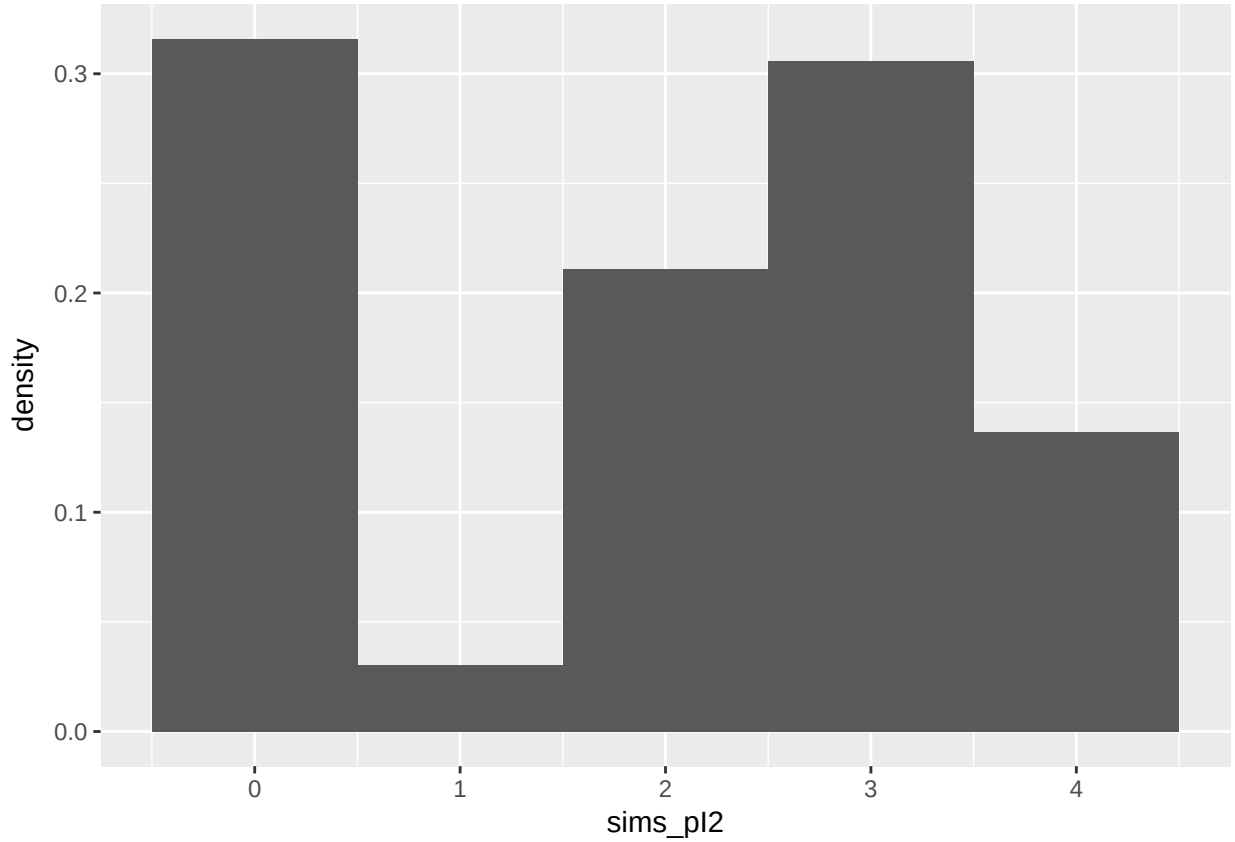
b. Utilizando la relación de recurrencia del inciso a., escriba un algoritmo de 5 pasos que genere una variable aleatoria binomial con parámetros n y p mediante el Método de Transformación Inversa Discreta.

1. Generar un número aleatorio U, tal que $U \in (0, 1)$
2. Inicializar: $i = 0, p = P$
3. Si $U \geq P$
4. $p = (p^i choose(n, i))(1 - p)^{(n-i)}$
5. Volver a 3.

c. Escriba en R una función que implemente el algoritmo del inciso b para $n = 10$ y $p = 0.3$

```
rI2 <- function(n = 1, p){
  U <- runif(1)
  i <- 0
  # p <- exp(-lambda)
  P <- p
```

```
  while(U >= P){
    p <- (choose(n,i)*p^i)*(1-p)^(n-i)
    P <- P + p
    i <- i + 1
  }
  i
}
sims_pI2 <- rerun(2000, rI2(n = 10, p = 0.3)) %>% flatten_dbl()
ggplot() +
  geom_histogram(aes(x = sims_pI2, y = ..density..), binwidth = 1)
```



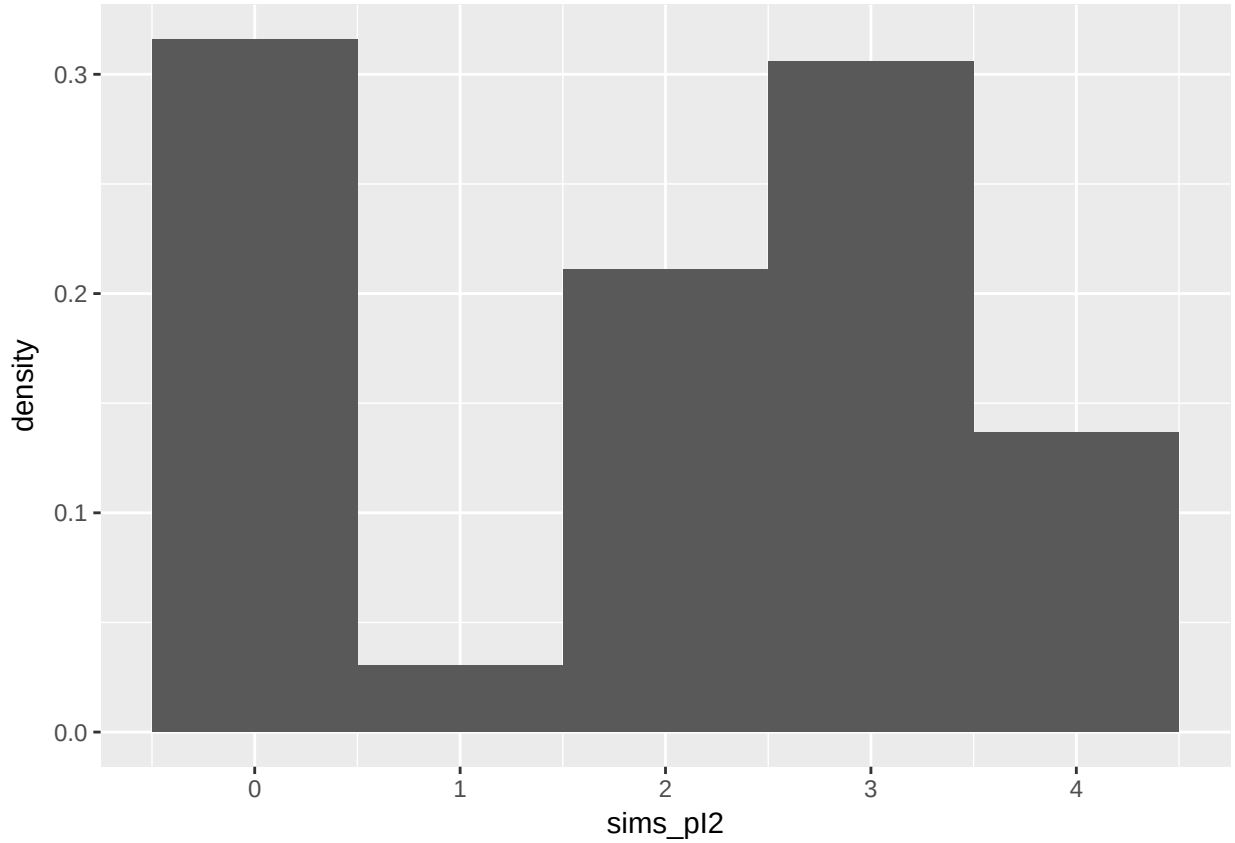
d. Realice 10,000 simulaciones utilizando la semilla 221285 y reporte las primeras simulaciones obtenidas.

```
set.seed(221285)
sims_pI3 <- rerun(10000, rI2(n = 10, p = 0.3)) %>% flatten_dbl()
head(sims_pI3)
```

```
## [1] 4 0 3 0 3 3
```

e. Genere un histograma con las 10,000 simulaciones anteriores y compárelo con una distribución construida utilizando la función `dbinom` de R.

```
ggplot() +
  geom_histogram(aes(x = sims_pI2, y = ..density..), binwidth = 1)
```



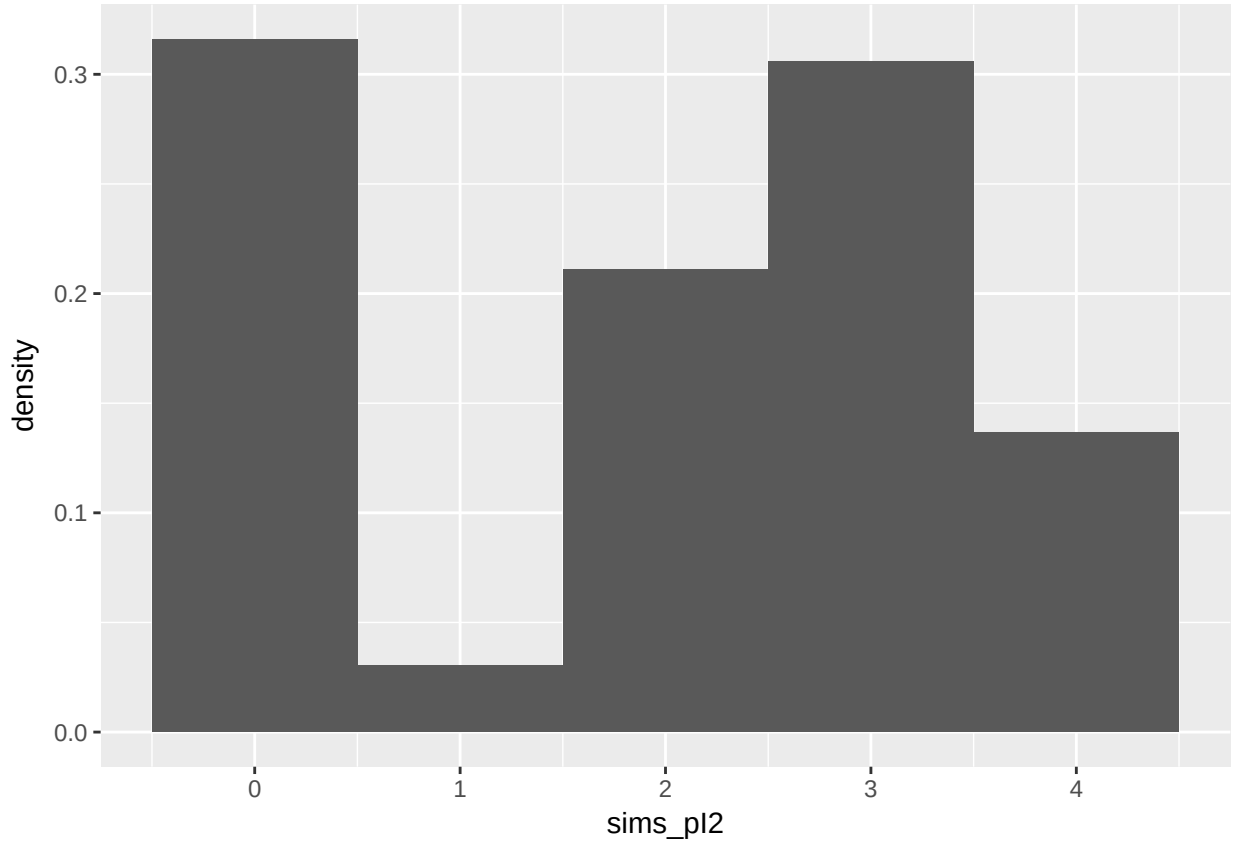
d. Realice 10,000 simulaciones utilizando la semilla 221285 y reporte las primeras simulaciones obtenidas.

```
set.seed(221285)
sims_pI3 <- rerun(10000, rI2(n = 10, p = 0.3)) %>% flatten_dbl()
head(sims_pI3)
```

```
## [1] 4 0 3 0 3 3
```

e. Genere un histograma con las 10,000 simulaciones anteriores y compárelo con una distribución construida utilizando la función `dbinom` de R.

```
ggplot() +
  geom_histogram(aes(x = sims_pI2, y = ..density..), binwidth = 1)
```



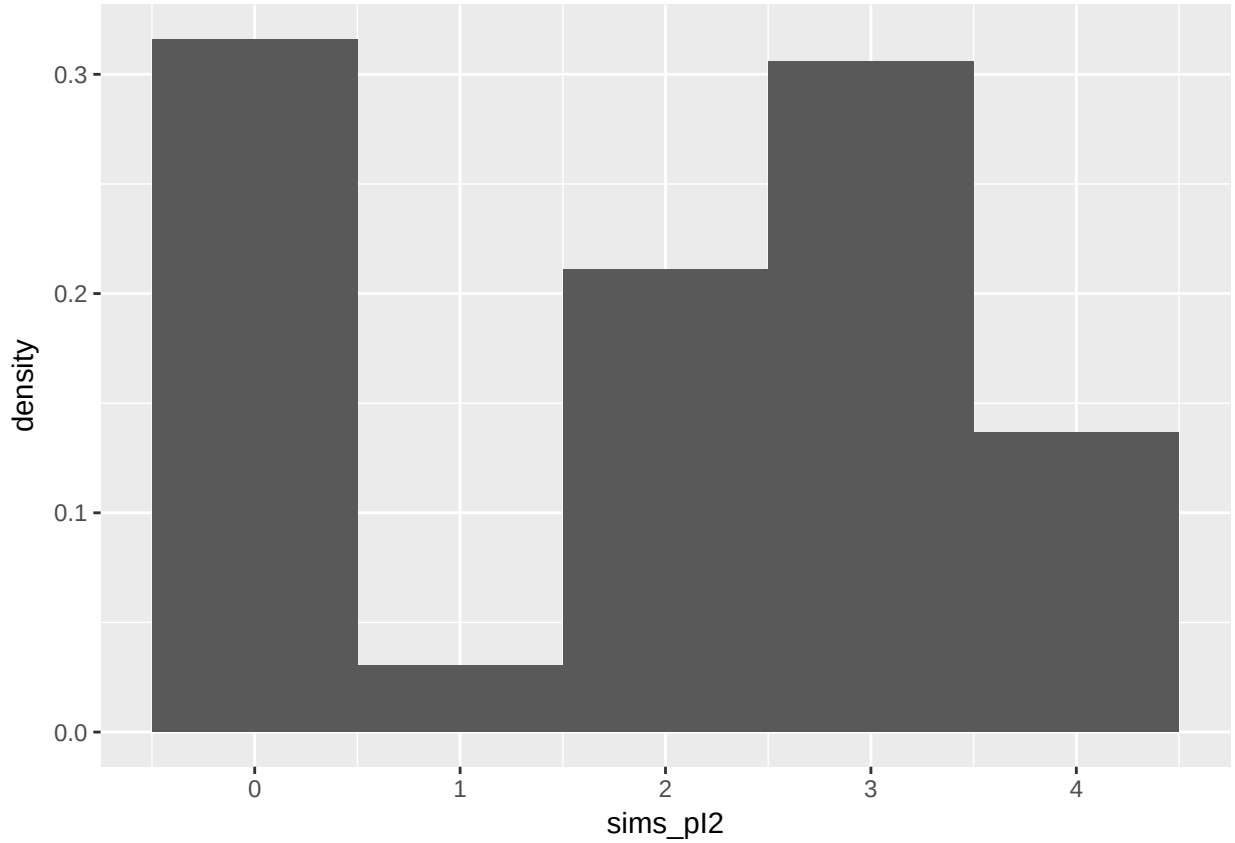
d. Realice 10,000 simulaciones utilizando la semilla 221285 y reporte las primeras simulaciones obtenidas.

```
set.seed(221285)
sims_pI3 <- rerun(10000, rI2(n = 10, p = 0.3)) %>% flatten_dbl()
head(sims_pI3)
```

```
## [1] 4 0 3 0 3 3
```

e. Genere un histograma con las 10,000 simulaciones anteriores y compárelo con una distribución construida utilizando la función `dbinom` de R.

```
ggplot() +
  geom_histogram(aes(x = sims_pI2, y = ..density..), binwidth = 1)
```



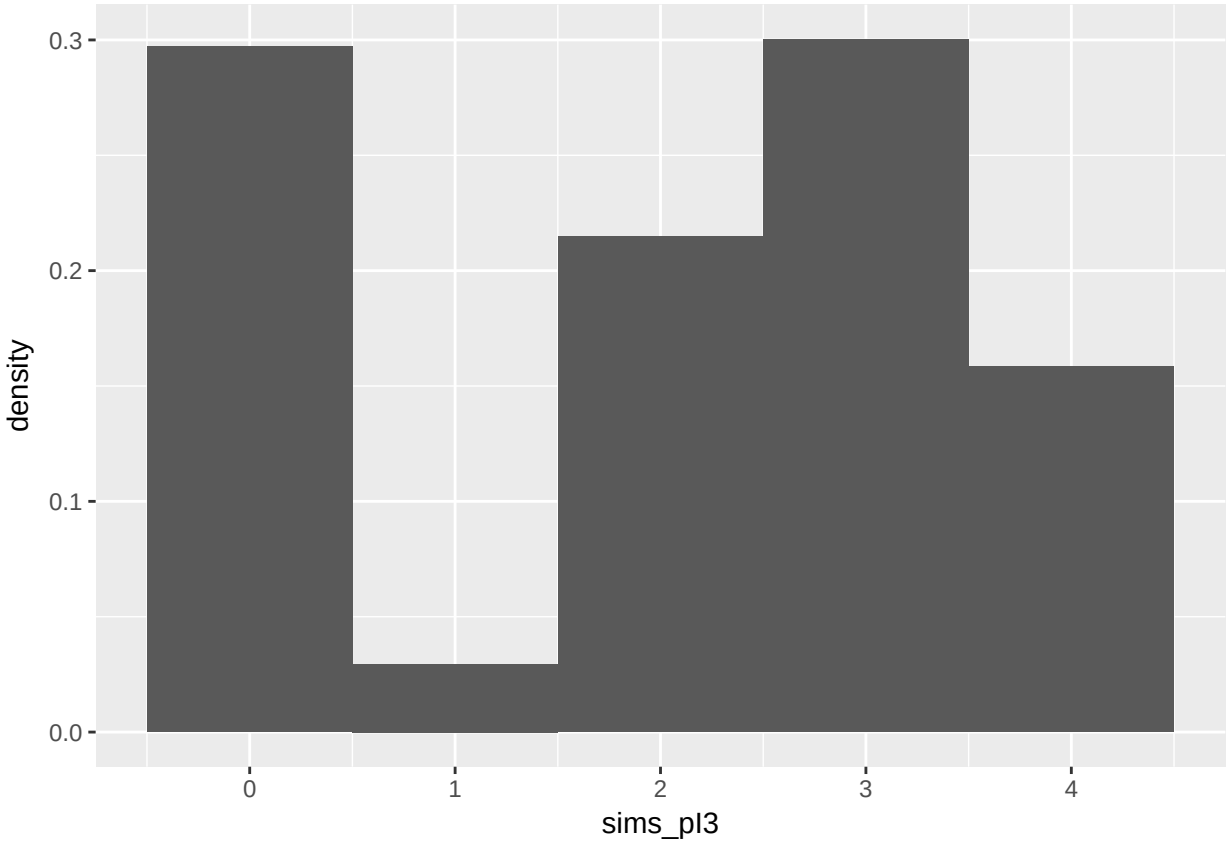
d. Realice 10,000 simulaciones utilizando la semilla 221285 y reporte las primeras simulaciones obtenidas.

```
set.seed(221285)
sims_pI3 <- rerun(10000, rI2(n = 10, p = 0.3)) %>% flatten_dbl()
head(sims_pI3)
```

```
## [1] 4 0 3 0 3 3
```

e. Genere un histograma con las 10,000 simulaciones anteriores y compárelo con una distribución construida utilizando la función `dbinom` de R.

```
ggplot() +
  geom_histogram(aes(x = sims_pI3, y = ..density..), binwidth = 1)
```


0.3
0.2
0.1
0.0
density
0
1
2
3
4
sims_pl3