Proyecto en R de probabilidad

Notas antes de empezar:

Las funciones de distribución en R se dividen en 4 tipos.

- ddistribución(parámetros): entrega la función de densidad calculada con los parámetros dados.
- pdistribución(parámetros): entrega la función de densidad acumulada con los parámetros dados.
- qdistribución(parámetros): entrega el valor de x para el que se acumula cierta probabilidad (cuantiles)
- rdistribución(parámetros): entrega N números que se distribuyen según lo elegido.
- par(mfrow=c(x, y)): me permite hacer dos gráficas en diferentes plots, útil para compararlas.
- x <- función_cualquiera(): para indicar que la variable es un arreglo utilizo <- en vez de =.

Problema 1.

Empecé declarando algunas variables para hacer mejor la legibilidad del código.

```
azules = 5
moradas = 3
rojas = 4
```

Inciso a)

Como pedía la probabilidad de cierto evento, podemos asumir que es el número de casos exitosos entre el número total de casos, dándonos el siguiente código.

```
p_morada = moradas/(azules + moradas + rojas)
print(p_morada)
```

Inciso b)

Esencialmente pedía la probabilidad complementaria del inciso a), entonces procedí a calcularla de la siguiente manera.

```
print(1 - p_morada)
```

Inciso c)

Pedía la probabilidad de que la primera canica sacada fuera de color morado, mientras que la segunda no lo fuera, esto sin reemplazo, por lo que podemos plantear que pedía $P(X_1 = Morada \cap X_2 \neq Morada)$. Podemos asumir que los dos eventos son independientes, ya que una tirada no depende de la otra, lo único que hace es disminuir el número de cánicas que hay en la caja.

```
p2_no_morada = (azules + rojas)/(azules + moradas + rojas - 1)
print(p_morada*p2_no_morada)
```

Problema 2.

Pedía calcular las probabilidades de una moneda justa, como lo vimos en clase, este tipo de problemas podemos resolverlos con una distribución binomial, por lo que en vez de hacerme la vida difícil, usé los comandos de R para distribuciones binomiales.

Inciso a)

Pedía la probabilidad de que el obtener cuatro aguilas en 6 tiradas, esto lo podemos hacer con el comando `dbinom(x, N, p)`, donde los parámetros significan:

- x: número deseado de éxitos.
- N: número total de eventos.
- p: probabilidad del éxito.

```
print(dbinom(4, 6, 0.5))
```

Inciso b)

Pedía algo similar al inciso a), solo que ahora pedía la acumulada en 4, para calcular esto, lo único que tenemos que cambiar es la función, en vez de `dbinom`, tenemos que utilizar `pbinom`, las dos funciones contienen los mismos parámetros.

```
print(pbinom(4, 6, 0.5))
```

Problema 3

Pide calcular una distribución normal con media 50 y varianza 25, por lo que lo primero que hice fue calcular el valor de la desviación estándar, que será un parámetro para las funciones que utilizaré más

adelante, esto lo hice con: `sd_norm = sqrt(abs(25))` *Tuve que meter el absoluto por que si no a R se le botaba la canica*.

Inciso a)

Pedía la acumulada en 48, esto lo procedí a calcular de la siguiente manera:

```
normal = pnorm(48, 50, sd_norm)
print(normal)
```

Inciso b)

Pedía el complemento de la acumulada en 48, aprovechando que ya tenía una variable con el valor de la acumulada en 48 guardado hice:

```
print(1 - normal)
```

Inciso c)

Pedía la acumulada entre 45 y 55, pedía que fuera mayor o igual a 45, por lo que tenemos que partir desde el 44 para poder incluir al 45 en los cálculos, además pedía que fuera menor que 55, pero como es una distribución continua podemos incluir al 55, procedí a calcular los dos valores y a restarlos:

```
normal_45 = pnorm(44, 50, sd_norm)
normal_55 = pnorm(55, 50, sd_norm)
print(normal_55 - normal_45)
```

Inciso d)

Pedía el valor en el que se acumula el 85% de los valores de la distribución, por lo que pedía el cuantil 85, esto se calcula de la siguiente manera:

```
quant_85 = qnorm(0.85, 50, sd_norm)
print(quant_85)
```

Inciso e)

Pedía generar 30 números aleatorios que siguieran la distribución normal con los parámetros ya mencionados. Esto se hace con la función:

```
norm_30 <- rnorm(30, 50, sd_norm)
print(norm_30)
```

Inciso f)

Pedía graficar la distribución normal con los valores ya vistos, después de eso, cambiar los parámetros e indicar como cambia la gráfica. Esto se hace de la siguiente manera:

```
x <- seq(0, 100) # Creamos un vector de 0 a 100, para tener números para calcular una gráfica de 100 puntos
y <- dnorm(x, 50, sd_norm) # Calculamos los valores de la normal como otro vector, para poder graficarla
y_20 <- dnorm(x, 20, sqrt(20)) # Calculamos un segundo vector con diferente media y desviación estándar
par(mfrow=c(1,2)) # Indicamos que van a estar en gráficas diferentes
plot(x, y) # Graficamos los valores obtenidos
plot(x, y_20) # Graficamos los valores comparados
```

Para obtener una gráfica diferente lo que hacemos es cambiar los parámetros de `y`, más específicamente, la media y la varianza.

Problema 4

Similar al problema 3, pero ahora con una distribución exponencial con media λ igual a 6. Calculé algunas variables de por medio para utilizarlas más fácilmente.

```
lamba = 1/6
lambda_10 = 1/10
```

Inciso a)

Pide calcular la acumulada en 3, ya que indica que debe ser estrictamente menor a 4, por lo que utilizamos el comando `pexp(x, lambda)`, con x el valor acumulado que queremos, y λ es el inverso de la media:

```
p6_4 = pexp(3, lambda)
print(p6_4)
```

Inciso b)

Pide calcular algo muy similar al inciso a), pero ahora es el complemento de la acumulada en 4, por lo que hacemos:

```
print(1 - pexp(4, lambda)
```

Inciso c)

Pide calcular la acumulada entre 3 y 6, siendo esto estrictamente mayor que 3 y menor que 6, por lo que aprovechando que ya tengo algo similar en el inciso a), puedo hacer:

```
print(pexp(5, lambda) - p6_4)
```

Inciso d)

Pide calcular el cuantil 75 de la función con los parámetros antes dados, simplemente hacemos, similar a la función `pexp`, pero ahora el primer parámetro indica el cuantil que queremos:

```
print(qexp(0.75, lambda))
```

Inciso e)

Similar al anterior problema, nos pide calcular treinta números que sigan la distribución exponencial con media igual a 6:

```
print(rexp(30, lambda))
```

Inciso f)

También como en el problema pasado, queremos graficar la distribución actual, además de compararla contra otra con el mismo tipo de distribución pero diferentes parámetros.

```
x <- seq(0, 40)
y <- dexp(x, lambda)
y_
```

Problema 5.

Similar a los dos problemas anteriores, ahora tenemos un problema de una distribución binomial com parametros N igual a 15 y p igual a 0.3. Primero que nada, algunas variables que utilizaremos:

```
acum = pbinom(5, 15, 0.3)
nums_bin <- rbinom(30, 15, 0.3)
```

Inciso a)

Nos pide la acumulada en 5, esto ya lo hemos hecho en la parte anterior donde definí algunas variables, la función `pbinom` usa parámetros:

- x: el valor que queremos acumular
- N: el número total de eventos
- p: la probabilidad de éxitos Van ingresados en el orden dado.

```
print(acum)
```

Inciso b)

Es el complemento del inciso a), por lo que podemos hacer lo siguiente:

```
print(1 - acum)
```

Inciso c)

Nos pide la acumulada entre 2 y 6, ya que nos indica que es estrictamente menor que 7, e inclusivo en 2, por lo que hacemos lo siguiente con la función de la binomial acumulada.

```
print(pbinom(6, 15, 0.3) - pbinom(1, 15, 0.3))
```

Inciso d)

Similar a los anteriores problemas, queremos generar 30 números con distribución binomial, esto ya lo hicimos cuando declaramos las variables originalmente con la función `rbinom(num, N, p)` donde:

- num: corresponde a la cantidad de números generados.
- N: a la máxima cantidad de eventos.
- p: a la probabilidad de éxito.

```
print(nums_bin)
```

Inciso e)

Otra vez volvemos a graficar la distribución normal que ya nos habían dado, volvemos a tener que compararla con otra distribución normal pero de parámetros diferentes, por lo que hacemos lo mismo que en los anteriores problemas, solo cambiando las funciones:

```
x <- seq(0, 15)
x_30 <- seq(0, 30)
y <- dbinom(x, 15, 0.3)
y_30 <- dbinom(x_30, 30, 0.7)
par(mfrow=c(1, 2))
plot(x, y)
plot(x_30, y_30)
```

Problema 6

Nos pedía simular una distribución uniforme continua con valores entre 3.7 y 5.8, además de calcular algunos datos como la media, varianza y desviación estándar. Hice esto de dos maneras:

1. Aprovechando las fórmula que teniamos en las tablas de distribuciones pude calcular las propiedades requeridas.
2. Usando métodos iterativos (ciclos for), hice sumas de los valores para luego calcular mediante las definiciones de valor esperado, varianza y desviación estándar, los valores necesarios.

Conclusiones: los valores fueron exactamente los esperados, hubo una mínima diferencia entre los dos métodos, que podemos adjudicar a redondeos hechos en las fórmulas, la cantidad de decimales que utiliza R, etcétera, el código es el siguiente:

```
x <- runif(1000, min = 3.7, max = 5.8)
print(x)
```

Inciso a)

```
# Método iterativo
mean_sum = 0
for(number in x){
  mean_sum = mean_sum+number
}
mean = mean_sum/length(x)
print(mean)

# Método por propiedades de la distribución
print((5.8+3.7)/2)
```

```
# Método iterativo
var_sum = 0
for(number in x){
  var_sum = var_sum+number^2
}
var_sum = var_sum/length(x)
var = var_sum - mean^2
print(var)

# Método por propiedades de la distribución
var_unif = (5.8-3.7)*2/12
print(var_unif)

# Resultados de la desviación estándar de los dos métodos
print(sqrt(var))
print(sqrt(var_unif))
```

Inciso b)

El inciso b) simplemente pedía el complemento de la acumulada en 3, por que pedía que fuera mayor o igual que 4, esto lo podemos hacer con la función `punif(x, mean, sd)` con:

- x: valor que queremos acumular.
- mean: media la distribución.
- sd: desviación estándar de la distribución.

```
print(1 - punif(4, 3.7, 5.8))
```

Problema 7

Nos dan una distribución Poisson con parámetro λ igual a 3.7, por lo que podemos guardar esto en una variable `lambda = 3.7` para tenerla lista para más tarde.

Inciso a)

El primer inciso lo único que nos pide es $P(X = 6)$, por lo que podemos usar la función `dpois(x, lambda)` con:

- x: valor de x que queremos obtener en la función de probabilidad.
- lambda: parámetro lambda de la distribución Poisson

```
print(dpois(6, lambda))
```

Inciso b)

Nos pide la probabilidad que haya menos de dos accidentes al año, por lo que podemos utilizar la acumulada en uno para calcular esto. Esto es la función `ppois(x, lambda)` con:

- x: el valor que queremos acumular
- lambda: el parámetro lambda de la distribución Poisson.

```
print(ppois(1, lambda))
```

Inciso c)

Nos pide la probabilidad de que haya estrictamente más de 8 accidentes en un año, por lo que podemos utilizar el complemento de la acumulada en 8 para calcular este valor:

```
print(1 - ppois(8, lambda))
```

Inciso d)

Nos pide el número máximo de accidentes en el cuantil 90, por lo que podemos utilizar la función `qpois(p, lambda)` para calcularlo, con:

- p: cuantil que queremos obtener.
- lambda: parámetro lambda de la distribución Poisson.

```
print(qpois(0.9, lambda))
```

Inciso e)

Nos pide simular el número de accidentes en 20 años, podemos utilizar la función `rpois(n, lambda)` para generar 20 números con distribución Poisson con parámetro 3.7, la función tiene parámetros:

- n: cantidad de números a simular.
- lambda: parámetro lambda de la distribución Poisson.

```
print(rpois(20, lambda))
```

Problema 8

Nos pide simular las similitudes entre las diferentes distribuciones, esto es:

- Binomial-Poisson.
- Normal-Poisson.
- Binomial-Normal.

Con las fórmulas ya comentadas en los problemas anteriores, además de la función `lines(x, y)` que permite graficar dos cosas en una misma gráfica, hice tres gráficas donde se puede observar como se relacionan las distribuciones cuando los parámetros manejan una cierta relación:

```
# Binomial-Poisson.
p_poi = 0.01
n_poi = 500
x_bin <- seq(0, 25)
y_bin <- dbinom(x_bin, n_poi, p_poi)
lambda_bin = n_poi*p_poi
y_poi_bin <- dpois(x_bin, lambda_bin)

# Normal-Poisson.
lambda_norm = 300
x_poi <- seq(lambda_norm - 100,lambda_norm + 100)
y_poi <- dpois(x_poi, lambda_norm)
sd_lambda_norm = sqrt(lambda_norm)
y_nor_poi <- dnorm(x_poi, lambda_norm, sd_lambda_norm)

# Binomial-Normal.
p_norm = 0.5
n_norm = 200
x_norm <- seq(0, n_norm)
y_norm <- dbinom(x_norm, n_norm, p_norm)
mean_norm = n_norm*p_norm
sd_norm = sqrt((n_norm*p_norm*(1 - p_norm)))
y_norm_bin <- dnorm(x_norm, mean_norm, sd_norm)
```

Inciso a)

Código de las gráficas de la relación Binomial-Poisson:

```
plot(x_bin, y_bin, col="red")
lines(x_bin, y_poi_bin, col="green")
```

Inciso b)

Código de las gráficas de la relación Normal-Poisson:

```
plot(x_poi, y_poi, col="blue")
lines(x_poi, y_nor_poi, col="green")
```

Inciso c)

Código de las gráficas de la relación Binomial-Normal:

```
plot(x_norm, y_norm, col="blue")
lines(x_norm, y_norm_bin, col="red")
```