Trabajo Juego Yatzy Probabilidad I

Abrahám Soto y Johanna Álvarez

2025-06-16

Primero determinamos las variables escenciales de nuestro estudio “caras y numero de dados” ademas del espacio muestral o total de posibles resultados

```
caras_dado <- 6
num_dados <- 5
```

Espacio Muestral para un solo lanzamiento:

```
espacio_muestral <- caras_dado^num_dados
espacio_muestral
```

```
## [1] 7776
```

Primero evaluaremos los casos favorables donde el usario obtiene puntos a favor y sus propiedades

Triple: Para el caso de Triple “(tres iguales y dos diferentes)” estamos hablando de permutaciones de un multiconjunto, o más simplemente, arreglos de objetos donde algunos son idénticos. Cuando lanzas 5 dados y quieres un “Triple” (por ejemplo: 4,4,4,1,6) lo que interesa es la disposición específica de los valores en los 5 dados. Cada dado es distinguible por su posición (Dado 1, Dado 2,..; Dado 5) Calculamos las diferentes maneras en que se puede obtener un Triple

Tenemos n = 5 (dados), n1 = 3 (dados con el mismo valor) , n2 = 1 (dado con un valor diferente) , n3 = 1 (dado con otro valor diferente)

1. Elegir el valor para el trio

Concepto de Combinación C(6, 1)

```
opciones_valor_triple_para_Triple <- choose(6, 1)
```

2. Elegir los posibles valores para el par diferente del numero escogido en el trio y entre ellos.

Concepto de Combinación C(5, 1)

```
opciones_valores_diferentes_Triple <- choose(5, 2)
```

3. Distribuir los 5 dados en las 5 posiciones

El orden seria (X,X,X,Y,Z) por ejemplo (1,1,1,2,3) Concepto de Coeficiente Multinomial (5! / (3! * 1! * 1!))

```
distribucion_posiciones_triple <- factorial(5) / (factorial(3) * factorial(1) * factorial(1))
```

4. Obtener los posibles valores de obtener un Triple

```
total_resultados_triple <- opciones_valor_triple_para_Triple* opciones_valores_diferentes_Triple * dist
```

Observamos el total de resultados de obtener un Triple

```
total_resultados_triple
```

```
## [1] 1200
```

Obteniendo así los posibles resultados de un triple como el producto de una combinación con el coeficiente multinomial

Explicacion: Primero se elige un valor de 6 lo cual es el concepto de combinacion, luego se seleccionan dos valores que deben ser diferentes entre sí y diferentes del primer valor, en esta seleccion el orden no importa para definir los valores "unicos" que aparecen en el lanzamiento de los dados. Para el arreglo final se debe evaluar los posibles resultados de ordenar 3 dados iguales y 2 diferentes entre sí, siendo esto un coeficiente binomial.

Casos para obtener Full

Un Full consiste en tres dados de un número y dos dados de un mismo número, los denotamos como (X,X,X,Y,Y) por ejemplo (1,1,1,2,2)

Para este caso contamos n = 5 dados, n1 = 3 dados iguales, n2 = dado de un numero diferente a n1 , n3 = dado igual a n2 Observamos que para este caso se trata de un problema de permutación con repetición o, más precisamente, de coeficiente multinomial. Es el número de formas de organizar 5 objetos, donde 3 son idénticos (el primer valor) y 2 son idénticos (el segundo valor).

1. Elegimos los posibles valores para el trio de Full

Concepto de combinación C(6, 1)

```
opciones_valor_triple_para_Full <- choose(6, 1)
```

2. Elegir los posibles valores para el par diferente del valor del trio

Concepto de Combinación C(5, 1)

```
opciones_valor_par_Full<- choose(5, 1)
```

3. Distribuir los valores en las 5 posiciones de los dados

Concepto de Coeficiente Multinomial (5! / (3! * 2!)) esto es equivalente a elegir 3 posiciones para el triplete de 5, C(5, 3)

```
distribucion_posiciones_full <- choose(5, 3)
```

4. Por ultimo hayamos el total de posibles resultados de obtener un Full (8 puntos)

```
total_resultados_full <- opciones_valor_triple_para_Full * opciones_valor_par_Full * distribucion_posici
```

Observamos el total de resultados de obtener un Full

```
total_resultados_full
```

```
## [1] 300
```

Obteniendo asi los posibles resultados de obtener un Full como el productos de una combinacion y el coeficiente multinomial

Explicacion: estas son combinaciones porque el orden en que elegimos los valores X y Y no importan para definir la "clase" de valores que formarán el Full. Una vez que tienes los valores específicos este coeficiente cuenta las formas de organizar los 5 números en los 5 dados distinguibles. Es una forma de "permutación con repetición" de un multiconjunto.

Casos de obtener una Escalera (Menor y Mayor)

Existen 2 casos de escalera: menor (1,2,3,4,5) y mayor (2,3,4,5,6) por lo cual podemos observar que se trata de una simple permutacion en ambos casos

1. Número de tipos de escaleras (menor y mayor)

Conteo directo

```
numero_tipos_escalera <- 2
```

2. Obtener los valores para cada tipo de escalera

Concepto de Permutación 5!

```
permutaciones_escalera <- factorial(5)
```

3. Obtener el total de resultados posibles de conseguir una escalera

```
total_resultados_escalera <- numero_tipos_escalera * permutaciones_escalera
```

Observamos el número total de resultados de obtener una Escalera

```
total_resultados_escalera
```

```
## [1] 240
```

Total de casos de no obtener ninguno de los 3 anteriores, es decir, de no conseguir puntos

Aplicamos propiedad de complemento para conseguir este resultado

```
total_no_obtener_puntos <- espacio_muestral - (total_resultados_escalera + total_resultados_full + total
```

Observamos el numero total de no obtener ningun punto

```
total_no_obtener_puntos
```

```
## [1] 6036
```

Vector de total de resultados obtenibles en un lanzamiento:

```
total_resultados_obtenibles <- c("Escaleras" = total_resultados_escalera, "Triple" = total_resultados_t
total_resultados_obtenibles
```

```
## Escaleras    Triple      Full   Ninguno
##       240      1200       300      6036
```

Apartado 1: Hallar la probabilidad de que una persona gane el juego

Para esto debemos conocer las probabilidades de cada suceso interpretados como puntos obtenidos, estos son obtener un triple, full o escalera, se dividen por el total de resultados posibles (Espacio muestral)

```
prob_full <- total_resultados_full / espacio_muestral
prob_triple <- total_resultados_triple / espacio_muestral
prob_escalera <- total_resultados_escalera / espacio_muestral
```

Observamos las probabilidades:

```
prob_escalera
```

```
## [1] 0.0308642
```

```
prob_triple
```

```
## [1] 0.154321
```

```
prob_full
```

```
## [1] 0.03858025
```

Estas son las probabilidades de obtener cada uno de los sucesos en un solo lanzamiento

Adicionalmente poidemos encontrar la probabilidad de no obtener ninguno de los anteriores aplicando propiedad de complemento

```r
prob_ninguno <- 1 - (prob_escalera + prob_full + prob_triple)
```

Ordenamos todas las probabilidades de cada suceso en un lanzamiento dentro de un vector:

```r
probabilidad_un_solo_lanzamiento <- c("Escalera" = prob_escalera, "Triple" = prob_triple, "Full" = prob

probabilidad_un_solo_lanzamiento
```

```
##   Escalera     Triple       Full    Ninguno
## 0.03086420 0.15432099 0.03858025 0.77623457
```

Ahora organizaremos los posibles eventos de ganar y sus probabilidades en vectores para facilitar los calculos

```r
puntos <- c("Escalera" = 10,"Triple" = 5,  "Full" = 8,"Ninguno" = 0)
puntos
```

```
## Escalera   Triple     Full  Ninguno
##       10        5        8        0
```

El usuario gana la partida si acumula 12 puntos o más, por lo que, ahora debemos determinar el resultado en base a un segundo lanzamiento, estos lanzamientos son indeoendientes entre sí

Creamos una matriz donde cada celda [i,j] representa la probabilidad de obtener la jugada i en la primera partida y la jugada j en la segunda partida:

```r
prob_combinada <- outer(probabilidad_un_solo_lanzamiento, probabilidad_un_solo_lanzamiento, FUN = "*")
colnames(prob_combinada) <- names(probabilidad_un_solo_lanzamiento)
rownames(prob_combinada) <- names(probabilidad_un_solo_lanzamiento)

prob_combinada
```

```
##              Escalera      Triple        Full    Ninguno
## Escalera 0.0009525987 0.004762993 0.001190748 0.02395786
## Triple   0.0047629934 0.023814967 0.005953742 0.11978929
## Full     0.0011907484 0.005953742 0.001488435 0.02994732
## Ninguno  0.0239578570 0.119789285 0.029947321 0.60254010
```

Ahora debemos determinar la cantidad de puntos sumados en los 2 lanzamientos, para esto creamos una matriz de combinacion de la suma de puntos de cada suceso:

```r
puntos_totales_combinados <- outer(puntos, puntos, FUN = "+")
colnames(puntos_totales_combinados) <- names(puntos)
rownames(puntos_totales_combinados) <- names(puntos)

puntos_totales_combinados
```

```
##          Escalera Triple Full Ninguno
## Escalera       20     15   18      10
## Triple         15     10   13       5
## Full           18     13   16       8
## Ninguno        10      5    8       0
```

Para que el usuario pueda ganar la partida se debe obtener un puntaje mayor o igual a 12, por lo que, usando las matrices creadas anteriormente y aplicando filtrado para sumar los puntajes, obtenemos los siguientes resultados:

```
Puntaje_Ganador <- puntos_totales_combinados >= 12
```

Por ultimo para hallar la probabilidad de ganar la partida sumamos sumamos las probabilidades que cumplen la condicion de ganar:

```
probabilidad_ganar <- sum(prob_combinada[Puntaje_Ganador])
```

La probabilidad de que una persona gane el juego es de 0.026256 obteniendo alguno de los resultados favorables que sumen 12 puntos o más en los dos lanzamientos.