



**FACULTAD DE
SISTEMAS**
EXPERIENCIA QUE DA RESULTADO



UNIVERSIDAD
AUTÓNOMA
DE COAHUILA

WeatherWiseMarkov: **Modelando transiciones climáticas con** **Markov.**

Equipo ClimatikGenius:

- ✚ Salvador Paniagua de la Fuente - 19278475
- ✚ Erick Misael Rodríguez Ramírez - 20302740
- ✚ Estrella Guadalupe López Armendáriz - 20305062
- ✚ Yair Alejandro Torres Siller - 18043044
- ✚ Darwin Dadier Solís Cedillo - 20310920

Maestra: Valeria Soto Mendoza.

ModelosComputacionales-828-6NSB-ValeriaSoto

15/11/2023

Descripción del proyecto:

Este proyecto trabajará usando cadenas de Markov para modelar y simular patrones climáticos basados en datos históricos recopilados durante varios meses; concretamente en Saltillo, Coahuila. El análisis incluye la transición entre 4 estados climáticos, el cálculo de probabilidades de cambio entre estos estados, la construcción del clima del día siguiente y la simulación del clima futuro para un número específico de días a partir de diferentes condiciones climáticas iniciales.

Metas:

- El código fuente debe crear conteos para los futuros cambios de estados, independiente cual sea.
- Utilizar los datos recopilados de los estados climáticos para comprender y modelar las transiciones entre diferentes estados (soleado, nublado, lluvioso, tormenta) en la ciudad de Saltillo.
- Que el código sea moldeable y adaptable a cualquier estado de clima; es decir, que no sea solo para un tipo concreto de datos.
- Asegurar que las frecuencias de transición se conviertan en probabilidades normalizadas para permitir una representación adecuada de la matriz de transición de Markov.
- Utilizar la matriz de transición de Markov para simular el estado del clima en un período futuro dado un estado climático inicial.
- Se buscará obtener la predicción del clima de todo el mes de diciembre tomando en cuenta los 4 estados como estado base con lo cual se mostrará los 31 días para cada estado y así determinar el conteo total y el promedio de días en cada estado climático a lo largo de múltiples simulaciones para entender las probabilidades relativas de cada estado.
- Se mostrarán varios gráficos de los diferentes estados climáticos donde se mostrará su frecuencia dependiendo su estado inicial y con esto describir y explicar los procedimientos utilizados, los resultados obtenidos y su relevancia para la comprensión y predicción del clima basado en la cadena de Markov.

Explicación:

En nuestro proyecto crearemos una variable de datos en la cual guardaremos los estados del clima desde enero hasta noviembre del presente año. En este caso para que no haya implicaciones que cambien al código se decidió optar que los datos fueran numéricos, es decir, los números elegidos del 0 al 3, serían implementados como estados del clima. Aquí es su topología:

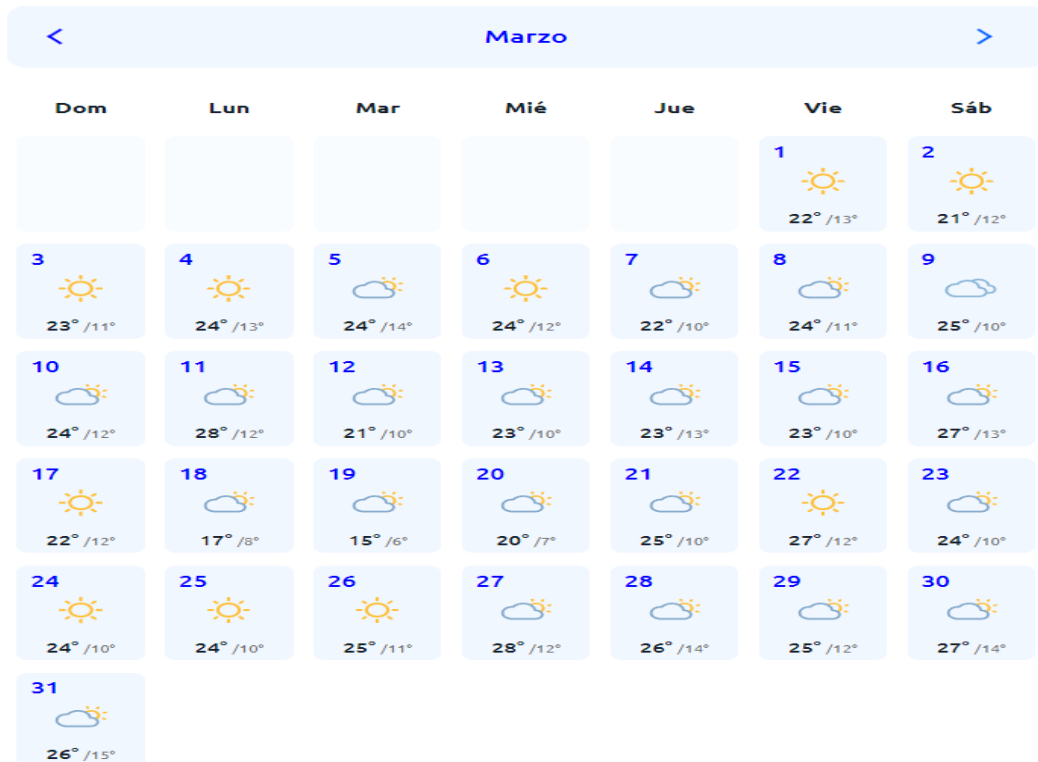
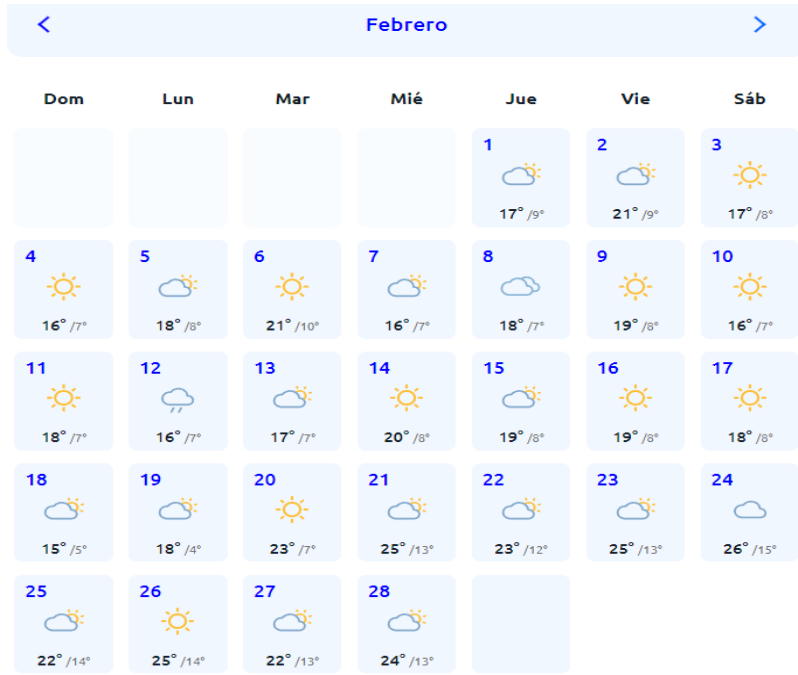
- 0: Soleado
- 1: Nublado
- 2: Lluvioso
- 3: Tormenta

Una vez ya seleccionado los estados y sus respectivos nombres, vamos a hacer la recopilación de los datos que se usaran para el proyecto. En este caso se tomaron de una página meteorológica en la internet llamada tiempo3.com; en donde cada estado se selecciona por la topología que maneja en el sumario de específicamente la ciudad de Saltillo.

Aquí se encuentran recopilados los datos iniciales que se usaran:

< Enero >						
Dom	Lun	Mar	Mié	Jue	Vie	Sáb
	1  20° / 10°	2  16° / 7°	3  17° / 6°	4  20° / 9°	5  20° / 9°	6  20° / 10°
7  21° / 10°	8  18° / 9°	9  18° / 9°	10  16° / 7°	11  14° / 6°	12  14° / 6°	13  15° / 5°
14  20° / 8°	15  17° / 7°	16  16° / 5°	17  19° / 10°	18  22° / 11°	19  20° / 11°	20  16° / 9°
21  16° / 7°	22  16° / 8°	23  20° / 8°	24  19° / 11°	25  20° / 9°	26  16° / 7°	27  15° / 6°
28  15° / 5°	29  18° / 8°	30  22° / 10°	31  19° / 10°			

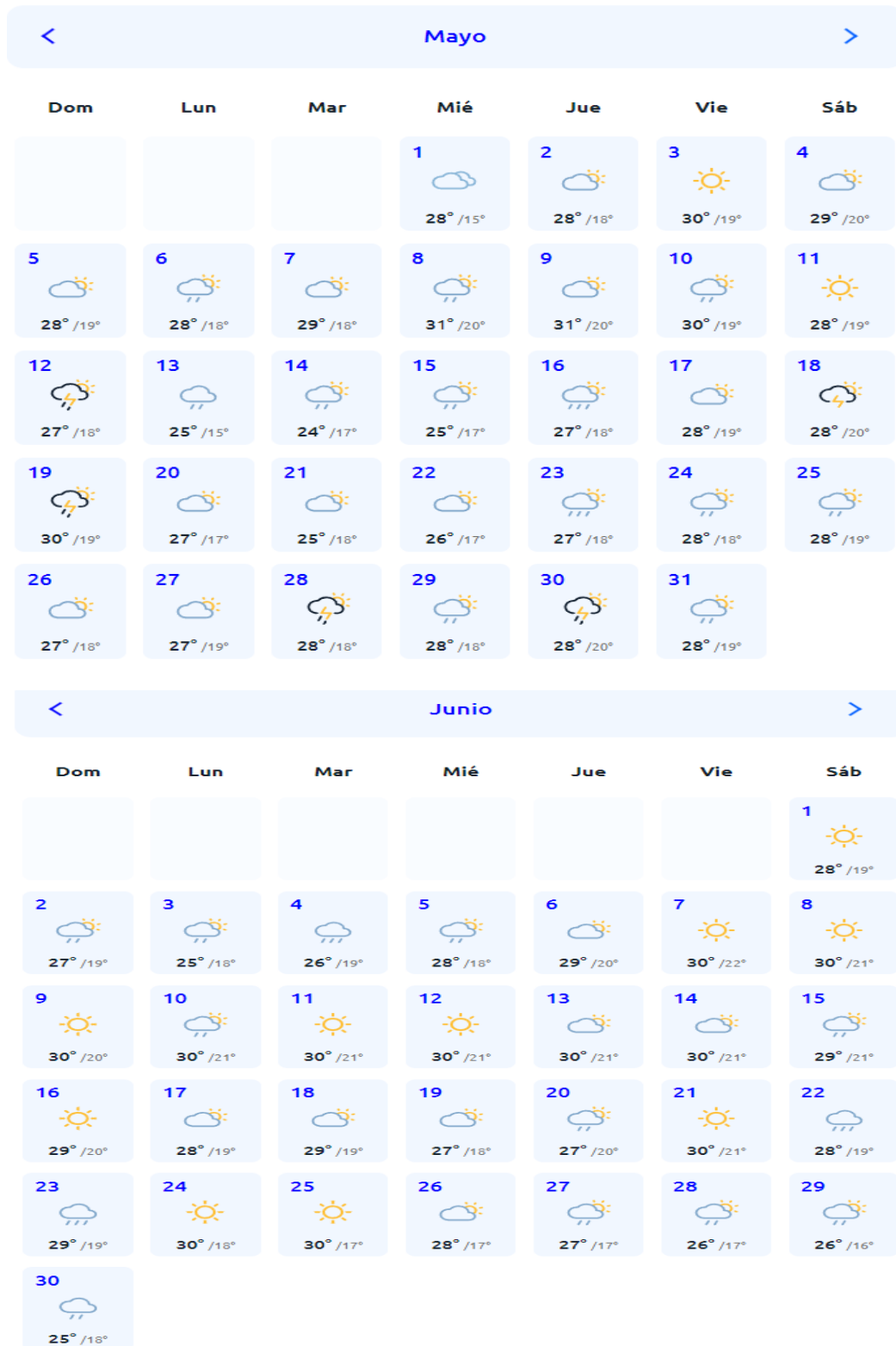
Proyecto Final



Proyecto Final



Proyecto Final



Proyecto Final



Julio

Dom	Lun	Mar	Mié	Jue	Vie	Sáb
	1 27° / 18°	2 28° / 17°	3 27° / 17°	4 28° / 18°	5 28° / 20°	6 27° / 18°
7 24° / 14°	8 26° / 15°	9 28° / 18°	10 28° / 18°	11 29° / 19°	12 29° / 19°	13 29° / 18°
14 28° / 18°	15 28° / 17°	16 27° / 17°	17 29° / 17°	18 29° / 17°	19 29° / 18°	20 30° / 19°
21 28° / 16°	22 27° / 15°	23 28° / 20°	24 29° / 17°	25 28° / 17°	26 29° / 18°	27 27° / 18°
28 25° / 16°	29 27° / 17°	30 27° / 16°	31 27° / 17°			

























Agosto

Dom	Lun	Mar	Mié	Jue	Vie	Sáb
				1 26° / 17°	2 27° / 18°	3 29° / 17°
4 29° / 19°	5 29° / 18°	6 27° / 19°	7 28° / 17°	8 30° / 18°	9 29° / 18°	10 29° / 18°
11 29° / 19°	12 28° / 19°	13 27° / 18°	14 26° / 18°	15 24° / 18°	16 26° / 17°	17 28° / 17°
18 28° / 18°	19 26° / 19°	20 28° / 17°	21 27° / 18°	22 26° / 18°	23 25° / 17°	24 26° / 18°
25 26° / 18°	26 25° / 19°	27 25° / 17°	28 28° / 18°	29 24° / 14°	30 26° / 18°	31 26° / 18°

Proyecto Final



Septiembre

Dom	Lun	Mar	Mié	Jue	Vie	Sáb
1  26° / 17°	2  26° / 17°	3  27° / 17°	4  27° / 17°	5  27° / 17°	6  26° / 17°	7  26° / 16°
8  26° / 16°	9  27° / 16°	10  28° / 16°	11  28° / 16°	12  26° / 15°	13  26° / 15°	14  26° / 16°
15  26° / 17°	16  25° / 17°	17  25° / 17°	18  25° / 16°	19  26° / 16°	20  30° / 15°	21  27° / 16°
22  24° / 15°	23  25° / 15°	24  28° / 15°	25  26° / 16°	26  25° / 15°	27  26° / 15°	28  25° / 15°
29  25° / 14°	30  25° / 15°					

Octubre

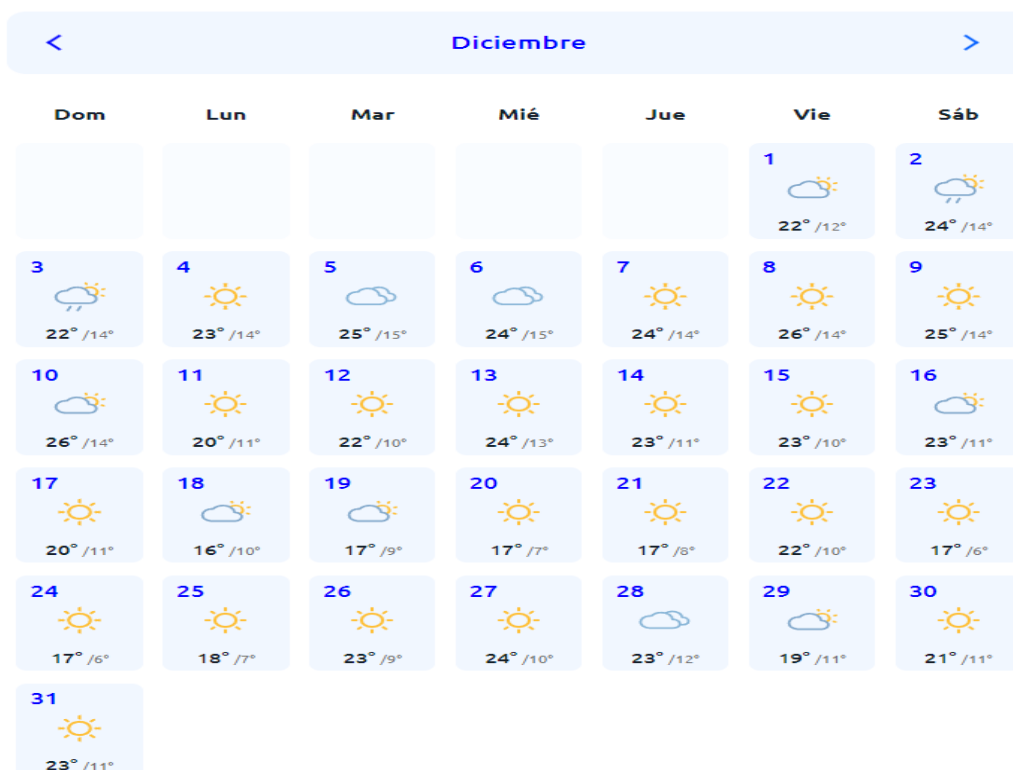
Dom	Lun	Mar	Mié	Jue	Vie	Sáb
		1  25° / 15°	2  24° / 15°	3  23° / 14°	4  22° / 15°	5  22° / 15°
6  20° / 14°	7  20° / 13°	8  19° / 13°	9  21° / 14°	10  23° / 17°	11  24° / 16°	12  28° / 16°
13  26° / 17°	14  24° / 16°	15  25° / 15°	16  22° / 12°	17  21° / 11°	18  22° / 11°	19  22° / 13°
20  23° / 13°	21  26° / 13°	22  26° / 15°	23  25° / 16°	24  27° / 16°	25  26° / 16°	26  24° / 15°
27  26° / 16°	28  24° / 14°	29  25° / 13°	30  22° / 10°	31  18° / 9°		

Proyecto Final



NOTA:

- Dado que solo se tomaron los datos de enero a noviembre, los datos meteorológicos de diciembre se tomarán solo como referencia para las simulaciones futuras y solo es representativo de los resultados que encontraremos.



Primero debemos almacenar los datos para concretar los datos elocuentes para la cadena de Markov:

```

{r}
# Estados:
## 0: soleado
## 1: nublado
## 2: lluvioso
## 3: tormenta
datos = c(0,1,0,0,0,0,1,1,1,0,1,1,0,0,0,0,0,0,1,2,1,1,1,0,1,1,1,0,0,
1,1,0,0,1,0,1,1,0,0,0,2,1,0,1,0,0,1,1,0,1,1,1,1,0,1,1,
0,0,0,0,1,0,1,1,1,1,1,1,1,1,1,0,1,1,1,1,0,0,1,1,1,1,1,
2,1,1,2,1,1,1,1,0,1,1,1,1,1,2,0,0,1,0,1,1,1,1,1,3,2,1,1,
1,1,0,1,1,2,1,2,1,2,0,3,2,2,2,2,1,3,3,1,1,1,2,2,2,1,1,3,2,3,2,
0,2,2,2,2,1,0,0,0,2,0,0,1,1,2,0,1,1,1,2,0,2,2,0,0,1,2,2,2,2,
2,1,2,2,2,2,2,2,2,2,0,2,2,2,3,0,2,2,2,2,2,2,1,1,1,2,2,3,1,0,
2,3,2,2,0,0,0,0,2,1,1,2,2,2,2,2,2,2,3,2,2,1,1,1,2,2,1,1,1,2,
2,3,2,1,1,1,1,0,1,1,2,1,2,1,2,1,1,1,1,2,0,2,2,1,1,2,0,2,
2,2,2,2,2,1,1,1,1,1,1,1,2,0,0,2,2,0,1,2,2,3,0,0,0,1,0,0,0,0,0,
2,0,0,1,0,0,0,0,1,1,1,1,1,1,0,0,1,0,0,0,0,0,0,0,2,1,1,2,1,0)

```

Con los datos debemos diferir en cómo se vería la matriz de transición usando una matriz de 4x4. Trabajaremos con la matriz más adelante pero ya se tiene una idea de cómo se trabajará. (Explicación más adelante en la explicación).

$$P = \begin{bmatrix} P(\text{Soleado}|\text{Soleado}) & P(\text{Nublado}|\text{Soleado}) & P(\text{Lluvioso}|\text{Soleado}) & P(\text{Tormenta}|\text{Soleado}) \\ P(\text{Soleado}|\text{Nublado}) & P(\text{Nublado}|\text{Nublado}) & P(\text{Lluvioso}|\text{Nublado}) & P(\text{Tormenta}|\text{Nublado}) \\ P(\text{Soleado}|\text{Lluvioso}) & P(\text{Nublado}|\text{Lluvioso}) & P(\text{Lluvioso}|\text{Lluvioso}) & P(\text{Tormenta}|\text{Lluvioso}) \\ P(\text{Soleado}|\text{Tormenta}) & P(\text{Nublado}|\text{Tormenta}) & P(\text{Lluvioso}|\text{Tormenta}) & P(\text{Tormenta}|\text{Tormenta}) \end{bmatrix}$$

Para esto se usa la línea código `datos = c(...)` para almacenarlos, una vez hecho esto ya recompilados los datos creamos un conteo para los posibles cambios partiendo del estado 0 (Día Soleado). Su propósito en concreto es contar las veces que siga soleado al día siguiente, las veces que de soleado sea nublado, de soleado a lluvioso y por último de soleado a tormenta utilizando métodos de for & if.

```

{r}
##Se contarán las veces que siga soleado al día siguiente,
##Las veces que de soleado cambió a nublado, de soleado a lluvioso y de
soleado a tormenta.
conteo0 = c(0, 0, 0, 0)
for(i in c(11:334)){
  if(datos[i-1]==0){
    if(datos[i]==0) conteo0[1]=conteo0[1]+1
    if(datos[i]==1) conteo0[2]=conteo0[2]+1
    if(datos[i]==2) conteo0[3]=conteo0[3]+1
    if(datos[i]==3) conteo0[4]=conteo0[4]+1
  }
}

```

Estos conteos se repiten 3 veces más para los cambios de estados restantes. En concreto lo que se hace es un ciclo en donde contamos las veces en que después del estado 0 exista otro estado 0; en donde después de un estado 0 se genere un estado 1, después de un

estado 0 haya un estado 2 y de un estado 0 se encuentre un estado 3. Esto se genera en la primera fila de las cadenas de Márkov, para que esto sea apropiado debemos implementar otros 3 ciclos adicionales para poder así hacer conteos de los otros estados.

De manera general estos bloques de código están realizando un conteo de las transiciones entre los estados climáticos. En particular, se están contando las veces que ocurren transiciones específicas entre los diferentes estados climáticos (soleado, nublado, lluvioso, tormenta) en un conjunto de datos históricos.

Cada bloque se encarga de contar las transiciones desde un estado particular hacia los otros estados posibles.

11 meses para 334 datos históricos en concreto.

##Se hace el mismo conteo para los demás cambios de estados.

```
{r}
conteo1 = c(0, 0, 0, 0)
for(i in c(11:334)){
  if(datos[i-1]==1){
    if(datos[i]==0) conteo1[1]=conteo1[1]+1
    if(datos[i]==1) conteo1[2]=conteo1[2]+1
    if(datos[i]==2) conteo1[3]=conteo1[3]+1
    if(datos[i]==3) conteo1[4]=conteo1[4]+1
  }
}
```

```
conteo2 = c(0, 0, 0, 0)
for(i in c(11:334)){
  if(datos[i-1]==2){
    if(datos[i]==0) conteo2[1]=conteo2[1]+1
    if(datos[i]==1) conteo2[2]=conteo2[2]+1
    if(datos[i]==2) conteo2[3]=conteo2[3]+1
    if(datos[i]==3) conteo2[4]=conteo2[4]+1
  }
}
```

```
conteo3 = c(0, 0, 0, 0)
for(i in c(11:334)){
  if(datos[i-1]==3){
    if(datos[i]==0) conteo3[1]=conteo3[1]+1
    if(datos[i]==1) conteo3[2]=conteo3[2]+1
    if(datos[i]==2) conteo3[3]=conteo3[3]+1
    if(datos[i]==3) conteo3[4]=conteo3[4]+1
  }
}
...
```

Una vez compilados los bucles, se sacan las probabilidades, estas se realizan dividiendo el número contado de cada uno entre el número total de cada estado usando la siguiente fórmula matemática:

$$(p(x) = \text{conteos}/n).$$

```
##Despues normaliza los conteos de transición para cada estado climático.
Línea por línea:
```

```
##n = sum(conteo0): Calcula la suma de los conteos de transición para el
estado 0 (soleado). Es decir, suma las veces que el estado 0 cambió a otros
estados.
```

```
##conteo0[1]=conteo0[1]/n: Normaliza la frecuencia de transición del estado 0
al estado 0 (soleado a soleado) dividiendo el conteo de esta transición por
el total de transiciones desde el estado 0.
```

```
##conteo0[2]=conteo0[2]/n: Hace lo mismo que el paso anterior pero para la
transición del estado 0 al estado 1 (soleado a nublado).
```

```
##conteo0[3]=conteo0[3]/n: Normaliza la frecuencia de transición del estado 0
al estado 2 (soleado a lluvioso).
```

```
##conteo0[4]=conteo0[4]/n: Normaliza la frecuencia de transición del estado 0
al estado 3 (soleado a tormenta).
```

```
{r}
n = sum(conteo0)
conteo0[1]=conteo0[1]/n
conteo0[2]=conteo0[2]/n
conteo0[3]=conteo0[3]/n
conteo0[4]=conteo0[4]/n

n = sum(conteo1)
conteo1[1]=conteo1[1]/n
conteo1[2]=conteo1[2]/n
conteo1[3]=conteo1[3]/n
conteo1[4]=conteo1[4]/n

n = sum(conteo2)
conteo2[1]=conteo2[1]/n
conteo2[2]=conteo2[2]/n
conteo2[3]=conteo2[3]/n
conteo2[4]=conteo2[4]/n

n = sum(conteo3)
conteo3[1]=conteo3[1]/n
conteo3[2]=conteo3[2]/n
conteo3[3]=conteo3[3]/n
conteo3[4]=conteo3[4]/n
```

Para resolver el problema se tuvieron que contar sistemáticamente cuantos estados de cada uno existen en la recopilación histórica. En el caso del estado 0 se encontraron 90 casualidades, del estado 1 el total de casualidades fueron de 140, para el estado 2 el total de casualidades fueron 92, y para el estado 3 el total fue de 12, sumando así los 334 datos que recopilamos inicialmente.

Ya con todos estos recabado el siguiente chunk está diseñado para dividir cada número de los conteos entre su respectivo total de estados de manera automática. Una vez asignadas

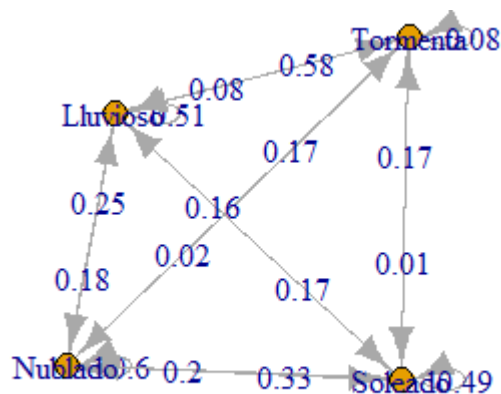
las probabilidades es donde se genera la cadena de Márkov y se completa guardando en la variable (*mc_p*) para imprimir la matriz y la tabla de transición usando la formula ya mencionada.

```
##Sacamos los porcentajes dada la siguiente formula:  $p(x) = \text{conteos}/n$ 
```{r}
statesNames = c("Soleado", "Nublado", "Lluvioso", "Tormenta")
mc_p = new("markovchain", transitionMatrix = matrix(c(conteo0, conteo1,
conteo2, conteo3), byrow=TRUE,
nrow=4, dimnames=list(statesNames, statesNames)))

print(mc_p)
plot(mc_p)
```
```

Una vez se imprime la variable *mc_p* y se entregan sus resultados de probabilidad. La primera imagen son los resultados de la matriz en bruto, mientras en la segunda representa la tabla de transición:

| | Soleado | Nublado | Lluvioso | Tormenta |
|----------|-----------|-----------|-----------|------------|
| Soleado | 0.4880952 | 0.3333333 | 0.1666667 | 0.01190476 |
| Nublado | 0.1985294 | 0.6029412 | 0.1764706 | 0.02205882 |
| Lluvioso | 0.1630435 | 0.2500000 | 0.5108696 | 0.07608696 |
| Tormenta | 0.1666667 | 0.1666667 | 0.5833333 | 0.08333333 |



Una vez generado los resultados podemos redondear los resultados para así crear la matriz que se usara para el proyecto a partir de la que se había diseñado al principio del proyecto. He aquí el resultado:

$$P = \begin{bmatrix} 0.488 & 0.333 & 0.167 & 0.012 \\ 0.199 & 0.603 & 0.176 & 0.022 \\ 0.163 & 0.250 & 0.511 & 0.076 \\ 0.076 & 0.167 & 0.583 & 0.083 \end{bmatrix}$$

La matriz de transición se construye a partir de los conteos de cambios de estados. En el código, se consideraron los cuatro estados iniciales preestablecidos (Soleado, Nublado, Lluvioso, Tormenta), por lo que la matriz de transición es de tamaño 4x4. Cada fila de la matriz representa el estado actual, y cada columna representa el estado siguiente.

Por lo tanto, la matriz de transición utilizada en el código es de tamaño 4x4. Puedes observar la construcción de esta matriz en la sección donde se crea el objeto “mc_p”:

```
mc_p = new("markovchain", transitionMatrix
          = matrix(c(conteo0, conteo1, conteo2, conteo3), byrow = TRUE, nrow
                    = 4, dimnames = list(statesNames, statesNames)))
```

Se denota en la línea de código que: conteo0, conteo1, conteo2, y conteo3 son vectores que contienen los conteos de transiciones para los estados 0, 1, 2, y 3, respectivamente. La función **matrix** se utiliza para construir la matriz de transición a partir de estos conteos. La matriz resultante tiene dimensiones 4x4, como se especifica con el argumento `nrow=4`.

Por lo tanto, la matriz de transición que se diseñó suponiendo la matriz iniciada fue de:

$$P = \begin{bmatrix} P(\text{Soleado}|\text{Soleado}) & P(\text{Nublado}|\text{Soleado}) & P(\text{Lluvioso}|\text{Soleado}) & P(\text{Tormenta}|\text{Soleado}) \\ P(\text{Soleado}|\text{Nublado}) & P(\text{Nublado}|\text{Nublado}) & P(\text{Lluvioso}|\text{Nublado}) & P(\text{Tormenta}|\text{Nublado}) \\ P(\text{Soleado}|\text{Lluvioso}) & P(\text{Nublado}|\text{Lluvioso}) & P(\text{Lluvioso}|\text{Lluvioso}) & P(\text{Tormenta}|\text{Lluvioso}) \\ P(\text{Soleado}|\text{Tormenta}) & P(\text{Nublado}|\text{Tormenta}) & P(\text{Lluvioso}|\text{Tormenta}) & P(\text{Tormenta}|\text{Tormenta}) \end{bmatrix}$$

Conociendo ahora los datos y los resultados completos, podemos continuar con la explicación de cómo funcionan las simulaciones y las pruebas.

Para esto se creó una función para aleatoriamente calcular un estado dependiendo del estado anterior, para así al haber creado una función para aleatoriamente elegir con las probabilidades cargadas, entonces con esto se estima el clima del siguiente día, el primer caso de prueba empieza con un solo día.

Lo que se realiza en el chunk es que la variable **estimarClima**, toma como argumento el estado climático actual (**climaAnterior**) y utiliza una cadena de Markov representada por la matriz **mc_p** para estimar el clima del siguiente día basado en las probabilidades cargadas en esa matriz.

Mas concretamente es esto lo que hace cada parte en 3 sencillos pasos:

1. **aleatorio = runif(1)**: Genera un número aleatorio uniforme entre 0 y 1. Esto simula la probabilidad de transición a diferentes estados climáticos.
2. **climaAnterior = climaAnterior+1**: Incrementa en uno el estado climático actual. Esto se hace para ajustar el índice del estado climático. En tu código, se usa una representación numérica para los estados climáticos (0 para Soleado, 1 para Nublado, 2 para Lluvioso y 3 para Tormenta). Aumentar en uno permite acceder a las filas correspondientes en la matriz **mc_p**.
3. Los **if** y **else if** siguientes comparan el número aleatorio generado con sumas acumulativas de las probabilidades en la matriz **mc_p**.
 - Si el número aleatorio es menor que la probabilidad de transición a Soleado desde el estado **climaAnterior**, devuelve 0 (representando Soleado).
 - Si es menor que la probabilidad de transición a Nublado, devuelve 1, y así sucesivamente para Lluvioso y Tormenta.

Este es el chunk que se explicó:

```

```{r}
estimarClima <- function(climaAnterior){
 aleatorio = runif(1)
 climaAnterior = climaAnterior+1
 if(aleatorio<sum(mc_p[climaAnterior][1:1])){
 return(0)
 }
 else if(aleatorio<sum(mc_p[climaAnterior][1:2])){
 return(1)
 }
 else if(aleatorio<sum(mc_p[climaAnterior][1:3])){
 return(2)
 }
 else if(aleatorio<sum(mc_p[climaAnterior][1:4])){
 return(3)
 }
}
```

```

Para que el programa nos otorgue el estado climático del día siguiente solo habría que imprimir la variable **estimarClima** para cada estado.

##Se supondra del estado que empieze, calcular aleatoriamente con las probabilidades de la cadena de markov, (Este no siempre será el mismo).

```

```{r}
##Recuerda que:
0: soleado
1: Nublado
2: Lluvioso
3: Tormenta
print(estimarClima(0))
print(estimarClima(1))
print(estimarClima(2))
print(estimarClima(3))
```

```

Gracias a esto podemos saber para solo un día que estado sería el siguiente, pero con motivo de la cantidad abrupta de datos se puede además simular en un intervalo de días 1 por 1 usando la temporalidad para así simular el mes de diciembre del presente año en un intervalo de 31 usando las mismas especificaciones de la primera simulación.

Para esto debemos crear una función que pueda simular la cadena de Markov con un intervalo de días usando la temporalidad como guía.

```
{R}
# Crear una función para simular la cadena de Markov por un número específico
de días usando temporalidad.

simularClima <- function(dias, climaInicial = 0) {
  climaActual <- climaInicial
  historial <- c(climaActual)

  for (i in 1:(dias - 1)) { #Bucle para asegurar que la longitud sea 'dias'.
    climaActual <- estimarClima(climaActual)
    historial <- c(historial, climaActual)
  }

  return(historial)
}
```

Dentro del chunk existe una función llamada **simularClima**. Su objetivo es simular la evolución del clima a lo largo de un número específico de días basado en la cadena de Markov y la función **estimarClima** que define cómo cambia el clima de un día a otro.

Dentro de la función, se crea un historial para registrar los estados climáticos en cada día. Se utiliza un bucle **for** para calcular el clima de cada día basándose en el clima del día anterior mediante la función **estimarClima**. Esta última función utiliza la información de transición (probabilidades) de la cadena de Markov para generar el clima del día siguiente basándose en el estado actual.

Con esto ya completado ahora podemos realizar las simulaciones correspondientes, para esto debemos utilizar la función para simular los 31 días de clima basado en la cadena de Markov usando temporalidad:


```
# Utilizar la función para simular 31 días de clima basado en la cadena de
# Markov usando temporalidad:
historialSimuladoSoleado <- simularClima(31, climaInicial = 0)
print("Suponiendo que hoy ha sido un día soleado:")
print(historialSimuladoSoleado)

historialSimuladoNublado <- simularClima(31, climaInicial = 1)
print("Suponiendo que hoy ha sido un día nublado:")
print(historialSimuladoNublado)

historialSimuladoLluvioso <- simularClima(31, climaInicial = 2)
print("Suponiendo que hoy ha sido un día lluvioso:")
print(historialSimuladoLluvioso)

historialSimuladoTormenta <- simularClima(31, climaInicial = 3)
print("Suponiendo que hoy ha sido un día de tormenta:")
print(historialSimuladoTormenta)
```

El chunk realiza la simulación correspondiente usando la cadena de Markov y simula los 31 días dependiendo de en qué día empieza el primer estado, 1 para cada 1.

Ya al final las líneas de código siguientes son para recabar estos resultados a través de gráficos de barras; 4 en total para cada estado, sumando de los 31 días cuantas iteraciones se lograron encontrar. Con esto creamos una función para obtener el conteo de días en cada estado y plotear los gráficos correspondientes.

```
# Función para obtener el conteo con todos los estados
obtenerConteo <- function(historial) {
  estados <- c(0, 1, 2, 3)
  conteo <- table(factor(historial, levels = estados))
  conteo[as.character(estados)] <- conteo[as.character(estados)]
  return(conteo)
}

# Función para obtener el gráfico con todos los estados
obtenerConteoYPlot <- function(historial, titulo) {
  conteo <- obtenerConteo(historial)
  datosConteo <- data.frame(
    Estado = as.factor(names(conteo)),
    Conteo = as.numeric(conteo)
  )
  ggplot(datosConteo, aes(x = Estado, y = Conteo, fill = Estado)) +
    geom_bar(stat = "identity") +
```

```

    geom_text(aes(label = Conteo), position = position_stack(vjust = 0.5),
size = 3) +
  labs(title = titulo, x = "Estado Climático", y = "Conteo") +
  theme_minimal()
}

# Obtener conteo y plotear para cada simulación
plotSoleado <- obtenerConteoYPlot(historialSimuladoSoleado, "Simulación de si
el día empieza en soleado; Estado 0")
plotNublado <- obtenerConteoYPlot(historialSimuladoNublado, "Simulación de si
el día empieza en nublado; Estado 1")
plotLluvioso <- obtenerConteoYPlot(historialSimuladoLluvioso, "Simulación de
si empieza en día lluvioso; Estado 2")
plotTormenta <- obtenerConteoYPlot(historialSimuladoTormenta, "Simulación de
si empieza en día de tormenta; Estado 3")

```

```

# Mostrar los gráficos
print(plotSoleado)
print(plotNublado)
print(plotLluvioso)
print(plotTormenta)

```

Con los datos ya recabados se realiza un análisis final visualizando los totales y promedios de todos los estados que se encontraron en las simulaciones a través del historial simulado para los 4 estados utilizando un gráfico de barras con etiquetas.

```

```{r}
Función para obtener el conteo con todos los estados
obtenerConteo <- function(historial) {
 # Crear un vector con los estados posibles
 estados <- c(0, 1, 2, 3)

 # Obtener el conteo para cada estado
 conteo <- table(factor(historial, levels = estados))

 # Convertir a un vector con todos los estados
 conteo[as.character(estados)] <- conteo[as.character(estados)]
 return(conteo)
}

Función para obtener el gráfico con todos los estados

```

```

obtenerConteoYPlot <- function(historial, titulo) {
 conteo <- obtenerConteo(historial)

 # Crear un marco de datos con todos los estados
 datosConteo <- data.frame(
 Estado = as.factor(names(conteo)),
 Conteo = as.numeric(conteo)
)

 # Trazar el gráfico con todos los estados
 ggplot(datosConteo, aes(x = Estado, y = Conteo, fill = Estado)) +
 geom_bar(stat = "identity") +
 geom_text(aes(label = Conteo), position = position_stack(vjust = 0.5),
size = 3) +
 labs(title = titulo,

```

```

 x = "Estado Climático",
 y = "Conteo") +
 theme_minimal()
}

Obtener el conteo para cada simulación
conteoSoleado <- obtenerConteo(historialSimuladoSoleado)
conteoNublado <- obtenerConteo(historialSimuladoNublado)
conteoLluvioso <- obtenerConteo(historialSimuladoLluvioso)
conteoTormenta <- obtenerConteo(historialSimuladoTormenta)

Calcular el total y promedio para todos los estados
totalEstados <- conteoSoleado + conteoNublado + conteoLluvioso +
conteoTormenta
promedioEstados <- totalEstados / 4

```

```

Crear un marco de datos con todos los estados para el gráfico
datosTotales <- data.frame(
 Estado = as.factor(names(totalEstados)),
 Total = as.numeric(totalEstados),
 Promedio = as.numeric(promedioEstados)
)

Trazar el gráfico con todos los estados
ggplot(datosTotales, aes(x = Estado)) +
 geom_bar(aes(y = Total, fill = "Total"), stat = "identity") +
 geom_bar(aes(y = Promedio, fill = "Promedio"), stat = "identity") +
 geom_text(aes(y = Total, label = Total), position = position_stack(vjust =
0.5), size = 3) +
 geom_text(aes(y = Promedio, label = round(Promedio, 2)), position =
position_stack(vjust = 0.5), size = 3) +
 labs(title = "Total y Promedio de Estados en las 4 simulaciones",
 x = "Estado Climático",

```

```

 y = "Conteo") +
 theme_minimal() +
 scale_fill_manual(values = c("Total" = "blue", "Promedio" = "green"))
}

```

## Uso y resultados:

Los usos que le podemos dar a los resultados que provienen del código son muy importantes para poder implementarlos en un posible proyecto de software relacionado a las probabilidades meteorológicas y como estos entregue resultados muchos más precisos, nuestro equipo conoce muy bien las limitaciones que esto conlleve a que los resultados en una prueba de campo pueden variar, pero con esto llevamos las raíces para un trabajo de este estilo.

Se debe recalcar que los resultados aquí mostrados en la documentación pueden variar, por lo tanto, solo son ejemplos y deben ser tomados con discreción, sin embargo, los datos que empiezan desde la simulación del clima usando temporalidad y sus respectivas graficas son relacionadas a esos resultados.

### 1. Estimación del clima inicial:

Los primeros bloques de código permiten estimar el clima inicial para diferentes estados climáticos (soleado, nublado, lluvioso, tormenta). Cada vez que se ejecuta ***estimarClima(estado)***, se genera una predicción del clima siguiente basada en la probabilidad de transición entre estados.

🚦 Interpretación de resultados:

Los resultados muestran la probabilidad estimada de cada estado climático después del estado inicial proporcionado. Por ejemplo, para el clima soleado (estado 0), se mostrarán las probabilidades estimadas de pasar a los diferentes estados climáticos.

#### RESULTADOS:

```
[1] 0 [1] 0
[1] 3 [1] 1
[1] 2 [1] 2
[1] 0 [1] 2
```

### 2. Simulación del clima a lo largo del tiempo:

En el chunk donde se encuentra la función ***simularClima***, simula el clima para un número específico de días, utilizando la cadena de Markov y un estado climático inicial dado.

🚦 Interpretación de resultados:

Se generan secuencias simuladas de estados climáticos para el período especificado. Cada número en la lista representa el estado climático correspondiente para cada día.

## RESULTADOS:

```
[1] "suponiendo que hoy ha sido un día soleado:"
[1] 0 1 1 1 1 1 1 1 1 1 2 2 3 2 1 2 1 0 0 0 1 1 2 2 1 2 1 1 1 1 1
[1] "suponiendo que hoy ha sido un día nublado:"
[1] 1 0 0 2 1 0 2 0 1 1 0 0 0 1 1 2 2 1 0 0 1 0 0 0 2 2 2 1 0 0 2
[1] "suponiendo que hoy ha sido un día lluvioso:"
[1] 2 2 3 2 2 1 2 2 2 3 2 0 1 1 2 3 2 3 0 0 0 0 2 0 2 2 3 0 1 1 2
[1] "suponiendo que hoy ha sido un día de tormenta:"
[1] 3 0 2 0 0 0 2 2 1 2 2 2 2 0 0 0 0 0 1 0 0 1 1 1 2 0 1 1 1 1 2
```

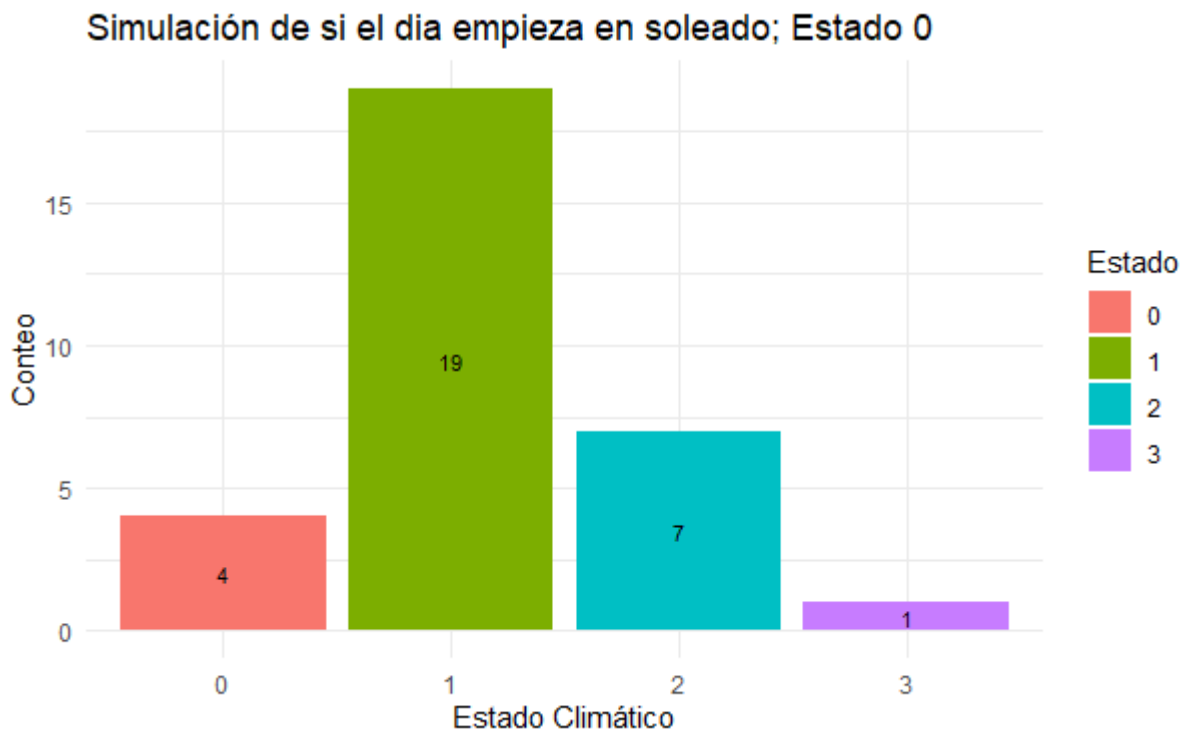
### 3. Análisis del clima simulado:

Los siguientes bloques de código aplican la simulación a cuatro estados climáticos iniciales diferentes, generando secuencias para cada uno. Luego, se cuenta la ocurrencia de cada estado y se genera un gráfico que muestra la distribución de estados climáticos.

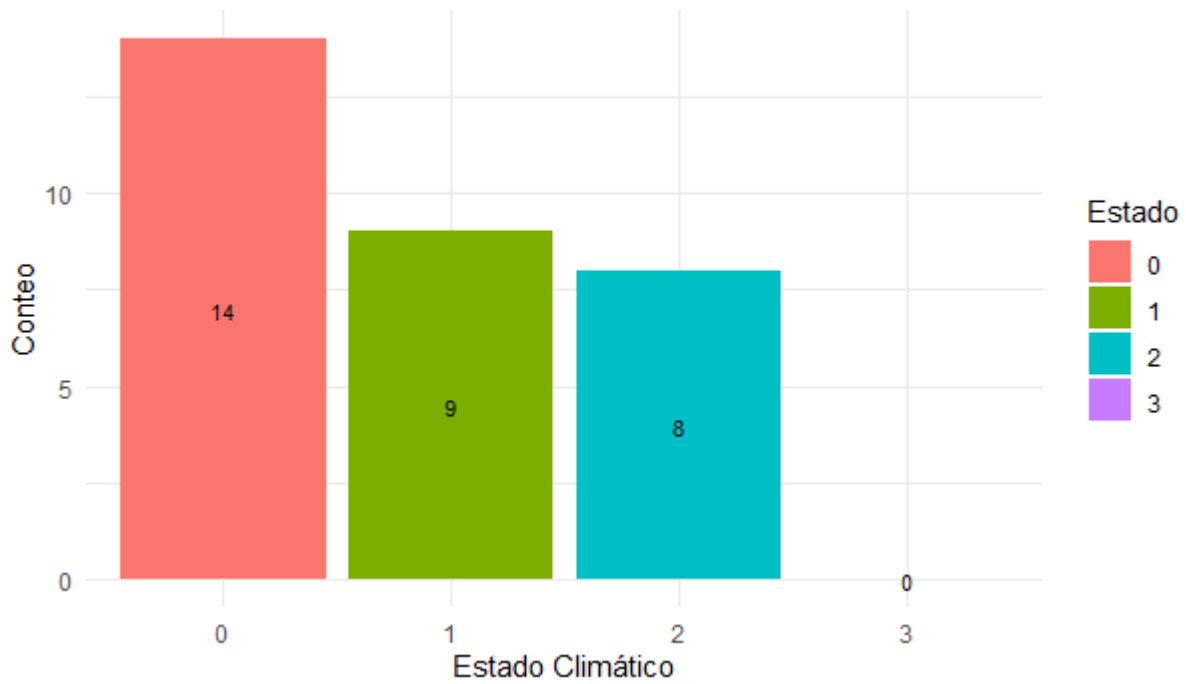
📊 Interpretación de resultados:

Los gráficos muestran la distribución de estados climáticos simulados para cada condición inicial.

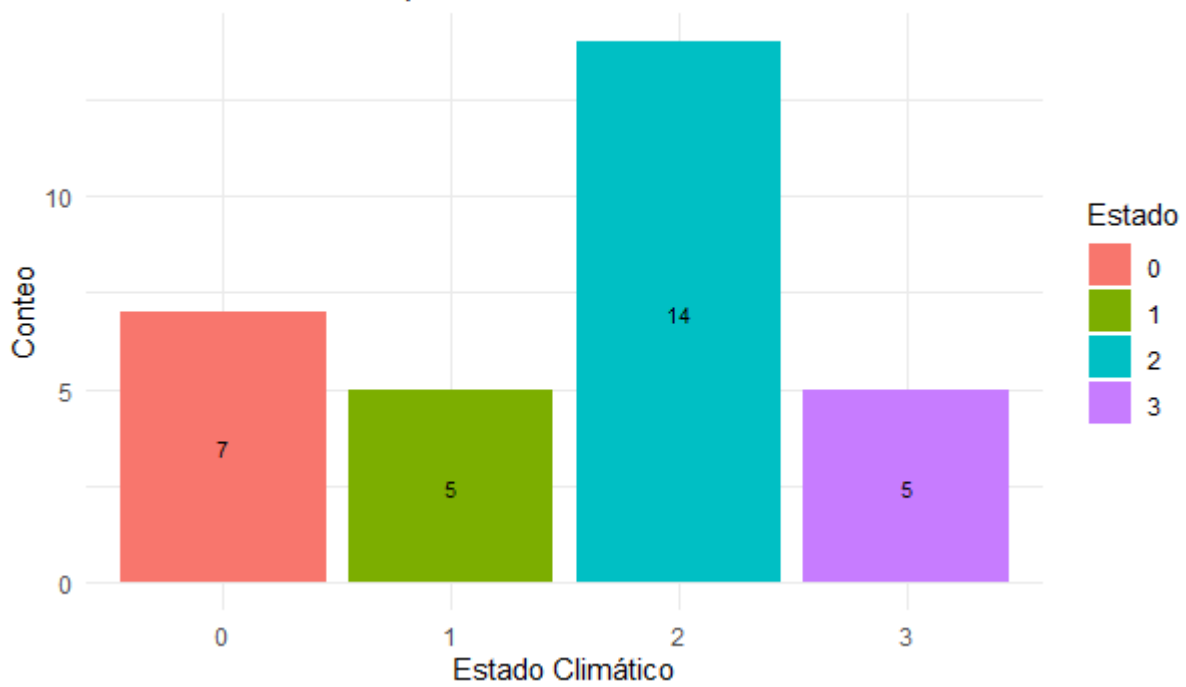
La altura de las barras representa la frecuencia de ocurrencia de cada estado climático durante el período simulado.

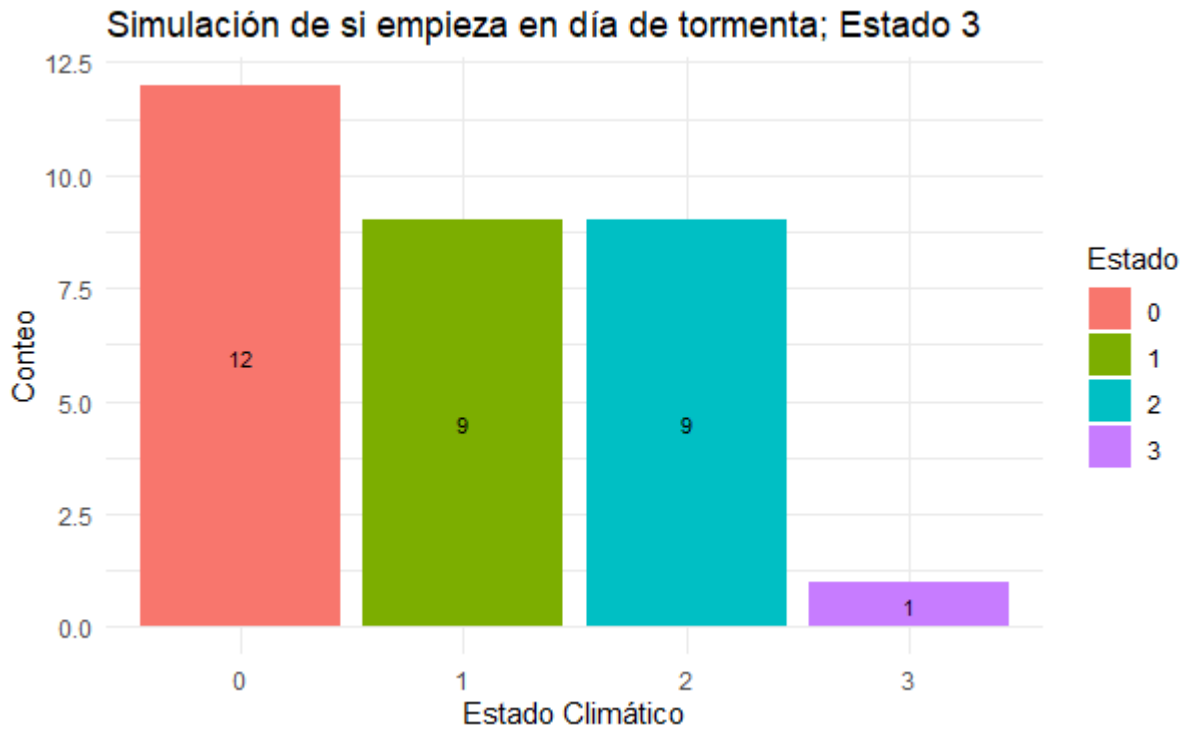


Simulación de si el dia empieza en nublado; Estado 1



Simulación de si empieza en día lluvioso; Estado 2





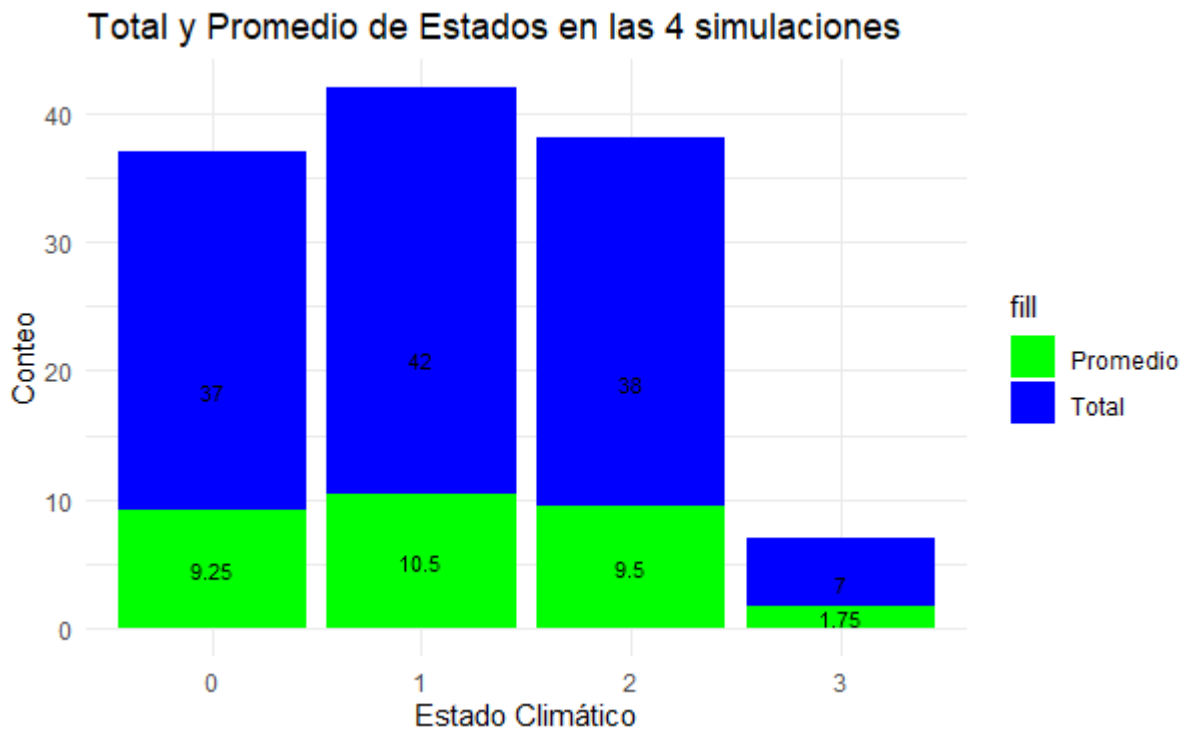
#### 4. Análisis agregado extra:

El último bloque de código calcula el conteo total y el promedio de los estados climáticos de todas las simulaciones, presentando esta información en un gráfico.

##### 📊 Interpretación de resultados:

El gráfico muestra la suma total y el promedio de la ocurrencia de cada estado climático a lo largo de todas las simulaciones.

Permite comparar la frecuencia promedio de cada estado climático entre las diferentes condiciones iniciales simuladas.



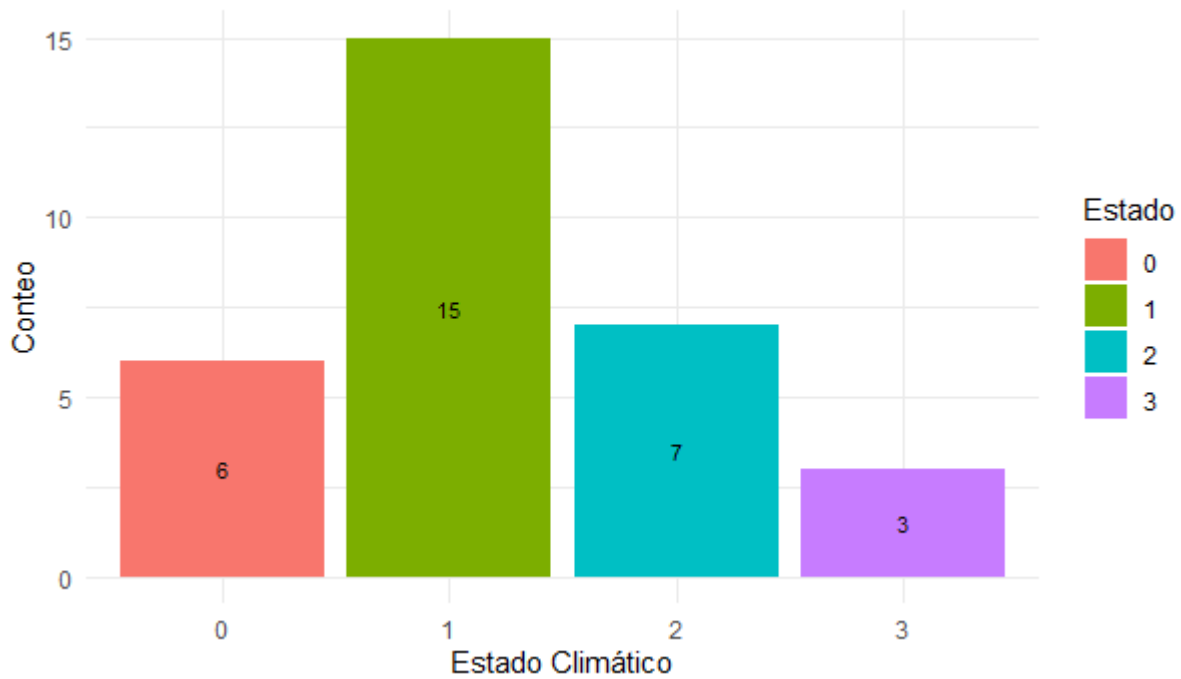
### Resultados extra:

```
[1] 1 [1] 0
[1] 0 [1] 1
[1] 2 [1] 1
[1] 0 [1] 3
```

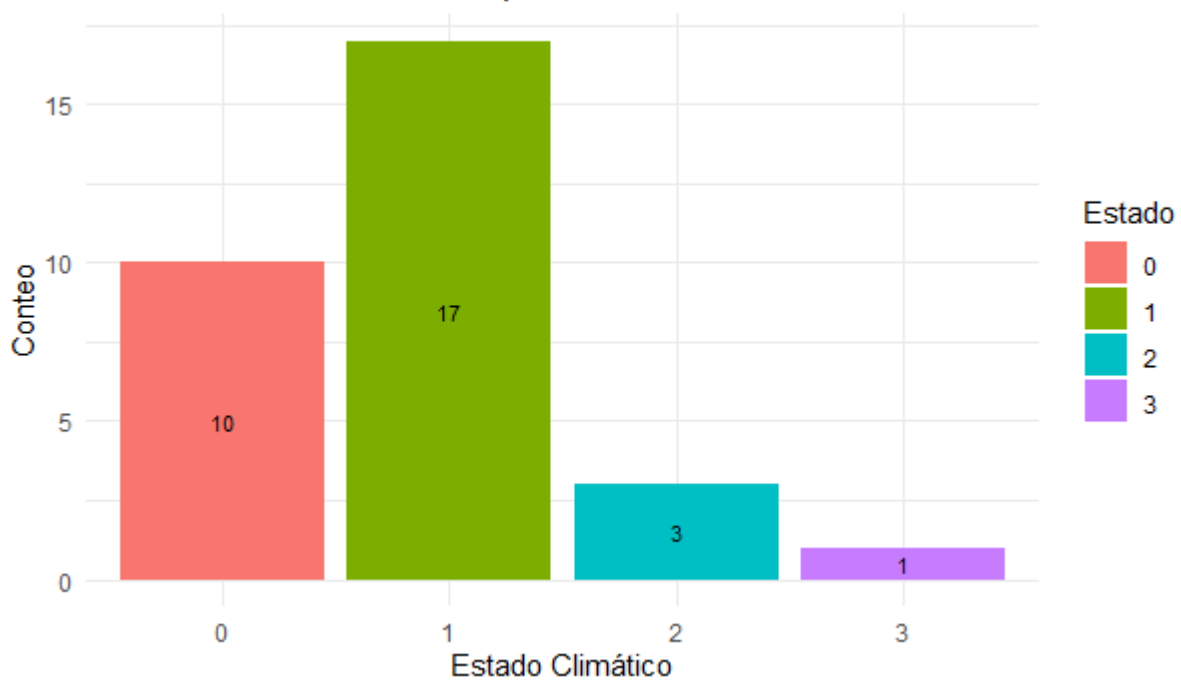
```
[1] "Suponiendo que hoy ha sido un día soleado:"
[1] 0 1 1 1 1 1 1 1 1 1 2 2 3 2 1 2 1 0 0 0 1 1 2 2 1 2 1 1 1 1 1
[1] "Suponiendo que hoy ha sido un día nublado:"
[1] 1 0 0 2 1 0 2 0 1 1 0 0 0 1 1 2 2 1 0 0 1 0 0 0 2 2 2 1 0 0 2
[1] "Suponiendo que hoy ha sido un día lluvioso:"
[1] 2 2 3 2 2 1 2 2 2 3 2 0 1 1 2 3 2 3 0 0 0 0 2 0 2 2 3 0 1 1 2
[1] "Suponiendo que hoy ha sido un día de tormenta:"
[1] 3 0 2 0 0 0 2 2 1 2 2 2 2 0 0 0 0 0 1 0 0 1 1 1 2 0 1 1 1 1 2
```



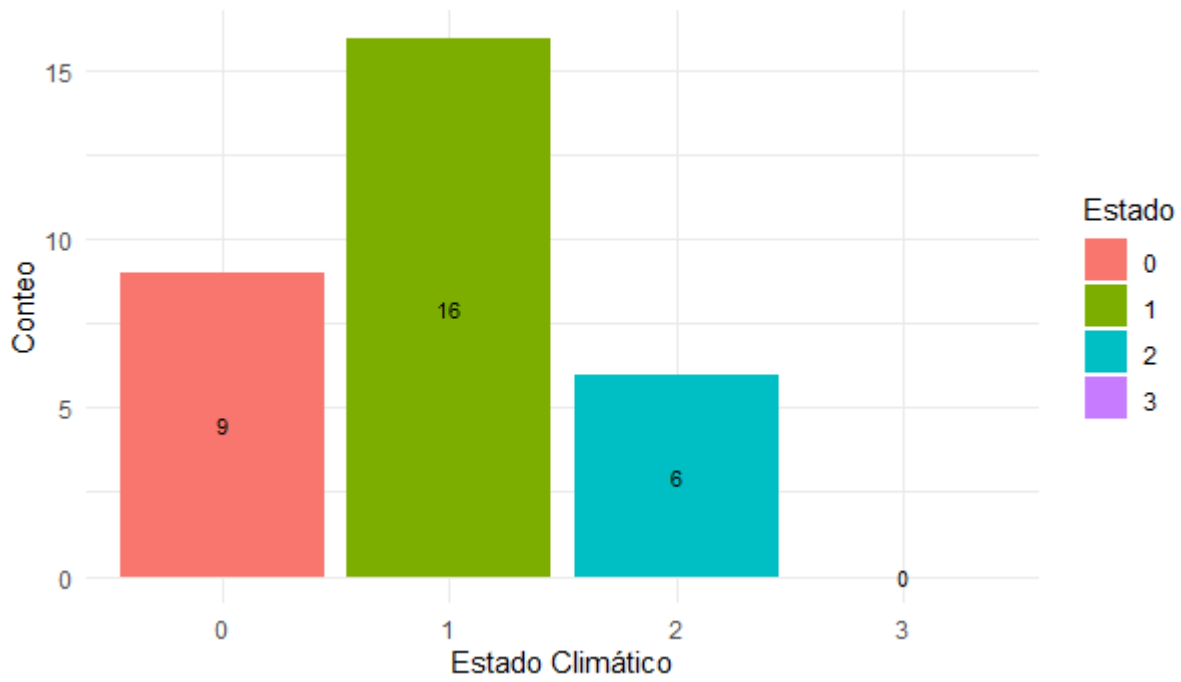
Simulación de si el día empieza en soleado; Estado 0



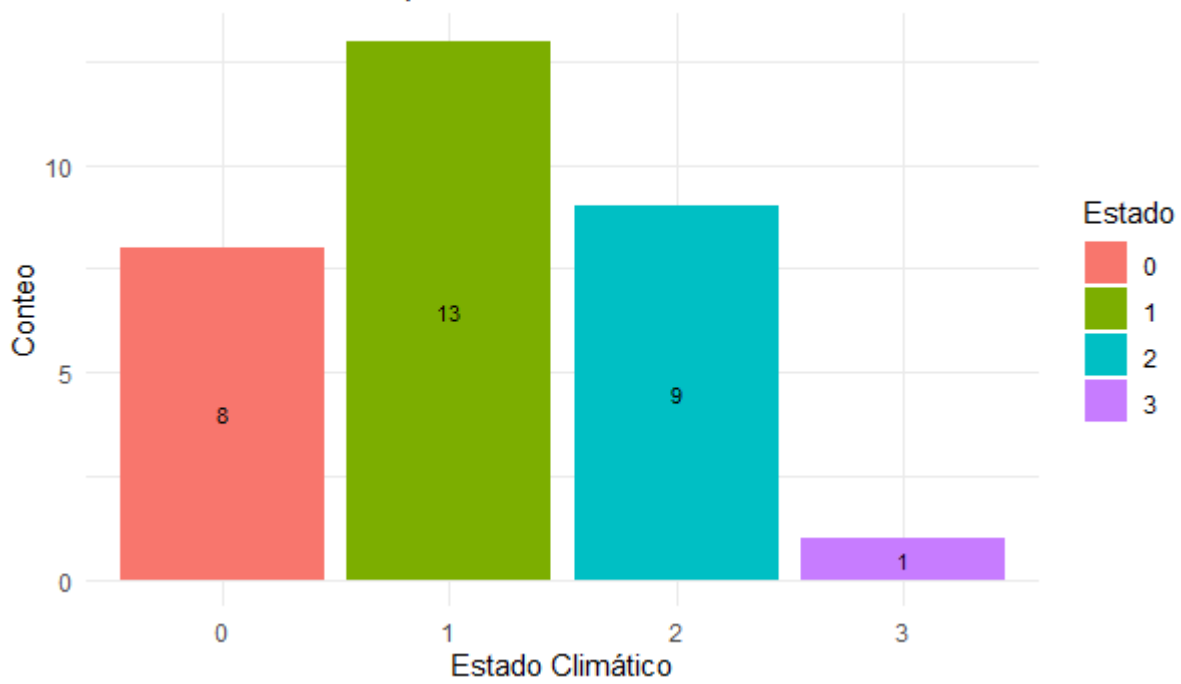
Simulación de si el día empieza en nublado; Estado 1

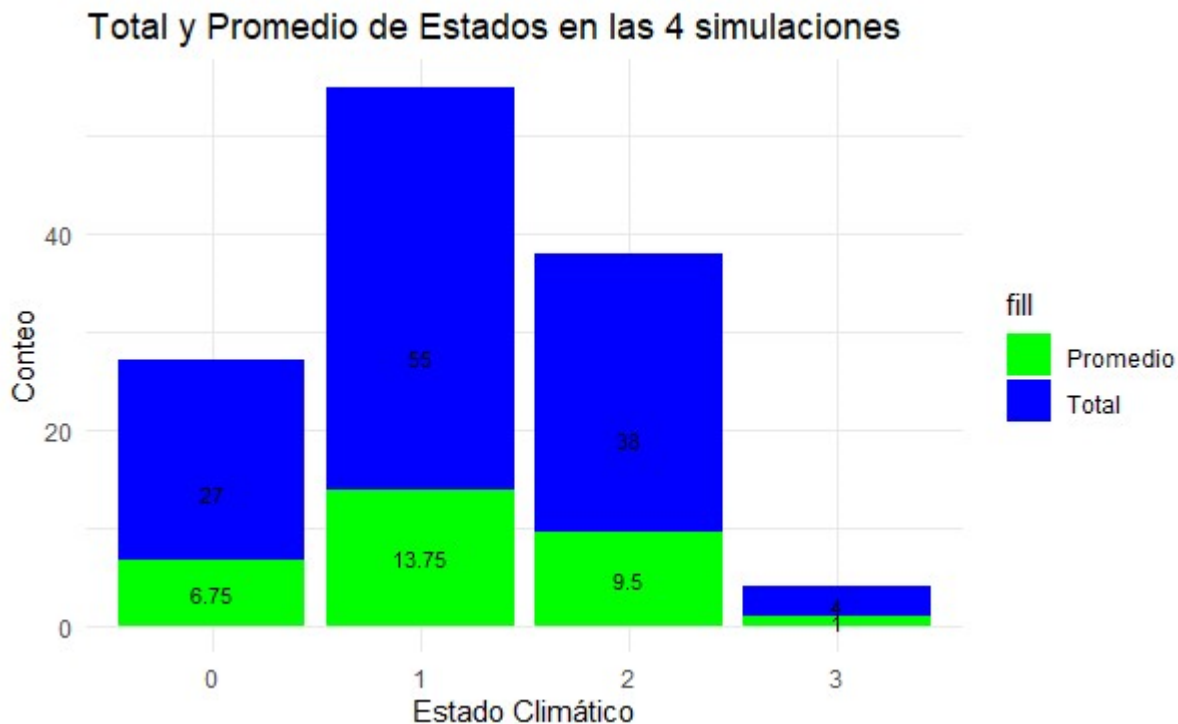


Simulación de si empieza en día lluvioso; Estado 2



Simulación de si empieza en día de tormenta; Estado 3





## Después de la recopilación ¿Qué se entendió?

Los resultados obtenidos a partir de los datos obtenidos ofrecen una visión detallada de las transiciones y la distribución probabilística de los estados climáticos. Estos datos tienen aplicaciones significativas en la planificación y toma de decisiones en diversas áreas. Por ejemplo, pueden ser fundamentales en la predicción meteorológica al proporcionar un entendimiento más preciso de las probabilidades de cambio climático a corto plazo, permitiendo a los meteorólogos y planificadores anticipar patrones climáticos. Además, en el ámbito agrícola, estas predicciones pueden ser vitales para tomar decisiones sobre siembras o cosechas, considerando la adaptación de cultivos a condiciones climáticas variables. También, en ingeniería y gestión de recursos, estos resultados pueden respaldar la planificación de infraestructuras resilientes ante posibles condiciones climáticas extremas, mejorando la preparación y respuesta a eventos meteorológicos adversos.

## Repositorio:

[vengeance422/WeatherWiseMarkov \(github.com\)](https://github.com/vengeance422/WeatherWiseMarkov)

## Código Fuente:

Salvador P

2023-11-14

```
library(markovchain)

Package: markovchain
Version: 0.9.4
Date: 2023-08-31 10:30:02 UTC
BugReport: https://github.com/spedygiorgio/markovchain/issues
```

```
library(ggplot2)
```

##Se tomaron los datos del estado de clima de los meses de Enero y Noviembre del 2023 en Saltillo, Coahuila, siendo los datos los siguientes. 334 en total.

```
Estados:
0: soleado
1: nublado
2: lluvioso
3: tormenta
datos = c(0,1,1,0,0,0,0,1,1,1,0,1,1,0,0,0,0,0,1,2,1,1,1,0,1,1,1,1,0,0,
1,1,0,0,1,0,1,1,0,0,0,2,1,0,1,0,0,1,1,0,1,1,1,1,1,0,1,1,
0,0,0,0,1,0,1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,0,1,0,0,0,1,1,1,1,1,
2,1,1,2,1,1,1,1,1,0,1,1,1,1,1,2,0,0,1,0,1,1,1,1,1,1,3,2,1,1,
1,1,0,1,1,2,1,2,1,2,0,3,2,2,2,2,1,3,3,1,1,1,2,2,2,1,1,3,2,3,2,
0,2,2,2,2,1,0,0,0,2,0,0,1,1,2,0,1,1,1,2,0,2,2,0,0,1,2,2,2,2,
2,1,2,2,2,2,2,2,2,0,2,2,0,2,2,3,0,2,2,2,2,2,1,1,1,2,2,3,1,0,
2,3,2,2,0,0,0,0,2,1,1,2,2,2,2,2,2,2,3,2,2,1,1,1,2,2,2,1,1,1,2,
2,3,2,1,1,1,1,1,0,1,1,2,1,2,1,2,1,1,1,1,2,2,0,2,2,1,1,2,0,2,
2,2,2,2,2,1,1,1,1,1,1,1,1,2,0,0,2,2,0,1,2,2,3,0,0,0,1,0,0,0,0,0,
2,0,0,1,0,0,0,0,1,1,1,1,1,1,0,0,1,0,0,0,0,0,0,0,0,2,1,1,2,1,0)
```

#Definición de la matriz de transición

$$P = \begin{bmatrix} P(\text{Soleado}|\text{Soleado}) & P(\text{Nublado}|\text{Soleado}) & P(\text{Lluvioso}|\text{Soleado}) & P(\text{Tormenta}|\text{Soleado}) \\ P(\text{Soleado}|\text{Nublado}) & P(\text{Nublado}|\text{Nublado}) & P(\text{Lluvioso}|\text{Nublado}) & P(\text{Tormenta}|\text{Nublado}) \\ P(\text{Soleado}|\text{Lluvioso}) & P(\text{Nublado}|\text{Lluvioso}) & P(\text{Lluvioso}|\text{Lluvioso}) & P(\text{Tormenta}|\text{Lluvioso}) \\ P(\text{Soleado}|\text{Tormenta}) & P(\text{Nublado}|\text{Tormenta}) & P(\text{Lluvioso}|\text{Tormenta}) & P(\text{Tormenta}|\text{Tormenta}) \end{bmatrix}$$

# Donde P(X/Y) representa la probabilidad de transición del estado Y al estado X.

Después de la recopilación de datos hacemos un conteo de los posibles cambios de estados partiendo de un día soleado.

*##Se contarán las veces que siga soleado al día siguiente,  
##Las veces que de soleado cambió a nublado, de soleado a ##Lluvioso y de soleado a tormenta.*

```
conteo0 = c(0, 0, 0, 0)
for(i in c(11:334)){
 if(datos[i-1]==0){
 if(datos[i]==0) conteo0[1]=conteo0[1]+1
 if(datos[i]==1) conteo0[2]=conteo0[2]+1
 if(datos[i]==2) conteo0[3]=conteo0[3]+1
 if(datos[i]==3) conteo0[4]=conteo0[4]+1
 }
}
```

Se hace el mismo conteo para los demás cambios de estados.

```
conteo1 = c(0, 0, 0, 0)
for(i in c(11:334)){
 if(datos[i-1]==1){
 if(datos[i]==0) conteo1[1]=conteo1[1]+1
 if(datos[i]==1) conteo1[2]=conteo1[2]+1
 if(datos[i]==2) conteo1[3]=conteo1[3]+1
 if(datos[i]==3) conteo1[4]=conteo1[4]+1
 }
}
```

```
conteo2 = c(0, 0, 0, 0)
for(i in c(11:334)){
 if(datos[i-1]==2){
 if(datos[i]==0) conteo2[1]=conteo2[1]+1
 if(datos[i]==1) conteo2[2]=conteo2[2]+1
 if(datos[i]==2) conteo2[3]=conteo2[3]+1
 if(datos[i]==3) conteo2[4]=conteo2[4]+1
 }
}
```

```
conteo3 = c(0, 0, 0, 0)
for(i in c(11:334)){
 if(datos[i-1]==3){
 if(datos[i]==0) conteo3[1]=conteo3[1]+1
 if(datos[i]==1) conteo3[2]=conteo3[2]+1
 if(datos[i]==2) conteo3[3]=conteo3[3]+1
 if(datos[i]==3) conteo3[4]=conteo3[4]+1
 }
}
```

##Despues normaliza los conteos de transición para cada estado climático. Línea por línea:

##n = sum(conteo0): Calcula la suma de los conteos de transición para el estado 0 (soleado). Es decir, suma las veces que el estado 0 cambió a otros estados.

##conteo0[1]=conteo0[1]/n: Normaliza la frecuencia de transición del estado 0 al estado 0 (soleado a soleado) dividiendo el conteo de esta transición por el total de transiciones desde el estado 0.

##conteo0[2]=conteo0[2]/n: Hace lo mismo que el paso anterior pero para la transición del estado 0 al estado 1 (soleado a nublado).

##conteo0[3]=conteo0[3]/n: Normaliza la frecuencia de transición del estado 0 al estado 2 (soleado a lluvioso).

##conteo0[4]=conteo0[4]/n: Normaliza la frecuencia de transición del estado 0 al estado 3 (soleado a tormenta).

```
n = sum(conteo0)
conteo0[1]=conteo0[1]/n
conteo0[2]=conteo0[2]/n
conteo0[3]=conteo0[3]/n
conteo0[4]=conteo0[4]/n
```

```
n = sum(conteo1)
conteo1[1]=conteo1[1]/n
conteo1[2]=conteo1[2]/n
conteo1[3]=conteo1[3]/n
conteo1[4]=conteo1[4]/n
```

```
n = sum(conteo2)
conteo2[1]=conteo2[1]/n
conteo2[2]=conteo2[2]/n
conteo2[3]=conteo2[3]/n
conteo2[4]=conteo2[4]/n
```

```
n = sum(conteo3)
conteo3[1]=conteo3[1]/n
conteo3[2]=conteo3[2]/n
conteo3[3]=conteo3[3]/n
conteo3[4]=conteo3[4]/n
```

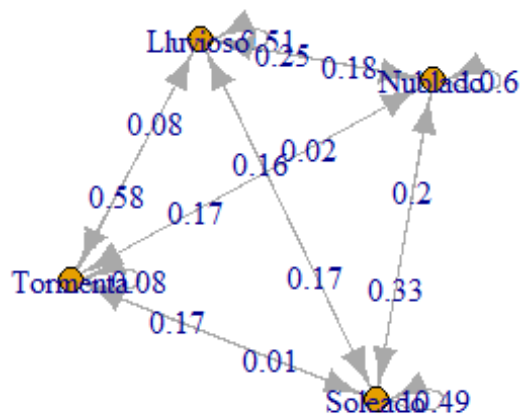
##Sacamos los porcentajes dada la siguiente formula:  $p(x) = \text{conteos}/n$

```
statesNames = c("Soleado", "Nublado", "Lluvioso", "Tormenta")
mc_p = new("markovchain", transitionMatrix = matrix(c(conteo0, conteo1, conteo2,
 conteo3), byrow=TRUE,
 nrow=4, dimnames=list(statesNames, statesNames)))
```

```
print(mc_p)
```

```
Soleado Nublado Lluvioso Tormenta
Soleado 0.4880952 0.3333333 0.1666667 0.01190476
Nublado 0.1985294 0.6029412 0.1764706 0.02205882
Lluvioso 0.1630435 0.2500000 0.5108696 0.07608696
Tormenta 0.1666667 0.1666667 0.5833333 0.08333333
```

```
plot(mc_p)
```



#Esta matriz muestra las probabilidades de transición entre los estados climáticos Soleado, Nublado, Lluvioso y Tormenta. Cada fila representa el estado inicial y cada columna representa la probabilidad de pasar a otro estado.

$$P = \begin{bmatrix} 0.488 & 0.333 & 0.167 & 0.012 \\ 0.199 & 0.603 & 0.176 & 0.022 \\ 0.163 & 0.250 & 0.511 & 0.076 \\ 0.076 & 0.167 & 0.583 & 0.083 \end{bmatrix}$$

#Hemos creado una función para aleatoriamente con las probabilidades cargadas, se estima el clima del siguiente día.

```
estimarClima <- function(climaAnterior){
 aleatorio = runif(1)
 climaAnterior = climaAnterior+1
 if(aleatorio<sum(mc_p[climaAnterior][1:1])){
 return(0)
 }
 else if(aleatorio<sum(mc_p[climaAnterior][1:2])){
 return(1)
 }
 else if(aleatorio<sum(mc_p[climaAnterior][1:3])){
 return(2)
 }
 else if(aleatorio<sum(mc_p[climaAnterior][1:4])){
 return(3)
 }
}
```

##Se supondra del estado que empieze, calcular aleatoriamente con las probabilidades de la cadena de markov, (Este no siempre será el mismo).

**##Recuerda que:**

**## 0: Soleado**

**## 1: Nublado**

**## 2: Lluvioso**

**## 3: Tormenta**

```
print(estimarClima(0))
```

```
[1] 0
```

```
print(estimarClima(1))
```

```
[1] 2
```

```
print(estimarClima(2))
```

```
[1] 0
```

```
print(estimarClima(3))
```

```
[1] 2
```

*# Crear una función para simular La cadena de Markov por un número específico de días usando temporalidad.*

```
simularClima <- function(dias, climaInicial = 0) {
 climaActual <- climaInicial
 historial <- c(climaActual)

 for (i in 1:(dias - 1)) { #Bucle para asegurar que la longitud sea 'dias'.
 climaActual <- estimarClima(climaActual)
 historial <- c(historial, climaActual)
 }

 return(historial)
}
```

*# Utilizar La función para simular 31 días de clima basado en La cadena de Markov usando temporalidad:*

```
historialSimuladoSoleado <- simularClima(31, climaInicial = 0)
```

```
print("Suponiendo que hoy ha sido un día soleado:")
```

```
[1] "Suponiendo que hoy ha sido un día soleado:"
```

```
print(historialSimuladoSoleado)
```

```
[1] 0 2 1 2 2 2 1 2 0 1 1 2 2 2 2 1 1 1 2 2 2 2 2 1 1 0 1 0 0 0 0
```

```
historialSimuladoNublado <- simularClima(31, climaInicial = 1)
```

```
print("Suponiendo que hoy ha sido un día nublado:")
```

```
[1] "Suponiendo que hoy ha sido un día nublado:"
```

```
print(historialSimuladoNublado)
```

```
[1] 1 1 1 1 2 0 0 1 0 0 0 3 3 2 1 1 0 0 1 3 2 2 1 0 1 0 0 0 0 0 1
```



```

historialSimuladoLluvioso <- simularClima(31, climaInicial = 2)
print("Suponiendo que hoy ha sido un día lluvioso:")

[1] "Suponiendo que hoy ha sido un día lluvioso:"

print(historialSimuladoLluvioso)

[1] 2 2 3 2 2 2 1 1 1 1 1 1 0 3 2 2 2 0 0 2 0 0 0 1 1 1 2 2 0 0 1

historialSimuladoTormenta <- simularClima(31, climaInicial = 3)
print("Suponiendo que hoy ha sido un día de tormenta:")

[1] "Suponiendo que hoy ha sido un día de tormenta:"

print(historialSimuladoTormenta)

[1] 3 2 0 2 0 1 2 1 1 2 2 1 0 0 1 0 0 0 0 0 1 1 2 2 2 1 0 0 0 0 1

Función para obtener el conteo con todos los estados
obtenerConteo <- function(historial) {
 estados <- c(0, 1, 2, 3)
 conteo <- table(factor(historial, levels = estados))
 conteo[as.character(estados)] <- conteo[as.character(estados)]
 return(conteo)
}

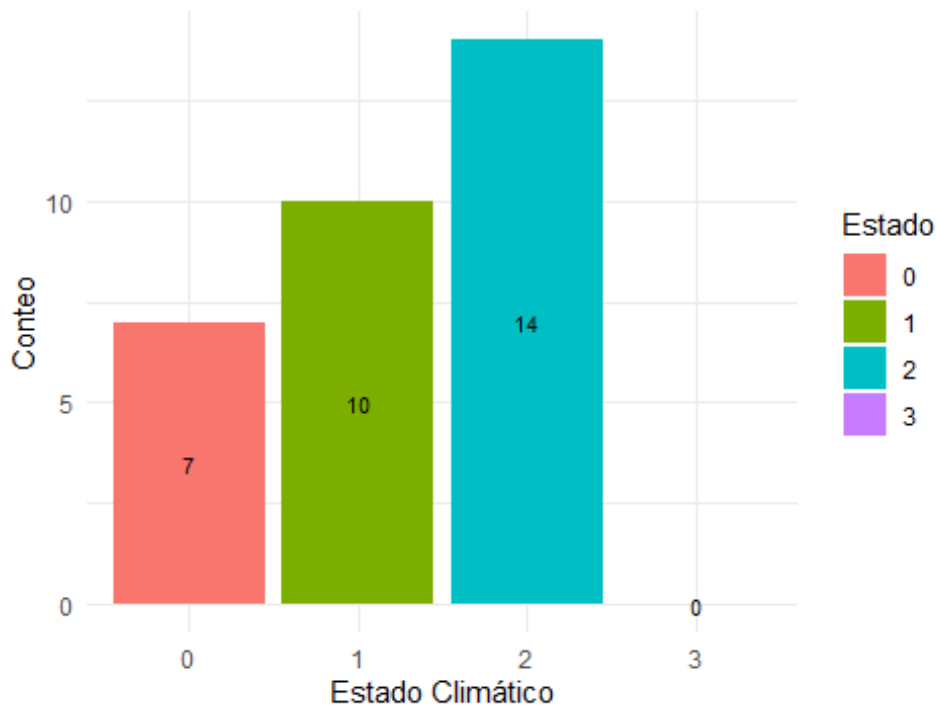
Función para obtener el gráfico con todos los estados
obtenerConteoYPlot <- function(historial, titulo) {
 conteo <- obtenerConteo(historial)
 datosConteo <- data.frame(
 Estado = as.factor(names(conteo)),
 Conteo = as.numeric(conteo)
)
 ggplot(datosConteo, aes(x = Estado, y = Conteo, fill = Estado)) +
 geom_bar(stat = "identity") +
 geom_text(aes(label = Conteo), position = position_stack(vjust = 0.5), size
= 3) +
 labs(title = titulo, x = "Estado Climático", y = "Conteo") +
 theme_minimal()
}

Obtener conteo y plotear para cada simulación
plotSoleado <- obtenerConteoYPlot(historialSimuladoSoleado, "Simulación de si el
día empieza en soleado; Estado 0")
plotNublado <- obtenerConteoYPlot(historialSimuladoNublado, "Simulación de si el
día empieza en nublado; Estado 1")
plotLluvioso <- obtenerConteoYPlot(historialSimuladoLluvioso, "Simulación de si
empieza en día lluvioso; Estado 2")
plotTormenta <- obtenerConteoYPlot(historialSimuladoTormenta, "Simulación de si
empieza en día de tormenta; Estado 3")

Mostrar Los gráficos
print(plotSoleado)

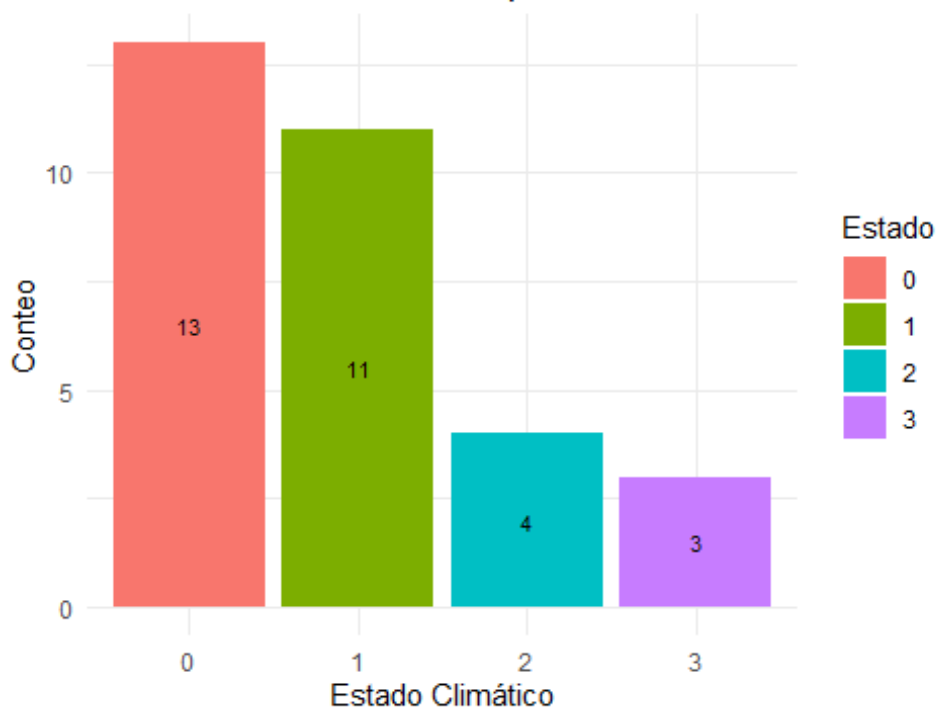
```

Simulación de si el día empieza en soleado; Estado 0



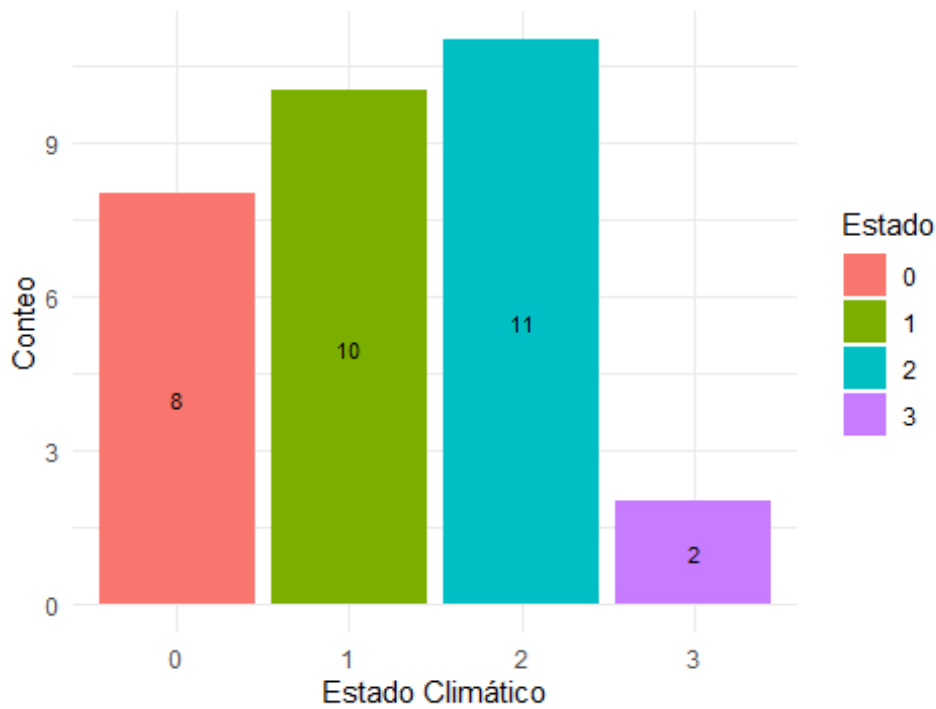
```
print(plotNublado)
```

Simulación de si el día empieza en nublado; Estado 1



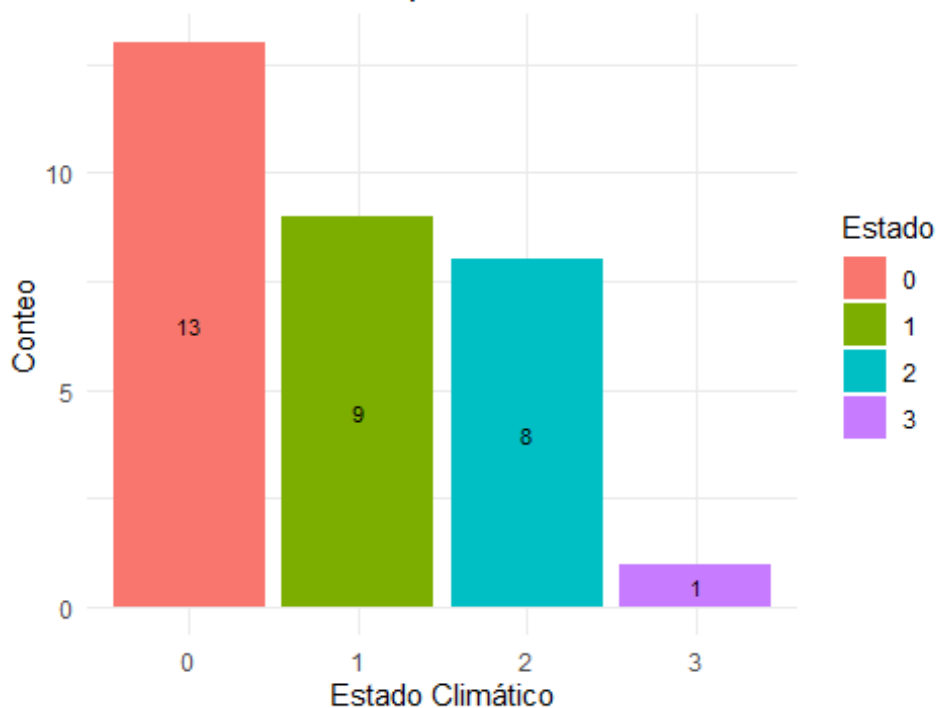
```
print(plotLluvioso)
```

## Simulación de si empieza en día lluvioso; Estado 2



```
print(plotTormenta)
```

## Simulación de si empieza en día de tormenta; Estado 3



```
Función para obtener el conteo con todos los estados
obtenerConteo <- function(historial) {
 # Crear un vector con los estados posibles:
 estados <- c(0, 1, 2, 3)
```

```

Obtener el conteo para cada estado:
conteo <- table(factor(historial, levels = estados))

Convertir a un vector con todos los estados:
conteo[as.character(estados)] <- conteo[as.character(estados)]
return(conteo)
}

Función para obtener el gráfico con todos los estados:
obtenerConteoYPlot <- function(historial, titulo) {
 conteo <- obtenerConteo(historial)

 # Crear un marco de datos con todos los estados:
 datosConteo <- data.frame(
 Estado = as.factor(names(conteo)),
 Conteo = as.numeric(conteo)
)

 # Trazar el gráfico con todos los estados
 ggplot(datosConteo, aes(x = Estado, y = Conteo, fill = Estado)) +
 geom_bar(stat = "identity") +
 geom_text(aes(label = Conteo), position = position_stack(vjust = 0.5), size
= 3) +
 labs(title = titulo,
 x = "Estado Climático",
 y = "Conteo") +
 theme_minimal()
}

Obtener el conteo para cada simulación
conteoSoleado <- obtenerConteo(historialSimuladoSoleado)
conteoNublado <- obtenerConteo(historialSimuladoNublado)
conteoLluvioso <- obtenerConteo(historialSimuladoLluvioso)
conteoTormenta <- obtenerConteo(historialSimuladoTormenta)

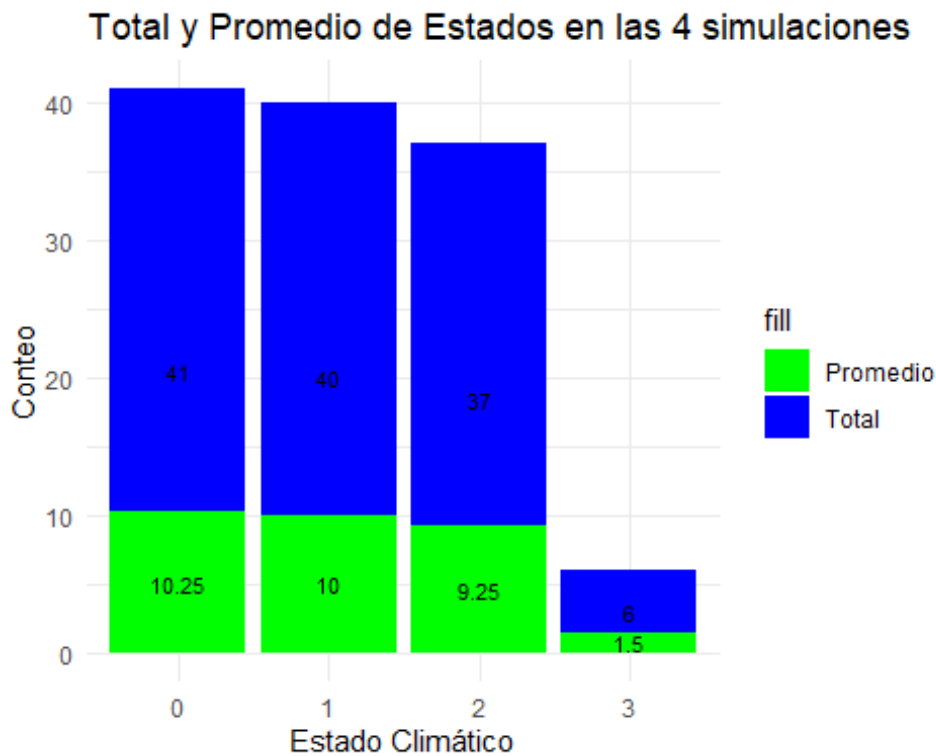
Calcular el total y promedio para todos los estados
totalEstados <- conteoSoleado + conteoNublado + conteoLluvioso + conteoTormenta
promedioEstados <- totalEstados / 4

Crear un marco de datos con todos los estados para el gráfico
datosTotales <- data.frame(
 Estado = as.factor(names(totalEstados)),
 Total = as.numeric(totalEstados),
 Promedio = as.numeric(promedioEstados)
)

Trazar el gráfico con todos los estados
ggplot(datosTotales, aes(x = Estado)) +
 geom_bar(aes(y = Total, fill = "Total"), stat = "identity") +
 geom_bar(aes(y = Promedio, fill = "Promedio"), stat = "identity") +
 geom_text(aes(y = Total, label = Total), position = position_stack(vjust = 0.5
), size = 3) +

```

```
geom_text(aes(y = Promedio, label = round(Promedio, 2)), position = position_s
tack(vjust = 0.5), size = 3) +
labs(title = "Total y Promedio de Estados en las 4 simulaciones",
x = "Estado Climático",
y = "Conteo") +
theme_minimal() +
scale_fill_manual(values = c("Total" = "blue", "Promedio" = "green"))
```



## Conclusión:

Las cadenas de Markov, un modelo estocástico que describe la transición de un estado a otro dentro de un sistema, han sido un pilar fundamental en la modelización de una amplia gama de fenómenos. El código implementado aquí, utilizando este enfoque, nos permitió simular y analizar el comportamiento del clima a través de diferentes estados climáticos: soleado, nublado, lluvioso y de tormenta. Estos estados, definidos en nuestra simulación, ofrecieron una representación simplificada pero poderosa de las fluctuaciones climáticas. La estructura de las cadenas de Markov nos permitió modelar las probabilidades de transición entre estos estados, reflejando la idea central de que el estado futuro depende únicamente del estado actual y no del historial pasado. A través del código, pudimos calcular y graficar el conteo de días en cada estado, así como el total y promedio de estados en múltiples simulaciones. Estos análisis proporcionaron una comprensión cuantitativa de la distribución y la frecuencia de los diferentes estados climáticos, ofreciendo una visión detallada del comportamiento simulado. Sin embargo, durante la implementación del código, surgieron algunas complicaciones. Uno de los problemas identificados fue la falta de definición de la función **obtenerConteo**. Esta función fue esencial para calcular el conteo de días en cada estado, y su falta generó errores en la ejecución del código. La resolución de este problema implicó definir y corregir la función para asegurar un cálculo preciso del conteo.

Además, se encontraron desafíos al generar los gráficos de barras que representan los conteos de cada estado. La generación adecuada de estos gráficos requirió una manipulación cuidadosa de los datos y la correcta utilización de la librería `ggplot2`. Al ajustar el código para reflejar las estructuras de datos adecuadas y los mapeos apropiados en el gráfico, pudimos obtener visualizaciones claras y comprensibles de los conteos de cada estado climático.

A pesar de estas complicaciones, los resultados obtenidos fueron significativos. Las simulaciones nos proporcionaron información valiosa sobre la distribución probabilística de los estados climáticos a lo largo del tiempo. Además, los gráficos generados nos permitieron visualizar de manera efectiva cómo estos estados se distribuyeron en las múltiples simulaciones, destacando las variaciones y tendencias en los diferentes escenarios climáticos.

En conclusión, general, las cadenas de Markov y su implementación a través del código analizado ofrecen un enfoque poderoso y flexible para modelar y comprender sistemas que exhiben comportamientos probabilísticos y de transición entre estados. A pesar de los desafíos en la implementación, la capacidad de estos modelos para proporcionar percepciones cuantitativas sobre la dinámica de los estados y sus transiciones es invaluable. Esta metodología y su aplicación práctica nos permiten explorar y comprender mejor una amplia gama de fenómenos que cambian con el tiempo, proporcionando una base sólida para análisis y predicciones en diversos campos, desde las ciencias naturales hasta la economía y la ingeniería.

## Bibliografías (Formato APA):

- Meyn, S., Tweedie, R. L., & Glynn, P. W. (2009). *Markov Chains and stochastic stability*. <https://doi.org/10.1017/cbo9780511626630>
- Brémaud, P. (1999). Markov Chains. En *Texts in applied mathematics*. <https://doi.org/10.1007/978-1-4757-3124-8>
- Khiatani, D., & Ghose, U. (2017, October). Weather forecasting using hidden Markov model. In *2017 International Conference on Computing and Communication Technologies for Smart Nation (IC3TSN)* (pp. 220-225). IEEE.
- *Datos meteorológicos*. (s. f.). tiempo3. Recuperado 14 de noviembre de 2023, de <https://www.tiempo3.com/north-america/mexico/coahuila/salttillo?page=month&month=December>