Probabilidad y estadística con R

ldurazo

28/11/20

```
# Libraries.
chooseCRANmirror(ind = 52)
install.packages(c("poisbinom", "purrr"))
```

```
##
## The downloaded binary packages are in
##  /var/folders/f3/2p9snhhj759g5l1f2ztjbwdw0000gn/T//Rtmph1F34a/downloaded_packages
```

```
library(poisbinom)
library(purrr)
```

Esta libreta de R contiene una serie de ejercicios con el fin de practicar lo aprendido en la materia de matemáticas para ciencia de datos (probabilidad y estadística) en el lenguaje R.

Problema 1: distribución binomial. Un equipo de desarrollo con un sistema de integración continua está experimentando fallas esporádicas en sus pruebas unitarias. Se sabe que hay una condición de carrera que aunque difícil de detectar, ocurre con una probabilidad de 7.5% en cada corrida.

1) Con qué probabilidad fallará 3 veces seguidas cuando Luis quiera subir sus cambios al repositorio, después de intentar subir 3 cambios?

   P(x = 3) si n = 3 y p = 0.075

```
   # 1)
   dbinom(3, 3, 0.075)
```

```
## [1] 0.000421875
```

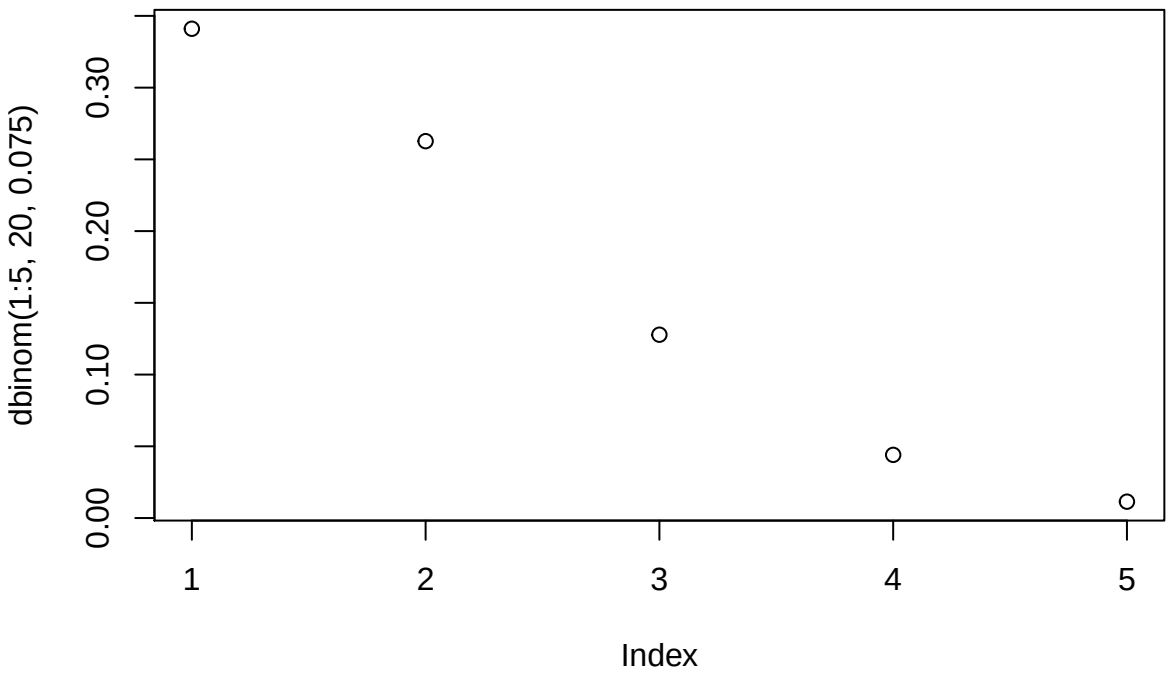
2) Con qué probabilidad fallará entre 1 y 5 veces seguidas cuando Luis quiera subir sus cambios al repositorio, después de intentar subir 20 cambios?

   P(1 <= x <= 5) si n = 20 y p = 0.075

```
   # 2)
   dbinom(1:5, 20, 0.075)
```

```
## [1] 0.34102340 0.26268019 0.12779036 0.04403587 0.01142552
```

```r
plot(dbinom(1:5, 20, 0.075))
```




```r
pbinom(0, 20, 0.075, lower.tail = FALSE) - pbinom(4, 20, 0.075, lower.tail = FALSE)
```

```
## [1] 0.7755298
```

3) Con qué probabilidad no fallará ninguna vez, si en promedio, Luis ejecuta 10 construcciones en el contenedor integración continua en cada ciclo de desarrollo diario?

```r
# 3)
1 - dbinom(0, 10, 0.075)
```

```
## [1] 0.5414177
```

4) Simula como se vería para desarrollador en un equipo de 20 personas de manera aleatoria si cada uno ejecuta 10 construcciones sobre integración continua por día, cuantas fallas podríamos observar?

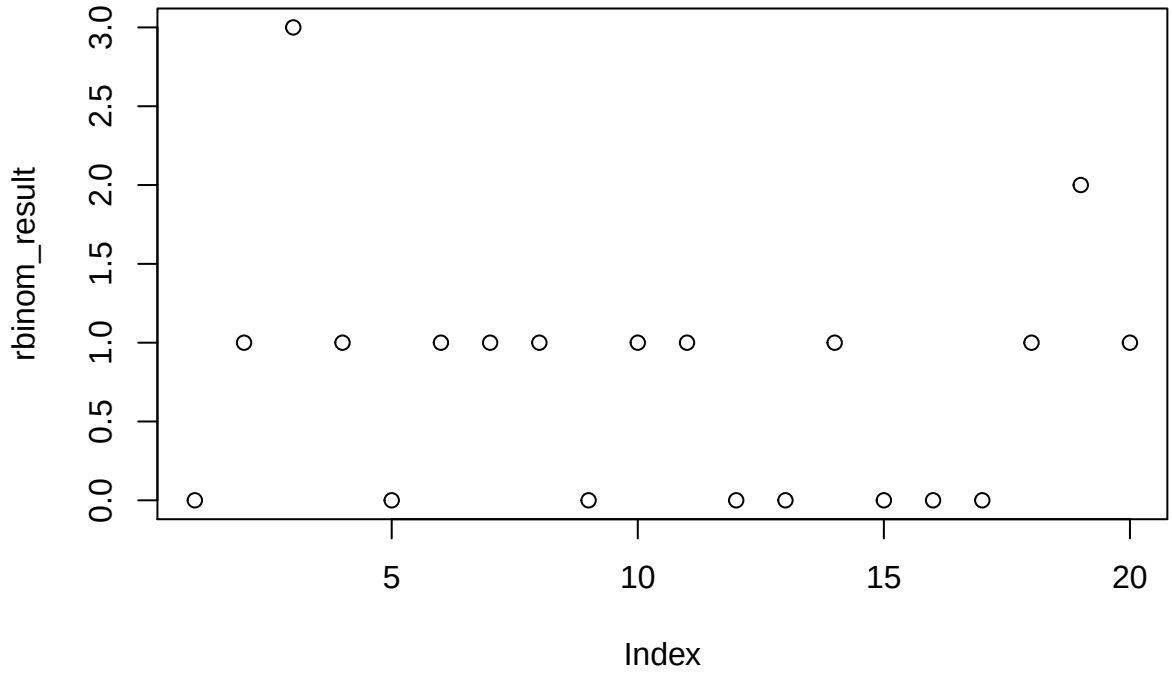
```r
# 4)
eventCount <- 0
# 20 desarrolladores
for(i in 1:20) {
  # Genera aleatoriamente 1 solo resultado de 10 ejecuciones de dicho desarrollador
    for(j in rbinom(1, 10, 0.075)) {
```

```
      eventCount <- eventCount + j
    }
  }
  print(eventCount)
```

```
## [1] 15
```

de una forma más simple, lo de arriba puede expresado de la siguiente manera:
genera resultados aleatorios para 20 (n) desarrolladores que ejecutan 10 (size) construcciones

```
  eventCount <- 0
  rbinom_result <- rbinom(20, 10, 0.075)
  plot(rbinom_result)
```



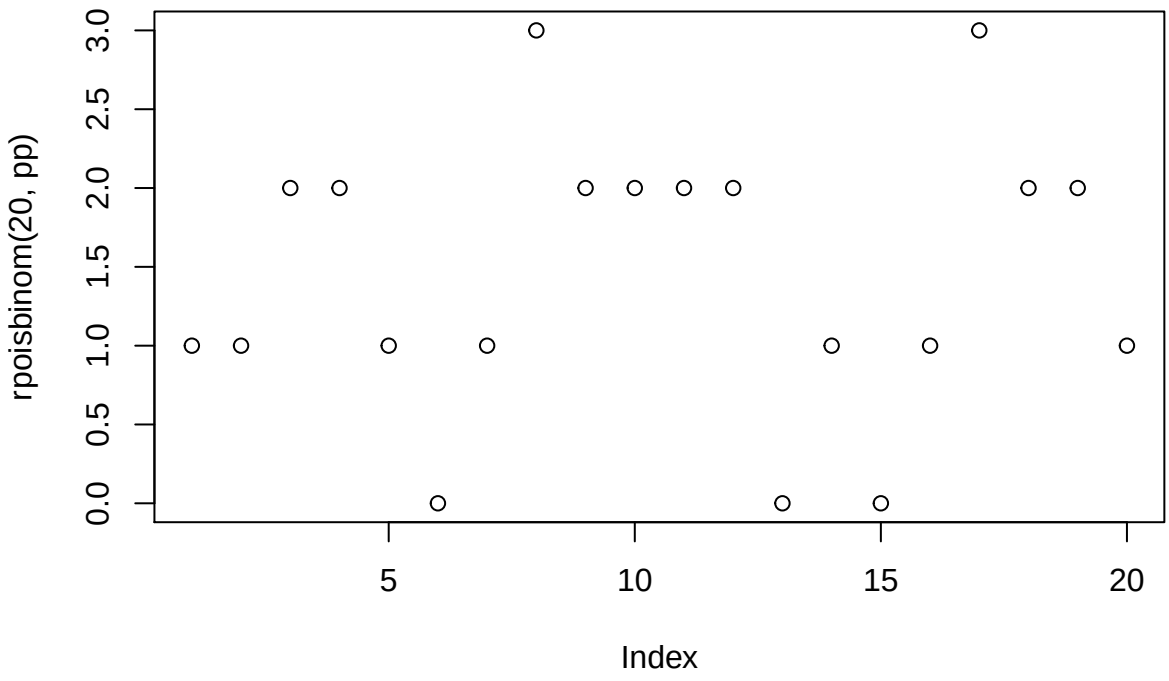

```
  for(i in rbinom_result) {
    eventCount <- eventCount + i
  }
  print(eventCount)
```

```
## [1] 15
```

Ambos resultados deben ser similares.

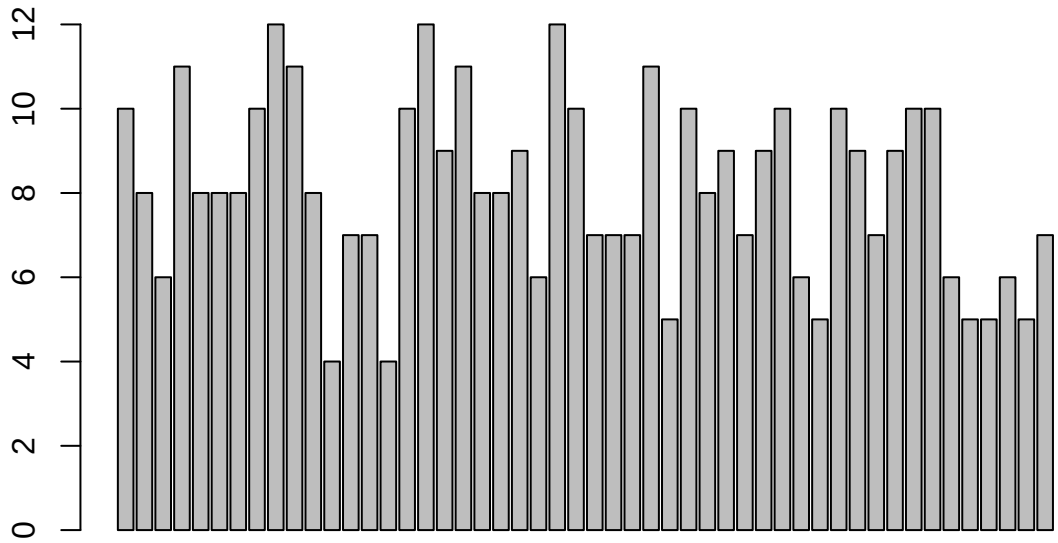
Sin embargo, esta no es la forma de sumar dos distribuciones binomiales, para hacer eso necesitaríamos l

```
pp <- vector()
for (i in rbinom_result) {
  p <- i / 10
  pp <- append(pp, p)
}

plot(rpoisbinom(20, pp))
```

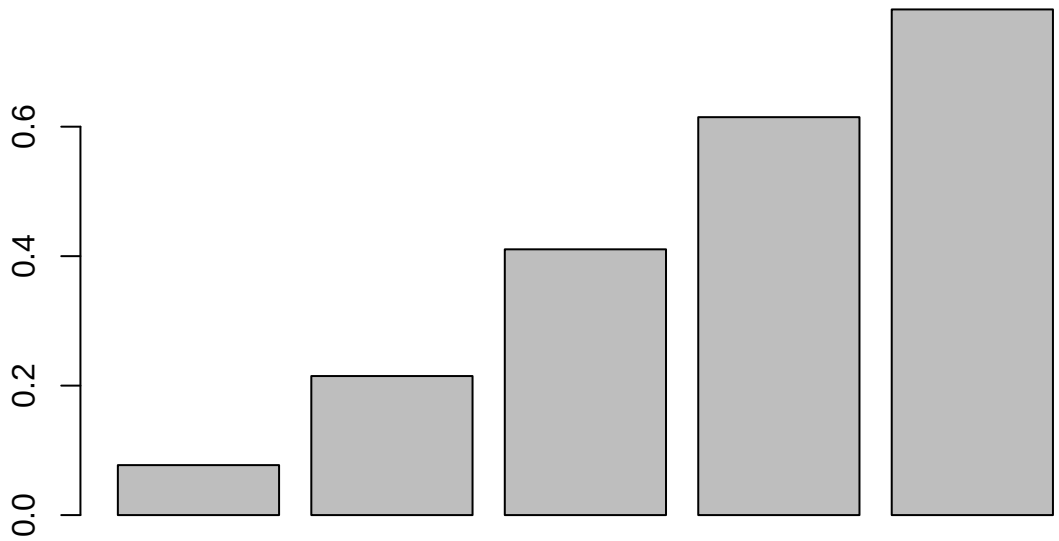


Lo anterior generado es uan distribución de desviaciones utilizando rpoisbinom, pero las muestras son muy pocas, incrementos los valores para observarlo de nuevo. Pongamos a 50 desarrolladores, a ejecutar 100 despliegues a integración continua por día.

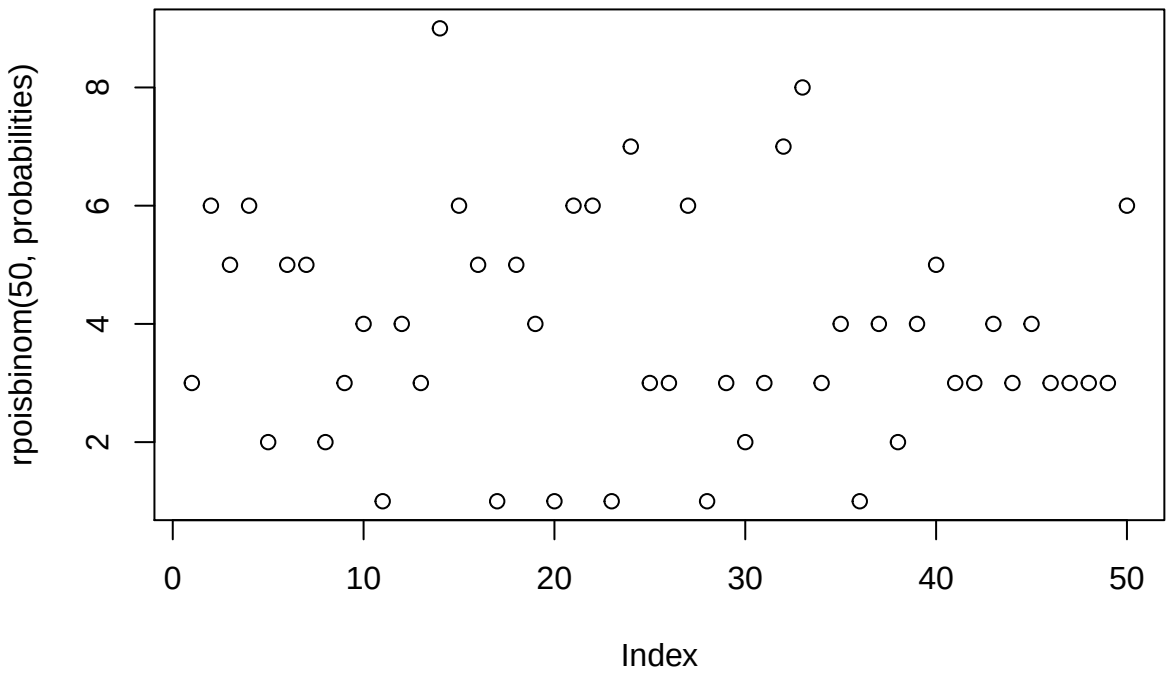
```
rbinom_result <- rbinom(50, 100, 0.075)
barplot(rbinom_result)
```



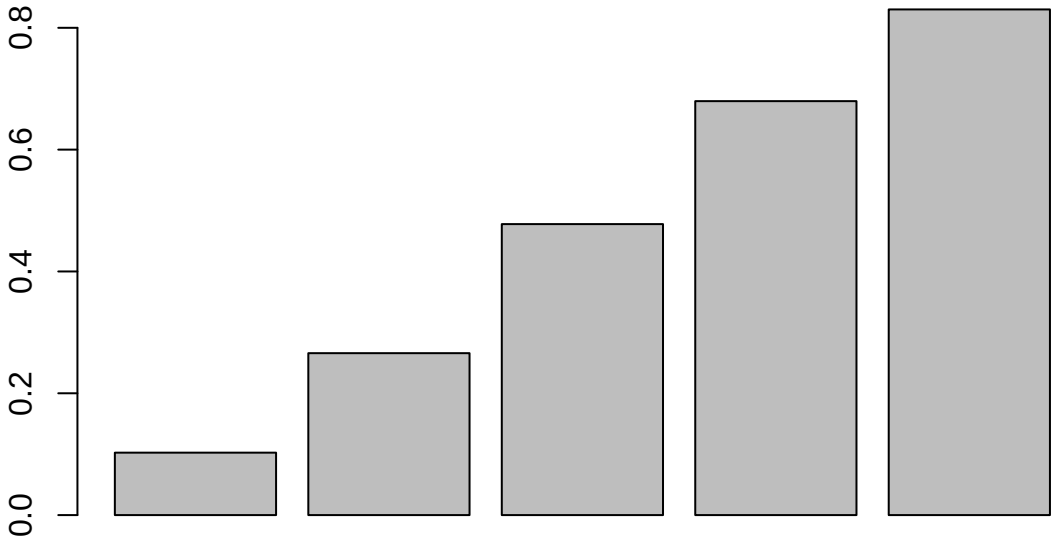

```
probabilities <- vector()
for (i in rbinom_result) {
  p <- i / 100
  probabilities <- append(probabilities, p)
}

barplot(ppoisbinom(1:5, probabilities))
```




```
plot(rpoisbinom(50, probabilities))
```




```
barplot(pbinom(1:5, 50, 0.075))
```



Lo anterior muestra la cumulativa para ambos casos de sumar un distinto número de eventos aleatorios (varias binomiales) versus la cumulativa para el cálculo tradicional de la distribución binomial.