Clase Modelado 1

2024-05-23

Modelos

```
# Corremos librerías
library(dplyr)
library(ggplot2)
library(purrr)
library(MASS)
library(fitdistrplus)
library(actuar)

# Datos
# Dataframe con la severidad de los siniestros
sevDatos <- data.frame(
  Rango = c("0-1000", "1000-3000", "3000-5000", "5000-10000", "+10000"),
  Frecuencia = c(79, 37, 4, 6, 7)
)

sevDatos
```

```
      Rango Frecuencia
1     0-1000         79
2  1000-3000         37
3  3000-5000          4
4 5000-10000          6
5     +10000          7
```

```
# Dataframe con la frecuencia de los siniestros
frecDatos <- data.frame(
```

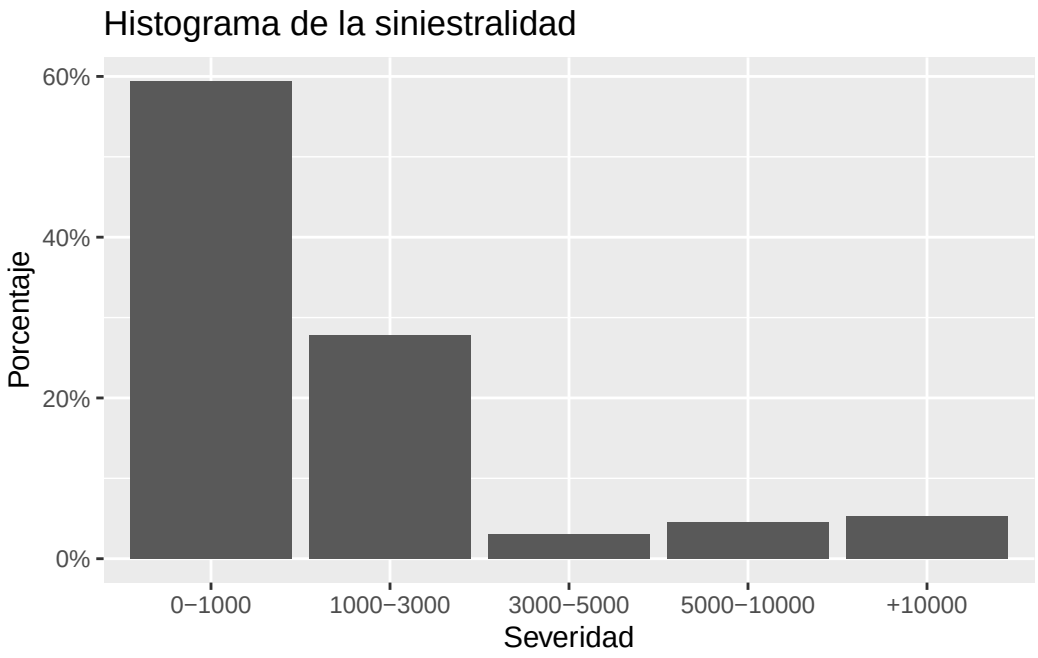
```
  Rango = c("0--1", "2--3", "3--4"),
  Completa = c(56, 12, 2)
)

frecDatos
```

```
  Rango Completa
1  0--1       56
2  2--3       12
3  3--4        2
```

Análisis exploratorio de datos

Siniestralidad

```
# Histograma de la severidad

# Calcular los porcentajes
sevDatos$Porcentaje <- sevDatos$Frecuencia / sum(sevDatos$Frecuencia) * 100

# Definir los niveles deseados en el orden correcto
niveles_deseados <- c("0-1000", "1000-3000", "3000-5000", "5000-10000", "+10000")

# Convertir Rango a factor con el orden deseado
sevDatos$Rango <- factor(sevDatos$Rango, levels = niveles_deseados)

# Crear la gráfica con ggplot2
library(ggplot2)
(plot1 <- ggplot(sevDatos, aes(x = Rango, y = Porcentaje)) +
  geom_bar(stat = "identity") +
  labs(title = "Histograma de la siniestralidad",
       x = "Severidad",
       y = "Porcentaje") +
  scale_y_continuous(labels = scales::percent_format(scale = 1)))
```



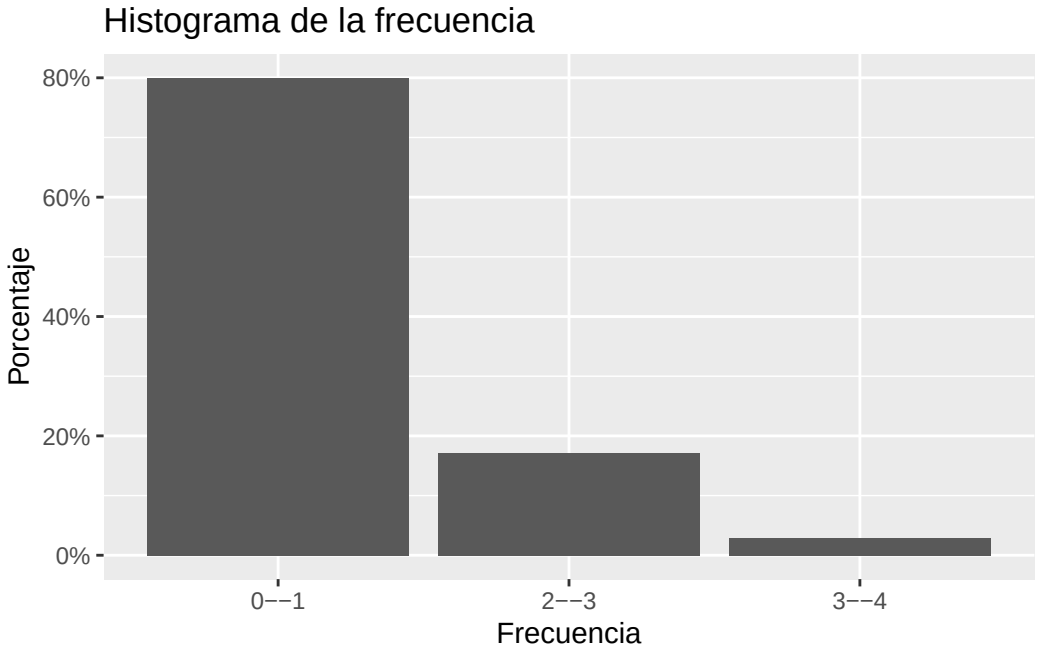

```
# Guardar el gráfico en un archivo JPG
ggsave(filename = "histograma_severidad_porc.jpg", plot = plot1, device = "jpeg")
```

Con lo cual notamos que para la siniestralidad de estos datos, los modelos que convienen deben de ser colas pesadas o medio pesadas, dado que se ve un ligero incremento final en la siniesralidad.

Frecuencia

```
# Histograma de la frecuencia

# Calcular los porcentajes
frecDatos$Porcentaje <- frecDatos$Completa / sum(frecDatos$Completa) * 100

# Definir los niveles deseados en el orden correcto

niveles_deseados <- c("0--1", "2--3", "3--4")

# Convertir Rango a factor con el orden deseado

frecDatos$Rango <- factor(frecDatos$Rango, levels = niveles_deseados)
```

```r
# Crear la gráfica con ggplot2

(plot2 <- ggplot(frecDatos, aes(x = Rango, y = Porcentaje)) +
  geom_bar(stat = "identity") +
  labs(title = "Histograma de la frecuencia",
       x = "Frecuencia",
       y = "Porcentaje") +
  scale_y_continuous(labels = scales::percent_format(scale = 1)))
```




```r
# Guardar el gráfico en un archivo JPG

ggsave(filename = "histograma_frecuencia_porc.jpg", plot = plot2, device = "jpeg")
```

```
Saving 5.5 x 3.5 in image
```

Dada la forma de los datos de frecuencia, los modelos que mejor pueden convenir son los modelos de Poisson o Binomial Negativa de la clase(a,b,0) para mantenerlo parsiarmónico.

Ajuste modelos de frecuencia

```
library(stats4)
# Definir los límites de los intervalos
intervalos <- list(c(-1, 1), c(1, 3), c(3,4))

# Crear una función de log-verosimilitud para la distribución de Poisson
logverosimilitud_poisson <- function(lambda) {
  result <- -sum(sapply(1:nrow(frecDatos), function(i) {
    frecDatos$Completa[i] * log(ppois(intervalos[[i]][2], lambda)-ppois(intervalos[[i]][1]
  }))
  return(result)
}

# Crear una función de log-verosimilitud para la distribución binomial negativa
logverosimilitud_negbin <- function(par) {
  size <- par[1]
  mu <- par[2]
  result <- -sum(sapply(1:nrow(frecDatos), function(i) {
    frecDatos$Completa[i] *log(pnbinom(intervalos[[i]][2], size = size, mu = mu)-pnbinom(i
  }))
  return(result)
}


# Ajuste a la distribución de Poisson
ajuste_poisson <- optimize(interval = c(.2,10), f = logverosimilitud_poisson)
(lambda_poisson <- ajuste_poisson$minimum)
```

```
[1] 0.8671189
```

```
# Ajuste a la distribución binomial negativa
ajuste_negbin <- optim(par = c(1, 1), fn = logverosimilitud_negbin)
size_negbin <- ajuste_negbin$par[1]
(mu_negbin <- ajuste_negbin$par[2])
```

```
[1] 0.8158993
```