Análisis Exploratorio de Datos

Francisco Paz

21/8/2019

```
library(pander)
library(fdth)
library(tidyverse)
```

Análisis Exploratorio de Datos

El libro por "excelencia" es 'Tukey, J. W. (1977). Exploratory Data Analysis. Massachusetts. Addison Wesley'. En el podemos encontart distintos ejemplos del uso del análisis exploratorio de datos

Como vimos la clase pasada, el conocer la materia prima con la que trabajamos puede y de hecho nos da una linea a seguir para obtener mayor información.

"The simple graph has brought more information to the data analyst's mind than any other device." — John Tukey

La idea es encontrar métodos/formas/trucos que nos ayuden a entender de manera más clara nuestros datos.

```
set.seed(649)
volado <- rbinom(10,1,0.5)
volado
```

```
##  [1] 1 0 1 1 1 0 0 1 0 0
```

```
table(volado)
```

```
## volado
## 0 1
## 5 5
```

En este caso vemos la funcionalidad de representar esta información en una tabla. Veremos algunos ejemplos en los que resumiremos y de esta forma probremos obtener más información.

Pensemos (de forma simple) en una población que probiene de una distribución $N(0, 1)$

```
set.seed(515)
datos1 <- rnorm(10)
pander(table(datos1))
```

Table 1: Table continues below

-2.01765898587435	-1.37058737817003	-0.480596092289985
1	1	1

Table 2: Table continues below

-0.345114132339731	0.300416041318492	0.474408785996932	0.844547488927244
1	1	1	1

0.949494030571598	0.951234362547441	1.13670069304068
1	1	1

Al contrario del ejemplo anterior organizar estos datos en una tabla resulta poco eficiente (si fueran más observaciones tendríamos n número de columnas) y por ello es más eficiente tener un rango en el cual podrían entrar las observaciones.

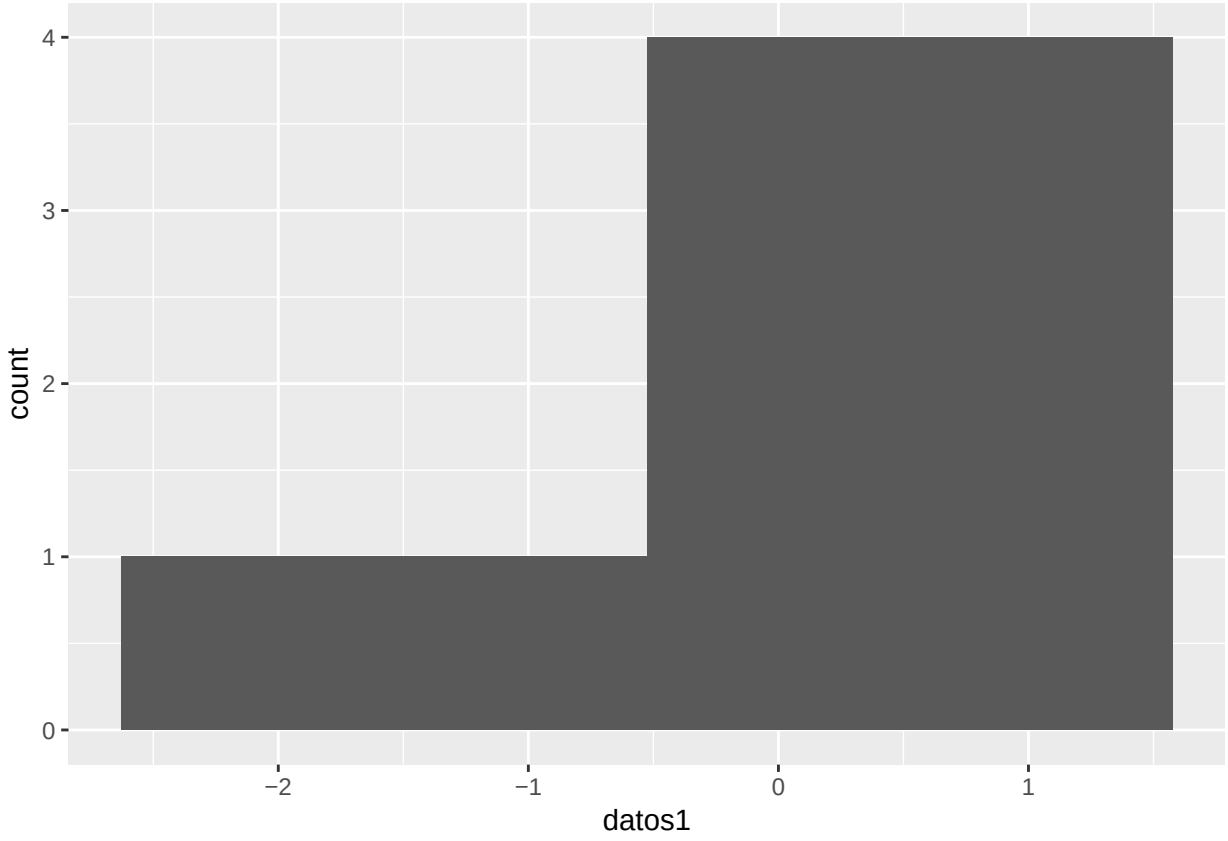
```
tabla_hist <- fdt(datos1,breaks="Sturges") # calcula la distribución de
#frecuencias utilizando la regla Sturges
tabla_hist
```

```
##       Class limits f  rf rf(%) cf cf(%)
##    [-2.038,-1.401) 1 0.1    10  1    10
##   [-1.401,-0.7635) 1 0.1    10  2    20
##  [-0.7635,-0.1263) 2 0.2    20  4    40
##   [-0.1263,0.5109) 2 0.2    20  6    60
##     [0.5109,1.148) 4 0.4    40 10   100
```

Sturges

$$k = 1 + (3.322 * log_{10}(n))$$

```
data1 <- as.data.frame(datos1)
#Mover el número de bins quizá 2,5,10,20,...
ggplot(data1,aes(datos1)) + geom_histogram(bins = 4)
```



¿Notas algo en el histograma?

Pero este ejemplo lo hicimos con pocos datos (10), ya que tenemos esta forma tan sencilla de tener la información, hagamoslo con un montón más

```
set.seed(4458)
datos <-  rnorm(10000,0,1)
```

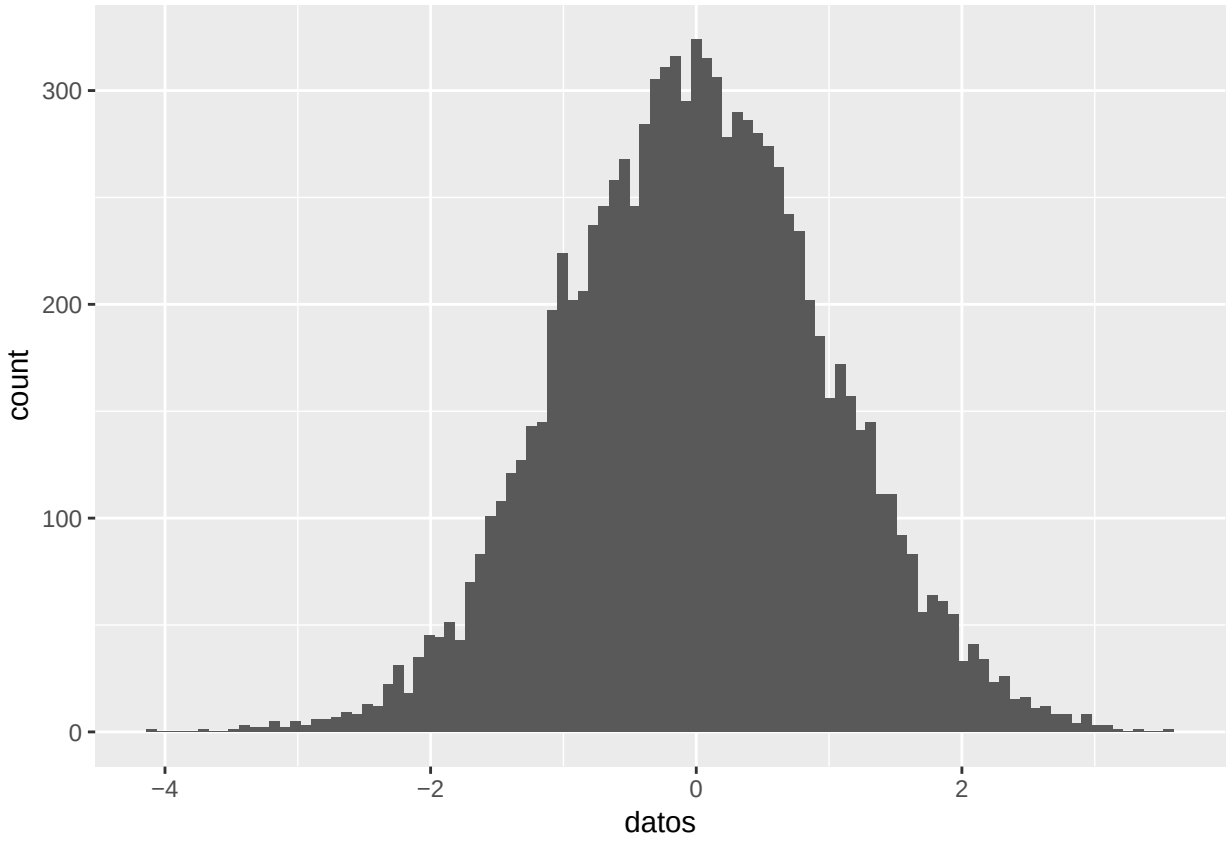
Análicemos la tabla

```
tabla_hist <- fdt(datos,breaks="Sturges")
tabla_hist
```

```
##        Class limits    f   rf rf(%)    cf  cf(%)
##     [-4.137,-3.621)    2 0.00  0.02     2   0.02
##     [-3.621,-3.106)   13 0.00  0.13    15   0.15
##      [-3.106,-2.59)   38 0.00  0.38    53   0.53
##      [-2.59,-2.074)  128 0.01  1.28   181   1.81
##     [-2.074,-1.559)  389 0.04  3.89   570   5.70
##     [-1.559,-1.043)  903 0.09  9.03  1473  14.73
##    [-1.043,-0.5273) 1544 0.15 15.44  3017  30.17
##  [-0.5273,-0.01164) 1969 0.20 19.69  4986  49.86
##    [-0.01164,0.504) 1965 0.20 19.65  6951  69.51
##        [0.504,1.02) 1496 0.15 14.96  8447  84.47
##        [1.02,1.535)  925 0.09  9.25  9372  93.72
##       [1.535,2.051)  413 0.04  4.13  9785  97.85
##       [2.051,2.567)  162 0.02  1.62  9947  99.47
##       [2.567,3.082)   48 0.00  0.48  9995  99.95
##       [3.082,3.598)    5 0.00  0.05 10000 100.00
```

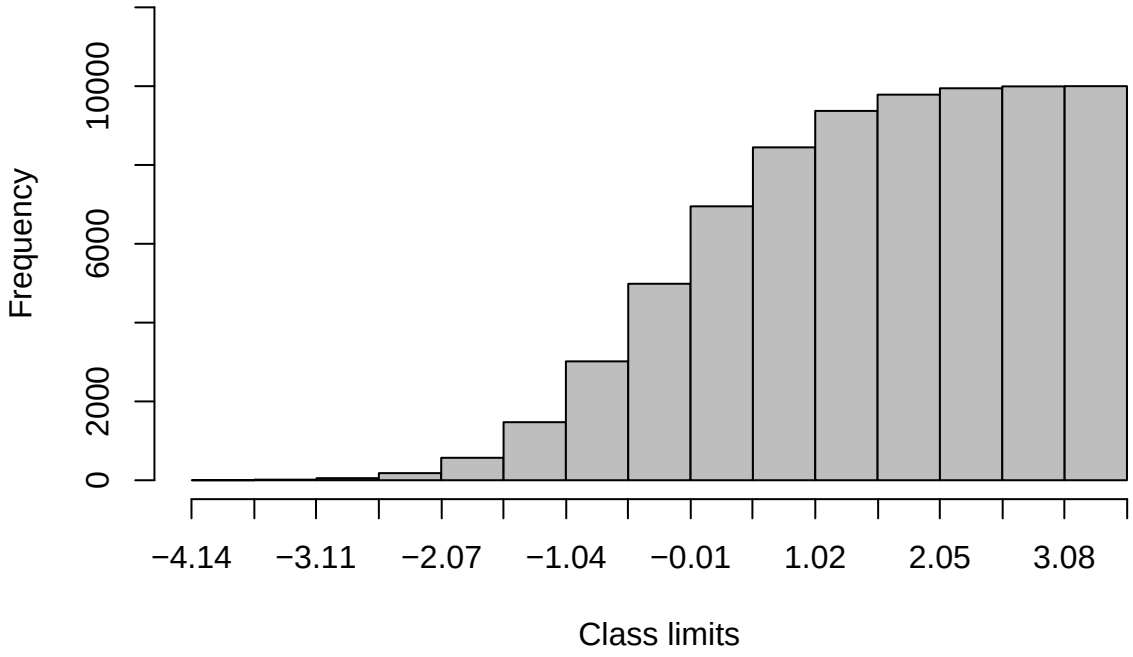
Hagamos un histograma y probemos que pasa con distinto número de bins

```
data <- as.data.frame(datos)
#Mover el número de bins quizá 2,5,10,20,...
ggplot(data,aes(datos)) + geom_histogram(bins = 100)
```

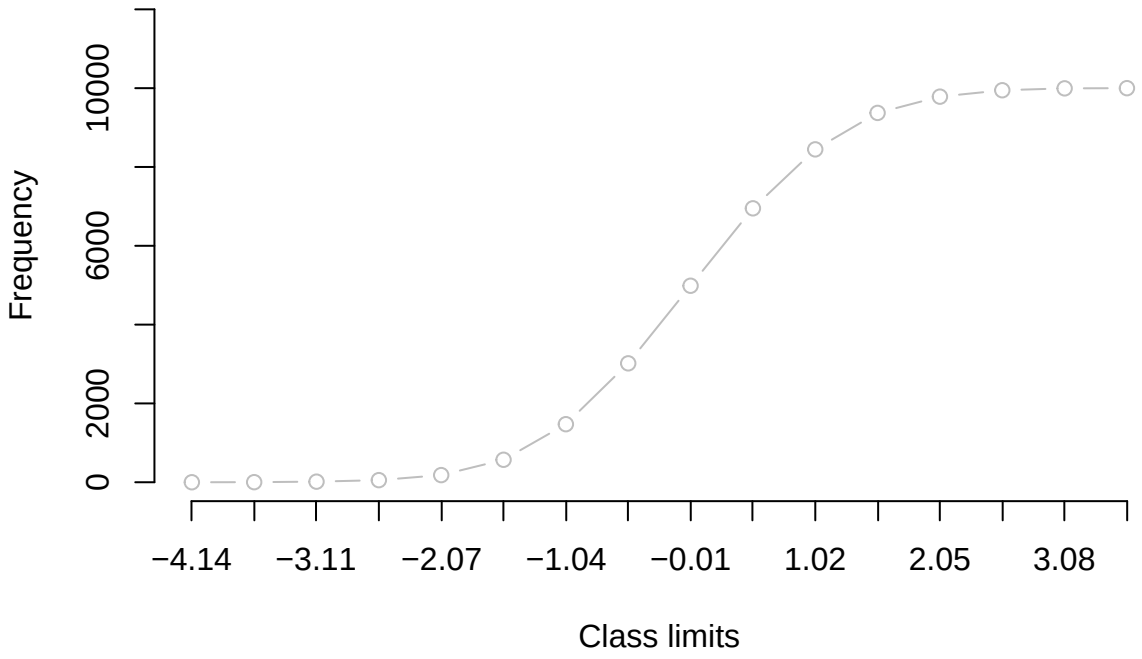


También podemos obtener el histograma y poligono de frecuencias acumuladas. Normalizando obtenemos ls Función de Distribución Acumulada (FDA).

```
plot(tabla_hist, type="cfh")
```




```
plot(tabla_hist, type="cfp")
```



Además de las tablas de frecuencia, los histogramas y las gráficas de frecuencia acumulada existen otras formas de conocer los datos. Siguiendo con nuestros datos generados de forma aleatoria siguiendo una distribución normal $N(0,1)$ definimos las siguientes funciones

```
prom <- function(x)
{
  sum(x)/length(x)
}

desv_est <- function(x){
  suma <- 0
  for (i in 1:length(x)) {
    suma = suma + (x[i]-prom(x))^2
  }
  return(sqrt(suma/length(x)))
}
```

Es importante recalcar que r ¡ya tiene implementadas estas funciones! aproche la sencilles de las mismas para mostrar como es que podemos definir funciones dentro de r.

```
prom(datos)
```

```
## [1] -0.002366132
```

```
desv_est(datos)
```

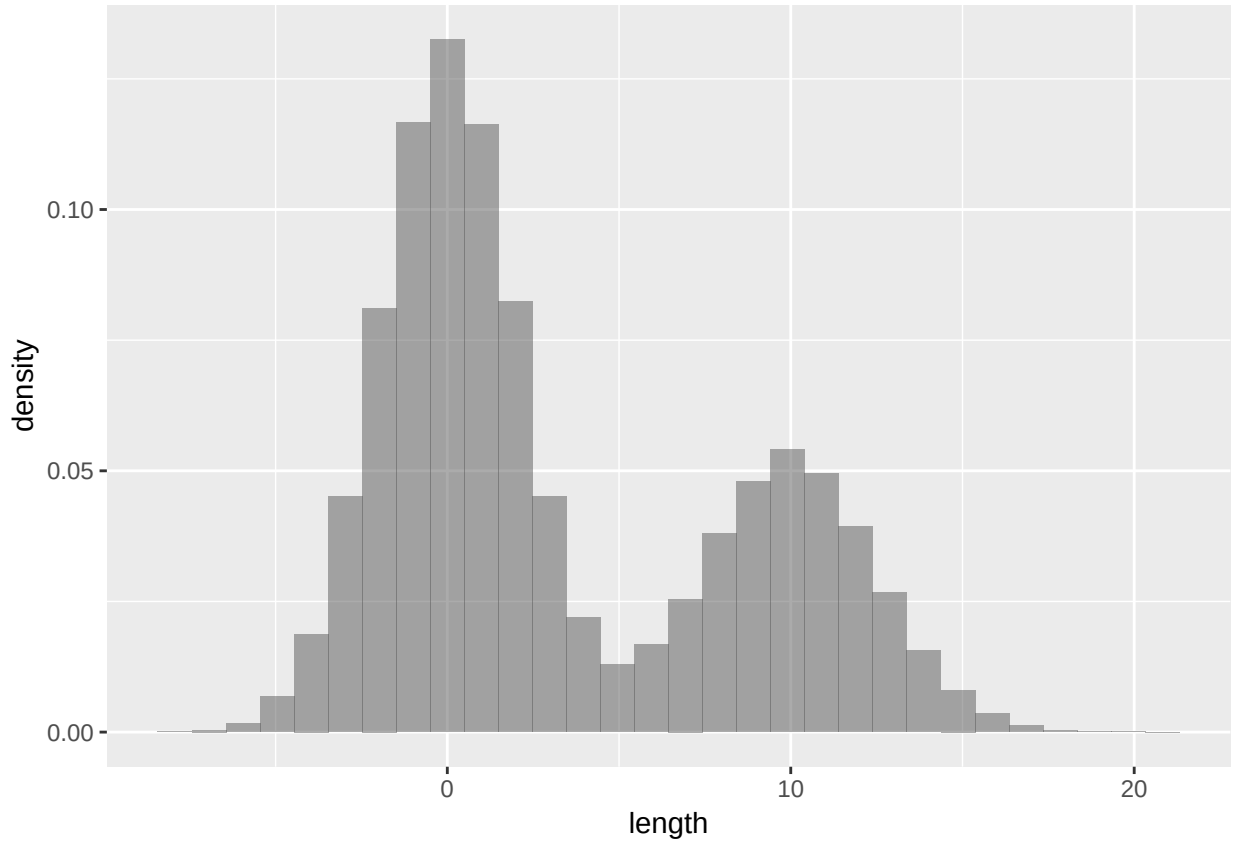
```
## [1] 0.9982666
```

Una forma más sencilla es utilizar el siguiente comando en r

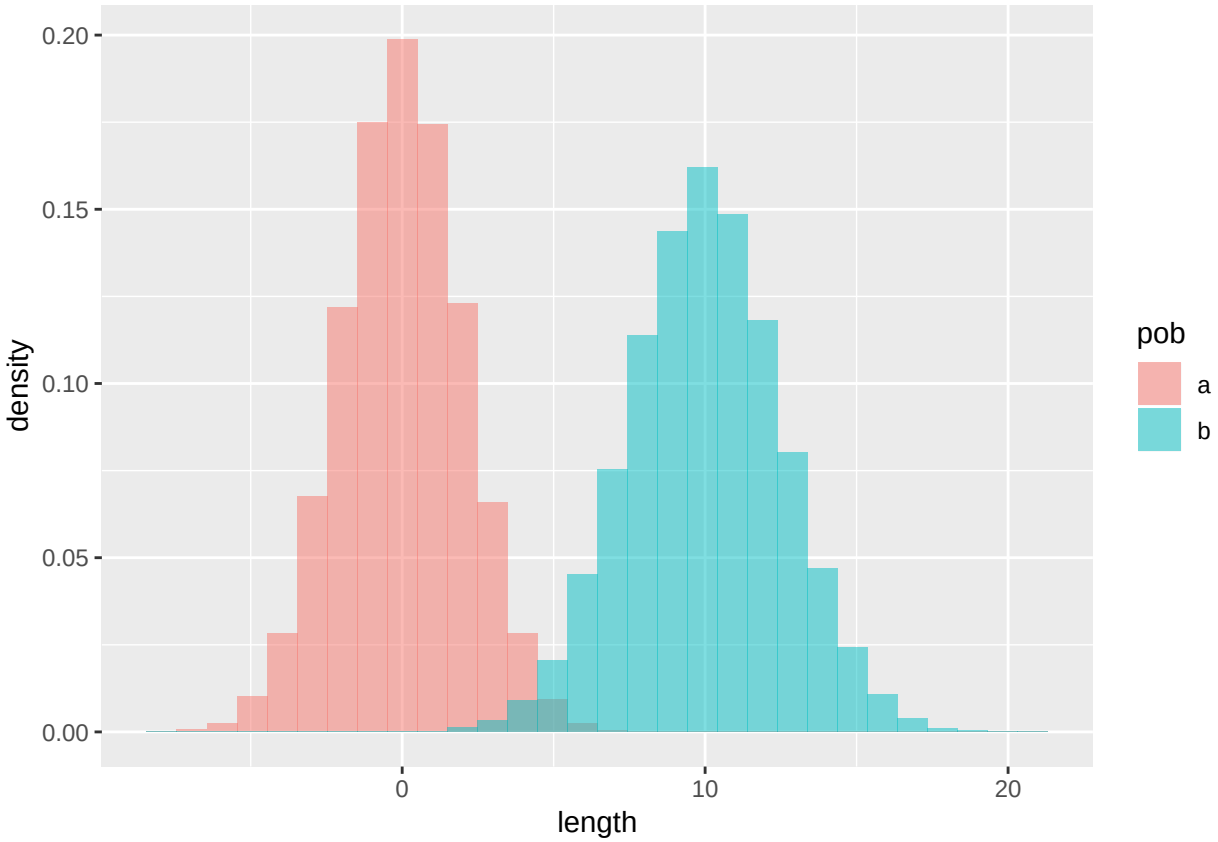
```
summary(datos)
```

```
##       Min.    1st Qu.     Median       Mean    3rd Qu.       Max.
## -4.096087 -0.680579 -0.007441 -0.002366  0.663837  3.562471
```

```
ggplot(vegLengths, aes(length)) +
   geom_histogram(alpha = 0.5, aes(y = ..density..), position = 'identity')
```




```
ggplot(vegLengths, aes(length,fill = pob)) +
   geom_histogram(alpha = 0.5, aes(y = ..density..), position = 'identity')
```


0.20
0.15
0.10
0.05
0.00
density
0
10
20
length
pob
a
b