

Coches

Imanol

6/3/2021

Análisis de los coches (mtcars)

Carga de datos

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from plotnine import ggplot
from plotnine.data import mtcars

data = mtcars
# Asignar a cada fila la columna name
data.index = mtcars["name"]

print(data.head())
```

```
##              name    mpg  cyl  disp  ...  vs  am  gear  carb
## name
## Mazda RX4          Mazda RX4  21.0   6  160.0  ...  0   1    4    4
## Mazda RX4 Wag      Mazda RX4 Wag  21.0   6  160.0  ...  0   1    4    4
## Datsun 710          Datsun 710  22.8   4  108.0  ...  1   1    4    1
## Hornet 4 Drive      Hornet 4 Drive  21.4   6  258.0  ...  1   0    3    1
## Hornet Sportabout  Hornet Sportabout  18.7   8  360.0  ...  0   0    3    2
##
## [5 rows x 12 columns]
```

Medidas de centralización

Dan una idea del valor mas tipico, donde se concentra la media y la mediana etc etc

```
# Media de cada una de las columnas
print(data.mean())

# Media de cada una de las filas (no tiene mucho sentido hacerlo)

## mpg      20.090625
## cyl       6.187500
## disp     230.721875
```

```
## hp      146.687500
## drat     3.596563
## wt       3.217250
## qsec     17.848750
## vs       0.437500
## am       0.406250
## gear     3.687500
## carb     2.812500
## dtype: float64
```

```
print(data.mean(axis = 1)) # axis = 1 media por filas
```

```
# Mediana de cada una de las columnas
```

```
## name
## Mazda RX4      29.907273
## Mazda RX4 Wag  29.981364
## Datsun 710     23.598182
## Hornet 4 Drive 38.739545
## Hornet Sportabout 53.664545
## Valiant        35.049091
## Duster 360     59.720000
## Merc 240D      24.634545
## Merc 230       27.233636
## Merc 280       31.860000
## Merc 280C      31.787273
## Merc 450SE     46.430909
## Merc 450SL     46.500000
## Merc 450SLC    46.350000
## Cadillac Fleetwood 66.232727
## Lincoln Continental 66.058545
## Chrysler Imperial 65.972273
## Fiat 128       19.440909
## Honda Civic    17.742273
## Toyota Corolla 18.814091
## Toyota Corona  24.888636
## Dodge Challenger 47.240909
## AMC Javelin    46.007727
## Camaro Z28     58.752727
## Pontiac Firebird 57.379545
## Fiat X1-9      18.928636
## Porsche 914-2  24.779091
## Lotus Europa   24.880273
## Ford Pantera L 60.971818
## Ferrari Dino   34.508182
## Maserati Bora  63.155455
## Volvo 142E     26.262727
## dtype: float64
```

```
print(data.median())
```

```
# Moda de cada una de las columnas
```

```
## mpg      19.200
```

```
## cyl      6.000
## disp    196.300
## hp      123.000
## drat     3.695
## wt       3.325
## qsec     17.710
## vs       0.000
## am       0.000
## gear     4.000
## carb     2.000
## dtype: float64
```

```
print(data.mode())
```

```
##          name  mpg  cyl  disp    hp  ...  qsec  vs  am  gear  carb
## 0      AMC Javelin  10.4  8.0  275.8  110.0  ...  17.02  0.0  0.0   3.0   2.0
## 1  Cadillac Fleetwood  15.2  NaN   NaN  175.0  ...  18.90  NaN  NaN  NaN   4.0
## 2      Camaro Z28  19.2  NaN   NaN  180.0  ...   NaN  NaN  NaN  NaN  NaN
## 3  Chrysler Imperial  21.0  NaN   NaN   NaN  ...   NaN  NaN  NaN  NaN  NaN
## 4      Datsun 710  21.4  NaN   NaN   NaN  ...   NaN  NaN  NaN  NaN  NaN
## 5   Dodge Challenger  22.8  NaN   NaN   NaN  ...   NaN  NaN  NaN  NaN  NaN
## 6      Duster 360  30.4  NaN   NaN   NaN  ...   NaN  NaN  NaN  NaN  NaN
## 7   Ferrari Dino   NaN  NaN   NaN   NaN  ...   NaN  NaN  NaN  NaN  NaN
## 8      Fiat 128   NaN  NaN   NaN   NaN  ...   NaN  NaN  NaN  NaN  NaN
## 9      Fiat X1-9   NaN  NaN   NaN   NaN  ...   NaN  NaN  NaN  NaN  NaN
## 10   Ford Pantera L   NaN  NaN   NaN   NaN  ...   NaN  NaN  NaN  NaN  NaN
## 11   Honda Civic   NaN  NaN   NaN   NaN  ...   NaN  NaN  NaN  NaN  NaN
## 12   Hornet 4 Drive   NaN  NaN   NaN   NaN  ...   NaN  NaN  NaN  NaN  NaN
## 13  Hornet Sportabout   NaN  NaN   NaN   NaN  ...   NaN  NaN  NaN  NaN  NaN
## 14 Lincoln Continental   NaN  NaN   NaN   NaN  ...   NaN  NaN  NaN  NaN  NaN
## 15   Lotus Europa   NaN  NaN   NaN   NaN  ...   NaN  NaN  NaN  NaN  NaN
## 16   Maserati Bora   NaN  NaN   NaN   NaN  ...   NaN  NaN  NaN  NaN  NaN
## 17   Mazda RX4   NaN  NaN   NaN   NaN  ...   NaN  NaN  NaN  NaN  NaN
## 18   Mazda RX4 Wag   NaN  NaN   NaN   NaN  ...   NaN  NaN  NaN  NaN  NaN
## 19   Merc 230   NaN  NaN   NaN   NaN  ...   NaN  NaN  NaN  NaN  NaN
## 20   Merc 240D   NaN  NaN   NaN   NaN  ...   NaN  NaN  NaN  NaN  NaN
## 21   Merc 280   NaN  NaN   NaN   NaN  ...   NaN  NaN  NaN  NaN  NaN
## 22   Merc 280C   NaN  NaN   NaN   NaN  ...   NaN  NaN  NaN  NaN  NaN
## 23   Merc 450SE   NaN  NaN   NaN   NaN  ...   NaN  NaN  NaN  NaN  NaN
## 24   Merc 450SL   NaN  NaN   NaN   NaN  ...   NaN  NaN  NaN  NaN  NaN
## 25   Merc 450SLC   NaN  NaN   NaN   NaN  ...   NaN  NaN  NaN  NaN  NaN
## 26  Pontiac Firebird   NaN  NaN   NaN   NaN  ...   NaN  NaN  NaN  NaN  NaN
## 27   Porsche 914-2   NaN  NaN   NaN   NaN  ...   NaN  NaN  NaN  NaN  NaN
## 28   Toyota Corolla   NaN  NaN   NaN   NaN  ...   NaN  NaN  NaN  NaN  NaN
## 29   Toyota Corona   NaN  NaN   NaN   NaN  ...   NaN  NaN  NaN  NaN  NaN
## 30      Valiant   NaN  NaN   NaN   NaN  ...   NaN  NaN  NaN  NaN  NaN
## 31   Volvo 142E   NaN  NaN   NaN   NaN  ...   NaN  NaN  NaN  NaN  NaN
##
## [32 rows x 12 columns]
```

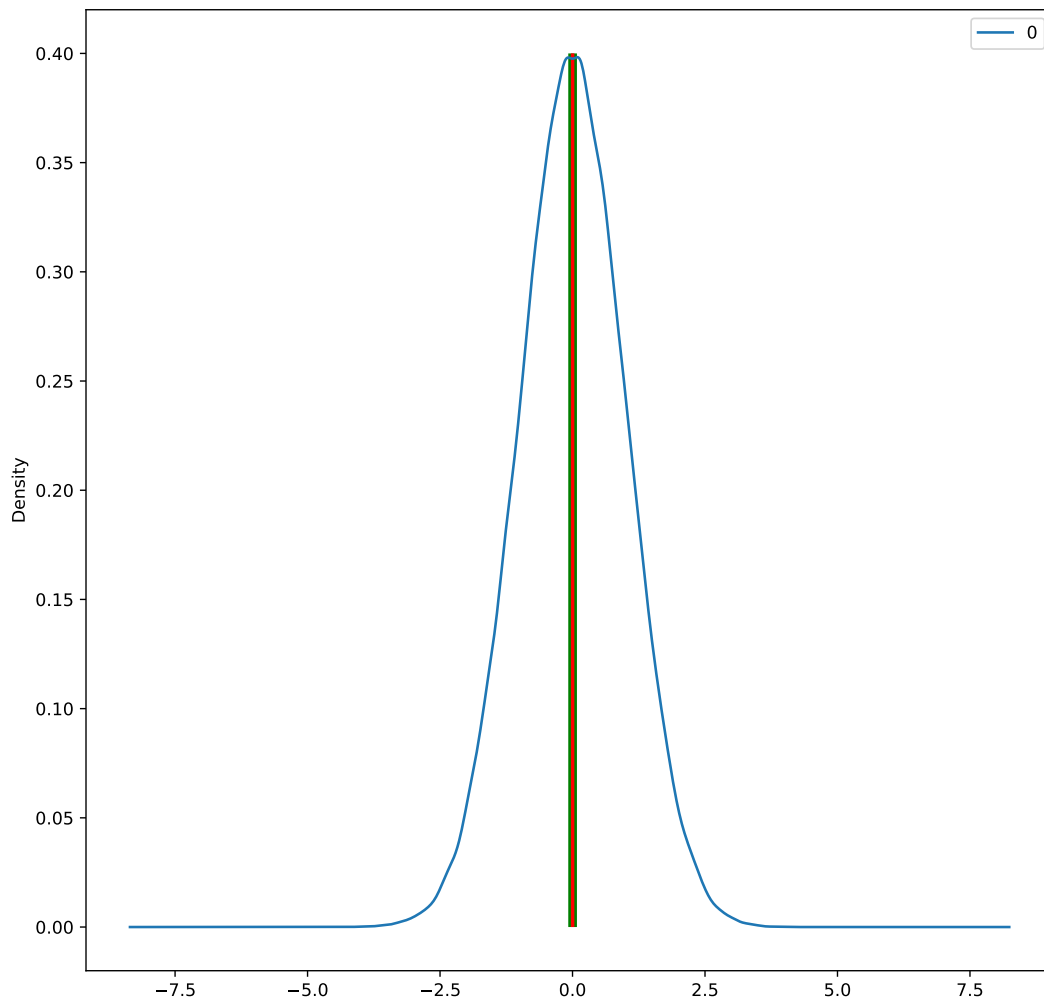
Medidas vs distribuciones

```
# Con distribución normal
norm_data = pd.DataFrame(np.random.normal(size=100000))
norm_data.plot(kind="density", figsize=(10,10))

## Media y mediana estan muy cerca de cero
plt.vlines(norm_data.mean(), ymin = 0, ymax = 0.4, linewidth=5.0, color = "green")
plt.vlines(norm_data.median(), ymin = 0, ymax = 0.4, linewidth=2.0, color = "red")

plt.show()

# Con distribución exponencial que tiene sesgo
```

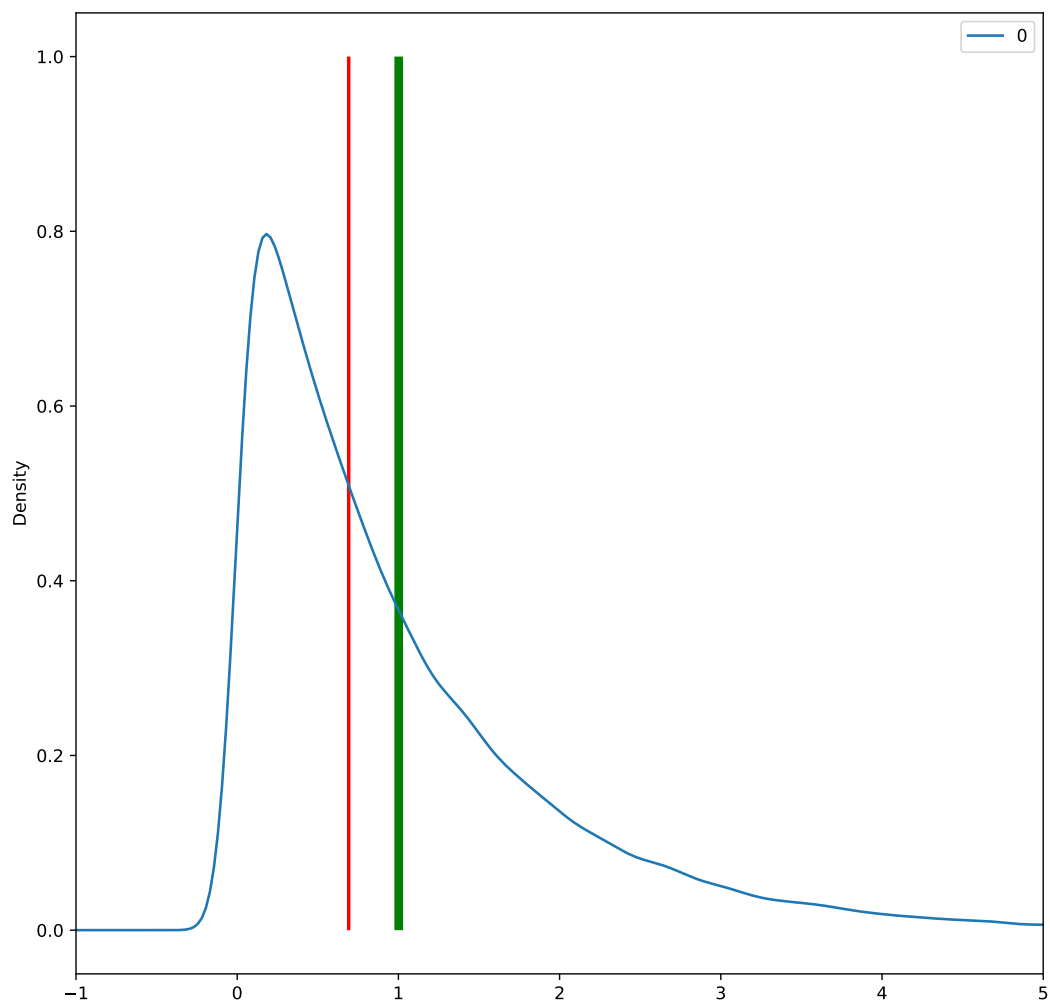


```
skewed_data = pd.DataFrame(np.random.exponential(size=100000))
skewed_data.plot(kind="density", figsize=(10,10), xlim = (-1,5))

# Media y mediana están mas alejadas
plt.vlines(skewed_data.mean(), ymin = 0, ymax = 1.0, linewidth=5.0, color = "green")
plt.vlines(skewed_data.median(), ymin = 0, ymax = 1.0, linewidth=2.0, color = "red")

plt.show()

# Con otra distribución normal
```



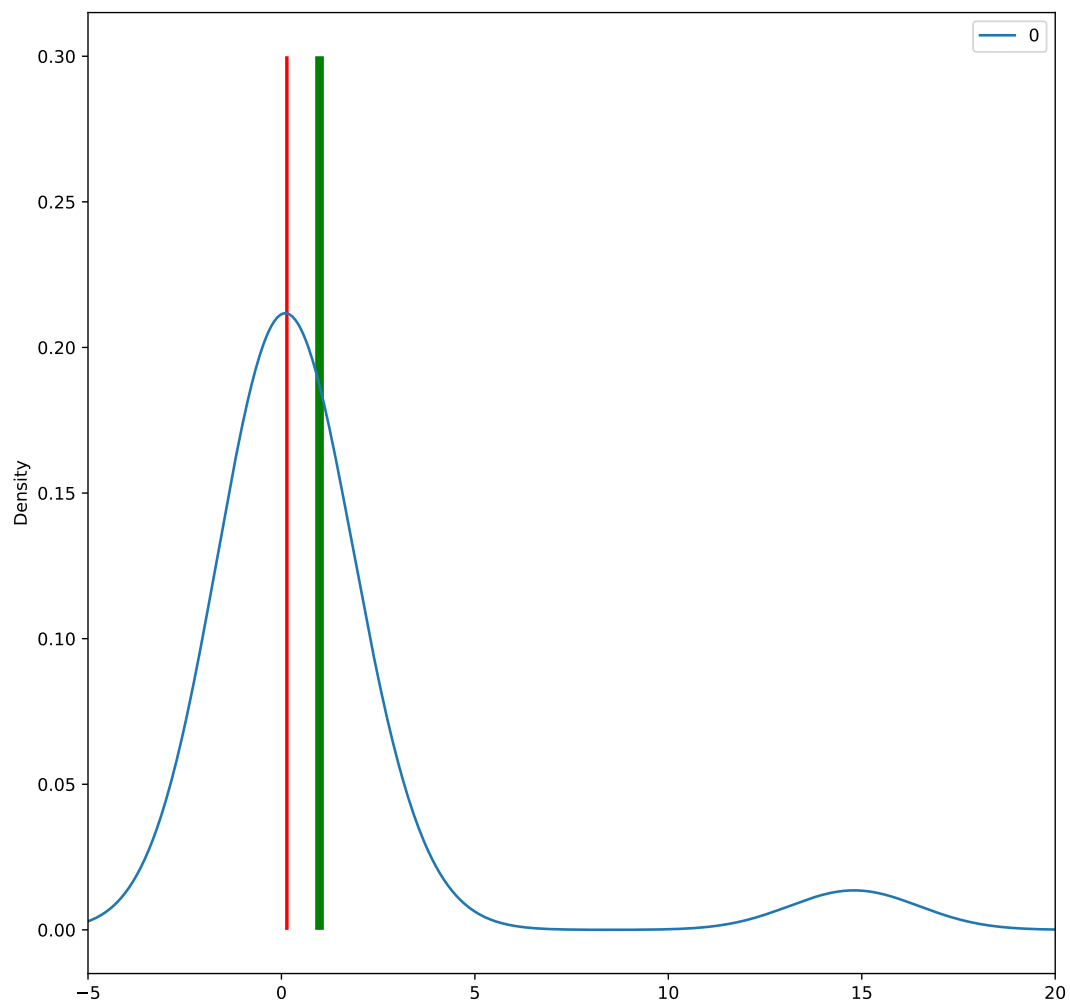
```
norm_data = np.random.normal(size=50)
outliers = np.random.normal(15, size=3)
```

```
combined_data = pd.DataFrame(np.concatenate((norm_data, outliers), axis = 0))

combined_data.plot(kind="density", figsize=(10,10), xlim = (-5,20))

# La media queda desplazada a la derecha, es muy sensible a los outliers
plt.vlines(combined_data.mean(), ymin = 0, ymax = 0.3, linewidth=5.0, color = "green")
plt.vlines(combined_data.median(), ymin = 0, ymax = 0.3, linewidth=2.0, color = "red")

plt.show()
```



Medidas de dispersión

Describen la variación de los datos

Rango de mpg, five nums, cuartiles

```
# Rango de milla por galon
rang = max(data["mpg"]) - min(data["mpg"])
print(rang)
```

```
# Los cinco percentiles que ayudan a describir la dispersión de una variable
```

```
## 23.5
```

```
five_nums = [mtcars["mpg"].quantile(0),
             mtcars["mpg"].quantile(0.25),
             mtcars["mpg"].quantile(0.5),
             mtcars["mpg"].quantile(0.75),
             mtcars["mpg"].quantile(1.0)
]
print(five_nums)
```

```
# Una función de R que da directamente esos five numbers
```

```
## [10.4, 15.425, 19.2, 22.8, 33.9]
```

```
print(mtcars["mpg"].describe())
```

```
# Rango intercuartilico
```

```
## count      32.000000
## mean       20.090625
## std        6.026948
## min        10.400000
## 25%        15.425000
## 50%        19.200000
## 75%        22.800000
## max        33.900000
## Name: mpg, dtype: float64
```

```
print(mtcars["mpg"].quantile(0.75) - mtcars["mpg"].quantile(0.25))
```

```
# El boxplot
```

```
## 7.375
```

```

data.boxplot(column = "mpg", return_type = "axes", figsize = (10,10))

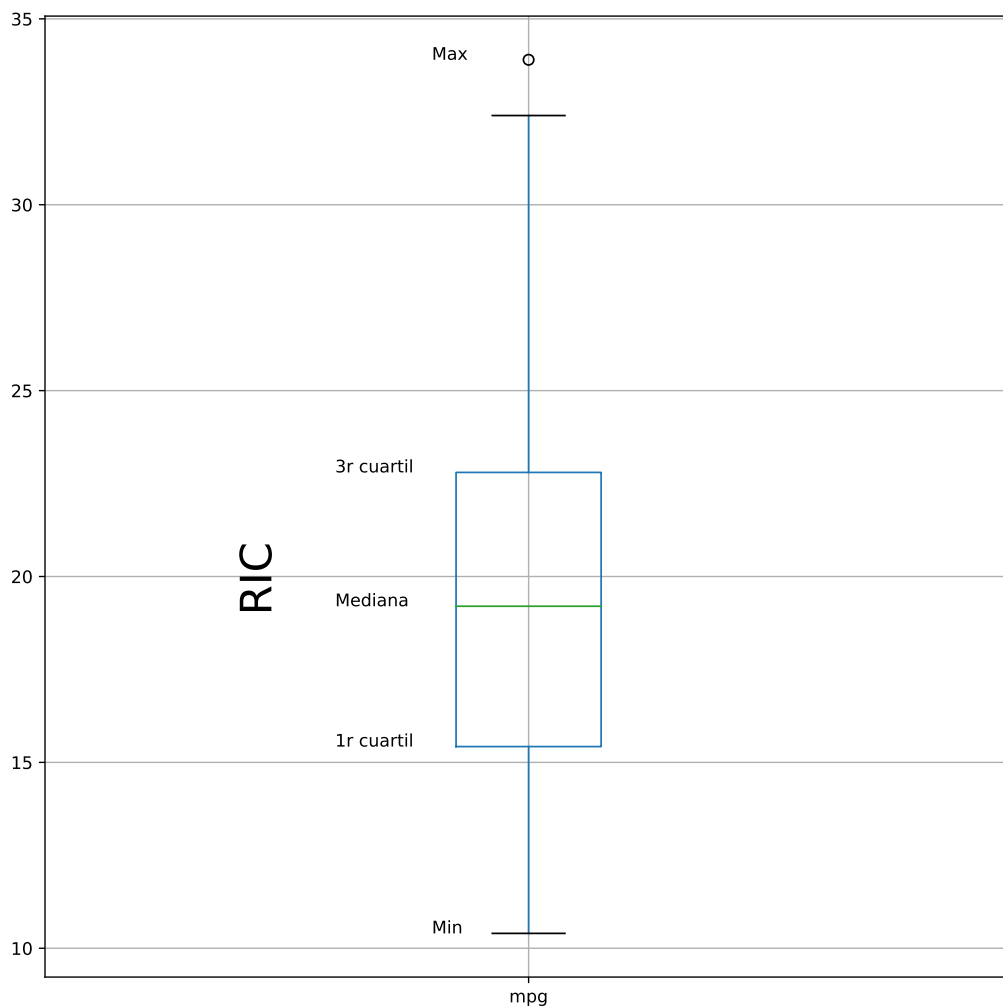
## Pongo nombres
plt.text(x=0.8, y = mtcars["mpg"].quantile(0.25), s = "1r cuartil")
plt.text(x=0.8, y = mtcars["mpg"].quantile(0.5), s = "Mediana")
plt.text(x=0.8, y = mtcars["mpg"].quantile(0.75), s = "3r cuartil")

plt.text(x=0.9, y = mtcars["mpg"].quantile(0), s = "Min")
plt.text(x=0.9, y = mtcars["mpg"].quantile(1), s = "Max")

plt.text(x = 0.7, y = mtcars["mpg"].quantile(0.5), s = "RIC", rotation = 90, size = 25) # El rotation r

plt.show()

```



Varianza y desviación típica

```
# Variacion
print(data["mpg"].var())

# Desviación típica
#cambia de manera rapida por influencia de sesgo o outliers
```

```
## 36.32410282258064
```

```
print(data["mpg"].std())

# Desviacion mediana absoluta
#es dificil que cambie de manera rapida por influencia de sesgo o outliers
```

```
## 6.026948052089104
```

```
mad = abs(data["mpg"] - data["mpg"].median())
k = 1.4826 # Es un valor deducido de la estadística (siempre es este)
print(mad.median()*k)
```

```
## 5.4114900000000001
```

El sesgo y la curtosis

El sesgo indica la simetria o asimetria de una distribución

La curtosis mide como de picuda es la distribucion.

```
# Sesgo
print(data["mpg"].skew()) # Tira mas hacia la derecha (positivo)

# Curtosis
```

```
## 0.6723771376290805
```

```
print(data["mpg"].kurt()) # Es bastante neutro porque se acerca a 0
```

```
## -0.0220062914240855
```

```
# Vemos como quedan el sesgo y la curtosis comparando dos distribuciones diferentes

## Distribución con sesgo bastante neutro
norm = np.random.normal(size=100000)

## Una distribución con sesgo a la izquierda
skew = np.concatenate((np.random.normal(size=35000)+2,
                        np.random.exponential(size=65000)),
                        axis = 0)
```

```

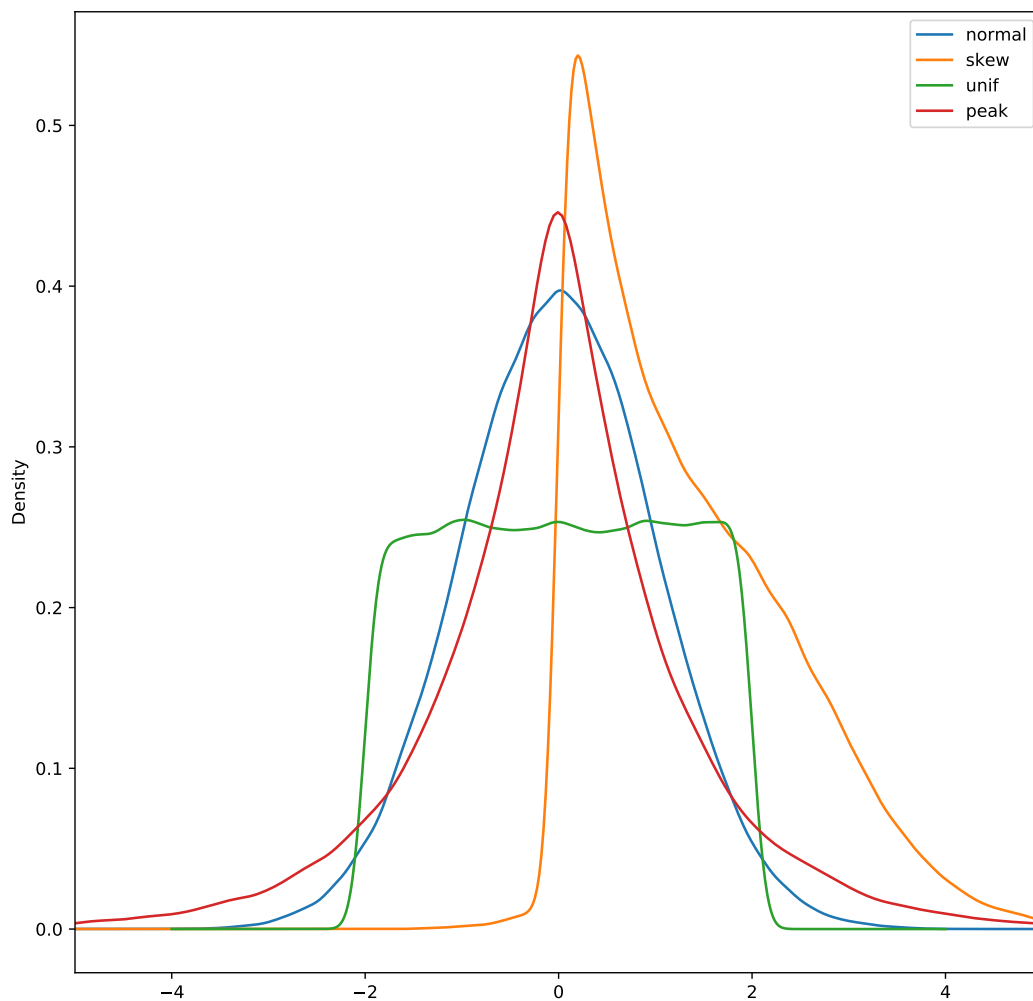
## Una distribucion planita, sesgo nulo y curtosis negativa por ser lo contrario a un pincho.
unif = np.random.uniform(-2,2,size = 100000)

## Acampanada, muy simetrica. Pero su curtosis es grande por tener un pico muy prominente.
peak = np.concatenate((np.random.exponential(size=50000),
                        np.random.exponential(size=50000)*(-1)),
                        axis = 0)

data = pd.DataFrame({
    "normal": norm,
    "skew": skew,
    "unif": unif,
    "peak": peak
})

data.plot(kind="density", figsize = (10,10), xlim = (-5,5))
plt.show()

```



```
print("Normal, Sesgo = %f, Curtosis = %f"%(data["normal"].skew(), data["normal"].kurt()))
```

```
## Normal, Sesgo = 0.010592, Curtosis = 0.006434
```

```
print("Normal+Exp, Sesgo = %f, Curtosis = %f"%(data["skew"].skew(), data["skew"].kurt()))
```

```
## Normal+Exp, Sesgo = 0.984666, Curtosis = 1.088310
```

```
print("Uniforme, Sesgo = %f, Curtosis = %f"%(data["unif"].skew(), data["unif"].kurt()))
```

```
## Uniforme, Sesgo = -0.007588, Curtosis = -1.199385
```

```
print("Suma de Exp, Sesgo = %f, Curtosis = %f"%(data["peak"].skew(), data["peak"].kurt()))
```

```
## Suma de Exp, Sesgo = 0.021037, Curtosis = 2.939710
```

La normal es una distribucion totalmente simetrica

La suma de la normal mas la exponencial tienen un sesgo positivo, por lo que la mayoria de datos a la izquierd.

La uniforme como la suma de las dos exponenciales, un sesgo muy cercano a 0. Mientras que con la curtosis a mas plana menos pico y cuanto mas afilado el pico mayor curtosis.