# Analisis de los coches (mtcars)

Ramon Ceballos

11/2/2021

## Análisis de los coches (mtcars)

### 1. Carga de datos

Vamos a trabajar con Python.

Importamos las librerías que vamos a usar en el análisis y cargamos los datos del dataset "mtcars" presente en ggplot.

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

```
## C:\Users\usuario\ANACON~1\lib\site-packages\ggplot\utils.py:81: FutureWarning: pandas.tslib is depre
## You can access Timestamp as pandas.Timestamp
##   pd.tslib.Timestamp,
## C:\Users\usuario\Documents\R\win-library\3.6\reticulate\python\rpytools\loader.py:24: FutureWarning:
##   level=level
```

Vemos si se han cargado los datos correctamente.

```
data = mtcars
print(data.head(4))
```

```
##              name   mpg  cyl   disp   hp  drat     wt   qsec  vs  am  gear  \
## 0       Mazda RX4  21.0    6  160.0  110  3.90  2.620  16.46   0   1     4
## 1   Mazda RX4 Wag  21.0    6  160.0  110  3.90  2.875  17.02   0   1     4
## 2      Datsun 710  22.8    4  108.0   93  3.85  2.320  18.61   1   1     4
## 3  Hornet 4 Drive  21.4    6  258.0  110  3.08  3.215  19.44   1   0     3
##
##    carb
## 0     4
## 1     4
## 2     1
## 3     1
```

## 2. Medidas de centralización

Vemos que el índice se asigna por números que pone Python. Vamos a asignar el índice según el nombre del modelo del coche.

```
data.index = data["name"]
```

A continuación hago la **media** para cada una de las columnas numéricas.

```
print(data.mean())
```

```
## 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
```

También se puede hacer la media por filas, aunque en este caso no tenga sentido.

```
print(data.mean(axis=1))
```

```
## 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
```

La mediana de la distribución para cada columna sería:

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

```
## 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
```

La moda sería:

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

```
##                    name   mpg  cyl   disp     hp  drat    wt   qsec   vs   am  \
## 0            AMC Javelin  10.4  8.0  275.8  110.0  3.07  3.44  17.02  0.0  0.0
## 1    Cadillac Fleetwood  15.2  NaN    NaN  175.0  3.92   NaN  18.90  NaN  NaN
## 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
```

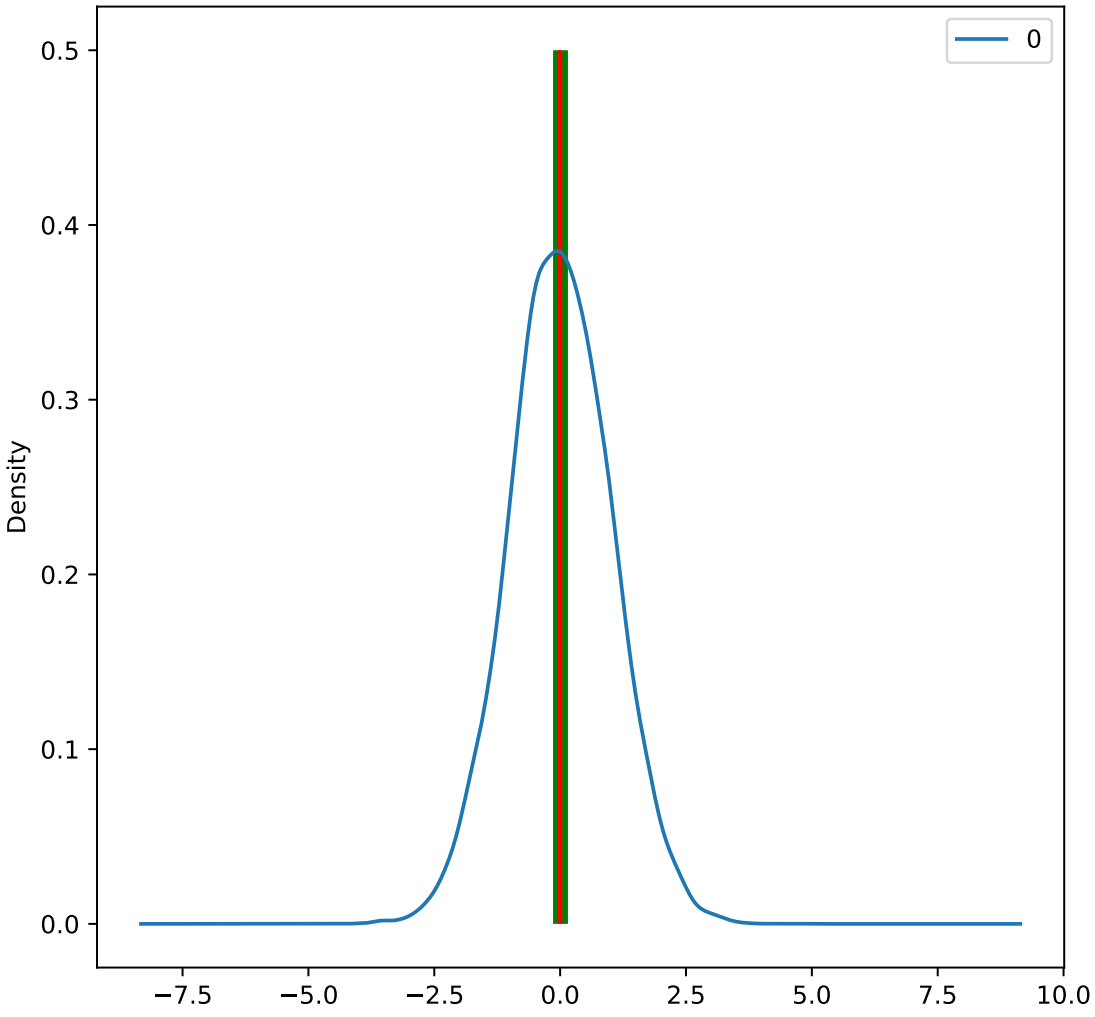
```
## 19             Merc 230   NaN  NaN    NaN    NaN   NaN   NaN    NaN  NaN  NaN
## 20            Merc 240D   NaN  NaN    NaN    NaN   NaN   NaN    NaN  NaN  NaN
## 21             Merc 280   NaN  NaN    NaN    NaN   NaN   NaN    NaN  NaN  NaN
## 22            Merc 280C   NaN  NaN    NaN    NaN   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
##
##     gear  carb
## 0    3.0   2.0
## 1    NaN   4.0
## 2    NaN   NaN
## 3    NaN   NaN
## 4    NaN   NaN
## 5    NaN   NaN
## 6    NaN   NaN
## 7    NaN   NaN
## 8    NaN   NaN
## 9    NaN   NaN
## 10   NaN   NaN
## 11   NaN   NaN
## 12   NaN   NaN
## 13   NaN   NaN
## 14   NaN   NaN
## 15   NaN   NaN
## 16   NaN   NaN
## 17   NaN   NaN
## 18   NaN   NaN
## 19   NaN   NaN
## 20   NaN   NaN
## 21   NaN   NaN
## 22   NaN   NaN
## 23   NaN   NaN
## 24   NaN   NaN
## 25   NaN   NaN
## 26   NaN   NaN
## 27   NaN   NaN
## 28   NaN   NaN
## 29   NaN   NaN
## 30   NaN   NaN
## 31   NaN   NaN
```

## 3. Medidas vs Distribuciones

La media y la mediana no siempre tienen que coincidir, y esto a su vez depende del sesgo de la distribución. Cuando la distribución es simétrica la mediana y la media coinciden. Cuando el sesgo es positivo o negativo, no coincidirán.

Genero unos datos aleatorios segun una normal y represento un plot de la función de densidad de dichos datos. A su vez, le añado una línea que coincida con la media y la mediana.

```
norm_data = pd.DataFrame(np.random.normal(size=10000))

norm_data.plot(kind="density", figsize=(7,7))

plt.vlines(norm_data.mean(), ymin=0, ymax=0.5, linewidth=6.0, color="green")

plt.vlines(norm_data.median(), ymin=0, ymax=0.5, linewidth=2.0, color="red")

plt.show()
```



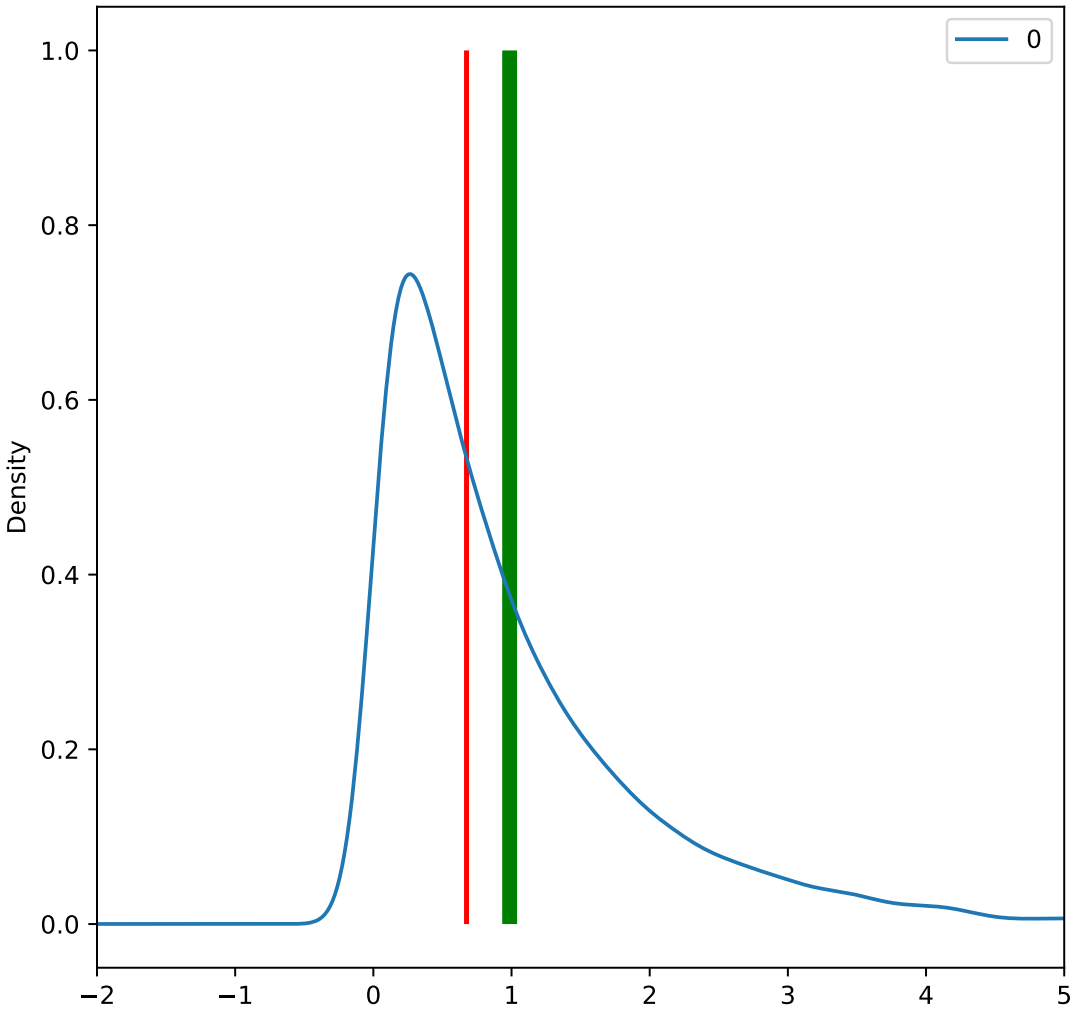

Vemos que la distribución es simétrica ya que la media y la mediana se superponen.

Limpio la figura con “plt.clf()” y represento unos datos de distribución exponencial de la misma forma que antes.

```
skewed_data = pd.DataFrame(np.random.exponential(size=10000))

skewed_data.plot(kind="density", figsize=(7,7))

plt.vlines(skewed_data.mean(), ymin=0, ymax=1, linewidth=6.0, color="green")

plt.vlines(skewed_data.median(), ymin=0, ymax=1, linewidth=2.0, color="red")

plt.xlim(-2,5)
```

```
## (-2.0, 5.0)
```

```
plt.show()
```




Se observa que hay un sesgo positivo (asimetría hacia la dcha), y la mediana es más pequeña que la media.

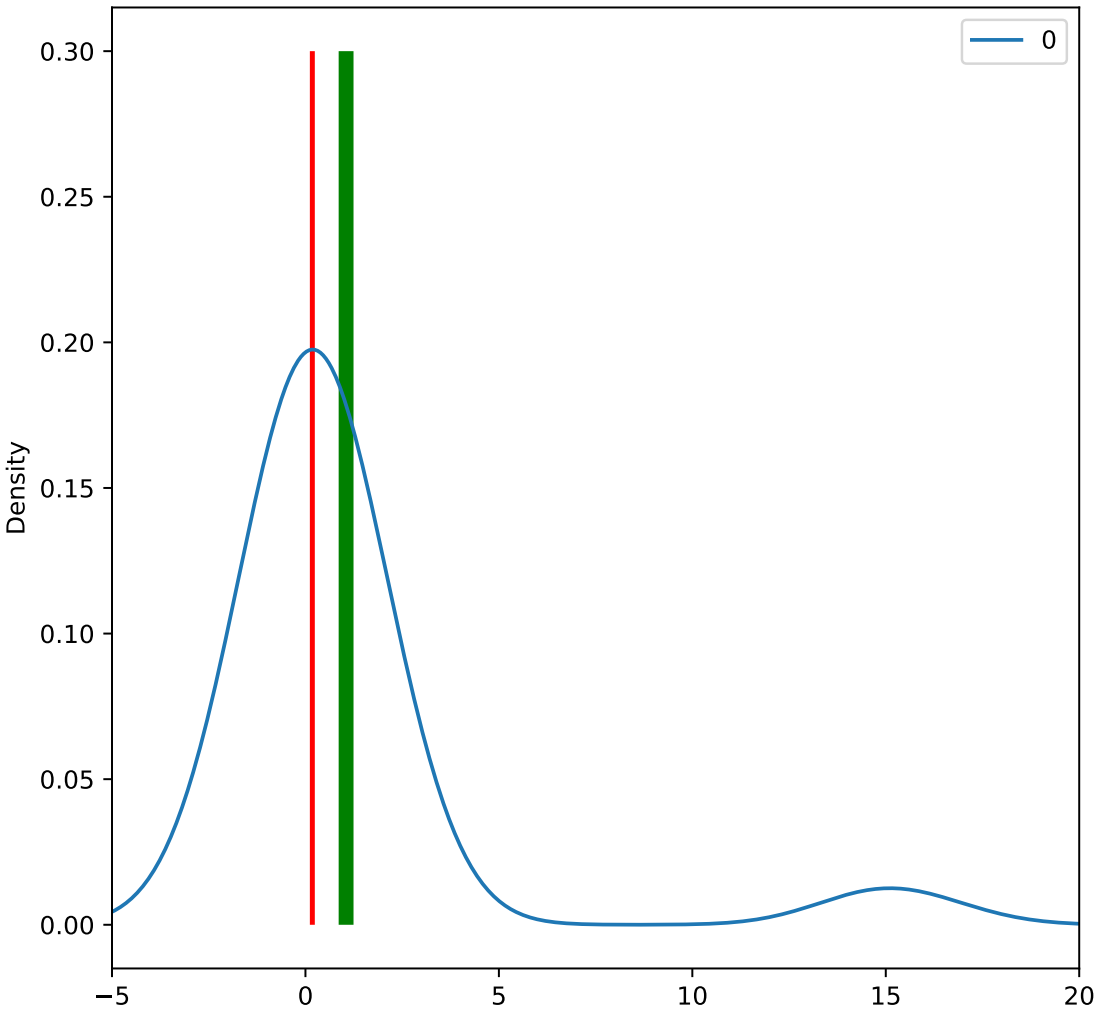
Ahora vamos a generar una distribución normal de 50 datos (norm_data) con una serie de outliers (outliers) que siguen una distribución normal con $\mu = 15$. Juntamos ambos datos en un Data Frame mediante np.concatenate, y para indicar que se haga por columnas se pone axis=0.

```
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=(7,7))

plt.vlines(combined_data.mean(), ymin=0, ymax=0.3, linewidth=6.0, color="green")
```

```
plt.vlines(combined_data.median(), ymin=0, ymax=0.3, linewidth=2.0, color="red")

plt.xlim(-5,20)
```

```
## (-5.0, 20.0)
```

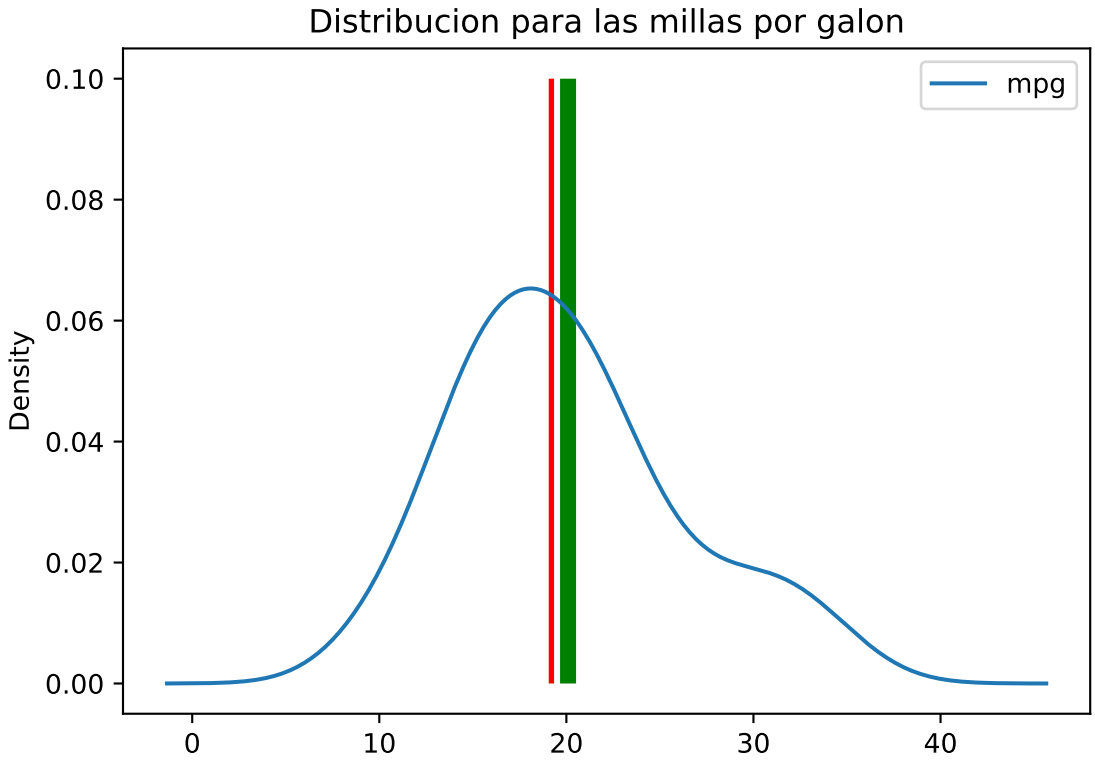
```
plt.show()
```




Ahora la media aparece desplaza a la dcha, indicando la existencia de un sesgo. Esto se debe a los outliers que hemos incorporado. La mediana, por tanto, es más robusta que la media debido a que se resiente menos frente a la presencia de outliers. La media en cambio es muy sensible.

Vamos a investigar si dentro del dataset de mtcars, la variable mpg (millas por galón) está sesgada o no.

```
data_mpg = data [["mpg"]]

data_mpg.plot(kind="density", title="Distribucion para las millas por galon")

plt.vlines(data_mpg.mean(),
color="green", linewidth=6,
ymin=0, ymax=0.1)

plt.vlines(data_mpg.median(),
color="red", linewidth=2,
ymin=0, ymax=0.1)

plt.show()
```

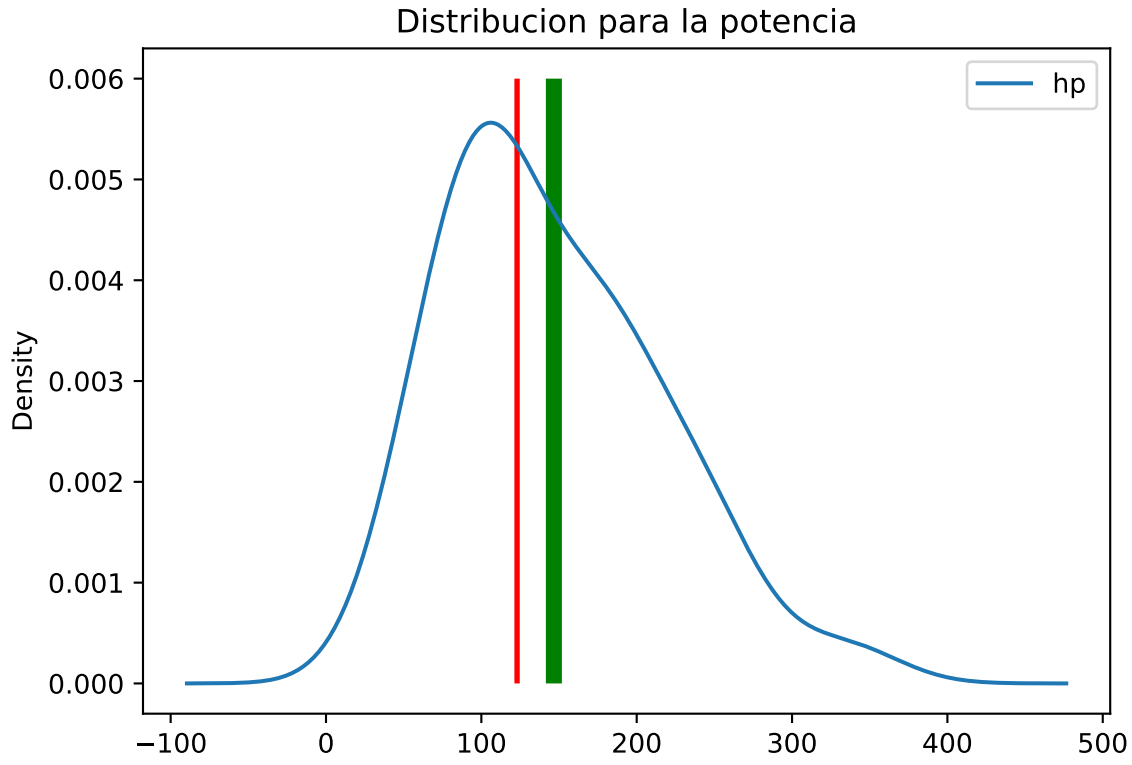



Vemos que la media se encuentra desplazada ligeramente a la derecha, lo cual indica que hay un ligero sesgo positivo.

Para al potencia (hp) lo desarrollaríamos de forma análoga.

```
data_hp = data [["hp"]]

data_hp.plot(kind="density", title="Distribucion para la potencia")

plt.vlines(data_hp.mean(),
color="green", linewidth=6,
ymin=0, ymax=0.006)
```

```
plt.vlines(data_hp.median(),
color="red", linewidth=2,
ymin=0, ymax=0.006)

plt.show()
```




Tambié presenta un sesgo positivo la distribución de la variable de la potencia.