analisis en phyton

Table of contents

ajuste de una distibucion

Ajuste de una distribución.

Ajustar una distribución paramétrica a partir de un conjunto de datos consiste en encontrar el valor de los parámetros con los que, con mayor probabilidad, dicha distribución puede haber generado los datos observados. Por ejemplo, la distribución normal tiene dos parámetros (media y varianza), una vez conocidos estos dos parámetros, se conoce toda la distribución.

Ejemplo:

En este ejemplo se procede a ajustar dos distribuciones, normal y gamma, con el objetivo de modelizar la distribución del precio de venta de diamantes. Además de realizar los ajustes, se representan gráficamente los resultados y se calculan las métricas de bondad de ajuste AIC, BIC y Log-Likelihood con el objetivo de comparar e identificar el mejor que distribución se ajusta mejor.

Librerias :

```
# Tratamiento de datos
# ==============================================================================
import pandas as pd
import numpy as np
import seaborn as sns
```

```
# Gráficos
# ==============================================================================
import matplotlib.pyplot as plt
from matplotlib import style
```

```
# Ajuste de distribuciones
# ==============================================================================
from scipy import stats
import inspect
from statsmodels.distributions.empirical_distribution import ECDF
```

```
# Configuración matplotlib
# ==============================================================================
```

```
#plt.rcParams['image.cmap'] = "bwr"
#plt.rcParams['figure.dpi'] = "100"
plt.rcParams['savefig.bbox'] = "tight"
style.use('ggplot') or plt.style.use('ggplot')
```

```
# Configuración warnings
# ==============================================================================
import warnings
warnings.filterwarnings('ignore')
```

DATOS:

Para esta demostración se emplean como datos el precio de los diamantes disponible en data set *diamonds* de la librería seaborn, en concreto, la columna *price*.

```
# Datos
# ==============================================================================
datos = sns.load_dataset('diamonds')
datos = datos.loc[datos.cut == 'Fair', 'price']
```

Dos de los primeros pasos a la hora de analizar una variable son: calcular los principales estadísticos descriptivos y representar las distribuciones observadas (empíricas).

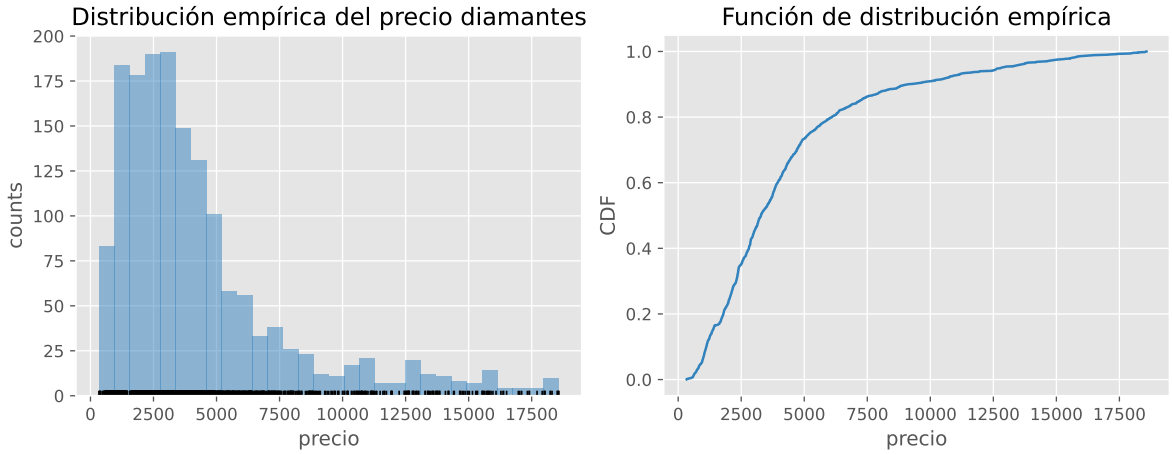
Si los datos se almacenan en un serie de Pandas, pueden obtenerse los principales estadísticos descriptivos con el método describe().

```
# Estadísticos descriptivos
# ==============================================================================
datos.describe()
```

```
count     1610.000000
mean      4358.757764
std       3560.386612
min        337.000000
25%       2050.250000
50%       3282.000000
75%       5205.500000
max      18574.000000
Name: price, dtype: float64
```

Gráficos distribución observada (empírica):

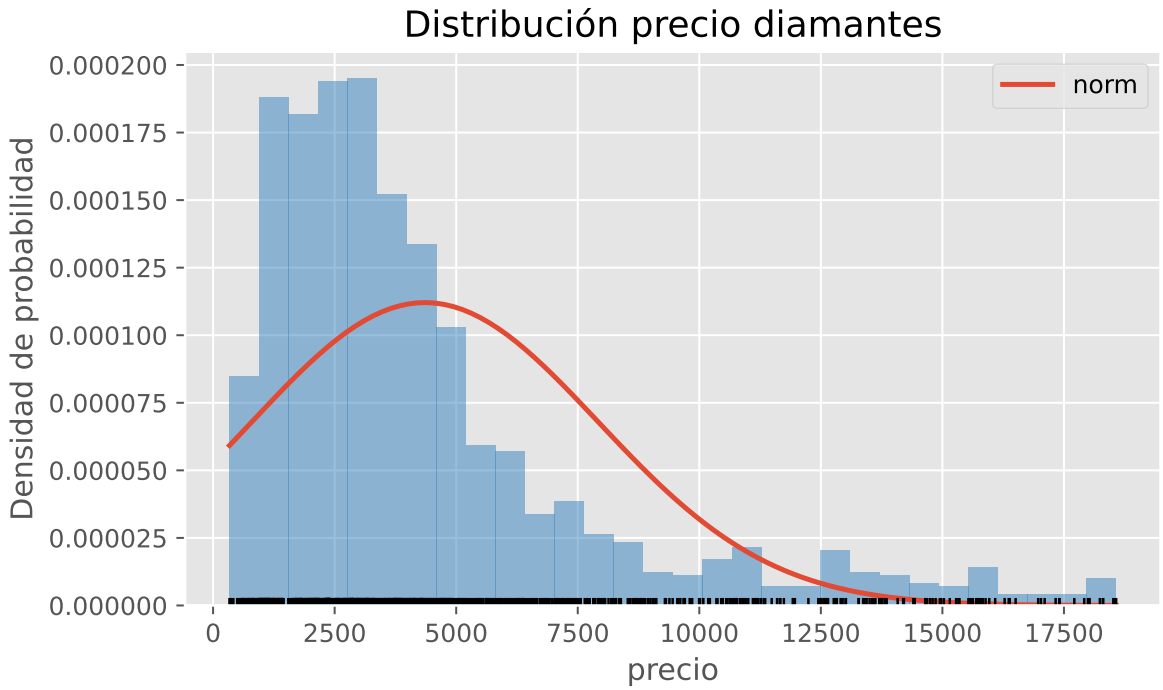
```
# Gráficos distribución observada (empírica)
# ==============================================================================
fig, axs = plt.subplots(nrows=1, ncols=2, figsize=(10, 4))

# Histograma
axs[0].hist(x=datos, bins=30, color="#3182bd", alpha=0.5)
axs[0].plot(datos, np.full_like(datos, -0.01), '|k', markeredgewidth=1)
axs[0].set_title('Distribución empírica del precio diamantes')
axs[0].set_xlabel('precio')
axs[0].set_ylabel('counts')

# Función de Distribución Acumulada
# ecdf (empirical cumulative distribution function)
ecdf = ECDF(x=datos)
axs[1].plot(ecdf.x, ecdf.y, color="#3182bd")
axs[1].set_title('Función de distribución empírica')
axs[1].set_xlabel('precio')
axs[1].set_ylabel('CDF')

plt.tight_layout();
```




```
# Ajuste distribución normal
#===============================================================================
# 1) Se define el tipo de distribución
```

```
distribucion = stats.norm

# 2) Con el método fit() se obtienen los parámetros
parametros = distribucion.fit(data=datos)

# 3) Se crea un diccionario que incluya el nombre de cada parámetro
nombre_parametros = [p for p in inspect.signature(distribucion._pdf).parameters \
                     if not p=='x'] + ["loc","scale"]
parametros_dict = dict(zip(nombre_parametros, parametros))

# 3) Se calcula el log likelihood
log_likelihood = distribucion.logpdf(datos.to_numpy(), *parametros).sum()

# 4) Se calcula el AIC y el BIC
aic = -2 * log_likelihood + 2 * len(parametros)
bic = -2 * log_likelihood + np.log(datos.shape[0]) * len(parametros)
```

```
# 5) Gráfico
x_hat = np.linspace(min(datos), max(datos), num=100)
y_hat = distribucion.pdf(x_hat, *parametros)
fig, ax = plt.subplots(figsize=(7,4))
ax.plot(x_hat, y_hat, linewidth=2, label=distribucion.name)
ax.hist(x=datos, density=True, bins=30, color="#3182bd", alpha=0.5)
ax.plot(datos, np.full_like(datos, -0.01), '|k', markeredgewidth=1)
ax.set_title('Distribución precio diamantes')
ax.set_xlabel('precio')
ax.set_ylabel('Densidad de probabilidad')
ax.legend();
```




```python
#6) Información del ajuste
print('----------------------')
print('Resultados del ajuste')
print('----------------------')
print(f"Distribución:   {distribucion.name}")
print(f"Dominio:        {[distribucion.a, distribucion.b]}")
print(f"Parámetros:     {parametros_dict}")
print(f"Log likelihood: {log_likelihood}")
print(f"AIC:            {aic}")
print(f"BIC:            {bic}")
```

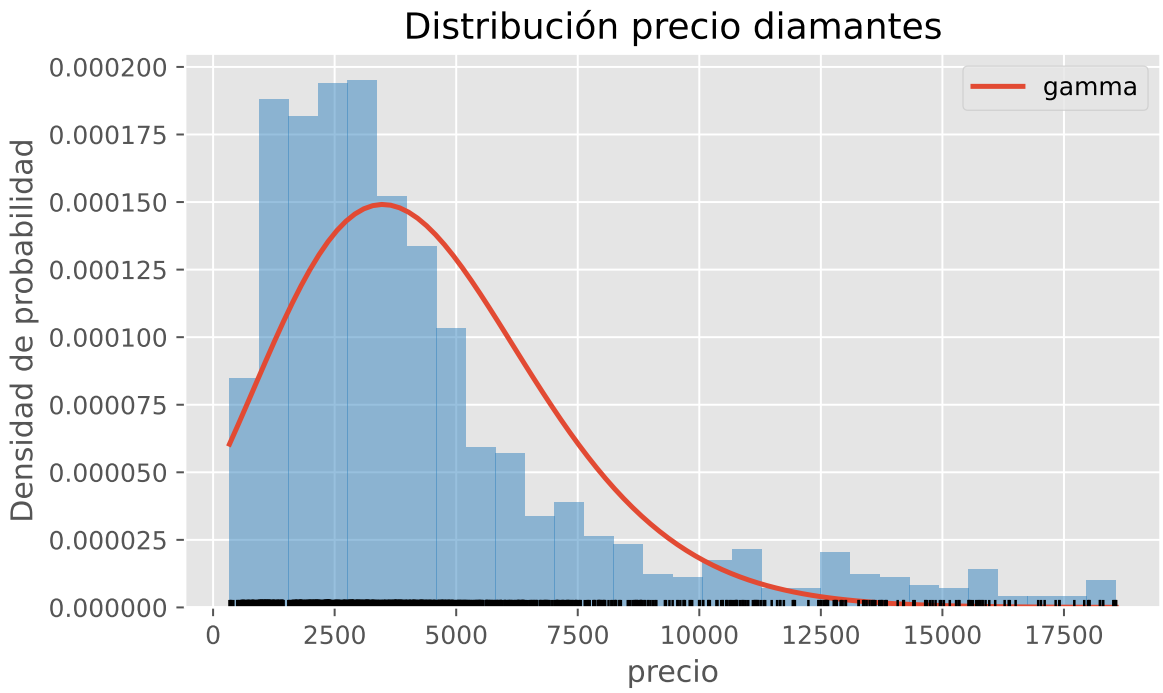
```
----------------------
Resultados del ajuste
----------------------
Distribución:   norm
Dominio:        [-inf, inf]
Parámetros:     {'loc': 4358.757763975155, 'scale': 3559.2807303891086}
Log likelihood: -15449.966194325283
AIC:            30903.932388650566
BIC:            30914.700367566522
```

```
# Ajuste distribución normal
#===============================================================================
# 1) Se define el tipo de distribución
distribucion = stats.gamma

# 2) Con el método fit() se obtienen los parámetros
parametros = distribucion.fit(data=datos)

# 3) Se crea un diccionario que incluya el nombre de cada parámetro
nombre_parametros = [p for p in inspect.signature(distribucion._pdf).parameters \
                     if not p=='x'] + ["loc","scale"]
parametros_dict = dict(zip(nombre_parametros, parametros))

# 3) Se calcula el log likelihood
log_likelihood = distribucion.logpdf(datos.to_numpy(), *parametros).sum()

# 4) Se calcula el AIC y el BIC
aic = -2 * log_likelihood + 2 * len(parametros)
bic = -2 * log_likelihood + np.log(datos.shape[0]) * len(parametros)
```

```
# 5) Gráfico
x_hat = np.linspace(min(datos), max(datos), num=100)
y_hat = distribucion.pdf(x_hat, *parametros)
fig, ax = plt.subplots(figsize=(7,4))
ax.plot(x_hat, y_hat, linewidth=2, label=distribucion.name)
ax.hist(x=datos, density=True, bins=30, color="#3182bd", alpha=0.5)
ax.plot(datos, np.full_like(datos, -0.01), '|k', markeredgewidth=1)
ax.set_title('Distribución precio diamantes')
ax.set_xlabel('precio')
ax.set_ylabel('Densidad de probabilidad')
ax.legend();

#6) Información del ajuste
print('----------------------')
print('Resultados del ajuste')
print('----------------------')
print(f"Distribución:   {distribucion.name}")
print(f"Dominio:        {[distribucion.a, distribucion.b]}")
print(f"Parámetros:     {parametros_dict}")
print(f"Log likelihood: {log_likelihood}")
print(f"AIC:            {aic}")
```

```
print(f"BIC:            {bic}")
```

```
----------------------
Resultados del ajuste
----------------------
Distribución:   gamma
Dominio:        [0.0, inf]
Parámetros:     {'a': 14.399103124683087, 'loc': -6244.786483560789, 'scale': 726.3476320693
Log likelihood: -15235.639828174484
AIC:            30477.27965634897
BIC:            30493.431624722904
```



Tanto la métrica AIC como la BIC coinciden en que la distribución gamma se ajusta mejor a los datos (valores de AIC y BIC más bajos). Esto se puede corroborar fácilmente con la inspección gráfica de los resultados.

En este caso, dado que los valores de precio solo pueden ser positivos y se tienen una notable cola derecha, la distribución gamma era una candidata mucho mejor que la normal. Sin embargo, hay otras posibles distribuciones, algunas de las cuales podrían ser mejores. En el siguiente ejemplo, se muestra cómo automatizar la búsqueda.