# analisis python

Marle

## Table of contents

## analisis python

---

---

# Correlacion lineal con Python.

### Ejemplo correlación lineal

Un estudio pretende analizar si existe una correlación lineal positiva entre la altura y el peso de las personas. Los datos utilizados en este ejemplo se han obtenido del libro *Statistical Rethinking by Richard McElreath*. El set de datos contiene información recogida por Nancy Howell a finales de la década de 1960 sobre el pueblo !Kung San, que viven en el desierto de Kalahari entre Botsuana, Namibia y Angola.

```
import pandas as pd
```

```
import numpy as np
from sklearn.datasets import load_diabetes
```

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

```
# Preprocesado y análisis
# ==============================================================================
import statsmodels.api as sm
import pingouin as pg
from scipy import stats
from scipy.stats import pearsonr
```

```
# Configuración matplotlib
# ==============================================================================
plt.style.use('ggplot')
```

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

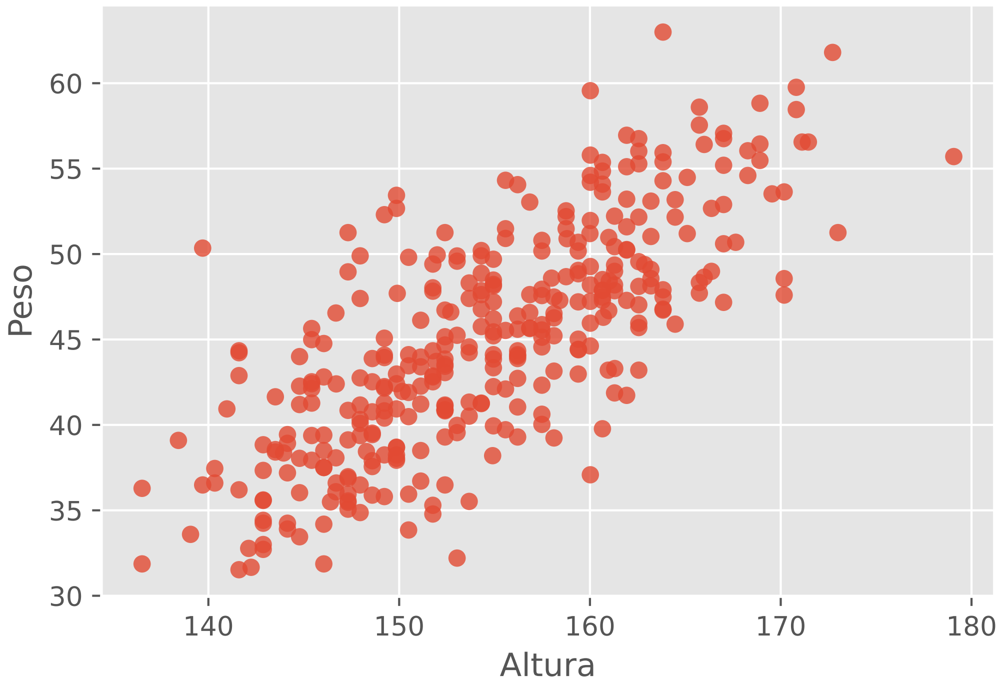
```python
url = ('https://raw.githubusercontent.com/JoaquinAmatRodrigo/' +
       'Estadistica-machine-learning-python/master/data/Howell1.csv')
datos = pd.read_csv(url)

# Se utilizan únicamente información de individuos mayores de 18 años.
datos = datos[datos.age > 18]

datos.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 346 entries, 0 to 543
Data columns (total 4 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   height  346 non-null    float64
 1   weight  346 non-null    float64
 2   age     346 non-null    float64
 3   male    346 non-null    int64  
dtypes: float64(3), int64(1)
memory usage: 13.5 KB
```

##**Análisis gráfico¶**

En primer lugar se representan las dos variables mediante un diagrama de dispersión (*scatterplot*) para intuir si existe relación lineal o monotónica. Si no la hay, no tiene sentido calcular este tipo de correlaciones.

```python
# Gráfico
# ==============================================================================
fig, ax = plt.subplots(1, 1, figsize=(6,4))
ax.scatter(x=datos.height, y=datos.weight, alpha= 0.8)
ax.set_xlabel('Altura')
ax.set_ylabel('Peso');
```



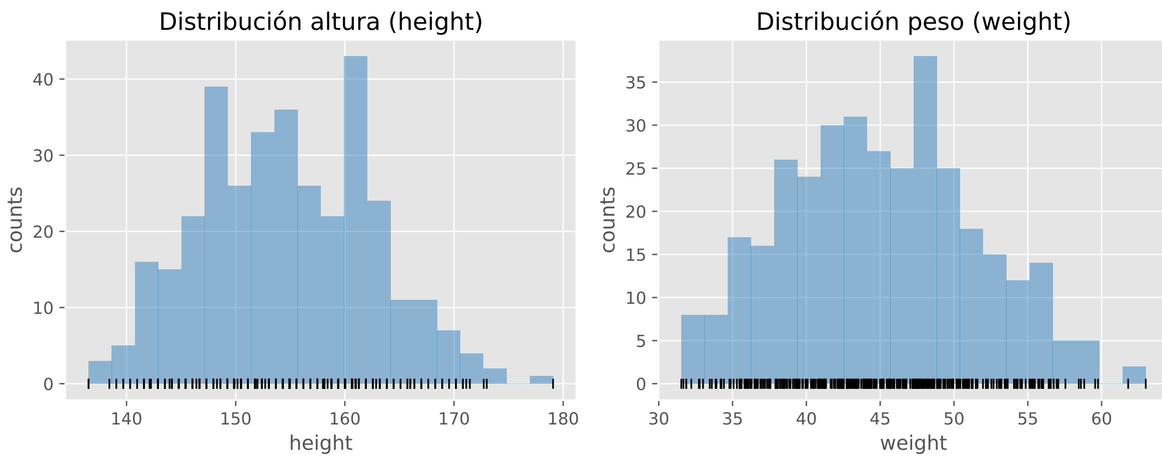

El diagrama de dispersión parece indicar una relación lineal positiva entre ambas variables.

Para poder elegir el coeficiente de correlación adecuado, se tiene que analizar el tipo de variables y la distribución que presentan. En este caso, ambas variables son cuantitativas continuas y pueden ordenarse para convertirlas en un ranking, por lo que, a priori, los tres coeficientes podrían aplicarse. La elección se hará en función de la distribución que presenten las observaciones: normalidad, homocedasticidad y presencia de *outliers*.

##**Normalidad**

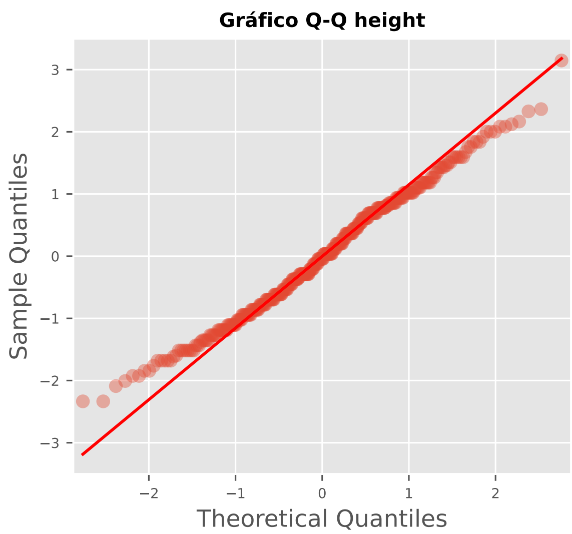
```
# Gráfico distribución variables
# ==============================================================================
fig, axs = plt.subplots(nrows=1, ncols=2, figsize=(10, 4))

axs[0].hist(x=datos.height, bins=20, color="#3182bd", alpha=0.5)
axs[0].plot(datos.height, np.full_like(datos.height, -0.01), '|k', markeredgewidth=1)
axs[0].set_title('Distribución altura (height)')
axs[0].set_xlabel('height')
axs[0].set_ylabel('counts')

axs[1].hist(x=datos.weight, bins=20, color="#3182bd", alpha=0.5)
axs[1].plot(datos.weight, np.full_like(datos.weight, -0.01), '|k', markeredgewidth=1)
axs[1].set_title('Distribución peso (weight)')
```

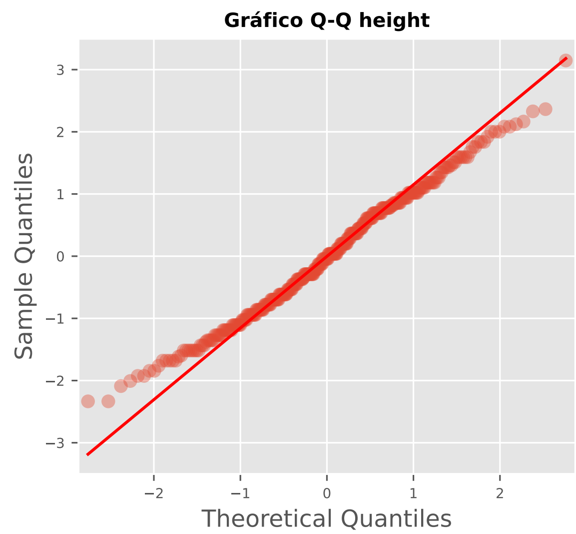
```
axs[1].set_xlabel('weight')
axs[1].set_ylabel('counts')


plt.tight_layout();
```




```
# Gráfico Q-Q
# ==============================================================================
fig, axs = plt.subplots(nrows=1, ncols=2, figsize=(10, 4))

sm.qqplot(
    datos.height,
    fit   = True,
    line  = 'q',
    alpha = 0.4,
    lw    = 2,
    ax    = axs[0]
)
axs[0].set_title('Gráfico Q-Q height', fontsize = 10, fontweight = "bold")
axs[0].tick_params(labelsize = 7)

sm.qqplot(
    datos.height,
    fit   = True,
    line  = 'q',
    alpha = 0.4,
    lw    = 2,
    ax    = axs[1]
```

```
)
axs[1].set_title('Gráfico Q-Q height', fontsize = 10, fontweight = "bold")
axs[1].tick_params(labelsize = 7)
```




Además del estudio gráfico, se recurre a dos test estadísticos que contrasten la normalidad de los datos: *Shapiro-Wilk test* y *D'Agostino's K-squared test*. Este último es el que incluye el *summary* de **statsmodels** bajo el nombre de *Omnibus*.

En ambos test, la hipótesis nula considera que los datos siguen una distribución normal, por lo tanto, si el *p-value* no es inferior al nivel de referencia *alpha* seleccionado, no hay evidencias para descartar que los datos se distribuyen de forma normal.

```
# Normalidad de los residuos Shapiro-Wilk test
# ==============================================================================
shapiro_test = stats.shapiro(datos.height)
print(f"Variable height: {shapiro_test}")
shapiro_test = stats.shapiro(datos.weight)
print(f"Variable weight: {shapiro_test}")
```

```
Variable height: ShapiroResult(statistic=0.9910705950686569, pvalue=0.034413216987494714)
Variable weight: ShapiroResult(statistic=0.9911816371339782, pvalue=0.0367282884335488)
```

```
# Normalidad de los residuos D'Agostino's K-squared test
# ==============================================================================
k2, p_value = stats.normaltest(datos.height)
print(f"Variable height: Estadítico = {k2}, p-value = {p_value}")
```

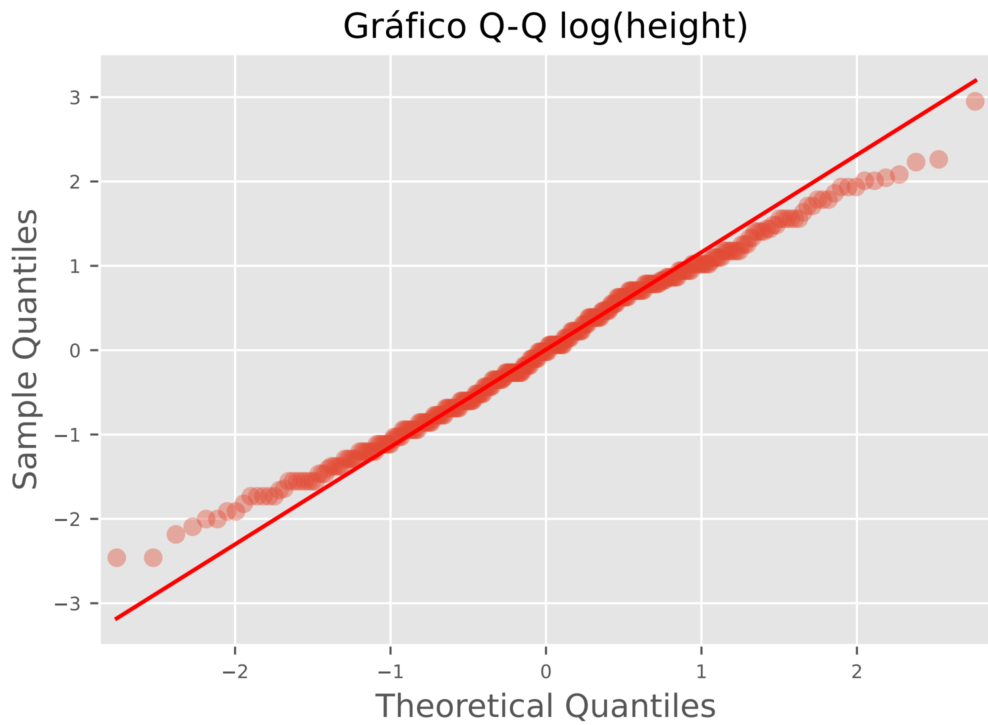
```python
k2, p_value = stats.normaltest(datos.weight)
print(f"Variable weight: Estadítico = {k2}, p-value = {p_value}")
```

```
Variable height: Estadítico = 7.210790495766356, p-value = 0.02717670115638557
Variable weight: Estadítico = 8.402628478646044, p-value = 0.014975881988444982
```

El análisis visual y las pruebas estadísticas indican que no podemos asumir normalidad en ninguna de las dos variables. Esto significa que no podemos usar el coeficiente de Pearson y, en su lugar, debemos considerar alternativas como el coeficiente de Spearman o Kendall. Sin embargo, dado que las distribuciones no se desvían significativamente de la normalidad y que el coeficiente de Pearson es relativamente robusto, podríamos usarlo en la práctica siempre y cuando seamos conscientes de esta limitación y la comuniquemos en los resultados. Otra opción sería intentar transformar las variables para mejorar su distribución, por ejemplo, aplicando el logaritmo.

```python
# Transformación logarítmica de los datos
# ==============================================================================
fig, ax = plt.subplots(nrows=1, ncols=1, figsize=(6, 4))

sm.qqplot(
    np.log(datos.height),
    fit   = True,
    line  = 'q',
    alpha = 0.4,
    lw    = 2,
    ax    = ax
)
ax.set_title('Gráfico Q-Q log(height)', fontsize = 13)
ax.tick_params(labelsize = 7)


shapiro_test = stats.shapiro(np.log(datos.height))
print(f"Variable height: {shapiro_test}")
```

```
Variable height: ShapiroResult(statistic=0.9922663896096799, pvalue=0.06946765640621282)
```




La trasformación logarítmica de la variable altura (*height*) consigue una distribución más próxima a la normal.

## Coeficientes correlación

Debido a la falta de normalidad, los resultados generados por Pearson no son del todo precisos. Sin embargo, dado que la desviación de la normalidad es leve y no se aprecian *outliers*, con fines ilustrativos, se procede a calcular los tres tipos de coeficientes.

De nuevo recordar que, cuando alguna de las condiciones asumidas por un modelo o test estadístico no se cumplen, no significa que obligatoriamente se tenga que descartar, pero hay que ser consciente de las implicaciones que tiene y reportarlo siempre en los resultados.

**Pandas**

Pandas permite calcular la correlación de dos Series (columnas de un DataFrame). El cálculo se hace por pares, eliminando automáticamente aquellos con valores NA/null. Una limitación de Pandas es que no calcula la significancia estadística.

```
# Cálculo de correlación con Pandas
# ==============================================================================
print('Correlación Pearson: ', datos['weight'].corr(datos['height'], method='pearson'))
print('Correlación spearman: ', datos['weight'].corr(datos['height'], method='spearman'))
print('Correlación kendall: ', datos['weight'].corr(datos['height'], method='kendall'))
```

```
Correlación Pearson:  0.7528177220327672
Correlación spearman:  0.7510966609219974
Correlación kendall:  0.5639709660523899
```

```
# Cálculo de correlación y significancia con Scipy
# ==============================================================================
r, p = stats.pearsonr(datos['weight'], datos['height'])
print(f"Correlación Pearson: r={r}, p-value={p}")

r, p = stats.spearmanr(datos['weight'], datos['height'])
print(f"Correlación Spearman: r={r}, p-value={p}")

r, p = stats.kendalltau(datos['weight'], datos['height'])
print(f"Correlación Pearson: r={r}, p-value={p}")
```

```
Correlación Pearson: r=0.7528177220327668, p-value=1.8941037794176386e-64
Correlación Spearman: r=0.7510966609219974, p-value=5.2882247217804375e-64
Correlación Pearson: r=0.5639709660523899, p-value=3.162649137764635e-54
```

Los test estadísticos muestran una correlación lineal entre moderada y alta, con claras evidencias estadísticas de que la relación observada no se debe al azar (pvalue 0      0).