1 El cuarteto de Anscombe

Objetivo. Visualizar el cuarteto de Anscombe usando herramientas de matplotlib, pandas y numpy, para demostrar las conclusiones del artículo del autor.

Anscombe, F. J. (1973). “Graphs in Statistical Analysis”. The American Statistician. 27 (1): 17–21. doi:10.2307/2682899. JSTOR 2682899.

Trabajo realizado con el apoyo del Programa UNAM-DGAPA-PAPIME PE101922

1.1 ¿Porqué Visualizar?

El proceso de transformar datos crudos en imágenes ayuda a mejorar las interpretaciones de grandes conjuntos de datos y eso permite obtener una perspectiva que podría pasarse por alto si se usarán solamente métodos estadísticos.

1.2 Ejemplo: Anscombe’s quartet

Refrencia en wikipedia: Anscombe’s quartet .

Consiste de cuatro conjuntos de datos que tienen las mismas propiedades estadísticas:

Propiedad	**Valor**
Media $\bar{x}$	9
Media $\bar{y}$	7.50
Varianza muestral s_x^2	11
Varianza muestral s_y^2	4.125
Correlación entre x y y	0.816
Regresión lineal	y = 3.00 + 0.500x
Coef. de determinación R^2	0.67

Cada conjunto consiste de once puntos (x, y) y fueron construidos por el estadístico F. J. Anscombe.

```
import numpy as np
import pandas as pd
from sklearn.linear_model import LinearRegression
import macti.visual
```

1.2.1 Paso 1.

Leer el archivo con la información y ponerla en un DataFrame

```
data = pd.read_csv('AnscombeQuartet.txt', sep='\t', header=None)
data
```

1.2.2 Paso 2.

Limpieza y organización de la información.

```
header = pd.MultiIndex.from_product([['Dataset 1','Dataset 2',
                                      'Dataset 3', 'Dataset 4'],
                                     ['x', 'y']],
                                    names=['dat','val'])
data.columns = header
data.index = [1,2,3,4,5,6,7,8,9,10,11]
data
```

1.2.3 Paso 3.

Calculo de algunas propiedades estadísticas.

```
data.mean(axis=0) # Calculamos la media de todos los conjuntos de datos
```

```
data.var(axis=0) # Calculamos la varianza muestral de todos los conjuntos de datc
```

```
data['Dataset 1'].corr() # Correlación
                         # Cambiar el número del dataset
```

1.2.4 Paso 4.

Realizamos una regresión lineal para ajustar una recta a los datos.

```
# Convertir los valores en un arreglo columna de numpy
X = data.iloc[:, 0].values.reshape(-1, 1)
X
```

La siguiente celda define una función para realizar el cálculo de la regresión lineal.

```
def regresionLineal(data, i):
    X = data.iloc[:, i].values.reshape(-1, 1)   # Vector columna X
    Y = data.iloc[:, i+1].values.reshape(-1, 1) # Vector columna Y

#    if i == 4:
#        X = np.delete(X, 2).reshape(-1,1)
#        Y = np.delete(Y, 2).reshape(-1,1)
```

```
    linear_regressor = LinearRegression() # Objeto para la regresión lineal
    linear_regressor.fit(X, Y)            # Se realiza la regresión lineal
    R2 = linear_regressor.score(X,Y)      # Coeficiente de determinación
    m = linear_regressor.coef_            # Coeficientes de la regresión
    b = linear_regressor.intercept_       # lineal y = mx + b

    X_pred = np.arange(0,21,1)
    X_pred.shape = (-1,1) # vector columna
    Y_pred = linear_regressor.predict(X_pred)  # Se realiza la predicción

    return X, Y, X_pred, Y_pred, R2, m[0][0], b[0]
```

```
# Cálculo de la regresión para el Dataset 1
X, Y, X_pred, Y_pred, R2, m, b = regresionLineal(data, 4)
print('R2 = {:0.3f} \t m = {:0.3f} \t b = {:0.3f}'.format(R2, m, b))
```

```
# Cálculo de la regresión para todos los Dataset's
for i in range(0,7,2):
    X, Y, X_pred, Y_pred, R2, m, b = regresionLineal(data, i)
    print('Dataset {:} : R2 = {:0.3f} \t m = {:0.3f} \t b = {:0.3f}'.format(i//2+
```

¿Qué se puede decir de estos resultados?

1.2.5 Paso 5.

Graficación de los resultados.

```
import matplotlib.pyplot as plt
params = {'legend.fontsize': 16,
          'axes.labelsize':16,
          'axes.titlesize':16,
          'xtick.labelsize':16,
          'ytick.labelsize':16}
plt.rcParams.update(params)

fig, axes = plt.subplots(nrows=2, ncols=2, figsize=(14,8))

num = 1
for a in axes:
    for ax in a:
        X, Y, X_pred, Y_pred, R2, m, b = regresionLineal(data, (num - 1)*2)
        ax.scatter(X, Y, marker = 'o', c='orange', s=75, edgecolor='red')
        leyenda = 'R2 : {:0.3f}, m : {:0.3f}, b : {:0.3f}'.format(R2, m, b)
        ax.plot(X_pred, Y_pred, lw=2.0, label=leyenda)
        ax.set_xlim(2,20)
        ax.set_aspect(aspect=1.0)
        ax.set_xlabel('x')
        ax.set_ylabel('y')
```

```
            ax.set_title('Dataset {}'.format(num))
            ax.legend()
            num += 1
    plt.tight_layout()
    #plt.savefig('anscombe.png', dpi=300)
```

¿Qué puede decir de estos gráficos

- Gráfica del Dataset 1: relación lineal simple entre dos variables correlacionadas.
- Gráfica del Dataset 2: se observa una relación entre x y y pero no parece ser lineal.
- Gráfica del Dataset 3: relación lineal pero la regresión obtenida se ve afectada por el dato extremo que influye en el resultado final y altera el coeficiente de correlación de 1 a 0.816.
- Gráfica del Dataset 4: se muestra como un valor atípico es suficiente para producir un coeficiente de correlación alto, aún cuando la relación entre las variables no es lineal.

Este cuarteto es usado todavía en la actualidad para ilustrar la importancia de graficar los datos antes de realizar cualquier análisis estadístico. También se muestra el efecto de los valores atípicos.

La intención fue cambiar la impresión de que **“los cálculos numéricos son exactos, pero los gráficos aproximados”**.

Edward Tufte usó el cuarteto en la primera página del primer capítulo de su libro *The Visual Display of Quantitative Information*, para enfatizar la importancia de mirar los datos antes de analizarlos.