2023-04-17_Example-Anscombe

April 18, 2023

[12]:
```
import numpy as np
import scipy
import imageio

import matplotlib
import matplotlib.pyplot as plt
import matplotlib.cm as cm

matplotlib.rc('image', interpolation='nearest')
matplotlib.rc('figure',facecolor='white')
matplotlib.rc('image',cmap='viridis')

prop_cycle = plt.rcParams['axes.prop_cycle']
colors = prop_cycle.by_key()['color']

%matplotlib inline

import scipy.stats
```

[13]:
```
# raw input data
x1=np.array([10.,8.,13.,9.,11.,14.,6.,4.,12.,7.,5.])
y1=np.array([8.04,6.95,7.58,8.81,8.33,9.96,7.24,4.26,10.84,4.82,5.68])
y2=np.array([9.14,8.14,8.74,8.77,9.26,8.10,6.13,3.10,9.13,7.26,4.74])
y3=np.array([7.46,6.77,12.74,7.11,7.81,8.84,6.08,5.39,8.15,6.42,5.73])
x2=np.array([8.,8.,8.,8.,8.,8.,8.,19.,8.,8.,8.])
y4=np.array([6.58,5.76,7.71,8.84,8.47,7.04,5.25,12.50,5.56,7.91,6.89])
```

[14]:
```
# print output for latex script
for x in [x1,y1,y2,y3,x2,y4]:
    for i,s in enumerate(x):
        if i>0:
            print(" & ",end="")
        print("{:.2f}".format(s),end="")
    print(" \\\\")
```

```
10.00 & 8.00 & 13.00 & 9.00 & 11.00 & 14.00 & 6.00 & 4.00 & 12.00 & 7.00 & 5.00
\\
8.04 & 6.95 & 7.58 & 8.81 & 8.33 & 9.96 & 7.24 & 4.26 & 10.84 & 4.82 & 5.68 \\
```

```
9.14 & 8.14 & 8.74 & 8.77 & 9.26 & 8.10 & 6.13 & 3.10 & 9.13 & 7.26 & 4.74 \\
7.46 & 6.77 & 12.74 & 7.11 & 7.81 & 8.84 & 6.08 & 5.39 & 8.15 & 6.42 & 5.73 \\
8.00 & 8.00 & 8.00 & 8.00 & 8.00 & 8.00 & 8.00 & 19.00 & 8.00 & 8.00 & 8.00 \\
6.58 & 5.76 & 7.71 & 8.84 & 8.47 & 7.04 & 5.25 & 12.50 & 5.56 & 7.91 & 6.89 \\
```

[15]:
```
# arrange into (x,y) pairs
data=np.array([[x1,y1],[x1,y2],[x1,y3],[x2,y4]])
```

[16]:
```
data.shape
```

[16]:
```
(4, 2, 11)
```

[17]:
```
# mean, variance of x sequences
for x in [x1,x2]:
    print("{:.2f}\t{:.2f}".format(np.mean(x),np.var(x)))
```

```
9.00    10.00
9.00    10.00
```

[18]:
```
# mean, variance of y sequences
for x in [y1,y2,y3,y4]:
    print("{:.2f}\t{:.2f}".format(np.mean(x),np.var(x)))
```

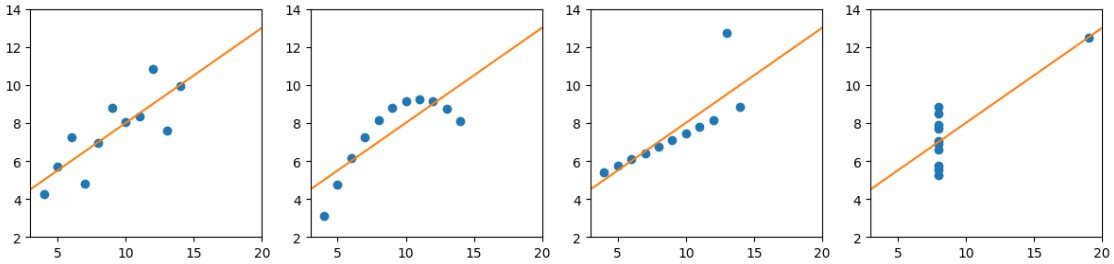
```
7.50    3.75
7.50    3.75
7.50    3.75
7.50    3.75
```

[19]:
```
# linear regression
regression=[scipy.stats.linregress(*dat) for dat in data]
print("slope\tintercept\tcorrelation\tstandard error slope")
for reg in regression:
    print("{: .2f} \t{: .2f}\t\t{: .2f}\t\t{: .2f}".format(reg.slope,reg.
 ↪intercept,reg.rvalue,reg.stderr))
```

```
slope   intercept       correlation     standard error slope
 0.50    3.00            0.82            0.12
 0.50    3.00            0.82            0.12
 0.50    3.00            0.82            0.12
 0.50    3.00            0.82            0.12
```

[20]:
```
# plots
xref=np.linspace(3,20,num=50)
fig=plt.figure(figsize=(12,3))
for i,(dat,reg) in enumerate(zip(data,regression)):
    fig.add_subplot(1,4,i+1)
    plt.scatter(dat[0],dat[1],c=colors[0])
    plt.plot(xref,reg.slope*xref+reg.intercept,c=colors[1])
```

```
    plt.xlim([3,20])
    plt.ylim([2,14])
plt.tight_layout()
plt.show()
```



[ ]:
```
fig.savefig("anscombe.pdf")
```