Aufgabe1

November 15, 2018

1 Aufgabe 1

Gegeben seien zwei Populationen P_0 und P_1. Die Punkte der ersten Population P_0 folgen einer bivariaten Normalverteilung mit den Parametern:

$$\mu_x = 0,\ \mu_y = 3,\ \sigma_x = 3{,}5,\ \sigma_y = 2{,}6 \text{ und Korrelationskoeffizient } \rho = 9\,. \tag{1}$$

Die zweite Population P_1 ist durch eine GauSSverteilung in x mit $\mu_x = 6$ und $\sigma_x = 3{,}5$ gegeben. In y liegt ebenso eine GauSSverteilung vor, wobei der bedingte Erwartungswert $\mathrm{E}(y|x) = \mu_{y|x} = a + bx$ mit $a = -0{,}5$ und $b = 0{,}6$ ist. Die bedingte Varianz von y bei gegebenem x ist konstant und damit unabhängig von x und beträgt $\mathrm{Var}(y|x) = \sigma^2_{y|x} = 1$.

1.1 Teilaufgabe a)

Zu zeigen ist, dass die zweite Population ebenfalls einer bivariaten Normalverteilung entspricht. Dazu ist die Formel für die bedingte Wahrscheinlichkeit der bivariaten Normalverteilung zu verwenden, welche

$$f(y|x) = \frac{1}{\sqrt{2\pi}\sigma_y\sqrt{1-\rho^2}} \cdot \exp\left(-\frac{1}{2(1-\rho^2)}\left[\frac{\tilde{y}}{\sigma_y} - \rho\frac{\tilde{x}}{\sigma_x}\right]^2\right) \tag{2}$$

ist, wobei anscheinend $\tilde{x} = x - \mu_x$ und $\tilde{y} = y - \mu_y$ ist.\ Für die Wahrscheinlichkeitsdichtefunktion einer bivariaten Normalverteilung gilt

$$f(x,y) = \frac{1}{2\pi\sigma_x\sigma_y\sqrt{1-\rho^2}} \exp\left(-\frac{1}{2(1-\rho^2)}\left[\frac{(x-\mu_x)^2}{\sigma_x^2} + \frac{(y-\mu_y)^2}{\sigma_y^2} - \frac{2\rho(x-\mu_x)(y-\mu_y)}{\sigma_x\sigma_y}\right]\right)\,, \tag{3}$$

was sich auch als Definition einer solchen Verteilung verstehen lässt. Es gilt auch $f(x,y) = g(x)f(y|x)$ mit der Randverteilung $g(x)$ in x und der bedingten Wahrscheinlichkeitsdichtefunktion $f(y|x)$. Es konnte auf einem vorherigen Blatt gezeigt werden, dass

$$g(x) = \frac{\sqrt{2(1-\rho^2)}}{2\pi\sigma_x\sigma_y\sqrt{1-\rho^2}}\sqrt{\pi}\sigma_y \cdot \exp\left(-\frac{1}{2}\left(\frac{x-\mu_x}{\sigma_x}\right)^2\right) \tag{4}$$

für die Randverteilung gilt. Das lässt sich einsetzen und nach kürzen folgt

$$f(x,y) = \frac{1}{2\pi\sigma_x\sigma_y\sqrt{1-\rho^2}} \cdot \exp\left(-\frac{1}{2}\left(\frac{x-\mu_x}{\sigma_x}\right)^2 - \frac{1}{2(1-\rho^2)}\left(\frac{y-\mu_y}{\sigma_y} - \rho\frac{x-\mu_x}{\sigma_x}\right)^2\right)\,. \tag{5}$$

Der Vorfaktor hat schon genau die richtige Form. Schaut man sich den Exponenten an, dann wird klar, dass auch schon der in y quadratische Term sowie der Mischterm genau wie in der bivariaten Normalverteilung sind. Also muss man sich noch den in x quadratischen Term ansehen:

$$
\begin{aligned}
& -\frac{(x-\mu_x)^2}{\sigma_x^2}\frac{\rho^2}{2(1-\rho^2)} - \frac{(x-\mu_x)^2}{2\sigma_x^2} && (6)\\
&= -\frac{(x-\mu_x)^2}{\sigma_x^2}\left(\frac{\rho^2}{1-\rho^2}+1\right) && (7)\\
&= -\frac{(x-\mu_x)^2}{\sigma_x^2}\frac{1}{2(1-\rho^2}\,. && (8)
\end{aligned}
$$

Das ist genau die Form wie in der Normalverteilung. Damit ist gezeigt, dass auch die Verteilung von Population 1 einer bivariaten Normalverteilung folgt.\ Nun wollen wir den Parametersatz der zweiten Verteilung vollständig bestimmen. Dazu wird die Gleichung $\sigma_{y|x}^2 = (1-\rho^2)\sigma_y^2$ benötigt, wobei σ_y^2 die Varianz der 2D-Verteilung bezeichnet. Dies ist anders in der Aufgabenstellung, wir finden es so angenehmer. Die Gleichung wurde auf https://math.stackexchange.com/questions/33993/bivariate-normal-conditional-variance, https://onlinecourses.science.psu.edu/stat414/node/118/ und http://athenasc.com/Bivariate-Normal.pdf gefunden. AuSSerdem wissen wir, dass $\mathrm{E}(y|x) = \rho\frac{\sigma_y}{\sigma_x}(x-\mu_x)+\mu_y$ gilt (Blatt 2). Aus einem Koffizientenvergleich dieser Gleichung und der vorherigen folgen drei Bestimmungsgleichungen für die fehlenden Parameter ρ, σ_y und μ_y:

$$
\begin{aligned}
b\sigma_x &= \rho\sigma_y && (9)\\
a &= -\rho\frac{\sigma_y}{\sigma_x}\mu_x + \mu_y && (10)\\
\sigma_{y|x}^2 &= (1-\rho^2)\sigma_y^2\,. && (11)
\end{aligned}
$$

Werden dann Gleichung 1 und Gleichung 3 miteinander kombiniert, so erhält man den Korrelationskoeffizienten ρ:

$$\rho^2 = \frac{\alpha}{1+\alpha^2} \text{ mit } \alpha = b\frac{\sigma_x}{\sigma_{y|x}}\,. \tag{12}$$

Hier konkret ergibt sich $\alpha = 0{,}6 \cdot 3{,}5 = 2{,}1$ und $\rho = \sqrt{210/541} \approx 0{,}6230$. Das lässt sich zurück in die dritte Gleichung einsetzen:

$$\sigma_y^2 = \frac{\sigma_{y|x}^2}{1-\rho^2} \approx 1{,}1774\,. \tag{13}$$

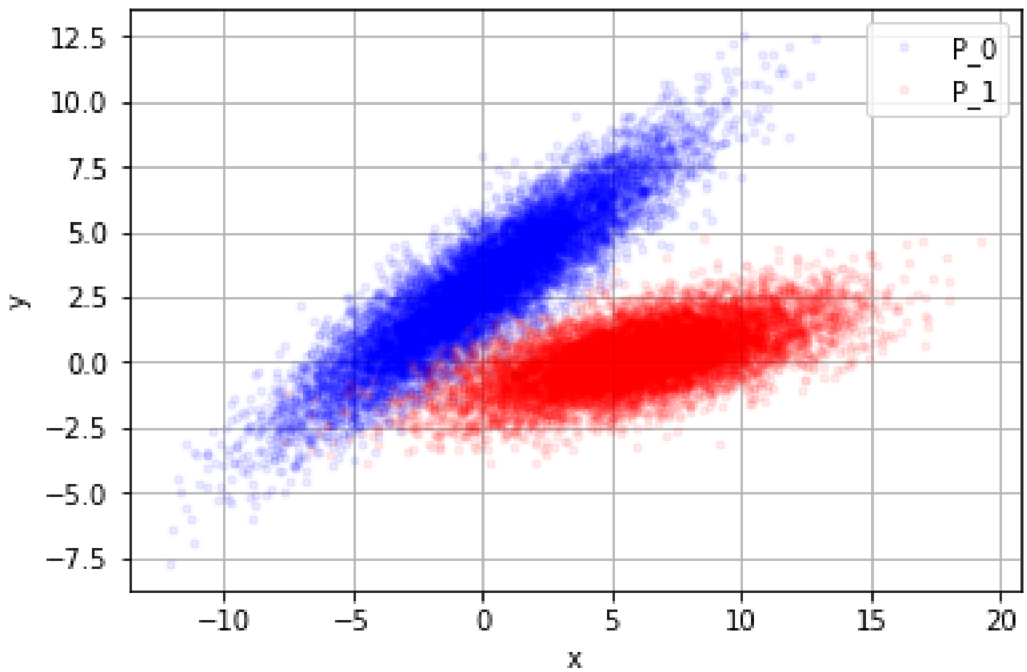
Nun hat man genügend Informationen, um μ_y aus der zweiten GLeichung zu ermitteln:

$$\mu_y = a + \frac{\rho\mu_x}{\sigma_x}\frac{\sigma_{y|x}}{\sqrt{1-\rho^2}} \approx 0{,}2221\,. \tag{14}$$

```
In [11]: # Teilaufgabe b)
         import numpy as np
         import matplotlib.pyplot as plt
         from numpy import random

         random.seed = 42
         covP0 = [[3.5**2, 0.9*3.5*2.6],[0.9*3.5*2.6, 2.6**2]]
```

```
P0 = random.multivariate_normal([0,3],covP0, size=10000)
x0, y0 = zip(*P0)
plt.plot(x0, y0, 'b.', label='P_0', alpha=0.07)
covP1 = [[3.5**2, 0.6230*3.5*1.1774],[0.6230*3.5*1.1774, 1.1774**2]]
P1 = random.multivariate_normal([6,0.2221],covP1, size=10000)
x1, y1 = zip(*P1)
plt.plot(x1, y1, 'r.', label='P_1', alpha=0.07)
#plt.hist2d(x1, y1, bins=(200,200))
plt.grid()
plt.legend()
plt.xlabel('x')
plt.ylabel('y')
plt.show()
```



In [4]:
```
# Teilaufgabe c)
print('Es werden nun die Werte für die einzelnen Populationen berechnet.')
print('Population 0:')
mu0 = np.array([np.mean(x0), np.mean(y0)])
print('mu_0 =',mu0)
varx0 = np.var(x0,ddof=1)
print('var_0x =', varx0)
vary0 = np.var(y0,ddof=1)
print('var_0y =', vary0)
cov0 = np.cov(x0,y0, ddof=1)[0][1]
print('cov(x0,y0) =', cov0)
```

```python
    rho0 = cov0/np.sqrt(varx0*vary0)
    print('rho0 =', rho0)
    print('')

    print('Population 1:')
    mu1 = np.array([np.mean(x1), np.mean(y1)])
    print('mu_1 = ',mu1)
    varx1 = np.var(x1,ddof=1)
    print('var_1x =', varx1)
    vary1 = np.var(y1,ddof=1)
    print('var_1y =', vary1)
    cov1 = np.cov(x1,y1, ddof=1)[0][1]
    print('cov(x1,y1) =', cov1)
    rho1 = cov1/np.sqrt(varx1*vary1)
    print('rho1 =', rho1)
    print('')

    print('Nun werden beide Populationen zu einer Gesamtpopulation zusammengefasst und die
    P = np.vstack((P0,P1))
    x, y = zip(*P)
    mu = np.array([np.mean(x), np.mean(y)])
    print('mu =',mu)
    varx = np.var(x,ddof=1)
    print('var_x =', varx)
    vary = np.var(y,ddof=1)
    print('var_y =', vary)
    cov = np.cov(x,y, ddof=1)[0][1]
    print('cov(x,y) =', cov)
    rho = cov/np.sqrt(varx*vary)
    print('rho =', rho)
    print('')
```

```
Es werden nun die Werte für die einzelnen Populationen berechnet.
Population 0:
mu_0 = [-0.05362058  2.96566614]
var_0x = 12.2480816929
var_0y = 6.70607267518
cov(x0,y0) = 8.15812103976
rho0 = 0.900164703016

Population 1:
mu_1 =  [ 6.02535121  0.23580991]
var_1x = 12.2882033538
var_1y = 1.39305587502
cov(x1,y1) = 2.52817623369
rho1 = 0.611053001109

Nun werden beide Populationen zu einer Gesamtpopulation zusammengefasst und die Werte erneut be
```

```
mu = [ 2.98586532  1.60073802]
var_x = 21.5064655541
var_y = 5.91248371345
cov(x,y) = 1.19399425202
rho = 0.105884584305
```

```
In [5]: import pandas as pd
        P0_1k = random.multivariate_normal([0,3],covP0, size=1000)
        P0_10k = P0
        P1_10k = P1
        df_P0_10k = pd.DataFrame({'x':P0_10k[:,0],'y':P0_10k[:,1]})
        # first column is number of point,
        # then x-value, then y-value of point
        df_P0_1k = pd.DataFrame({'x':P0_1k[:,0],'y':P0_1k[:,1]})
        df_P1_10k = pd.DataFrame({'x':P1_10k[:,0],'y':P1_10k[:,1]})

        hdf = pd.HDFStore('three_populations.h5')
        hdf.put('P0_10k', df_P0_10k) # key, then value
        hdf.put('P0_1k', df_P0_1k)
        hdf.put('P1_10k', df_P1_10k)
```

```
In [6]: # Just a quick check if everything worked out correctly
        reread = pd.read_hdf('three_populations.h5', key='P0_1k')
        print('P0_1k read from file:', reread.head(3))
        print('P0_1k original:', df_P0_1k.head(3))
        reread = pd.read_hdf('three_populations.h5', key='P0_10k')
        print('P0_10k read from file:', reread.head(3))
        print('P0_10k original:', df_P0_10k.head(3))
        reread = pd.read_hdf('three_populations.h5', key='P1_10k')
        print('P1_10k read from file:', reread.head(3))
        print('P1_10k original:', df_P1_10k.head(3))
        # That looks pretty good
```

```
P0_1k read from file:           x         y
0  0.927491  5.044736
1 -5.882805 -3.448108
2 -3.856405 -0.780843
P0_1k original:           x         y
0  0.927491  5.044736
1 -5.882805 -3.448108
2 -3.856405 -0.780843
P0_10k read from file:           x         y
0 -3.700531  0.216635
1  7.182575  7.229019
2 -5.875437 -1.285343
P0_10k original:           x         y
```

```
0 -3.700531  0.216635
1  7.182575  7.229019
2 -5.875437 -1.285343
P1_10k read from file:            x         y
0  3.330385  1.198667
1  3.760275 -0.290981
2  1.975719  0.263442
P1_10k original:            x         y
0  3.330385  1.198667
1  3.760275 -0.290981
2  1.975719  0.263442
```