blatt06_nico

June 8, 2021

1 Abgabe SMD Blatt 06

1.0.1 von Nico Guth, David Venker, Jan Jäkel

1.1 Aufgabe 13 Fisher-Diskriminante: Implementation

[1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib as mpl
```

[2]:
```
mpl.rcParams['font.size'] = 15
```

[3]:
```
# Daten einlesen
p0 = pd.read_hdf('zwei_populationen.h5', key='P_0_10000')
p1 = pd.read_hdf('zwei_populationen.h5', key='P_1')
p0_1000 = pd.read_hdf('zwei_populationen.h5', key='P_0_1000')
```

[4]:
```
print(p0.describe())
print(p1.describe())
print(p0_1000.describe())
```

```
                  x             y
count  10000.000000  10000.000000
mean      -0.007300      2.963676
std        3.543263      2.615033
min      -13.311864     -6.828857
25%       -2.398417      1.245018
50%        0.000262      2.965853
75%        2.348454      4.699709
max       15.451051     13.777433
                  x             y
count  10000.000000  10000.000000
mean       6.096272      3.174674
std        3.471670      2.308153
min       -5.672639     -5.123931
25%        3.763049      1.634243
50%        6.062341      3.139753
75%        8.488399      4.733547
```

```
max       20.355251     11.910529
                 x            y
count  1000.000000  1000.000000
mean     -0.026781     3.015787
std       3.473659     2.579017
min      -9.817560    -5.590628
25%      -2.373706     1.337761
50%       0.069495     2.981503
75%       2.400815     4.815889
max      10.001318    11.843885
```

1.2 a) Mittelwerte

[5]:
```
mu_p0 = p0.mean()
mu_p1 = p1.mean()
```

[6]:
```
print(f'mu_p0 = \n{mu_p0}')
print(f'mu_p1 = \n{mu_p1}')
```

```
mu_p0 =
x   -0.007300
y    2.963676
dtype: float64
mu_p1 =
x    6.096272
y    3.174674
dtype: float64
```

1.3 b) Kovarianzmatrix

[7]:
```
V_p0 = p0.cov()
V_p1 = p1.cov()
print(f'V_p0 = \n{V_p0}')
print(f'V_p1 = \n{V_p1}')
```

```
V_p0 =
           x         y
x  12.554716  8.360258
y   8.360258  6.838396
V_p1 =
           x        y
x  12.052492  7.21376
y   7.213760  5.32757
```

[8]:
```
V_p0p1 = V_p0 + V_p1
print(f'V_p0p1 = \n{V_p0p1}')
```

```
V_p0p1 =
           x          y
```

```
x  24.607208  15.574018
y  15.574018  12.165966
```

1.4 c) lineare Fisher-Diskriminante und Geradengleichung

$\vec{\lambda} = \lambda \cdot \vec{e_{\vec{\lambda}}}$

$\vec{\lambda} = S_W^{-1} \cdot (\vec{\mu}_1 - \vec{\mu}_2)$

[9]:
```
V_p0p1_inverse = np.linalg.inv(V_p0p1)
print(f'V_p0p1_inverse = \n{V_p0p1_inverse}')
```

```
V_p0p1_inverse =
[[ 0.21411259 -0.27409195]
 [-0.27409195  0.43306984]]
```

[10]:
```
lambda_ = V_p0p1_inverse @ (mu_p0-mu_p1)
print(f'lambda = \n{lambda_}')
```

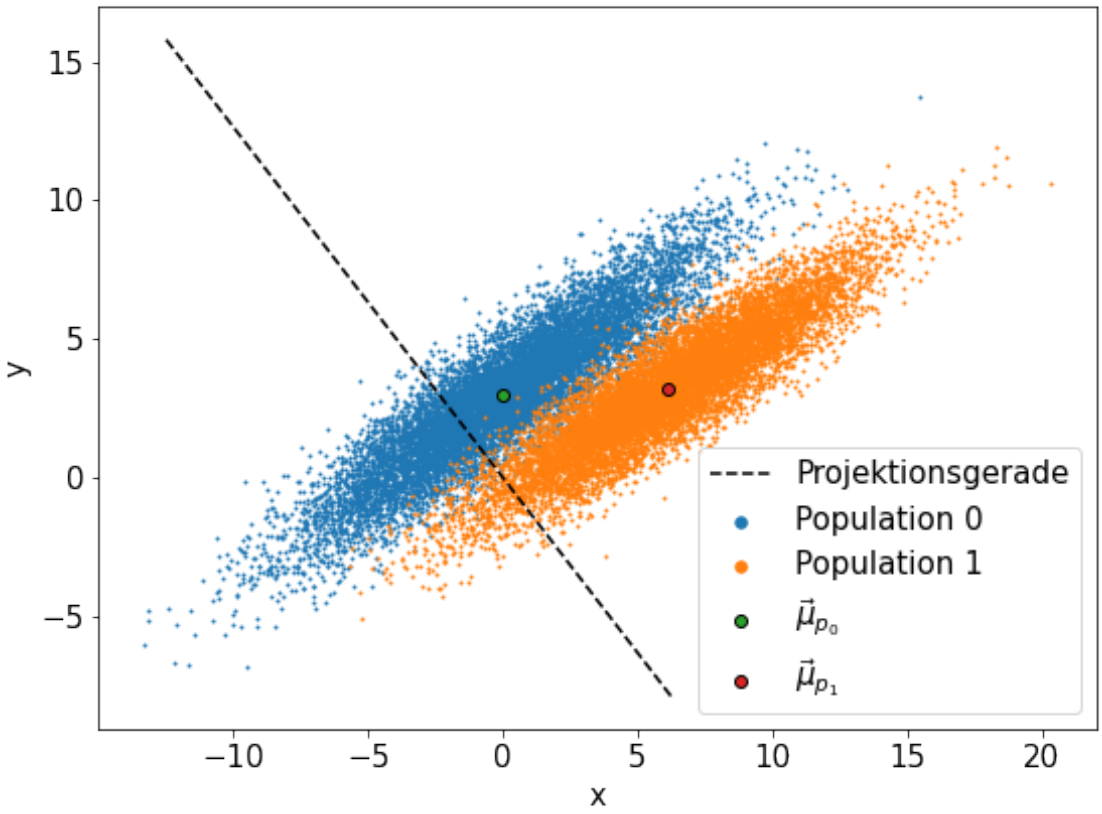
```
lambda =
[-1.24901882  1.58156321]
```

Geradengleichung: $\vec{x} = x' \cdot \vec{\lambda}$

[11]:
```
def gerade(x,lambda_):
    return x*lambda_[0],x*lambda_[1]
```

1.4.1 Plot der Populationen

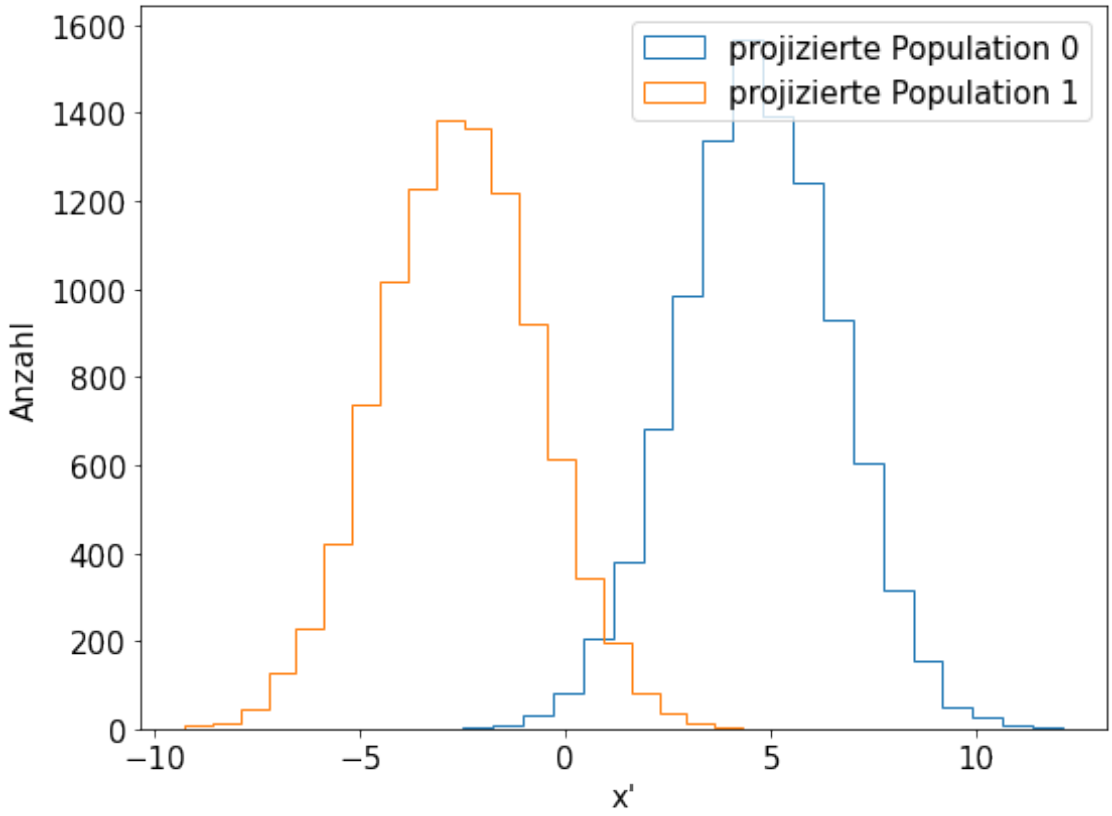
[12]:
```
plt.figure(figsize=(8,6))

plt.scatter(p0['x'],p0['y'],s=1,label='Population 0')
plt.scatter(p1['x'],p1['y'],s=1,label='Population 1')
plt.scatter(mu_p0['x'],mu_p0['y'], edgecolor='k',label=r'$\vec{\mu}_{p_0}$')
plt.scatter(mu_p1['x'],mu_p1['y'], edgecolor='k',label=r'$\vec{\mu}_{p_1}$')

projection_linspace = np.linspace(-5,10,100)
projection_line = gerade(projection_linspace,lambda_)
plt.plot(projection_line[0],projection_line[1],'k--',label='Projektionsgerade')

plt.xlabel('x')
plt.ylabel('y')

legend = plt.legend()
legend.legendHandles[1]._sizes = [30]
legend.legendHandles[2]._sizes = [30]

plt.tight_layout()
plt.show()
```



1.5 d) Projektion der Populationen

[13]:
```
p0_projected = np.dot(p0,lambda_)
p1_projected = np.dot(p1,lambda_)
```

[14]:
```
plt.figure(figsize=(8,6))
plt.hist(p0_projected,histtype='step',bins=20,label='projizierte Population 0')
plt.hist(p1_projected,histtype='step',bins=20,label='projizierte Population 1')
plt.xlabel('x\'')
plt.ylabel('Anzahl')
plt.legend()
plt.tight_layout()
plt.show()
```



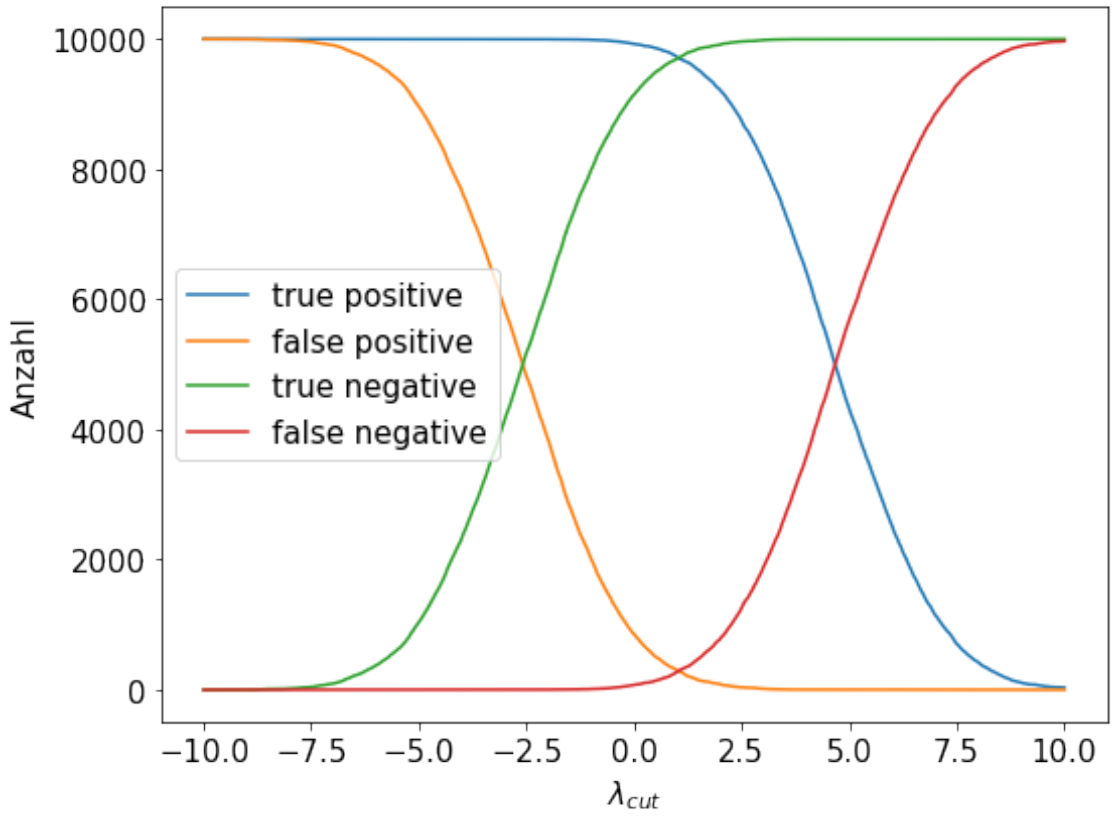
1.6 e) Vergleich der verschiedenen λ_{cut}

[15]:
```
lambda_cut_linspace = np.linspace(-10,10,10000)
```

[16]:
```
# p0 ist positiv (Signal), p1 ist negativ (Untergrund)
# true and false, positive and negative
tp = np.array([np.sum(p0_projected > lambda_cut) for lambda_cut in␣
 ↪lambda_cut_linspace])
fp = np.array([np.sum(p1_projected > lambda_cut) for lambda_cut in␣
 ↪lambda_cut_linspace])
tn = np.array([np.sum(p1_projected <= lambda_cut) for lambda_cut in␣
 ↪lambda_cut_linspace])
fn = np.array([np.sum(p0_projected <= lambda_cut) for lambda_cut in␣
 ↪lambda_cut_linspace])
```

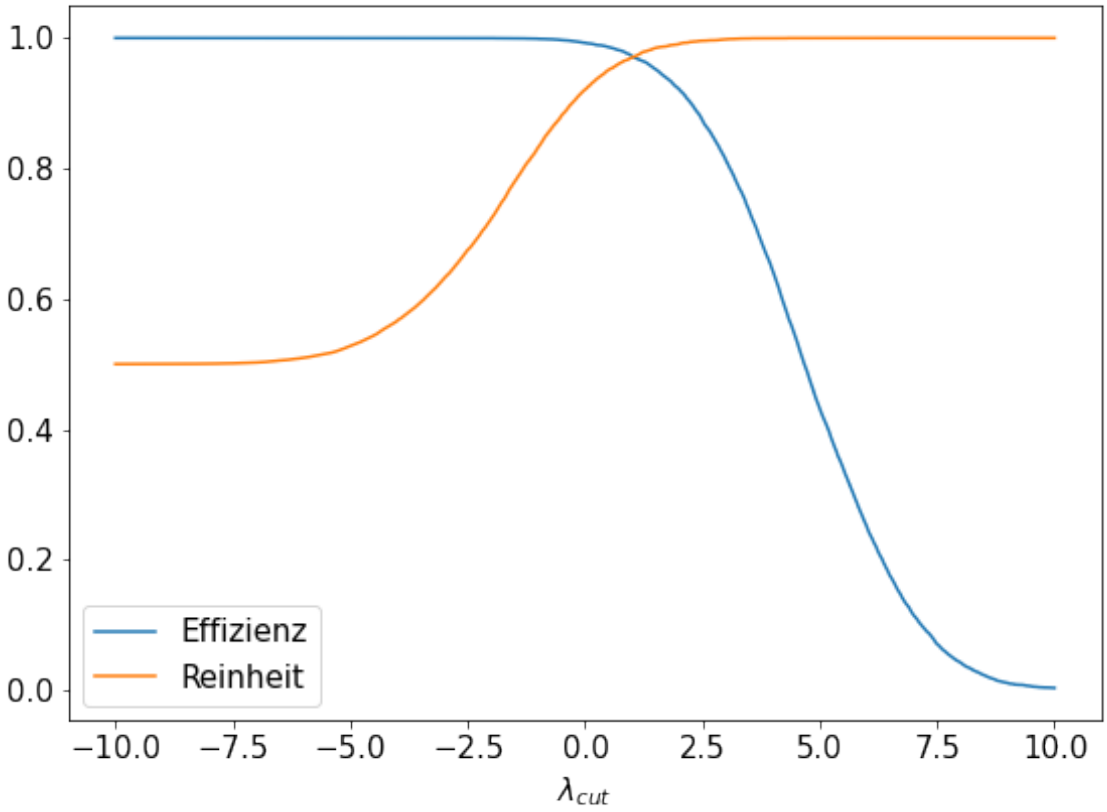
[17]:
```
plt.figure(figsize=(8,6))

plt.plot(lambda_cut_linspace, tp, label='true positive')
plt.plot(lambda_cut_linspace, fp, label='false positive')
plt.plot(lambda_cut_linspace, tn, label='true negative')
```

```
plt.plot(lambda_cut_linspace, fn, label='false negative')

plt.xlabel(r'$\lambda_{cut}$')
plt.ylabel('Anzahl')

plt.legend()
plt.tight_layout()
plt.show()
```



[18]:
```
efficiency = tp/(tp+fn)
purity = tp/(tp+fp)

plt.figure(figsize=(8,6))

plt.plot(lambda_cut_linspace,efficiency,label='Effizienz')
plt.plot(lambda_cut_linspace,purity,label='Reinheit')

plt.xlabel(r'$\lambda_{cut}$')

plt.legend()
plt.tight_layout()
```

```
plt.show()
```



[19]:
```
intersect_mask = np.isclose(efficiency,purity)
print(f'Schnittpunkt:
 ↪\n{lambda_cut_linspace[intersect_mask][0]}\n{efficiency[intersect_mask][0]}')
```

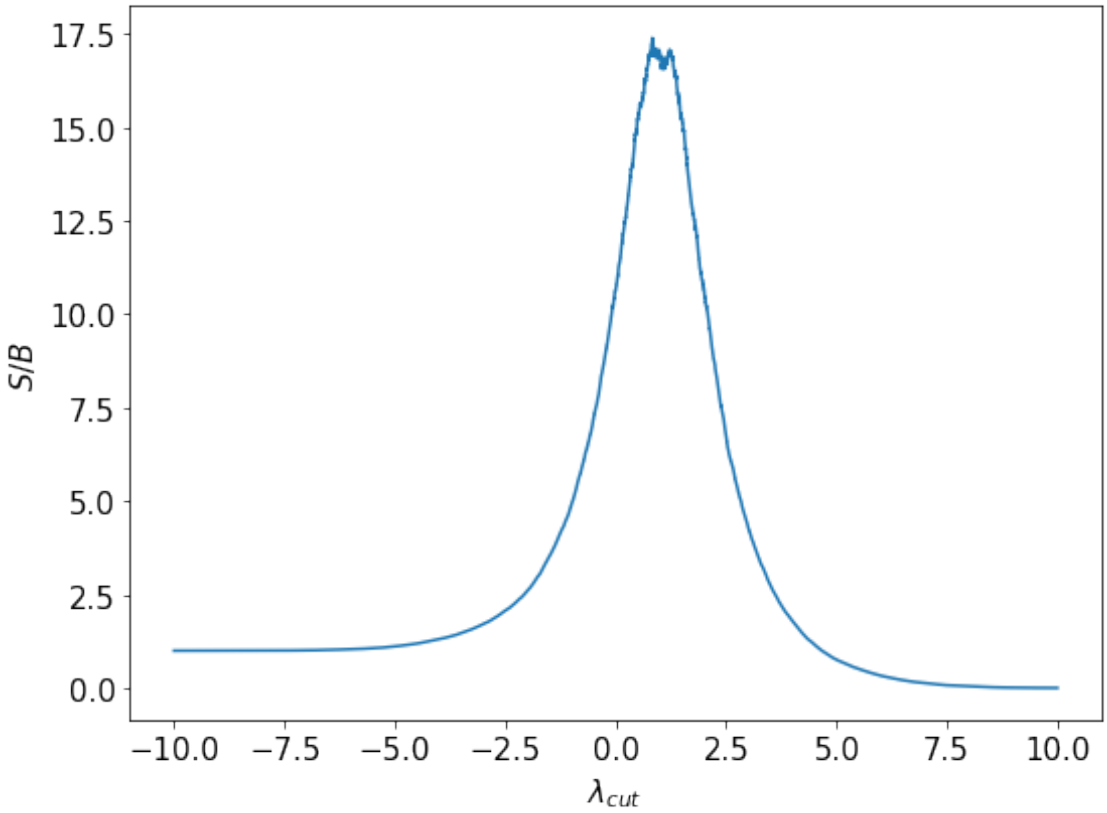
```
Schnittpunkt:
1.0331033103310325
0.9708
```

1.7 f) Bei welchem Wert von λ_{cut} wird nach der Trennung das Signal-zu-Untergrundverhältnis S/B maximal?

[20]:
```
S = tp
B = fn+fp
```

[21]:
```
SB = S/B

plt.figure(figsize=(8,6))
plt.plot(lambda_cut_linspace,S/B)
plt.xlabel(r'$\lambda_{cut}$')
```

```
plt.ylabel(r'$S/B$')
plt.tight_layout()
plt.show()
```



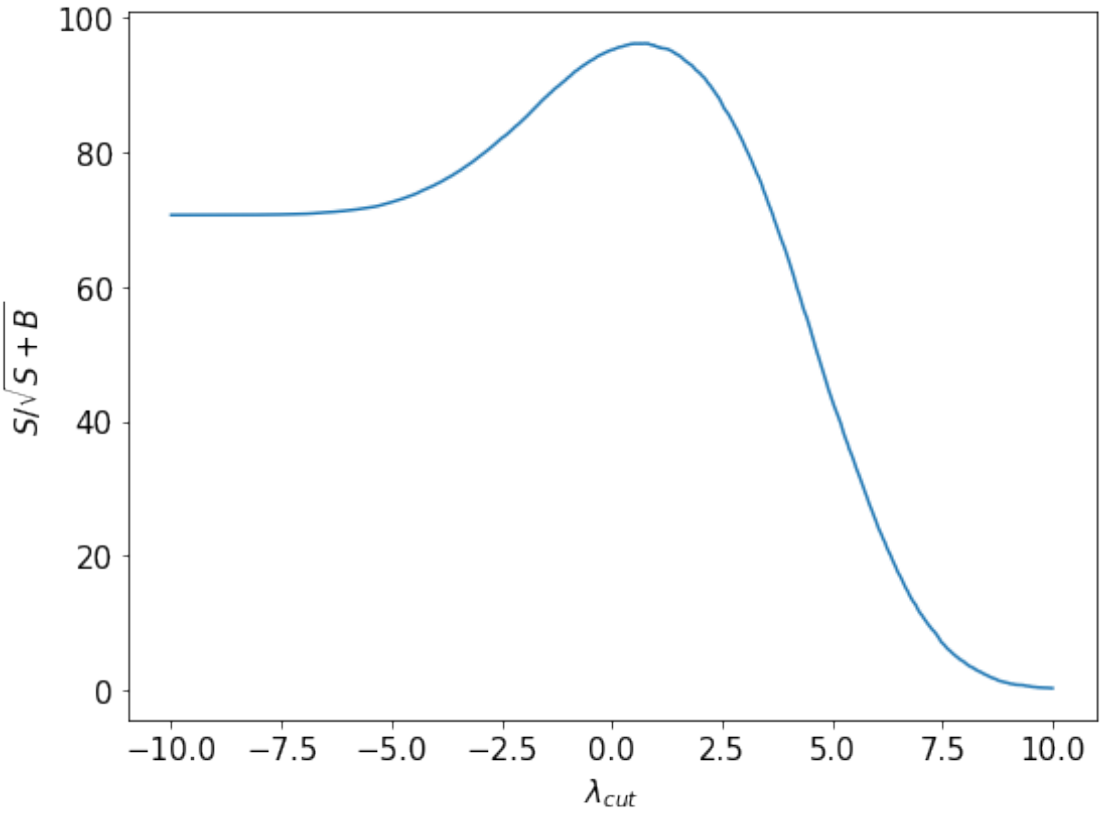
[22]:
```
lambda_cut_max_f = lambda_cut_linspace[np.argmax(SB)]
SB_max = SB[np.argmax(SB)]
print(f'Maximum von S/B:\n{lambda_cut_max_f}\n{SB_max}')
```

```
Maximum von S/B:
0.8270827082708276
17.392539964476022
```

1.8 g) Bei welchem Wert von λ_{cut} wird nach der Trennung die Signifikanz $S/\sqrt{S+B}$ maximal?

[23]:
```
significance = S/np.sqrt(S+B)

plt.figure(figsize=(8,6))
plt.plot(lambda_cut_linspace,significance)
plt.xlabel(r'$\lambda_{cut}$')
plt.ylabel(r'$S/\sqrt{S+B}$')
```

```
plt.tight_layout()
plt.show()
```



[24]:
```
lambda_cut_max_g = lambda_cut_linspace[np.argmax(significance)]
significance_max = significance[np.argmax(significance)]
print(f'Maximum von S/sqrt(S+B):\n{lambda_cut_max_g}\n{significance_max}')
```

```
Maximum von S/sqrt(S+B):
0.7630763076307634
96.240151511467
```

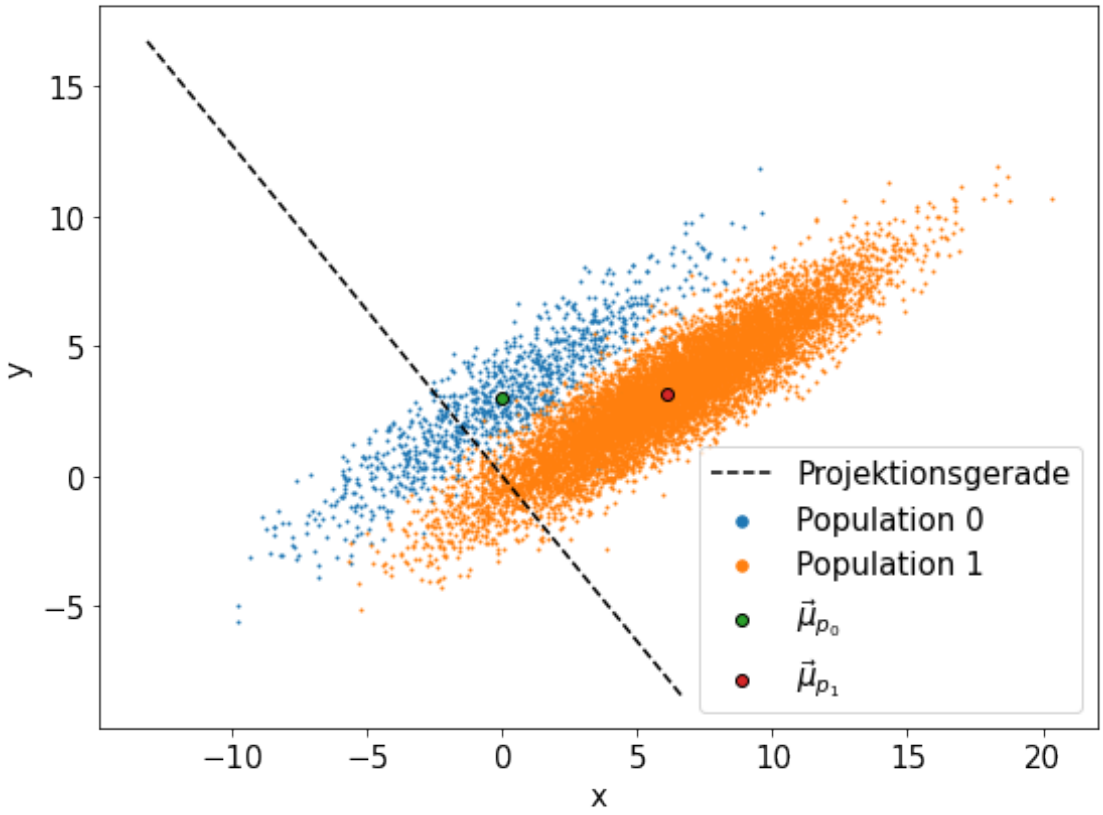
1.9 h) Wiederhole a) bis g) für $p_{0,1000}$ statt p_0

Code von vorher kopiert und an jede Variable ein '_h' geschrieben.

[25]:
```
p0_h = p0_1000
mu_p0_h = p0_1000.mean()
V_p0_h = p0_1000.cov()
V_p0p1_h = V_p0_h + V_p1
V_p0p1_inverse_h = np.linalg.inv(V_p0p1_h)
lambda_h = V_p0p1_inverse_h @ (mu_p0_h-mu_p1)
```

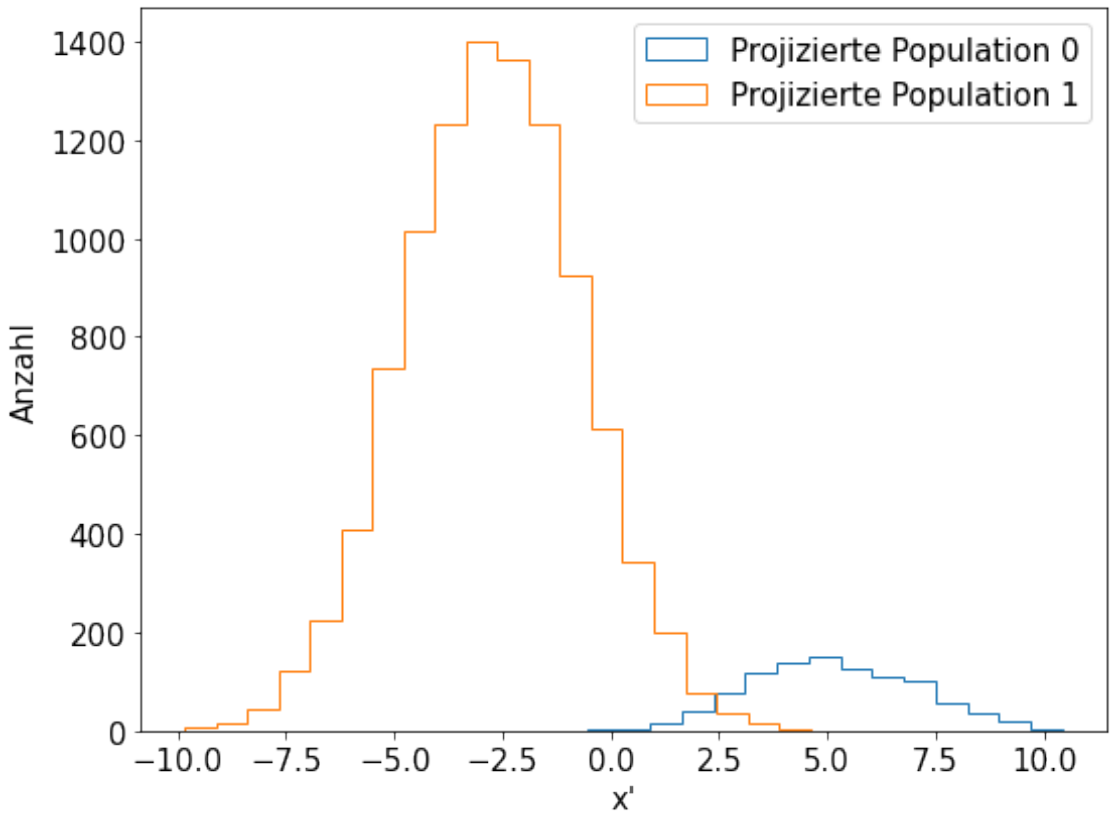
1.9.1 Plot der Populationen

[26]:
```
plt.figure(figsize=(8,6))

plt.scatter(p0_h['x'],p0_h['y'],s=1,label='Population 0')
plt.scatter(p1['x'],p1['y'],s=1,label='Population 1')
plt.scatter(mu_p0_h['x'],mu_p0_h['y'], edgecolor='k',label=r'$\vec{\mu}_{p_0}$')
plt.scatter(mu_p1['x'],mu_p1['y'], edgecolor='k',label=r'$\vec{\mu}_{p_1}$')

projection_linspace_h = np.linspace(-5,10,100)
projection_line_h = gerade(projection_linspace_h,lambda_h)
plt.
 ↪plot(projection_line_h[0],projection_line_h[1],'k--',label='Projektionsgerade')

plt.xlabel('x')
plt.ylabel('y')

legend = plt.legend()
legend.legendHandles[1]._sizes = [30]
legend.legendHandles[2]._sizes = [30]

plt.tight_layout()
plt.show()
```



1.9.2 Plot der Projektion

[27]:
```
p0_projected_h = np.dot(p0_h,lambda_h)
p1_projected_h = np.dot(p1,lambda_h)
plt.figure(figsize=(8,6))
plt.hist(p0_projected_h,histtype='step',bins=20,label='Projizierte Population␣
↪0')
plt.hist(p1_projected_h,histtype='step',bins=20,label='Projizierte Population␣
↪1')
plt.xlabel('x\'')
plt.ylabel('Anzahl')
plt.legend()
plt.tight_layout()
plt.show()
```

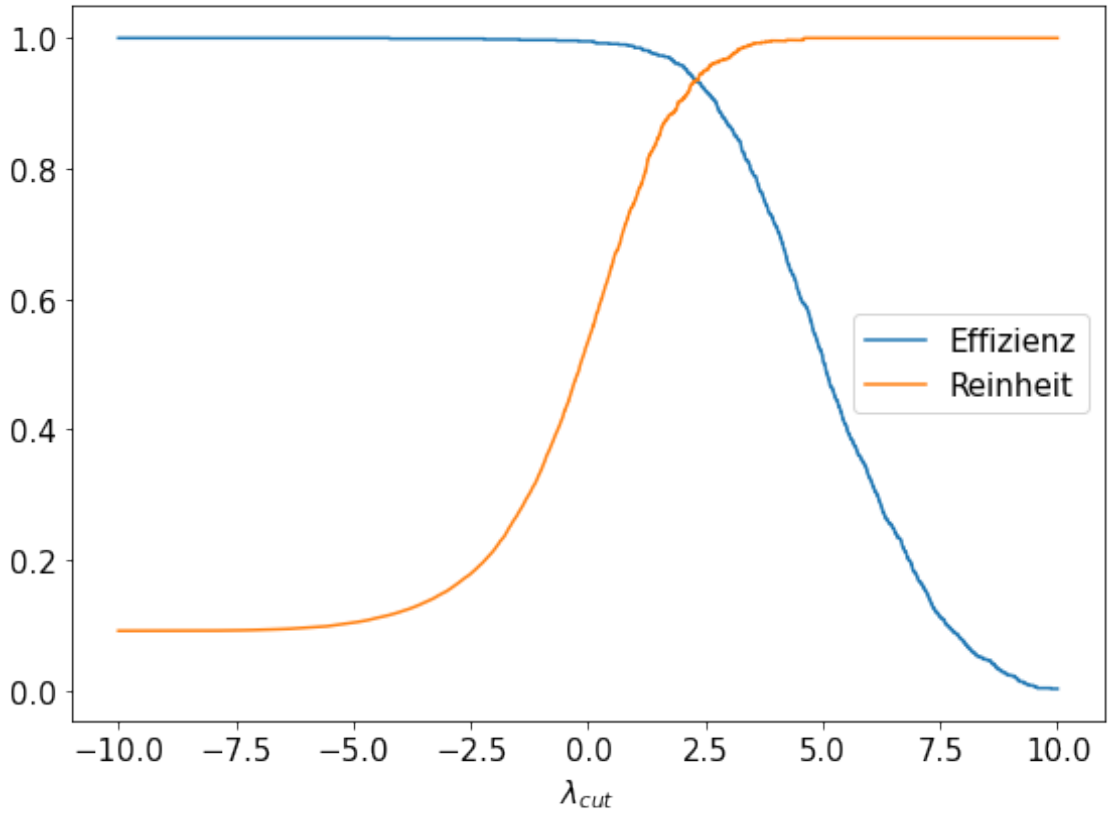


1.9.3 Vergleich von verschiedenen λ_{cut}

[28]:
```
lambda_cut_linspace_h = np.linspace(-10,10,10000)
```

[29]:
```
# p0 ist positiv (Signal), p1 ist negativ (Untergrund)
# true and false, positive and negative
tp_h = np.array([np.sum(p0_projected_h > lambda_cut) for lambda_cut in␣
 ↪lambda_cut_linspace_h])
fp_h = np.array([np.sum(p1_projected_h > lambda_cut) for lambda_cut in␣
 ↪lambda_cut_linspace_h])
tn_h = np.array([np.sum(p1_projected_h <= lambda_cut) for lambda_cut in␣
 ↪lambda_cut_linspace_h])
fn_h = np.array([np.sum(p0_projected_h <= lambda_cut) for lambda_cut in␣
 ↪lambda_cut_linspace_h])
```

[30]:
```
efficiency_h = tp_h/(tp_h+fn_h)
purity_h = tp_h/(tp_h+fp_h)

plt.figure(figsize=(8,6))

plt.plot(lambda_cut_linspace_h,efficiency_h,label='Effizienz')
plt.plot(lambda_cut_linspace_h,purity_h,label='Reinheit')

plt.xlabel(r'$\lambda_{cut}$')

plt.legend()
plt.tight_layout()
plt.show()
```

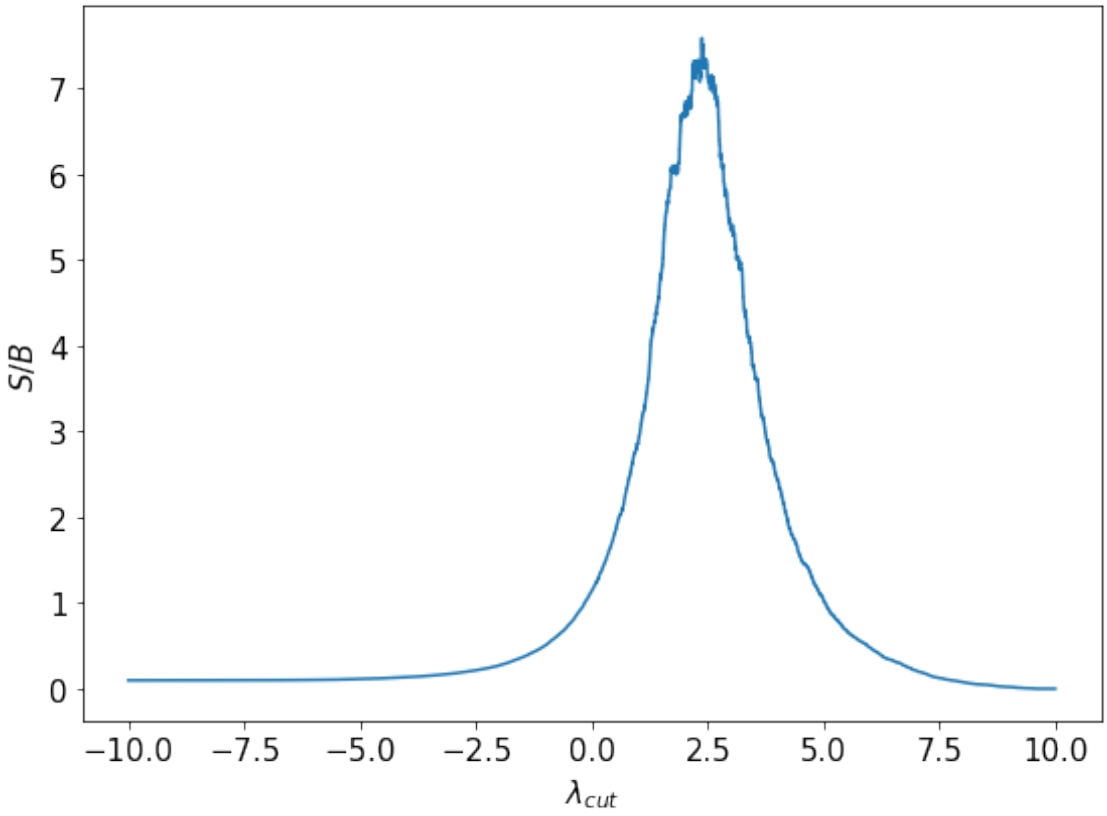


[31]:
```
intersect_mask_h = np.isclose(efficiency_h,purity_h)
print(f'Schnittpunkt:
 ↪\n{lambda_cut_linspace_h[intersect_mask_h][0]}\n{efficiency_h[intersect_mask_h][0]}')
```

```
Schnittpunkt:
2.3052305230523054
0.935
```

[32]:
```
S_h = tp_h
B_h = fn_h+fp_h
```

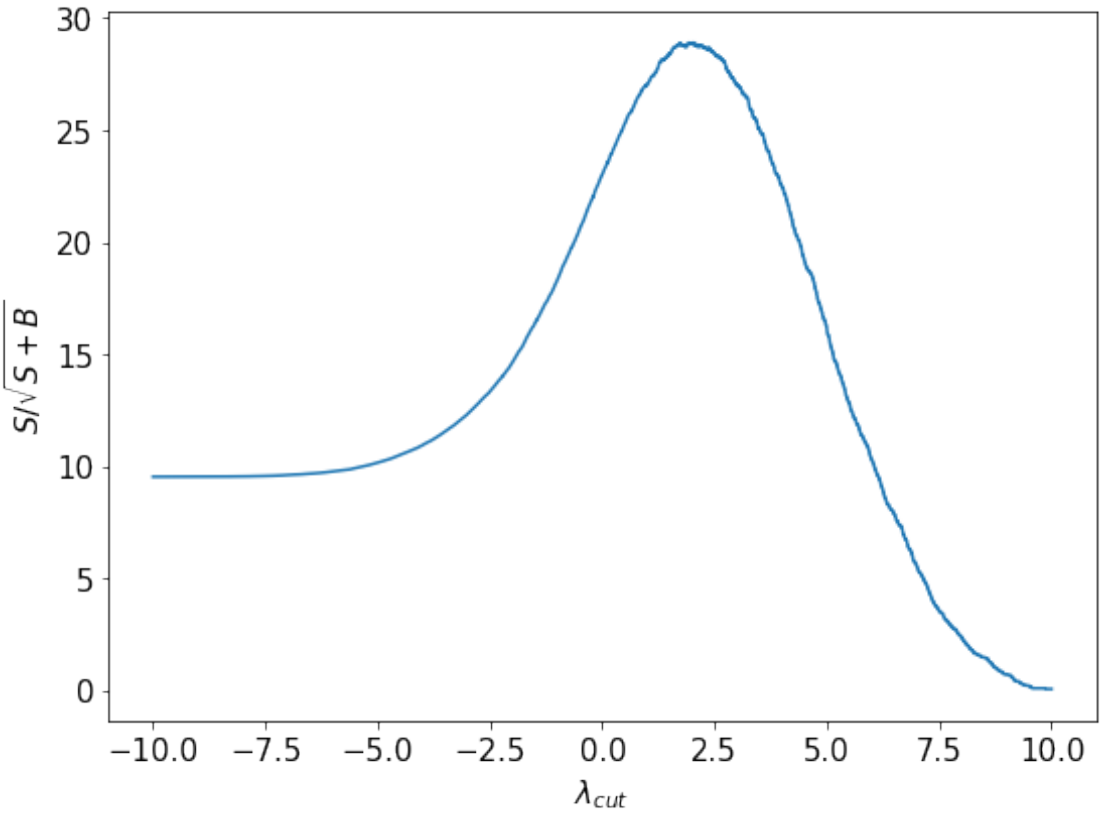
[33]:
```
SB_h = S_h/B_h

plt.figure(figsize=(8,6))
plt.plot(lambda_cut_linspace_h,SB_h)
plt.xlabel(r'$\lambda_{cut}$')
plt.ylabel(r'$S/B$')
plt.tight_layout()
plt.show()
```



[34]:
```
lambda_cut_max_f_h = lambda_cut_linspace_h[np.argmax(SB_h)]
SB_max_h = SB_h[np.argmax(SB_h)]
print(f'Maximum von S/B:\n{lambda_cut_max_f_h}\n{SB_max_h}')
```

```
Maximum von S/B:
2.373237323732374
7.5772357723577235
```

[35]:
```
significance_h = S_h/np.sqrt(S_h+B_h)

plt.figure(figsize=(8,6))
plt.plot(lambda_cut_linspace_h,significance_h)
plt.xlabel(r'$\lambda_{cut}$')
plt.ylabel(r'$S/\sqrt{S+B}$')
plt.tight_layout()
plt.show()
```



[36]:
```
lambda_cut_max_g_h = lambda_cut_linspace_h[np.argmax(significance_h)]
significance_max_h = (S/np.sqrt(S+B))[np.argmax(significance_h)]
print(f'Maximum von S/sqrt(S+B):\n{lambda_cut_max_g_h}\n{significance_max_h}')
```

```
Maximum von S/sqrt(S+B):
2.0252025202520247
91.5744319518368
```

1.9.4 Vergleich von der Analyse mit 10000 Samples zu 1000 Samples

Durch weniger Werte hat sich der Mittelpunkt der Population 0 etwas verschoben und die Projektionsgerade hat eine etwas andere Steigung.
Im Projektions-Histogram ist Population 0 deutlich flacher, da natürlich nur 1/10 der Punkte zur Verfügung stehen.

Schnittpunkte von Effizienz und Reinheit:
p_0 : $(1.033, 0.971)$
$p_{0,1000}$: $(2.305, 0.935)$
Also ist mit 1000 Samples ein geringerer Schnittpunkt von Effizienz und Reinheit zu finden.
Das heißt mit weniger Werten kann auch nur ein schlechterer Kompromiss zwischen Effizienz und Reinheit gefunden werden.
Außerdem ist der Schnittpunkt weiter rechts und schneidet somit mehr Punkte von der Signal

Population ab.

Auch S/B und $S/\sqrt{S+B}$ haben sich verschoben.
Maximum von S/B:
$p_0 : (0.827, 17.393)$
$p_{0,1000} : (2.373, 7.577)$
Maximum von $S/\sqrt{S+B}$:
$p_0 : (0.763, 96.240)$
$p_{0,1000} : (2.025, 91.574)$
Auch hier sind die Maxima deutlich geringer bei weniger Samples.
Somit zeigt sich auch hier, dass mehr Samples zu besseren Ergebnissen führen.

[ ]:

Blatt 6

Freitag, 4. Juni 2021 13:15

Aufgabe 12:

Population 0:

$(1;1), (2;1), (1.5;2), (2;2), (2;3), (3,3)$

Population 1:

$(1.5;1), (2.5;1), (3.5;1), (2.5,2), (3.5;2), (4.5;2)$

a)

Mittelwerte:

$$\vec{\mu}_i = \frac{1}{N_i} \sum_{j=1}^{N_i} \vec{x}_{i;j}$$

$$N_0 = N_1 = 6$$

$$\vec{\mu}_0 = \frac{1}{6} \begin{pmatrix} 11.5 \\ 12 \end{pmatrix} = \begin{pmatrix} \frac{11.5}{6} \\ 2 \end{pmatrix}$$

$$\vec{\mu}_1 = \frac{1}{6} \begin{pmatrix} 18 \\ 9 \end{pmatrix} = \begin{pmatrix} 3 \\ \frac{3}{2} \end{pmatrix}$$

Streumatrizen:

$$S_j = \sum_{i=1}^{N_j} (\vec{x}_{j,i} - \vec{\mu}_j)(\vec{x}_{j,i} - \vec{\mu}_j)^T$$

$$S_0 = \begin{pmatrix} -\frac{11}{12} \\ -1 \end{pmatrix} \begin{pmatrix} -\frac{11}{12} & -1 \end{pmatrix} + \begin{pmatrix} \frac{1}{12} \\ -1 \end{pmatrix} \begin{pmatrix} \frac{1}{12} & -1 \end{pmatrix} + \begin{pmatrix} -\frac{5}{12} \\ 0 \end{pmatrix} \begin{pmatrix} -\frac{5}{12} & 0 \end{pmatrix}$$
$$+ \begin{pmatrix} \frac{1}{12} \\ 0 \end{pmatrix} \begin{pmatrix} \frac{1}{12} & 0 \end{pmatrix} + \begin{pmatrix} \frac{1}{12} \\ 1 \end{pmatrix} \begin{pmatrix} \frac{1}{12} & 1 \end{pmatrix} + \begin{pmatrix} \frac{13}{12} \\ 1 \end{pmatrix} \begin{pmatrix} \frac{13}{12} & 1 \end{pmatrix}$$

$$= \begin{pmatrix} \frac{121}{144} & \frac{11}{12} \\ \frac{11}{12} & 1 \end{pmatrix} + \begin{pmatrix} \frac{1}{144} & -\frac{1}{12} \\ -\frac{1}{12} & 1 \end{pmatrix} + \begin{pmatrix} \frac{25}{144} & 0 \\ 0 & 0 \end{pmatrix}$$
$$+ \begin{pmatrix} \frac{1}{144} & 0 \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} \frac{1}{144} & \frac{1}{12} \\ \frac{1}{12} & 1 \end{pmatrix} + \begin{pmatrix} \frac{169}{144} & \frac{13}{12} \\ \frac{13}{12} & 1 \end{pmatrix}$$

$$S_0 = \begin{pmatrix} \frac{53}{24} & 2 \\ 2 & 4 \end{pmatrix}$$

$$\begin{pmatrix} -\frac{3}{2} \end{pmatrix} \begin{pmatrix} -\frac{1}{2} \end{pmatrix} + \begin{pmatrix} -\frac{1}{2} \end{pmatrix} \begin{pmatrix} -\frac{1}{2} \end{pmatrix} + \begin{pmatrix} \frac{1}{2} \end{pmatrix} \begin{pmatrix} \frac{1}{2} \end{pmatrix}$$

$$S_1 = \begin{pmatrix} -\frac{3}{2} \\ -\frac{1}{2} \end{pmatrix} \begin{pmatrix} -\frac{3}{2} & -\frac{1}{2} \end{pmatrix} + \begin{pmatrix} -\frac{1}{2} \\ -\frac{1}{2} \end{pmatrix} \begin{pmatrix} -\frac{1}{2} & -\frac{1}{2} \end{pmatrix} + \begin{pmatrix} \frac{1}{2} \\ -\frac{1}{2} \end{pmatrix} \begin{pmatrix} \frac{1}{2} & -\frac{1}{2} \end{pmatrix}$$
$$+ \begin{pmatrix} -\frac{1}{2} \\ \frac{1}{2} \end{pmatrix} \begin{pmatrix} -\frac{1}{2} & \frac{1}{2} \end{pmatrix} + \begin{pmatrix} \frac{1}{2} \\ \frac{1}{2} \end{pmatrix} \begin{pmatrix} \frac{1}{2} & \frac{1}{2} \end{pmatrix} + \begin{pmatrix} \frac{3}{2} \\ \frac{1}{2} \end{pmatrix} \begin{pmatrix} \frac{3}{2} & \frac{1}{2} \end{pmatrix}$$

$$= \begin{pmatrix} \frac{9}{4} & \frac{3}{4} \\ \frac{3}{4} & \frac{1}{4} \end{pmatrix} + \begin{pmatrix} \frac{1}{4} & \frac{1}{4} \\ \frac{1}{4} & \frac{1}{4} \end{pmatrix} + \begin{pmatrix} \frac{1}{4} & -\frac{1}{4} \\ -\frac{1}{4} & \frac{1}{4} \end{pmatrix}$$
$$+ \begin{pmatrix} \frac{1}{4} & -\frac{1}{4} \\ -\frac{1}{4} & \frac{1}{4} \end{pmatrix} + \begin{pmatrix} \frac{1}{4} & \frac{1}{4} \\ \frac{1}{4} & \frac{1}{4} \end{pmatrix} + \begin{pmatrix} \frac{9}{4} & \frac{3}{4} \\ \frac{3}{4} & \frac{1}{4} \end{pmatrix}$$

$$S_1 = \begin{pmatrix} \frac{11}{2} & \frac{3}{2} \\ \frac{3}{2} & \frac{3}{2} \end{pmatrix}$$

$$S_W = S_0 + S_1 = \begin{pmatrix} \frac{53}{24} & 2 \\ 2 & 4 \end{pmatrix} + \begin{pmatrix} \frac{11}{2} & \frac{3}{2} \\ \frac{3}{2} & \frac{3}{2} \end{pmatrix}$$
$$= \begin{pmatrix} \frac{185}{24} & \frac{7}{2} \\ \frac{7}{2} & \frac{11}{2} \end{pmatrix}$$

$$S_B = (\vec{\mu}_0 - \vec{\mu}_1)(\vec{\mu}_0 - \vec{\mu}_1)^T$$

$$\vec{\mu}_0 = \begin{pmatrix} \frac{11.5}{6} \\ 2 \end{pmatrix} \qquad \vec{\mu}_1 = \begin{pmatrix} 3 \\ \frac{3}{2} \end{pmatrix}$$

$$S_B = \begin{pmatrix} -\frac{13}{12} \\ \frac{1}{2} \end{pmatrix} \begin{pmatrix} -\frac{13}{12} & \frac{1}{2} \end{pmatrix}$$

$$= \begin{pmatrix} \frac{169}{144} & -\frac{13}{24} \\ -\frac{13}{24} & \frac{1}{4} \end{pmatrix}$$

b) Wie lautet $\vec{\lambda}$?

$$\vec{\lambda}^* = S_W^{-1} (\vec{\mu}_0 - \vec{\mu}_1)$$

Inverse von 2×2 Matrix $\begin{pmatrix} a & b \\ c & d \end{pmatrix}^{-1} = \frac{1}{ad-bc} \begin{pmatrix} d & -b \\ -c & a \end{pmatrix}$

$$S_W = \begin{pmatrix} \frac{185}{24} & \frac{7}{2} \\ \frac{7}{2} & \frac{11}{2} \end{pmatrix} \qquad \vec{\mu}_0 = \begin{pmatrix} \frac{11.5}{6} \\ 2 \end{pmatrix} \qquad \vec{\mu}_1 = \begin{pmatrix} 3 \\ \frac{3}{2} \end{pmatrix}$$

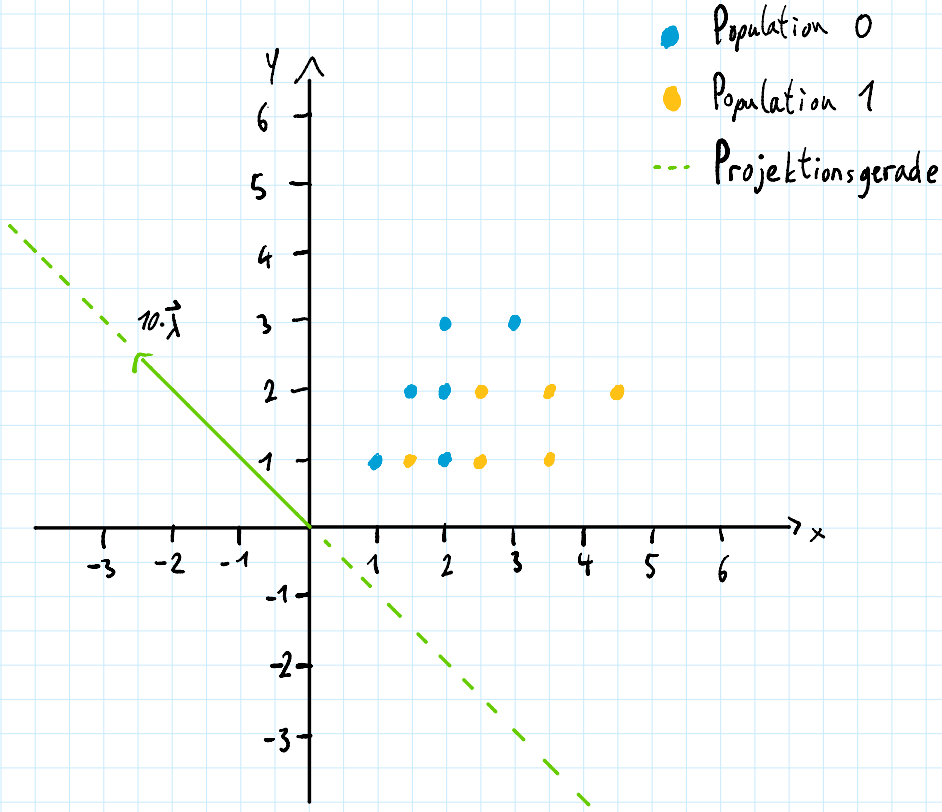
$$S_W^{-1} = \frac{1}{\frac{185}{24} \cdot \frac{11}{2} - \frac{7}{2} \cdot \frac{7}{2}} \begin{pmatrix} \frac{11}{2} & -\frac{7}{2} \\ -\frac{7}{2} & \frac{185}{24} \end{pmatrix}$$

$$= \frac{48}{1447}\begin{pmatrix} \frac{11}{2} & -\frac{7}{2} \\ -\frac{7}{2} & \frac{185}{24} \end{pmatrix}$$

$$\vec{\lambda} = \frac{48}{1447}\begin{pmatrix} \frac{11}{2} & -\frac{7}{2} \\ -\frac{7}{2} & \frac{185}{24} \end{pmatrix}\begin{pmatrix} -\frac{11}{12} \\ \frac{1}{2} \end{pmatrix}$$

$$= \frac{48}{1447}\begin{pmatrix} -\frac{185}{24} \\ \frac{367}{48} \end{pmatrix} = \frac{1}{1447}\begin{pmatrix} -370 \\ 367 \end{pmatrix} \approx \begin{pmatrix} -0.2557 \\ +0.2536 \end{pmatrix}$$

c)



d)

Projektion: $x'_{j,i} = \vec{\lambda} \cdot \vec{x}_{j,i}$

$$\vec{\lambda} = \frac{1}{1447}\begin{pmatrix} -370 \\ 367 \end{pmatrix}$$

Population 0:

$(1;1)$, $(2;1)$, $(1.5;2)$, $(2;2)$, $(2;3)$, $(3,3)$

Population 1:

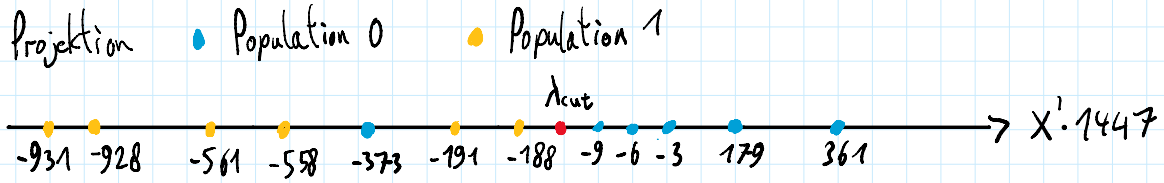
$(1.5;1)$, $(2.5;1)$, $(3.5;1)$, $(2.5,2)$, $(3.5;2)$, $(4.5;2)$

$$x'_{0,1} = -\frac{3}{1447} \quad x'_{0,2} = -\frac{373}{1447} \quad x'_{0,3} = \frac{179}{1447}$$

$$x'_{0,4} = -\frac{6}{1447} \quad x'_{0,5} = \frac{361}{1447} \quad x'_{0,6} = -\frac{9}{1447}$$

$$x'_{1,1} = -\frac{188}{1447} \quad x'_{1,2} = -\frac{558}{1447} \quad x'_{1,3} = -\frac{928}{1447}$$

$$x'_{1,4} = -\frac{191}{1447} \quad x'_{1,5} = -\frac{561}{1447} \quad x'_{1,6} = -\frac{931}{1447}$$



e) $\lambda_{cut} = -\frac{98,5}{1447}$

$$\text{Effizienz} = \frac{tp}{tp+fn} = \frac{6}{6+0} = 1$$

$$\text{Reinheit} = \frac{tp}{tp+fp} = \frac{6}{6+1} = \frac{6}{7} = 0.857$$

Den Parameter haben wir so gewählt, dass möglichst viele Punkte richtig klassifiziert wurden.