SML HW1

29-176004 奥村 恭平 *†

May 6, 2018

1 パターン認識の実例: 対テロ警備

今，東京と鳥取という 2 つの街をテロリストが攻めようとしており，警察はテロリストから街を守ろうとしている状況を考える．x で，テロリストに対する情報パターン (過去の活動・国際情勢 etc.) を表すとする．警察の人員には限りがあるので，どちらか一つの街しか守れないとする．$p(y \mid x)$ で,「パターン x の下で，テロリストが街 y を攻める確率」とする．x を所与とし，いま，

$$p(y \mid x) := \begin{cases} 0.7 & (y = \text{鳥取}) \\ 0.3 & (y = \text{東京}) \end{cases}$$

だったとする．

損失を以下のように定義する．

$$l_{\text{東京, 鳥取}} := -10000000,\ l_{\text{鳥取, 東京}} := -500000$$

これは，東京の方が鳥取より人口が多いことに対応している．これは確かに，誤って識別した場合の損失がカテゴリによって異なるようなパターン認識の実例になっている．

このとき，例えば，期待損失を計算してみると，

$$R(y = \text{東京} \mid x) = 0.3 \cdot (-10000000) = -3000000$$
$$R(y = \text{鳥取} \mid x) = 0.7 \cdot (-500000) = -350000$$

となり，$\widehat{y} := \arg\min_y R(y \mid x) = \text{東京}$ となることがわかる．

2 計算機演習

2.1 宿題 1

ソースコード 1: correlation

```
import numpy as np
n = 1000

x = np.random.randn(n)
y1 = x + np.random.randn(n)
y2 = -x + np.random.randn(n)
y3 = np.random.randn(n)
y4 = x**2 + np.random.randn(n)

print(np.corrcoef(x, y1)[0, 1])
print(np.corrcoef(x, y2)[0, 1])
print(np.corrcoef(x, y3)[0, 1])
print(np.corrcoef(x, y4)[0, 1])
```

*E-mail: kyohei.okumura@gmail.com
†東京大学大学院 経済学研究科 M2

```
14
15 Output:
16 0.729195186123
17 -0.731054853932
18 0.020960020396
19 0.0511090973279
```

相関係数の正負は大まかには，プロットしたときに散布図が右上がりか右下がりかを表している．しかし，3 つ目と 4 つ目の例を比べてみるとわかるように，相関係数が低いことは，二つの変量の間に規則的な関係がないことを必ずしも意味しない．(4 つ目の例においては，x, y の間の相関係数はほぼ 0 だが，y は「x の 2 乗 + 誤差項」という形で生成されており，明らかに両者の間には規則的な関係がある．しかし，相関係数はこの関係性までは捉えられない．)

2.2 宿題 2

以下のようなコードを用いて，二次元正規分布の密度関数の等高線をプロットした．言語は python.

ソースコード 2: 2d Gaussian

```
1  import numpy as np
2  import matplotlib.pyplot as plt
3  from mpl_toolkits.mplot3d import axes3d
4  from matplotlib import cm
5
6  fig = plt.figure()
7  ax = fig.add_subplot(111, projection='3d')
8
9  x = y = np.arange(-5, 5, 0.1)
10 X, Y = np.meshgrid(x, y)
11
12 mu = np.array([0,0])
13
14 sigma = np.array([[2, 1],
15                    [1, 4]])
16
17 det = np.linalg.det(sigma)
18 inv_sigma = np.linalg.inv(sigma)
19
20 def gauss_2d(x, y):
21     x_c = np.array([x, y]) - mu
22     return np.exp(- x_c.dot(inv_sigma).dot(x_c[np.newaxis, :].T) / 2.0) /
           (2*np.pi*np.sqrt(det))
23
24 Z = np.vectorize(gauss_2d)(X, Y)
25
26 ax.plot_surface(X, Y, Z, rstride=1, cstride=1, cmap=cm.coolwarm)
27 plt.show()
```

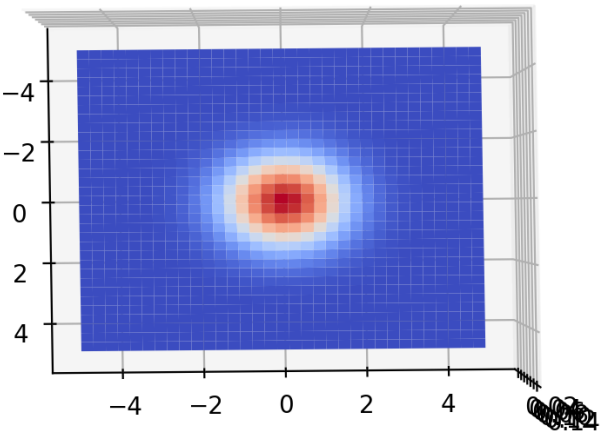

Figure 1: Σ_1

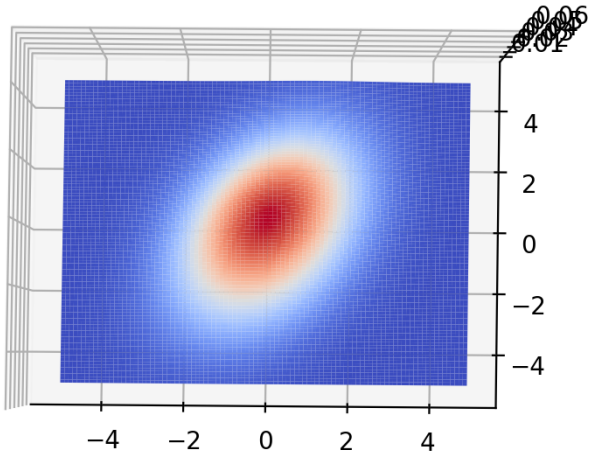

Figure 2: Σ_2

上の図は，それぞれ，以下のように分散共分散行列を定めた場合の等高線の図である．

$$\Sigma_1 := \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \Sigma_2 := \begin{bmatrix} 2 & 1 \\ 1 & 4 \end{bmatrix}.$$

Σ_2 に対応するグラフに着目する．計算すると，Σ_2^{-1} の固有値は $\lambda_1 := 0.63060194, \lambda_2 := 0.22654092$, 対応する固有ベクトルは，$x_1 := [0.92387953, -0.38268343]^\top, x_2 := [0.38268343, 0.92387953]^\top$ となる．

- 確率密度関数の裾野が，固有ベクトルの方向に向かって伸びている (等高線の楕円の軸が固有ベクトルを座標軸とみたものになっている) こと．
- 固有値が大きい固有ベクトルに対応する方向ほど，山の傾斜が急であること．

が観察される．以下，任意の分散共分散行列に対して，これらの性質 (*) が成立することをもう少しキチンと示す．

簡単のため，$\mu = 0$ とする．Σ を 2×2 の半正定値対称行列とし，固有値を $\lambda_1 \geq \lambda_2 \geq 0$, 対応する固有ベクトル (長さ 1 に正規化済み) を ϕ_1, ϕ_2 とする．いま，$x \in \mathbb{R}^2$ を任意に固定する．固有ベクトルは基底を為すので，

$$x = x_1\phi_1 + x_2\phi_2$$

と表せることに注意すると，$\Sigma = \lambda_1\phi_1\phi_1^\top + \lambda_2\phi_2\phi_2^\top$ であることを用いると，

$$x^\top \Sigma x = x_1^2\lambda_1 + x_2^2\lambda_2$$

となることが簡単な計算により示せる．

二次元正規分布の密度関数の，指数関数の中身の部分に着目する:

$$-\frac{1}{2}x^\top\Sigma^{-1}x = -\frac{1}{2}(x_1^2\lambda_1 + x_2^2\lambda_2)$$

これは，(*) が確かに成立することを示している．例えば，x の ϕ_1 成分 x_1 が大きくなると，λ_1 に対応する速さで確率密度関数の値は減少する．