Exercises 2

Antti Härkönen

1.

Start by expanding the formula for $ \sigma_{XY} $ $ \sigma_{XY} = E[(X − EX)(Y − EY)] = E[XY - Y(EX) - X(EY) + (EY)(EX)] = E(XY) - E[YE(X)] - E[XE(X)] + E[(EY)(EX)] $

If X and Y are independent, then E(XY) = E(X)E(Y), therefore $ E(XY) - E[YE(X)] - E[XE(X)] + E[(EY)(EX)] = E(X)E(Y) - E[YE(X)] - E[XE(X)] + E[(EY)(EX)] $

Since E(cX) = cE(X) when c is a constant, and expected value is a constant, $ E(X)E(Y) - E[YE(X)] - E[XE(X)] + E[(EY)(EX)] = E(X)E(Y) - E(X)E(Y) - E(X)E(Y) + E(X)E(Y) = 0 $.

Since $ \rho_{XY} = \frac{\sigma_{XY}}{{\sigma_{XX} \sigma_{YY}}} $, if $ \sigma_{XY} = 0 $ then $ \rho_{XY} = 0 $.

2.

In this case the reasonable starting point is to incorporate the disease base rate in the population as prior information in Bayesian analysis. The probability to be determined is $ P(C|T) $, where C is "A has cancer" and T means "A tests positive". Using Bayes's theorem, this can be calculated like this $ P(C|T) = \frac{P(C)P(T|C)}{P(T)} = \frac{P(C)P(T|C)}{P(C)P(T|C) + P(\neg C)P(T| \neg C)} $ Using the probabilities provided, the posterior probability is $ \frac{0.01 \times (1-0.05)}{0.01 \times (1-0.05) + (1-0.01) \times 0.1} = \frac{0.0095}{0.0095 + 0.099} = 0.0876 $

The true probability of having cancer is 8.8 %. However, this problem, as it is commonly presented, assumes that there is no information about the person other than the test result. In reality, things such as age, sex or home location would be used for more accurate prior information.

3.

(1)

In this case $ E(g(X)) = g(E(X)) $ is true.

Therefore, $ E_Y [E_X [X|Y]] = E_X [E_Y [X|Y]] = E[X] $.

(2)

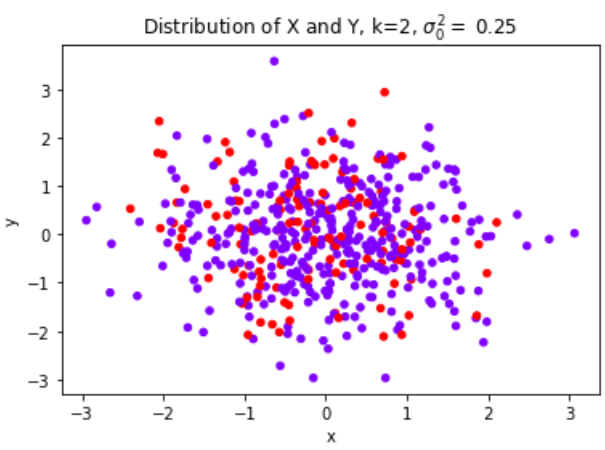
$ Var[X] = E[(X-EX)^2] $ $ = E[(Y-E(Y|X)+E(Y|X)-E(X))^2] $ $ = E_Y[E_X [Y-E(Y|X)+E(Y|X)-E(X)]^2] $ $ = E_Y[(E_X[Y|X] - E_X[X])^2] + E_Y[(E_X[Y|X] - E_X[X])^2] $ $ = E_Y[Var_X[Y|X]] + Var_Y[E_X[X|Y]] $

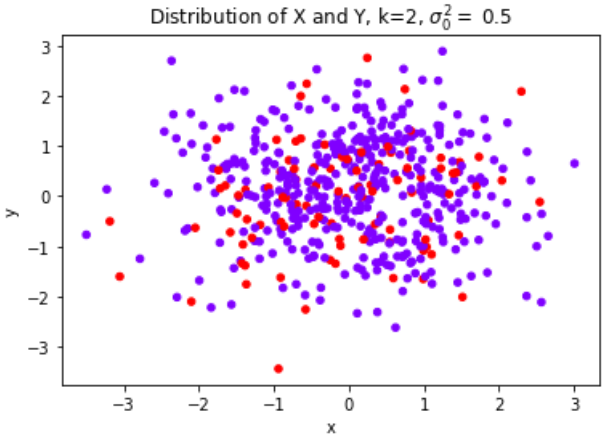
4.

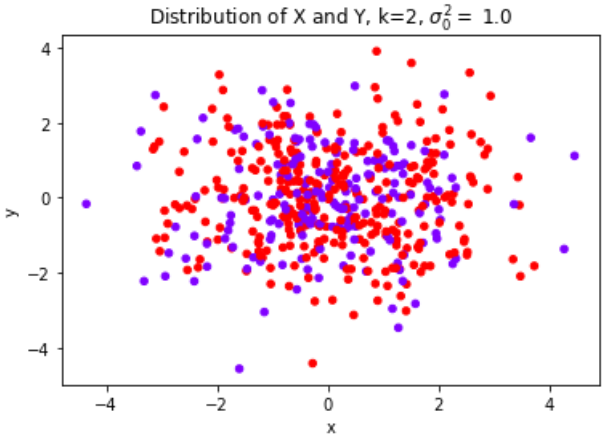
In [45]:

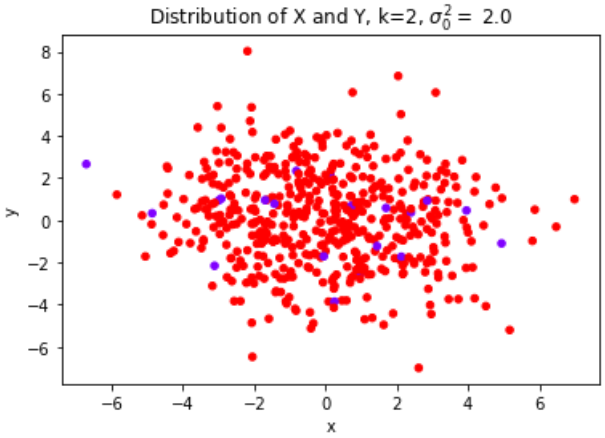
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import tensorflow_probability as tfp
tfd = tfp.distributions

n = 500
K = 2, 4, 16, 256
sigmas = 0.25, 0.5, 1., 2., 4.

for k in K:
    for sigma_0 in sigmas:
        alpha = np.ones(k, dtype=np.float)
        theta = tfd.Dirichlet(alpha).sample(1).numpy()
        z = tfd.Categorical(probs=theta).sample(n).numpy()

        mu_x = tfd.Normal(np.zeros(k), np.repeat(sigma_0, k)).sample(n).numpy()
        mu_zx = np.squeeze(np.take_along_axis(mu_x, z, 1))
        x = np.squeeze(tfd.Normal(mu_zx, np.ones(n)).sample(1).numpy())

        mu_y = tfd.Normal(np.zeros(k), np.repeat(sigma_0, k)).sample(n).numpy()
        mu_zy = np.squeeze(np.take_along_axis(mu_y, z, 1))
        y = np.squeeze(tfd.Normal(mu_zy, np.ones(n)).sample(1).numpy())
```

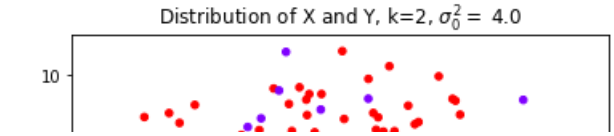
```
xy = pd.DataFrame({'x': x, 'y': y, 'k': np.squeeze(z)})
fig = xy.plot(x='x', y='y', c='k', kind='scatter', colormap='rainbow', colorbar=False)
fig.set_title(f"Distribution of X and Y, k={k}, $ \sigma^2_0=$ {sigma_0}")
plt.show()
```

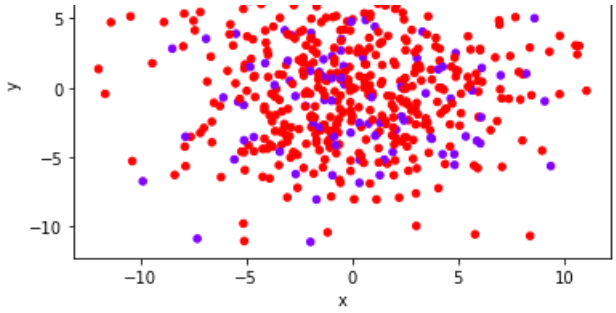


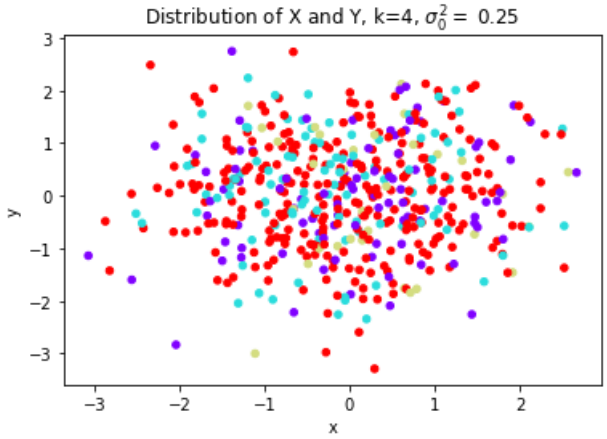


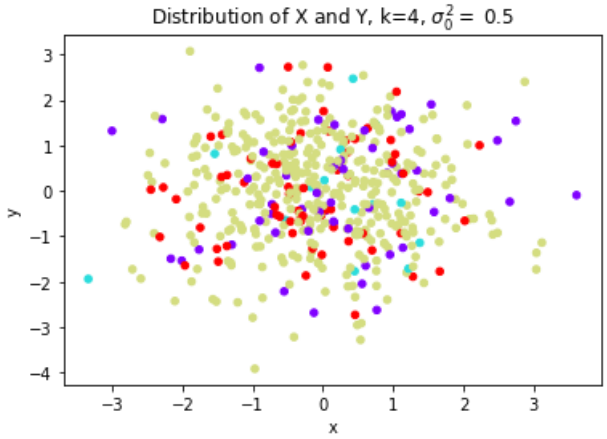


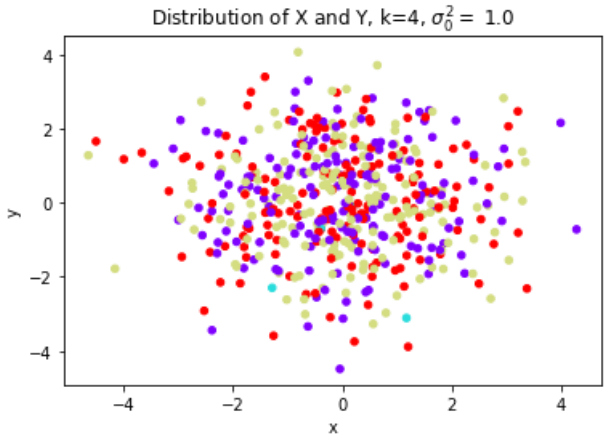


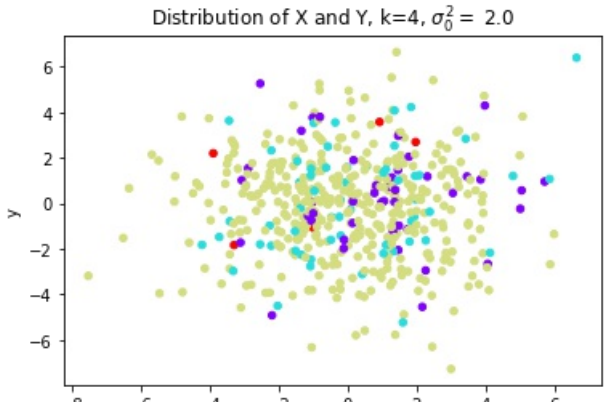


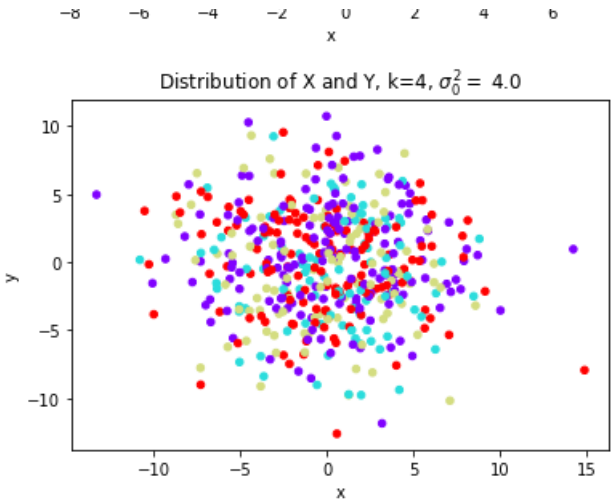

5
0
−5
−10
y
−10
−5
0
5
10
x

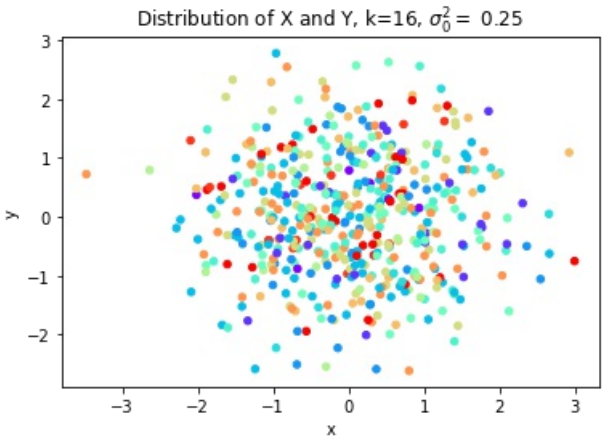

Distribution of X and Y, k=4, $\sigma_0^2 =$ 0.25
3
2
1
0
−1
−2
−3
y
−3
−2
−1
0
1
2
x

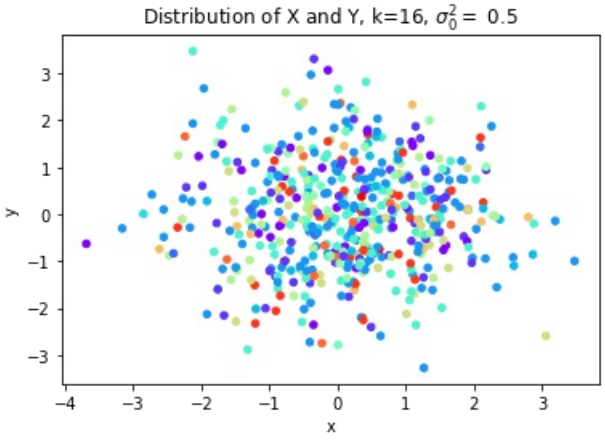

Distribution of X and Y, k=4, $\sigma_0^2 =$ 0.5
3
2
1
0
−1
−2
−3
−4
y
−3
−2
−1
0
1
2
3
x

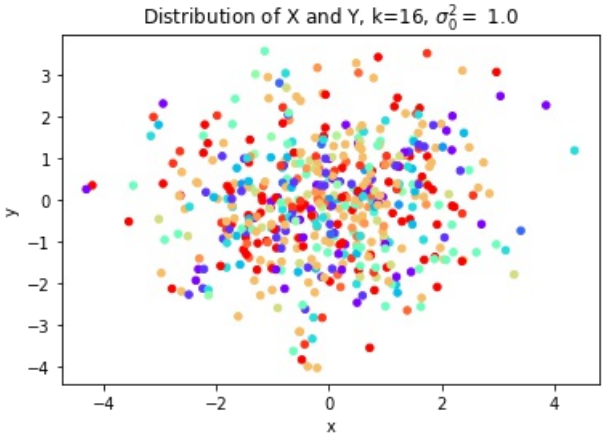

Distribution of X and Y, k=4, $\sigma_0^2 =$ 1.0
4
2
0
−2
−4
y
−4
−2
0
2
4
x

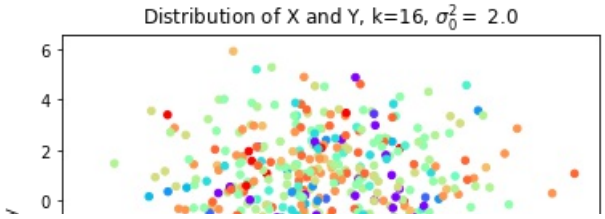

Distribution of X and Y, k=4, $\sigma_0^2 =$ 2.0
6
4
2
0
−2
−4
−6
y

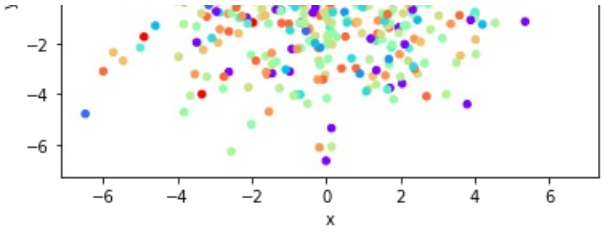

Distribution of X and Y, k=4, $\sigma_0^2 =$ 4.0
x
y

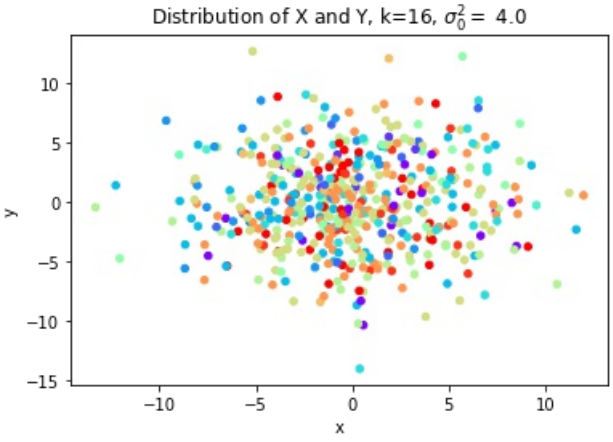

Distribution of X and Y, k=16, $\sigma_0^2 =$ 0.25
x
y

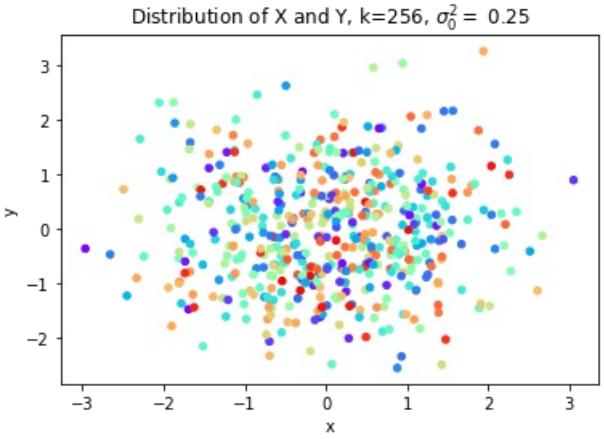

Distribution of X and Y, k=16, $\sigma_0^2 =$ 0.5
x
y

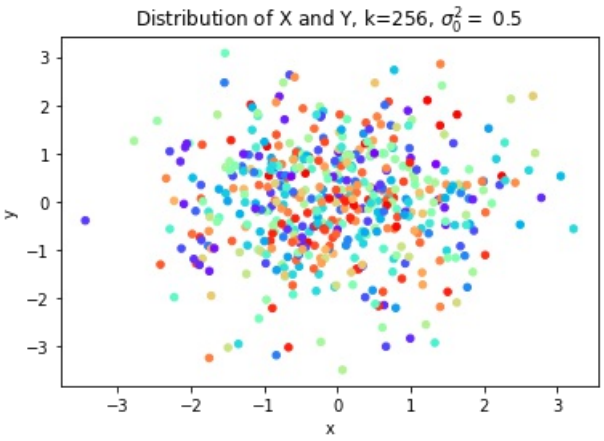

Distribution of X and Y, k=16, $\sigma_0^2 =$ 1.0
x
y

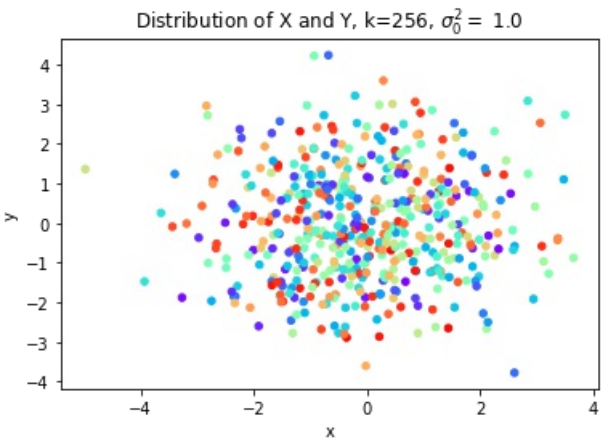

Distribution of X and Y, k=16, $\sigma_0^2 =$ 2.0



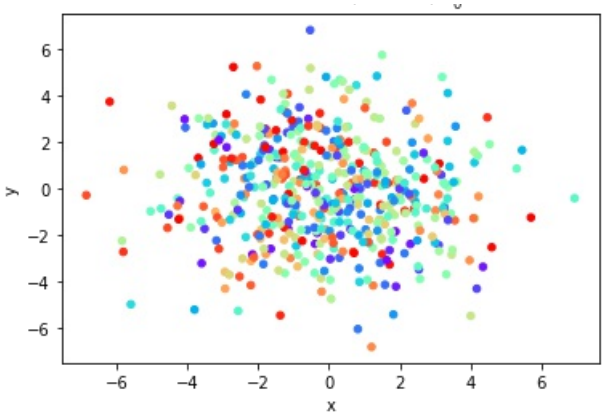


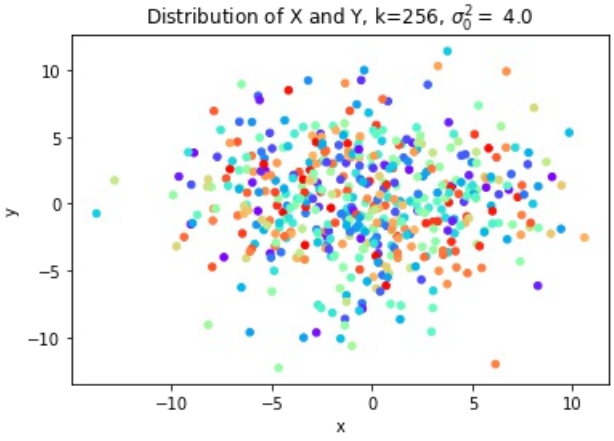






Distribution of X and Y, k=256, $\sigma_0^2 =$ 2.0


6
4
2
0
−2
−4
−6
y
−6
−4
−2
0
2
4
6
x


Distribution of X and Y, k=256, $\sigma_0^2 = 4.0$
10
5
0
−5
−10
y
−10
−5
0
5
10
x