COMP90051

Workshop Week 08

About the Workshops

- 7 sessions in total
  - Tue 12:00-13:00 AH211
  - Tue 12:00-13:00 AH108 *
  - Tue 13:00-14:00 AH210
  - Tue 16:15-17:15 AH109
  - Tue 17:15-18:15 AH236 *
  - Tue 18:15-19:15 AH236 *
  - Fri 14:15-15:15 AH211

About the Workshops

- Homepage
  - https://trevorcohn.github.io/comp90051-2017/workshops

- Solutions will be released on next Friday (a week later).

Syllabus

1	Introduction; Probability theory	Probabilistic models; Parameter fitting	
2	Linear regression; Intro to regularization	Logistic regression; Basis expansion	
3	Optimization; Regularization	Perceptron	
4	Backpropagation	CNNs; Auto-encoders	
5	Hard-margin SVMs	Soft-margin SVMs	
6	Kernel methods	Ensemble Learning	
7	Clustering	EM algorithm	←
8	Dimensionality reduction; Principal component analysis	Multidimensional scaling; Spectral clustering	
9	Bayesian fundamentals	Bayesian inference with conjugate priors	
10	PGMs, fundamentals	Conditional independence	
11	PGMs, inference	Belief propagation	
12	Statistical inference; Apps	Subject review	

Outline

- Review the lecture, background knowledge, etc.
  - Multivariate Gaussian distribution
    - Estimate parameters (for 1-d, 2-d Gaussian)
  - Probabilistic graphical models (PGM)
    - Parameters and variables
    - An example: Gaussian mixture models (GMM)
      - Generative process, joint distribution factorization, plate notation
  - Expectation maximization (EM) algorithm for GMM
- IPython notebook task: GMM

Outline

- Review the lecture, background knowledge, etc.
  - Multivariate Gaussian distribution
    - Estimate parameters (for 1-d, 2-d Gaussian)
  - Probabilistic graphical models (PGM)
    - Parameters and variables
    - An example: Gaussian mixture models (GMM)
      - Generative process, joint distribution factorization, plate notation
  - Expectation maximization (EM) algorithm for GMM

- IPython notebook task: GMM

Mean, variance, and covariance

- $X_1 = [1,2,3,4,5], \quad X_2 = [1,3,4,5,7]$
- $\mu_1 = 3, \quad \mu_2 = 4$
- $\mathrm{Var}(X_1) = \frac{1}{5}[(-2)^2 + (-1)^2 + 0^2 + 1^2 + 2^2] = 2 = \sigma_1^2$
- $\mathrm{Var}(X_2) = \frac{1}{5}[(-3)^2 + (-1)^2 + 0^2 + 1^2 + 3^2] = 4 = \sigma_2^2$
- $$\begin{aligned}\mathrm{Cov}(X_1, X_2) &= \frac{1}{5}[(-2)(-3) + (-1)(-1) + 0 \cdot 0 + 1 \cdot 1 + 2 \cdot 3] \\ &= 2.8 = \rho\sigma_1\sigma_2\end{aligned}$$
- *Standard deviation $\sigma_i = \sqrt{\mathrm{Var}(X_i)}$
- *Correlation coefficient $\rho = \frac{\mathrm{Cov}(X_1,X_2)}{\sqrt{\mathrm{Var}(X_1)}\sqrt{\mathrm{Var}(X_2)}} = \frac{\mathrm{Cov}(X_1,X_2)}{\sigma_1\sigma_2}$

Multivariate Gaussian distribution

- Univariate

$$p(x|\mu,\sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

- where σ is the standard deviation, σ^2 is the variance

- Multivariate (k-d)

$$p(\boldsymbol{x}|\boldsymbol{\mu},\boldsymbol{\Sigma}) = \frac{1}{\sqrt{(2\pi)^k|\boldsymbol{\Sigma}|}} e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})^T}$$

- where $\boldsymbol{\Sigma}$ is the covariance matrix

Bivariate ($k = 2$)

$$p(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{2\pi\sqrt{|\boldsymbol{\Sigma}|}} e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})^T}$$

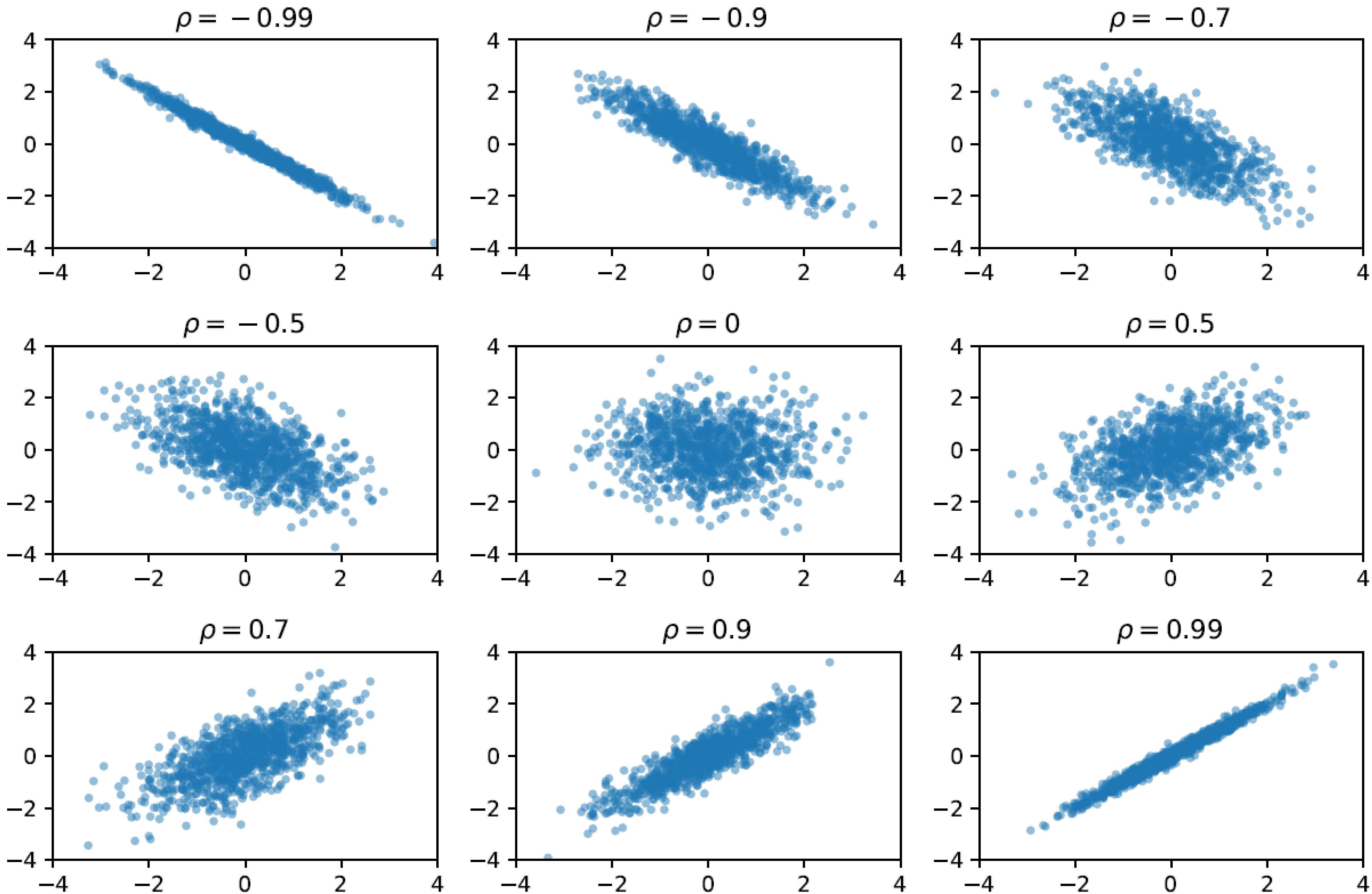
- 2-d point: $\boldsymbol{x} = [x_1 \quad x_2]$
- Parameters:
  - Mean: $\boldsymbol{\mu} = [\mu_1 \quad \mu_2]$
  - Covariance matrix: $\boldsymbol{\Sigma} = \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix} = \begin{bmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 \\ \rho\sigma_1\sigma_2 & \sigma_2^2 \end{bmatrix}$
    - σ_1 and σ_2 are the standard deviations
    - ρ is the correlation coefficient

The covariance matrix $\boldsymbol{\Sigma}$ and σ_1, σ_2, ρ

- $\Sigma = \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix} = \begin{bmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 \\ \rho\sigma_1\sigma_2 & \sigma_2^2 \end{bmatrix}$

- For example:
- $\boldsymbol{\Sigma} = \begin{bmatrix} 9 & 5 \\ 5 & 4 \end{bmatrix}$
- $\Sigma_{11} = 9, \Sigma_{12} = \Sigma_{21} = 5, \Sigma_{22} = 4$
- $\sigma_1 = \sqrt{\Sigma_{11}} = 3, \sigma_2 = \sqrt{\Sigma_{22}} = 2$
- $\Sigma_{12} = \Sigma_{21} = \rho\sigma_1\sigma_2 \rightarrow \rho = \frac{5}{3\times 2} = \frac{5}{6}$

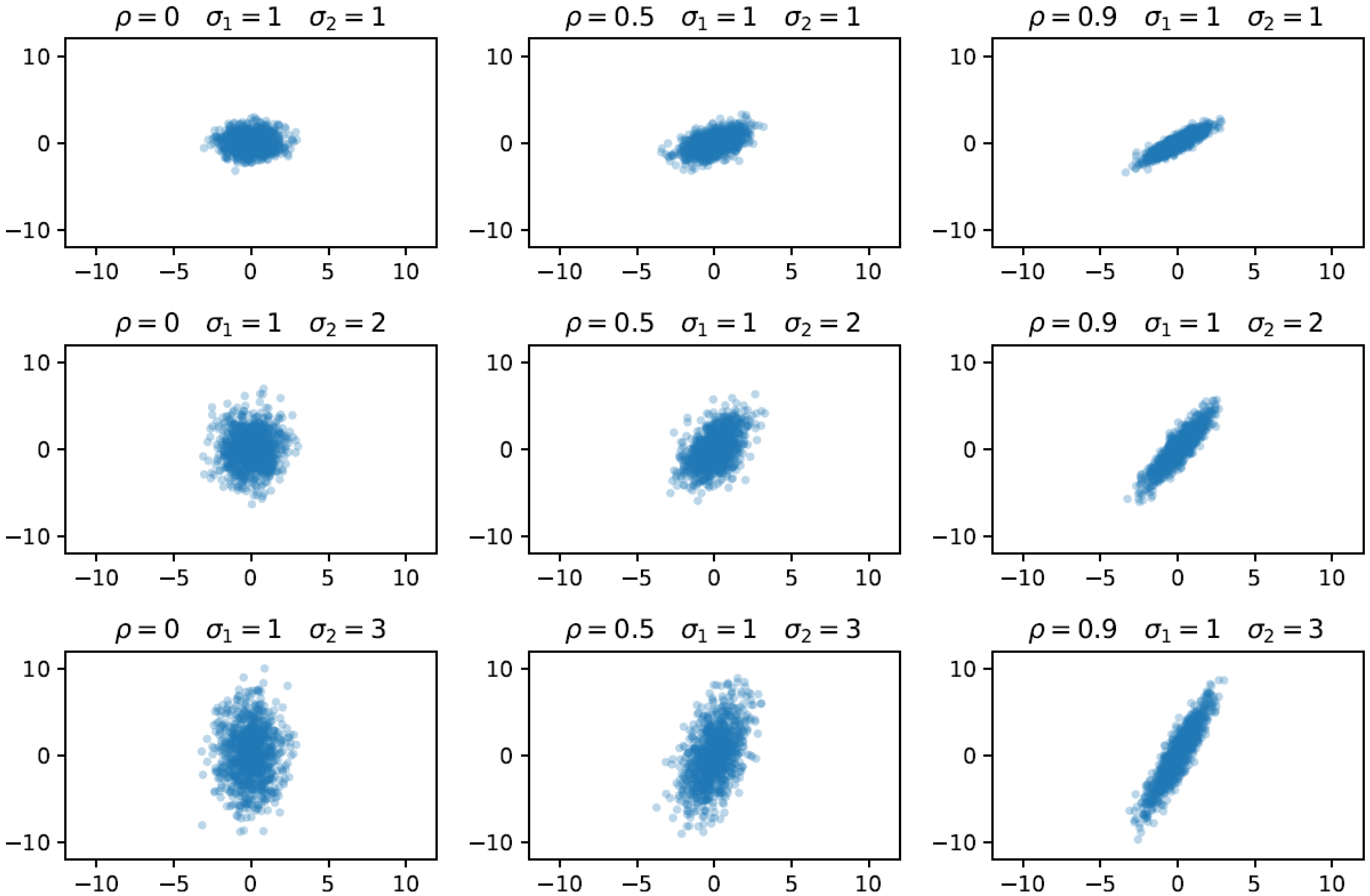
2-d Gaussian with $\sigma_1, \sigma_2 = 1$ & different ρ



Generate 2-d Gaussian points

```
fig = plt.figure(figsize=(9, 6))

rhos = [-0.99, -0.9, -0.7, -0.5, 0, 0.5, 0.7, 0.9, 0.99]
for i, rho in enumerate(rhos):
    mean = [0, 0]
    cov = [[1, rho], [rho, 1]]

    X1, X2 = np.random.multivariate_normal(mean, cov, 1000).T

    ax = fig.add_subplot(3, 3, i+1)
    ax.plot(X1, X2, '.', alpha=0.5)
    ax.set_title(r'$\rho=%g$'%rho)
    ax.set_xlim([-4, 4])
    ax.set_ylim([-4, 4])

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

2-d Gaussian with $\sigma_1 = 1$ & different ρ, σ_2



How to estimate parameters for 1-d points

```
mean, std = 3, 2
X = np.random.normal(mean, std, 20)
--------------------------------
[1.45592203  3.68408712  3.24950642  2.15079211
 5.68247789  3.44566904  6.05949948  5.15271825
 3.81870043  1.35680408  2.73188372  2.89284301
 5.74163191 -1.77632601  0.61871797  4.13751951
 2.89829252  1.90861471  0.27220436  3.10042466]
```

$$\mu = ? \quad \sigma = ?$$

Maximum likelihood estimates (1-d points)

```
mean, std = 3, 2
X = np.random.normal(mean, std, 20)
--------------------------------
[1.45592203  3.68408712  3.24950642  2.15079211
 5.68247789  3.44566904  6.05949948  5.15271825
 3.81870043  1.35680408  2.73188372  2.89284301
 5.74163191 -1.77632601  0.61871797  4.13751951
 2.89829252  1.90861471  0.27220436  3.10042466]
```

$$L = \prod_{i=1}^{20} N(x_i|\mu, \sigma) = \prod_{i=1}^{20} \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{1}{2}(x_i-\mu)^2}$$

- Find μ and σ that maximizes the likelihood L, let $\frac{\partial L}{\partial \mu} = 0$ and $\frac{\partial L}{\partial \sigma} = 0$

$$\mu = \frac{1}{20}\sum_{i=1}^{20} x_i \quad \sigma^2 = \frac{1}{20}\sum_{i=1}^{20} (x_i - \mu)^2 = \frac{1}{20}(\boldsymbol{x} - \mu)^T(\boldsymbol{x} - \mu)$$

Maximum likelihood estimates (1-d points)

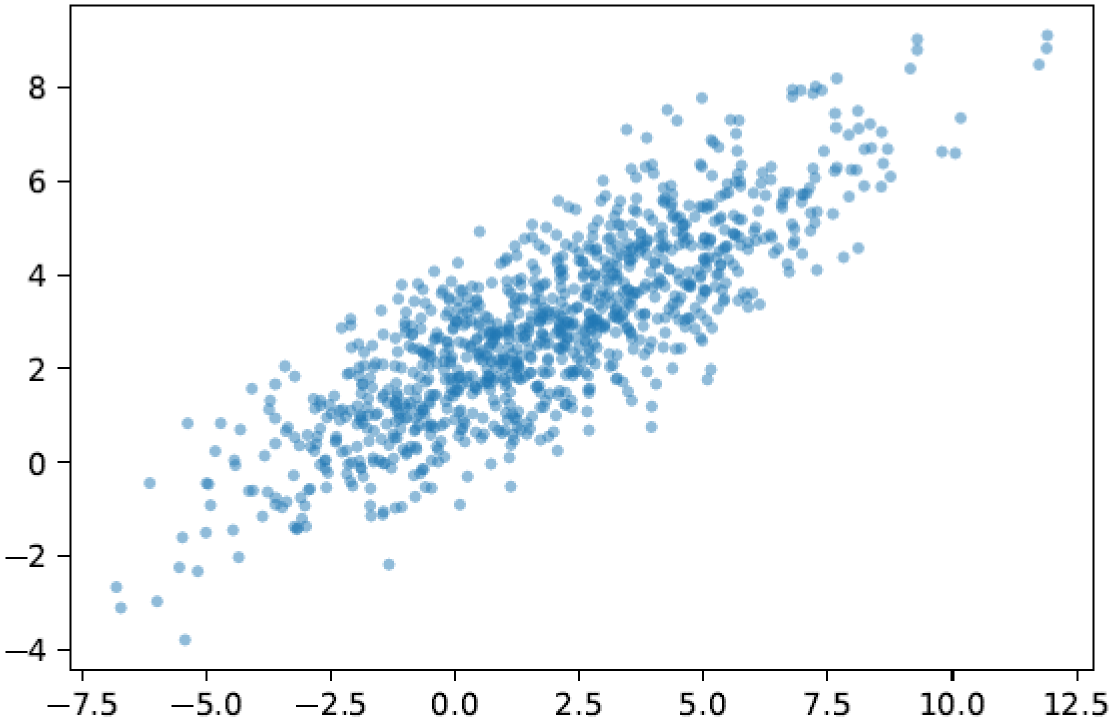
```
mean, std = 3, 2
X = np.random.normal(mean, std, 20)
--------------------------------
[1.45592203  3.68408712  3.24950642  2.15079211
 5.68247789  3.44566904  6.05949948  5.15271825
 3.81870043  1.35680408  2.73188372  2.89284301
 5.74163191 -1.77632601  0.61871797  4.13751951
 2.89829252  1.90861471  0.27220436  3.10042466]

mu    = X.sum() / 20
sigma = np.sqrt((X-mu).dot(X-mu) / 20)
print('mu    =', mu)
print('sigma =', sigma)
--------------------------------
mu    = 2.58609767501
sigma = 2.17869973073
```

$$\mu = \frac{1}{20}\sum_{i=1}^{20} x_i$$

$$\sigma^2 = \frac{1}{20}(\boldsymbol{x} - \mu)^T(\boldsymbol{x} - \mu)$$

What about MLE for 2-d points?

```
mean = [2, 3]
cov = [[9, 5], [5, 4]]

X = np.random.multivariate_normal(mean, cov, 1000)

plt.plot(X[:, 1], X[:, 2], '.', alpha=0.5)
```

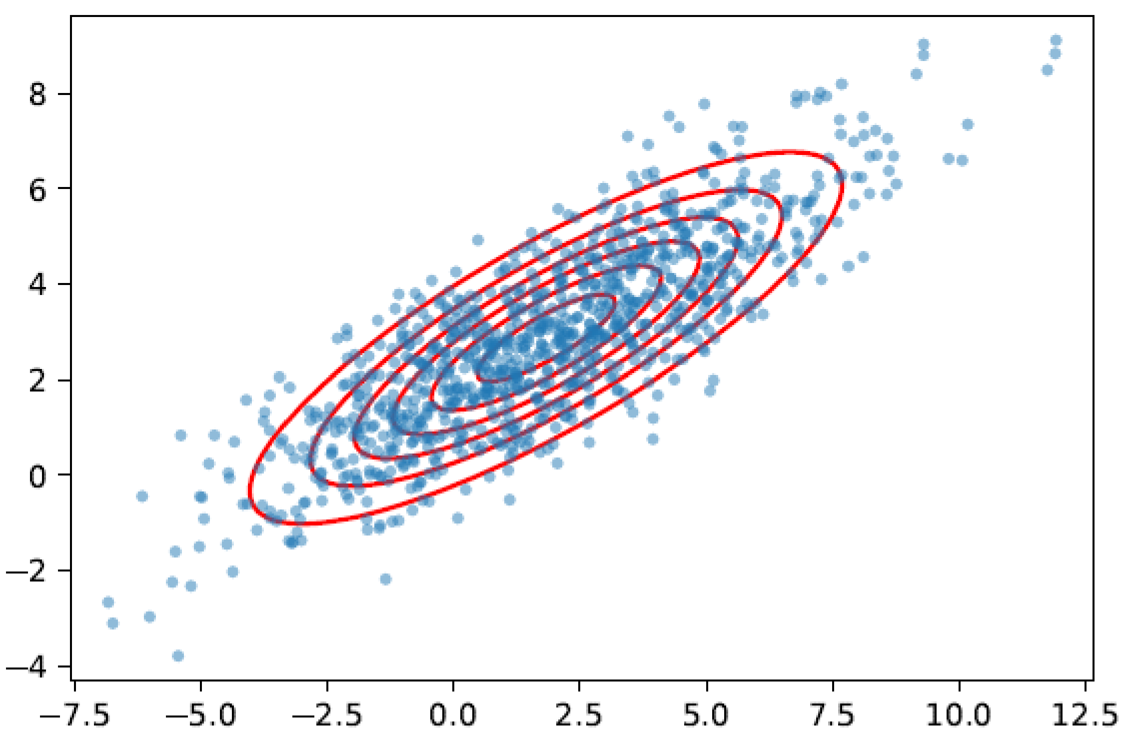


$$\boldsymbol{\mu} = \frac{1}{1000}\left[\sum_{i=1}^{1000} x_{i,1} \quad \sum_{i=1}^{1000} x_{i,2}\right]$$

$$\boldsymbol{\Sigma} = \frac{1}{1000}(\boldsymbol{X}-\boldsymbol{\mu})^T(\boldsymbol{X}-\boldsymbol{\mu})$$

What about MLE for 2-d points?

```
mean = [2, 3]
cov = [[9, 5], [5, 4]]

X = np.random.multivariate_normal(mean, cov, 1000)

plt.plot(X[:, 1], X[:, 2], '.', alpha=0.5)
```




```
Mu = X.mean(axis=0)
Sigma = (X-Mu).T.dot(X-Mu)/1000
print('Mu:', Mu)
print('Sigma:')
print(Sigma)
---------------------------------
Mu: [ 1.83023938  2.8712775 ]
Sigma:
[[ 9.04242839  4.95851301]
 [ 4.95851301  3.9911504 ]]
```

Outline

- Review the lecture, background knowledge, etc.
  - Multivariate Gaussian distribution
    - Estimate parameters (for 1-d, 2-d Gaussian)
  - Probabilistic graphical models (PGM)
    - Parameters and variables
    - An example: Gaussian mixture models (GMM)
      - Generative process, joint distribution factorization, plate notation
  - Expectation maximization (EM) algorithm for GMM

- IPython notebook task: GMM

Parameters and variables

- Note: there are no standard definitions for them, but you can understand them in the following way.
- Parameters
  - Known parameters
  - Unknown parameters
- (random) Variables
  - Observable variables
  - Latent variables

Parameters and variables

- Note: there are no standard definitions for them, but you can understand them in the following way.
- Input $\boldsymbol{x}$ and output y are usually considered as variables
- Variables are drawn from distributions
- Parameters are fixed, although could be unknown

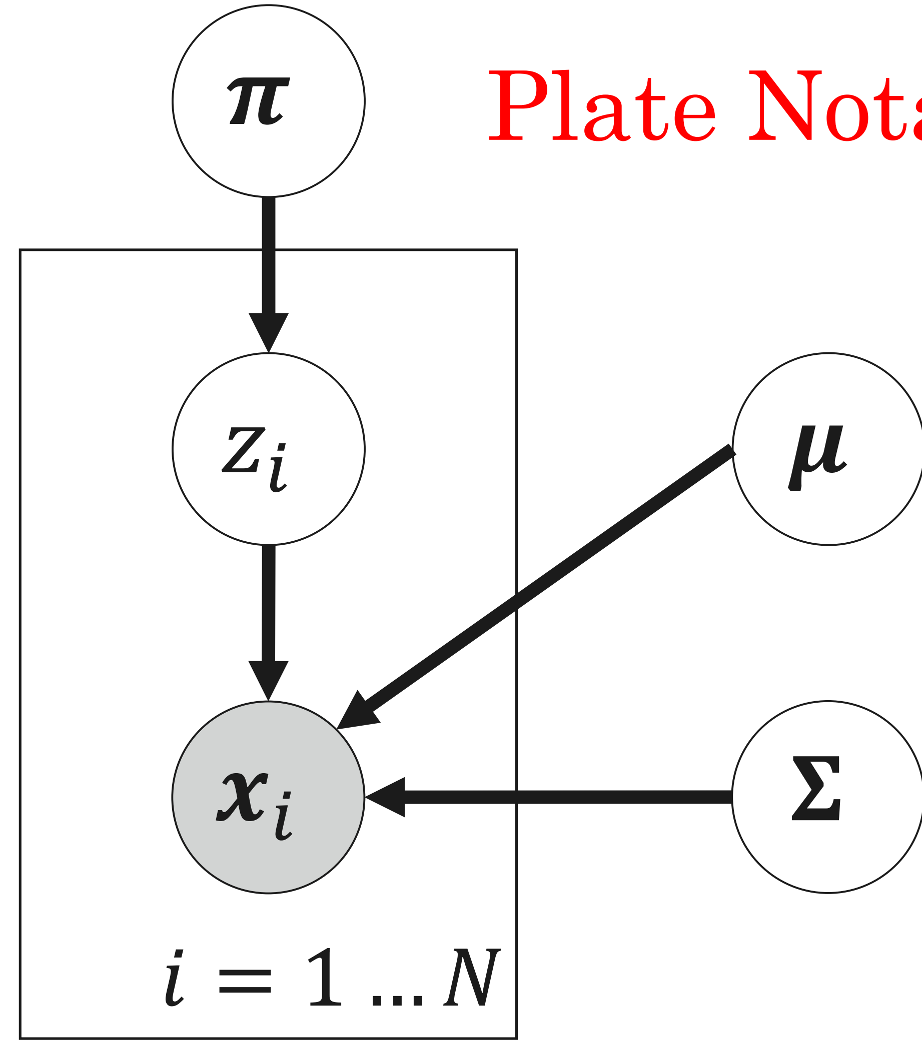
Parameters and variables

- Note: there are no standard definitions for them, but you can understand them in the following way.
- In logistic regression, $p(y = 1|\boldsymbol{x}; \boldsymbol{w}, b) = \sigma(\boldsymbol{w} \cdot \boldsymbol{x} + b)$
- $\boldsymbol{w}$, b are unknown parameters
- MLE for $\boldsymbol{w}$ and b: $\max_{\boldsymbol{w},b} p(\boldsymbol{y}|\boldsymbol{X}; \boldsymbol{w}, b)$
- But we can add a prior distribution for $\boldsymbol{w}$: $\boldsymbol{w} \sim N(0, \lambda^{-1}\boldsymbol{I})$
- $\boldsymbol{w}$ is a random variable, b is still an unknown parameter
- MAP estimate for $\boldsymbol{w}$: $\max_{\boldsymbol{w},b} p(\boldsymbol{y}|\boldsymbol{X}; \boldsymbol{w}, b)p(\boldsymbol{w})$

Three ways to define the GMM

- Generative process
- Parameters: $\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}$
- For i in $1 \ldots N$:
  - $z_i \sim \text{Categorical}(\boldsymbol{\pi})$
  - $\boldsymbol{x}_i \sim N\left(\boldsymbol{\mu}_{z_i}, \boldsymbol{\Sigma}_{z_i}\right)$

Plate Notation



- Factorization of the joint distribution
- $$p(\boldsymbol{X}, \boldsymbol{Z} | \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \prod_{i=1}^{N} p(z_i | \boldsymbol{\pi}) p(\boldsymbol{x}_i | z_i, \boldsymbol{\mu}, \boldsymbol{\Sigma})$$
  $$= \prod_{i=1}^{N} \boldsymbol{\pi}_{z_i} N\left(\boldsymbol{x}_i \middle| \boldsymbol{\mu}_{z_i}, \boldsymbol{\Sigma}_{z_i}\right)$$

To solve a GMM

- To maximize the log-likelihood

$$\max_{\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma}} \log p(\boldsymbol{X}|\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma})$$

- But how to calculate the log-likelihood?

To solve a GMM

- To maximize the log-likelihood

$$\max_{\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma}} \log p(\boldsymbol{X}|\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma})$$

- But how to calculate the log-likelihood?
  - – by marginalization

$$p(\boldsymbol{X}|\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma}) = \sum_{\boldsymbol{Z}} p(\boldsymbol{X},\boldsymbol{Z}|\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma})$$

- We know how to calculate the joint distribution

$$p(\boldsymbol{X}, \boldsymbol{Z} | \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \prod_{i=1}^{N} \boldsymbol{\pi}_{z_i} N\left(\boldsymbol{x}_i \middle| \boldsymbol{\mu}_{z_i}, \boldsymbol{\Sigma}_{z_i}\right)$$

- So the likelihood can be calculated as

$$\log p(\boldsymbol{X} | \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \log \sum_{\boldsymbol{Z}} p(\boldsymbol{X}, \boldsymbol{Z} | \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$$

$$= \log \sum_{\boldsymbol{Z}} \prod_{i=1}^{N} \boldsymbol{\pi}_{z_i} N\left(\boldsymbol{x}_i \middle| \boldsymbol{\mu}_{z_i}, \boldsymbol{\Sigma}_{z_i}\right) = \log \prod_{i=1}^{N} \sum_{z_i=1}^{C} \boldsymbol{\pi}_{z_i} N\left(\boldsymbol{x}_i \middle| \boldsymbol{\mu}_{z_i}, \boldsymbol{\Sigma}_{z_i}\right)$$

$$= \sum_{i=1}^{N} \log \sum_{z_i=1}^{C} \boldsymbol{\pi}_{z_i} N\left(\boldsymbol{x}_i \middle| \boldsymbol{\mu}_{z_i}, \boldsymbol{\Sigma}_{z_i}\right)$$

To solve a GMM

- To maximize the log-likelihood

$$\max_{\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma}} \log p(\boldsymbol{X}|\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma}) = \sum_{i=1}^{N} \log \sum_{z_i=1}^{C} \boldsymbol{\pi}_{z_i} N\left(\boldsymbol{x}_i \middle| \boldsymbol{\mu}_{z_i}, \boldsymbol{\Sigma}_{z_i}\right)$$

- Two options to solve
  - Gradient-based algorithms
  - Expectation maximization (EM) algorithm

An example: 2-D, 3 clusters, 4 points

- $\boldsymbol{\pi} = [\pi_1 \quad \pi_2 \quad \pi_3]$

- $\boldsymbol{\mu}_1 = [\mu_{1,1} \quad \mu_{1,2}]$ $\boldsymbol{\mu}_2 = [\mu_{2,1} \quad \mu_{2,2}]$ $\boldsymbol{\mu}_3 = [\mu_{3,1} \quad \mu_{3,2}]$
- $\boldsymbol{\Sigma}_1 = \begin{bmatrix} \Sigma_{1,11} & \Sigma_{1,12} \\ \Sigma_{1,21} & \Sigma_{1,22} \end{bmatrix}$ $\boldsymbol{\Sigma}_2 = \begin{bmatrix} \Sigma_{2,11} & \Sigma_{2,12} \\ \Sigma_{2,21} & \Sigma_{2,22} \end{bmatrix}$ $\boldsymbol{\Sigma}_3 = \begin{bmatrix} \Sigma_{3,11} & \Sigma_{3,12} \\ \Sigma_{3,21} & \Sigma_{3,22} \end{bmatrix}$

- $\boldsymbol{x}_1 = [x_{1,1} \quad x_{1,2}]$ $\boldsymbol{x}_2 = [x_{2,1} \quad x_{2,2}]$
- $\boldsymbol{x}_3 = [x_{3,1} \quad x_{3,2}]$ $\boldsymbol{x}_4 = [x_{4,1} \quad x_{4,2}]$

$$\log p(\boldsymbol{X}|\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma}) = \sum_{i=1}^{N} \log \sum_{z_i=1}^{C} \boldsymbol{\pi}_{z_i} N\left(\boldsymbol{x}_i \middle| \boldsymbol{\mu}_{z_i}, \boldsymbol{\Sigma}_{z_i}\right)$$

$$= \log\left(\pi_1 N(\boldsymbol{x}_1|\boldsymbol{\mu}_1,\boldsymbol{\Sigma}_1) + \pi_2 N(\boldsymbol{x}_1|\boldsymbol{\mu}_2,\boldsymbol{\Sigma}_2) + \pi_3 N(\boldsymbol{x}_1|\boldsymbol{\mu}_3,\boldsymbol{\Sigma}_3)\right)$$
$$+ \log\left(\pi_1 N(\boldsymbol{x}_2|\boldsymbol{\mu}_1,\boldsymbol{\Sigma}_1) + \pi_2 N(\boldsymbol{x}_2|\boldsymbol{\mu}_2,\boldsymbol{\Sigma}_2) + \pi_3 N(\boldsymbol{x}_2|\boldsymbol{\mu}_3,\boldsymbol{\Sigma}_3)\right)$$
$$+ \log\left(\pi_1 N(\boldsymbol{x}_3|\boldsymbol{\mu}_1,\boldsymbol{\Sigma}_1) + \pi_2 N(\boldsymbol{x}_3|\boldsymbol{\mu}_2,\boldsymbol{\Sigma}_2) + \pi_3 N(\boldsymbol{x}_3|\boldsymbol{\mu}_3,\boldsymbol{\Sigma}_3)\right)$$
$$+ \log\left(\pi_1 N(\boldsymbol{x}_4|\boldsymbol{\mu}_1,\boldsymbol{\Sigma}_1) + \pi_2 N(\boldsymbol{x}_4|\boldsymbol{\mu}_2,\boldsymbol{\Sigma}_2) + \pi_3 N(\boldsymbol{x}_4|\boldsymbol{\mu}_3,\boldsymbol{\Sigma}_3)\right)$$

- where

$$N(\boldsymbol{x}_i|\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k) = \frac{1}{2\text{pi}\sqrt{|\boldsymbol{\Sigma}_k|}} e^{-\frac{1}{2}(\boldsymbol{x}_i-\boldsymbol{\mu}_k)\boldsymbol{\Sigma}_k^{-1}(\boldsymbol{x}_i-\boldsymbol{\mu}_k)^T}$$

$$= \frac{1}{2\text{pi}\sqrt{\Sigma_{k,11}\Sigma_{k,22} - \Sigma_{k,12}\Sigma_{k,21}}} e^{-\frac{\begin{bmatrix} x_{i,1}-\mu_{k,1} & x_{i,1}-\mu_{k,2} \end{bmatrix} \begin{bmatrix} \Sigma_{k,22} & -\Sigma_{k,12} \\ -\Sigma_{k,21} & \Sigma_{k,11} \end{bmatrix} \begin{bmatrix} x_{i,1}-\mu_{k,1} \\ x_{i,1}-\mu_{k,2} \end{bmatrix}}{2\left(\Sigma_{k,11}\Sigma_{k,22}-\Sigma_{k,12}\Sigma_{k,21}\right)}}$$

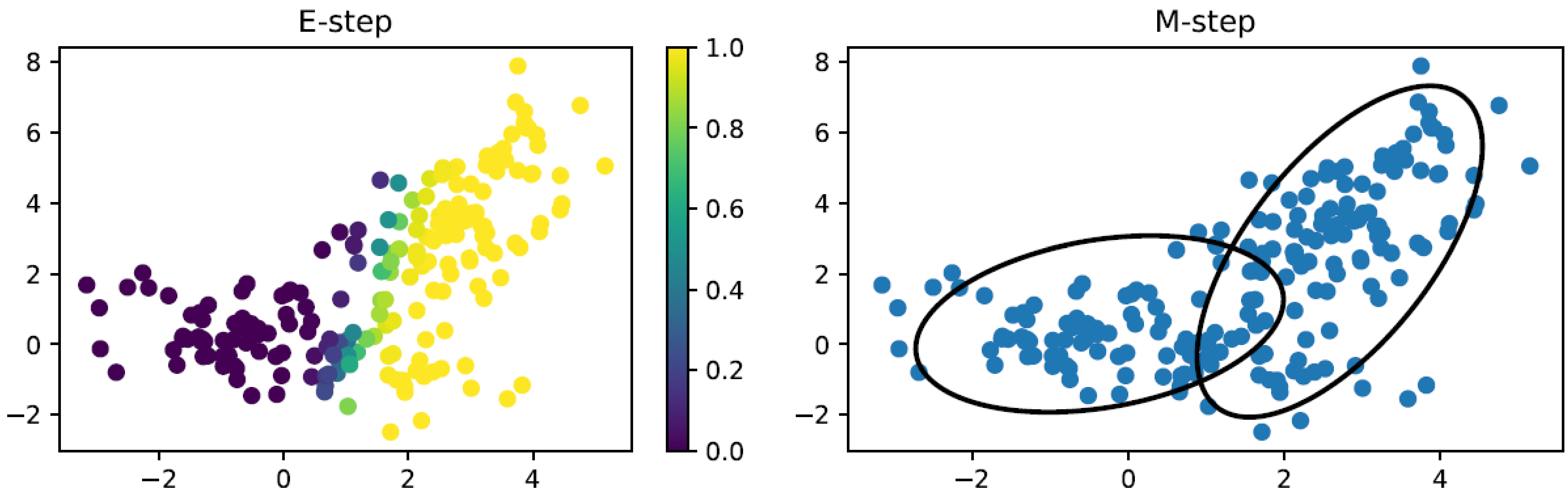
To solve a GMM

- To maximize the log-likelihood

$$\max_{\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma}} \log p(\boldsymbol{X}|\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma}) = \sum_{i=1}^{N} \log \sum_{z_i=1}^{C} \boldsymbol{\pi}_{z_i} N\left(\boldsymbol{x}_i \middle| \boldsymbol{\mu}_{z_i}, \boldsymbol{\Sigma}_{z_i}\right)$$

- Two options to solve
  - ~~Gradient-based algorithms~~
  - Expectation maximization (EM) algorithm ←

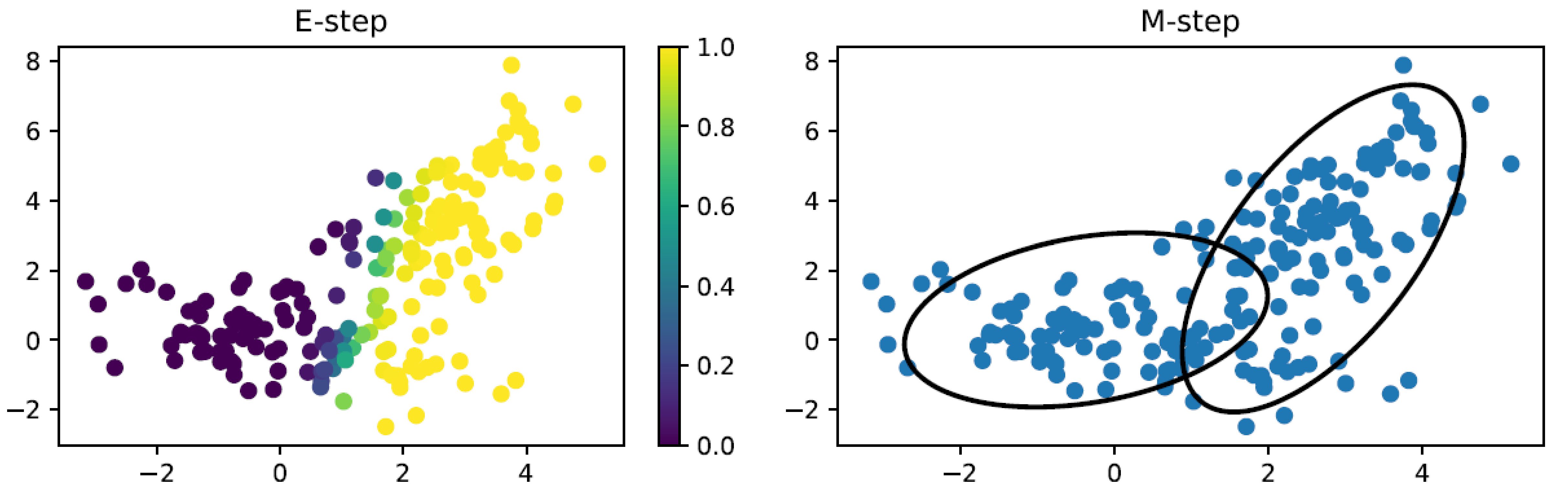
EM for GMM



EM for GMM v.s. k-means

- EM for GMM considers the probabilities of a point belonging to different clusters

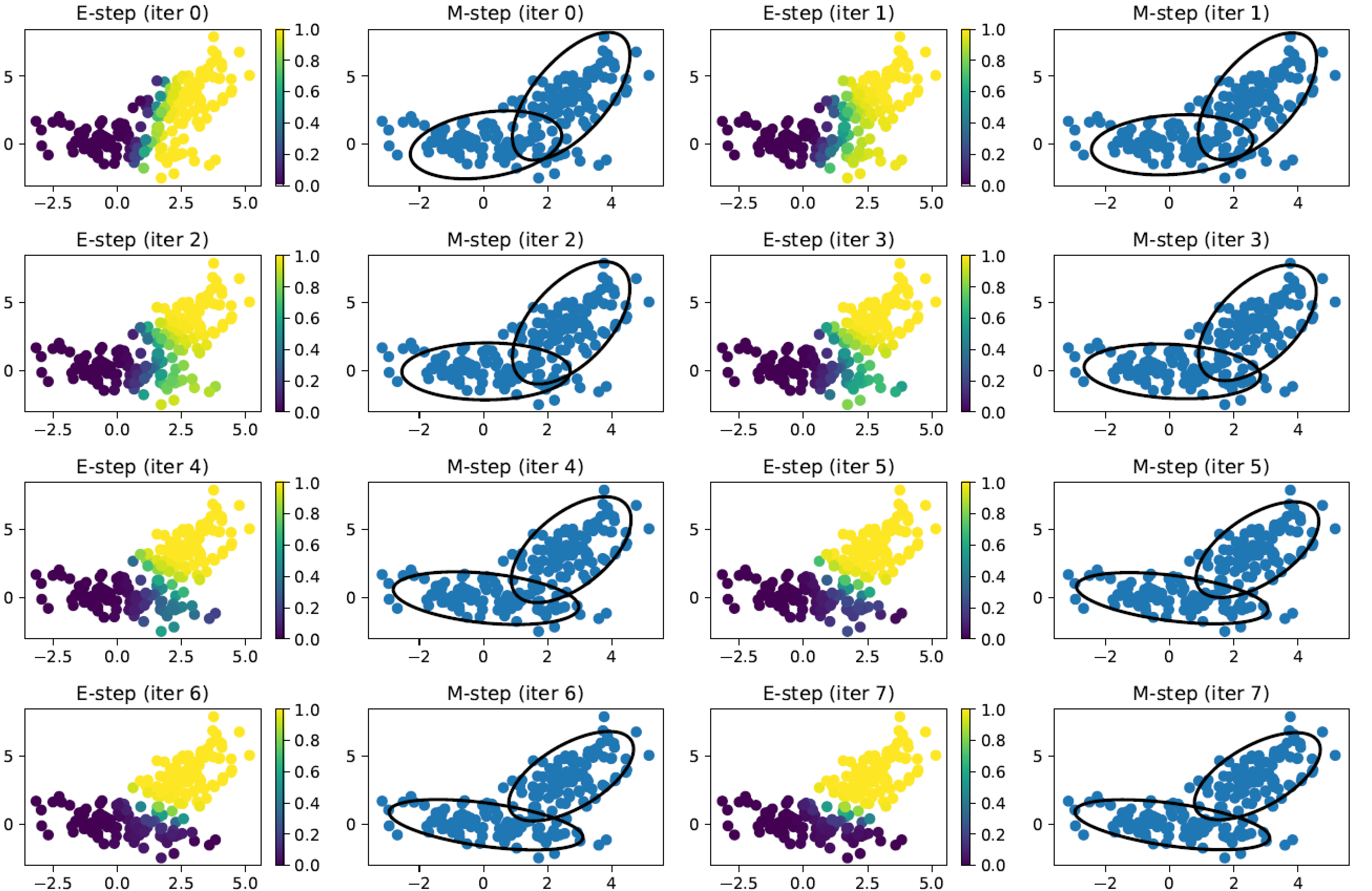
- K-means assigns every point to one cluster



- EM for GMM estimates $\boldsymbol{\mu}$, $\boldsymbol{\Sigma}$, $\boldsymbol{\pi}$

- K-means only estimates $\boldsymbol{\mu}$

EM for GMM



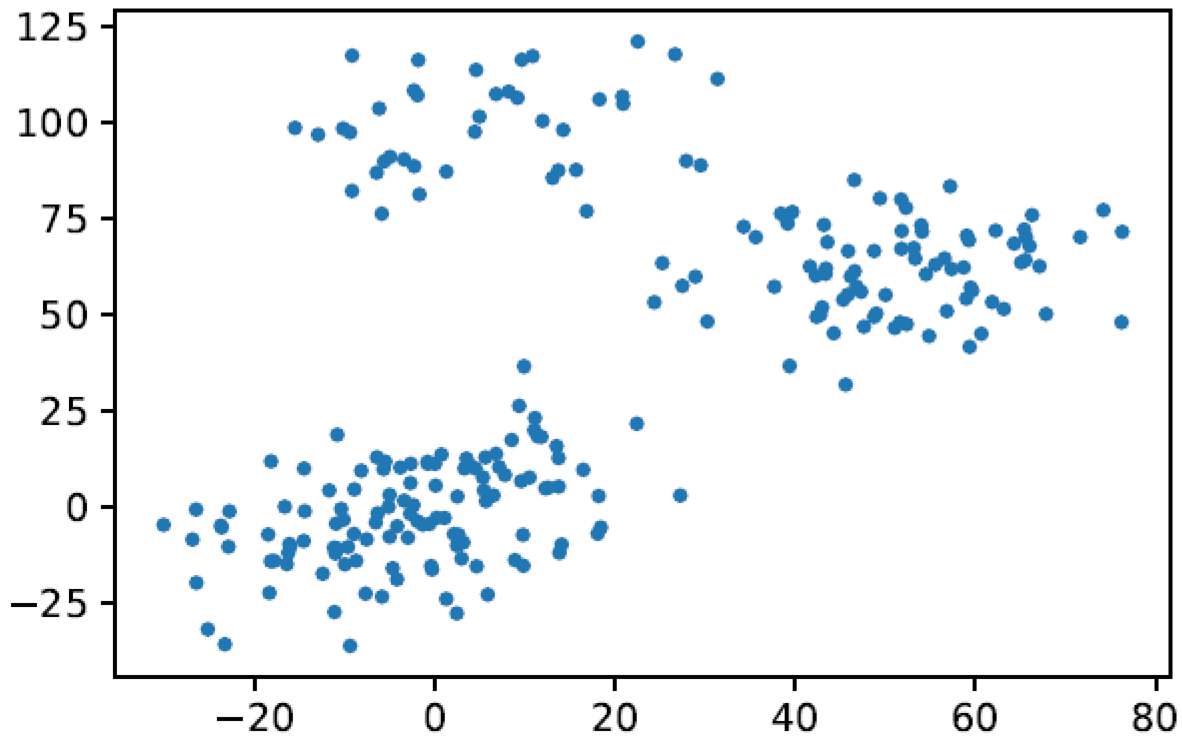
E-step (iter 0)
M-step (iter 0)
E-step (iter 1)
M-step (iter 1)
E-step (iter 2)
M-step (iter 2)
E-step (iter 3)
M-step (iter 3)
E-step (iter 4)
M-step (iter 4)
E-step (iter 5)
M-step (iter 5)
E-step (iter 6)
M-step (iter 6)
E-step (iter 7)
M-step (iter 7)

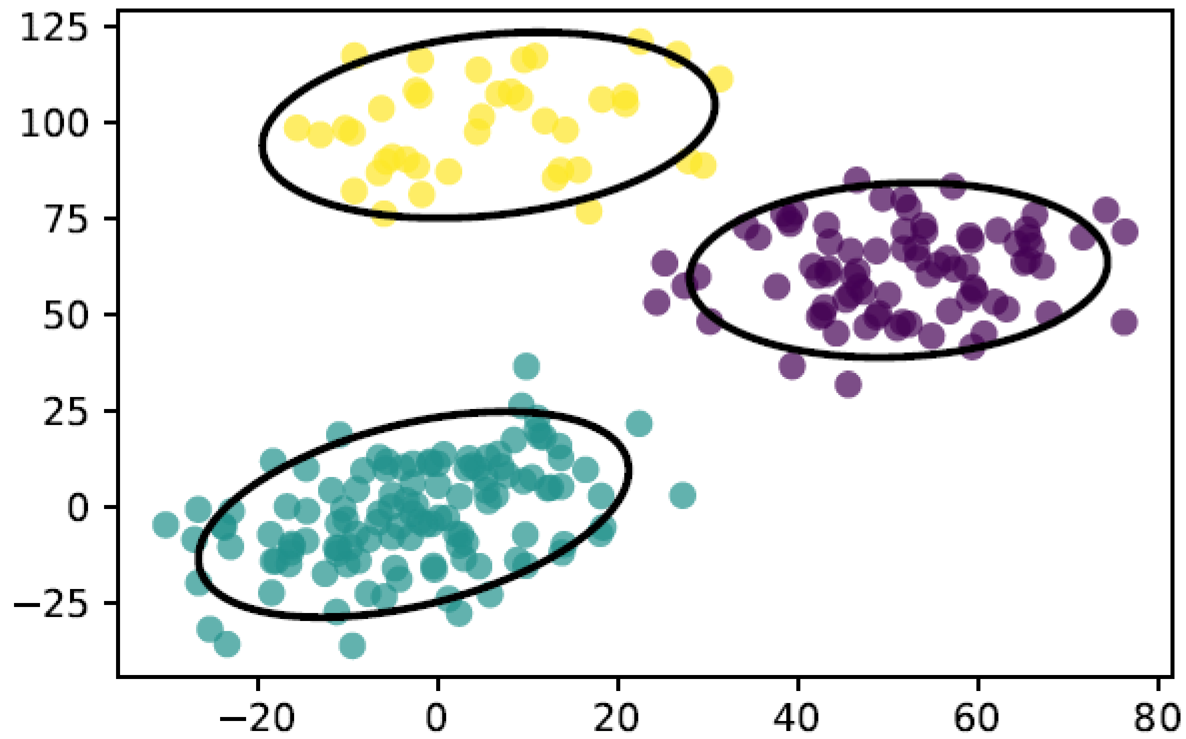
Outline

- Review the lecture, background knowledge, etc.
  - Multivariate Gaussian distribution
    - Estimate parameters (for 1-d, 2-d Gaussian)
  - Probabilistic graphical models (PGM)
    - Parameters and variables
    - An example: Gaussian mixture models (GMM)
      - Generative process, joint distribution factorization, plate notation
  - Expectation maximization (EM) algorithm for GMM

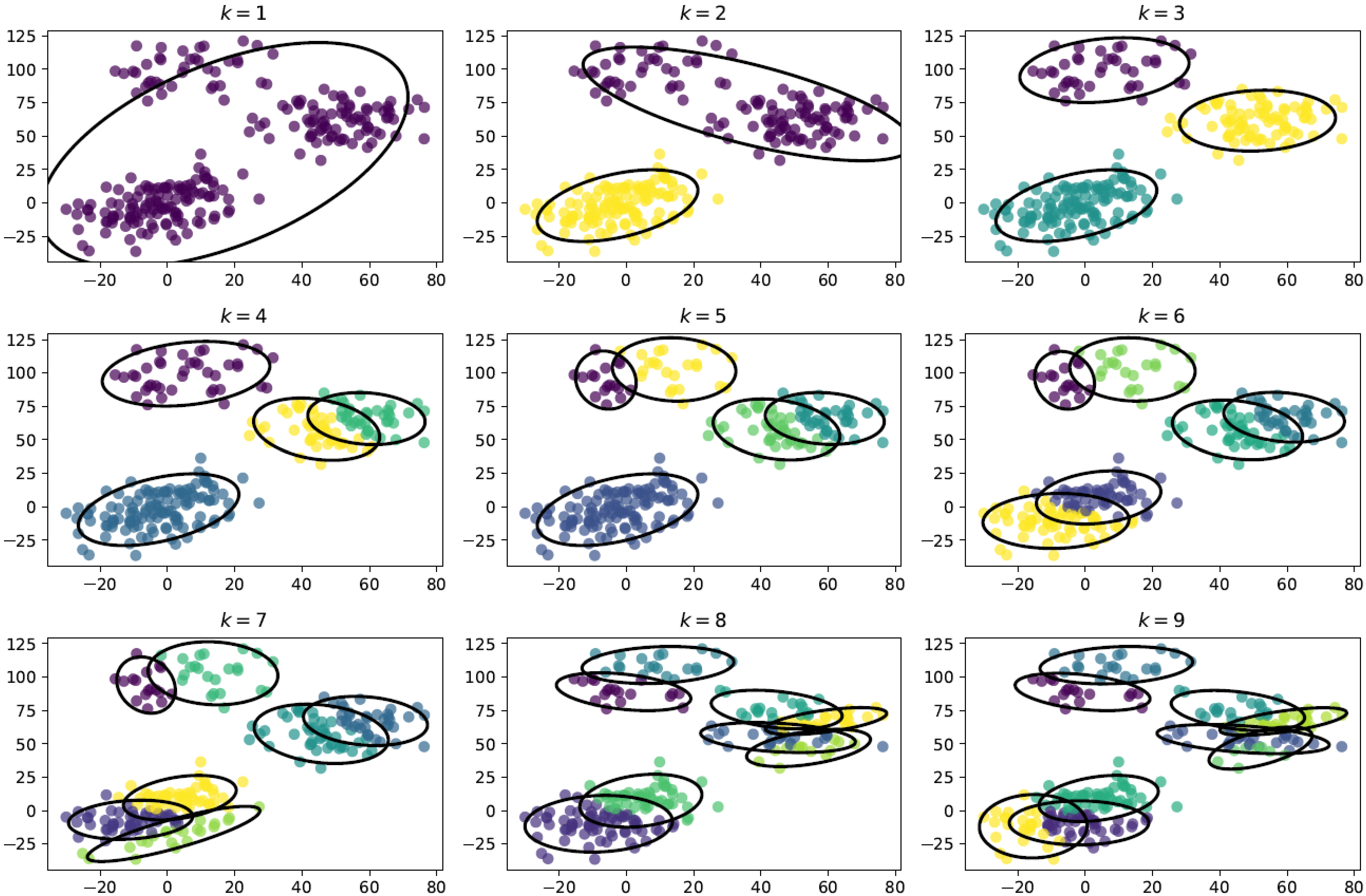
- IPython notebook task: GMM

GMM – clustering (EM, $k = 3$)

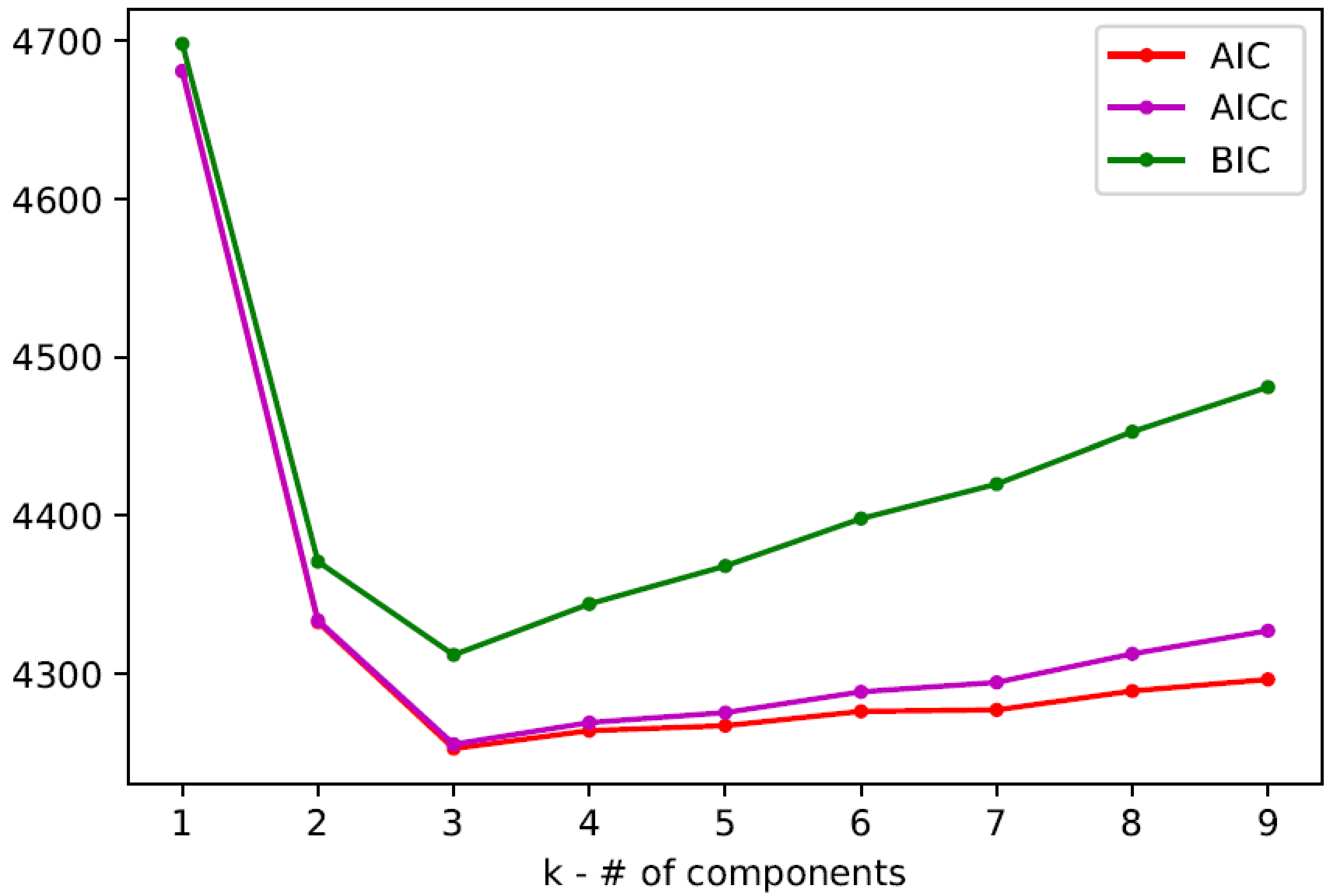




GMM – clustering with different k



GMM – choose k by AIC, AICc, BIC



GMM – EM generates different solutions