Chapter 10
Overview of Generative Models

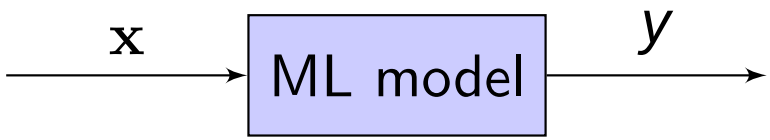
supplementary slides to
Machine Learning Fundamentals

published by Cambridge University Press

August 2020

Outline

Discriminative Models in ML: Review

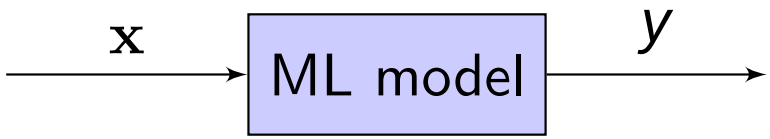


- input $\mathbf{x}$ is a random vector: $\mathbf{x} \quad p(\mathbf{x})$
- output y is generated by an unknown but *deterministic target* function $y = f(\mathbf{x})$ for each input $\mathbf{x}$
- our goal: estimate $f(\ )$ in a model space H
- use a training set: $D = f(\mathbf{x}_1; y_1); (\mathbf{x}_2; y_2); \quad ; (\mathbf{x}_N; y_N)g$, where $\mathbf{x}_i \quad p(\mathbf{x})$ and $y_i = f(\mathbf{x}_i)$
- choose a loss function $l(y; y')$, and minimize the empirical risk:

$$f^* = \arg\min_{f \in \mathrm{H}} R_{\mathrm{emp}}(f\,jD) = \arg\min_{f \in \mathrm{H}} \sum_{i=1}^{N} l(y_i; f(\mathbf{x}_i))$$

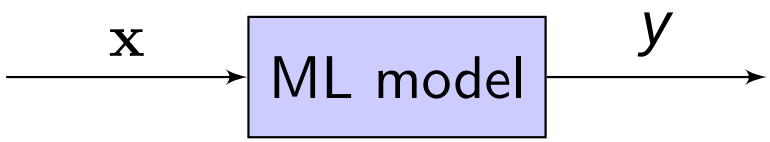
- finial performance depends on the generalization bound

Generative Models in ML



- input $\mathbf{x}$ and output y are random variables drawn from an unknown joint distribution, i.e. $(\mathbf{x}; y)$ $p(\mathbf{x}; y)$
- the relation $\mathbf{x}$! y is stochastic, solely relies on $p(y|\mathbf{x})$
- our goal: estimate $p(\mathbf{x}; y)$ using a probabilistic model $\hat{p}(\mathbf{x}; y)$
- use a training set: $D = f(\mathbf{x}_1; y_1); (\mathbf{x}_2; y_2); \quad ; (\mathbf{x}_N; y_N)g$, where $(\mathbf{x}_i; y_i)$ $p(\mathbf{x}; y)$
- the relation $\mathbf{x}$! y may be approximated by $\hat{p}(y|\mathbf{x})$
- final performance relies on the gap between $p(\mathbf{x}; y)$ and $\hat{p}(\mathbf{x}; y)$, e.g. $\mathrm{KL}\big(p(\,)\,||\,\hat{p}(\,)\big)$

Discriminative vs. Generative Models: Recap



the goal: to estimate an ML model to predict output y from input $\mathbf{x}$ based on some samples $D = f(\mathbf{x}_1; y_1); (\mathbf{x}_2; y_2); \quad ; (\mathbf{x}_N; y_N)g$

discriminative models

- data generation assumption: $\mathbf{x}_i \quad p(\mathbf{x})$ and $y_i = f(\mathbf{x}_i)$
- $\mathbf{x} \; ! \; y$ is deterministic: $y = f(\mathbf{x})$
- use D to estimate the target function: $y = f(\mathbf{x})$

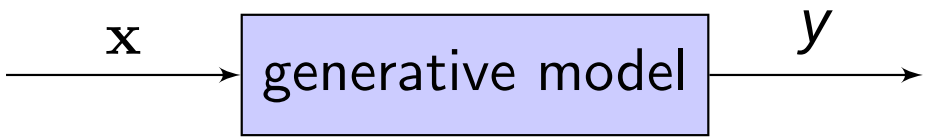
generative models

- data generation assumption: $(\mathbf{x}_i; y_i) \quad p(\mathbf{x}; y)$
- $\mathbf{x} \; ! \; y$ is stochastic: $p(y|\mathbf{x})$
- use D to estimate the joint distribution: $p(\mathbf{x}; y)$

Deterministic vs. Stochastic

- **deterministic**: given the same input $\mathbf{x}$, the output y is always the same as $y = f(\mathbf{x})$
- **stochastic**: given the same input $\mathbf{x}$, the output y is still a random variable following $p(y|\mathbf{x})$
- stochasticity may come from noises, parameter variations, etc.
- discriminative models focus on function estimation
- generative models focus on density estimation
- generative models are a more generic setting but also more challenging to learn in general

Generative Models for Classification



- input $\mathbf{x}$: feature vectors (continuous or discrete)
- output $y = f\,!_1; !_2; \quad ; !_K g$: **discrete**, called class labels
- the joint distribution $p(\mathbf{x}; y) = p(y)p(\mathbf{x}jy)$ breaks down to:

  prior probabilities: $p(y = !_k) = \Pr(!_k)$ $(8k = 1; 2; \quad ; K)$

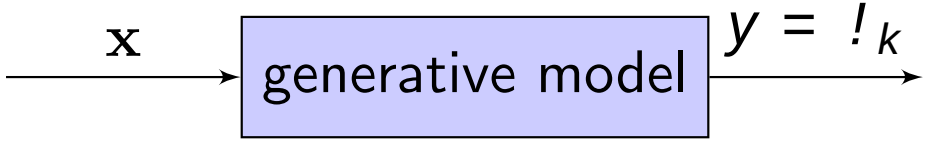
  class-conditional distributions: $p(\mathbf{x}jy = !_k) = p(\mathbf{x}j!_k)$ $(8k = 1; 2; \quad ; K)$
- probabilistic distribution constraints:

  priors satisfy $\sum_{k=1}^{K} Pr(!_k) = 1$

  if $\mathbf{x}$ is continuous, $\int_{\mathbf{x}} p(\mathbf{x}j!_k)d\mathbf{x} = 1 \quad (8k = 1; 2; \quad ; K)$

  if $\mathbf{x}$ is discrete, $\sum_{\mathbf{x}} p(\mathbf{x}j!_k) = 1 \quad (8k = 1; 2; \quad ; K)$

Bayesian Decision Theory (I): Classification



- given any $\mathbf{x}$, determine the best class in $f\,!_1;\quad !_K g$
- any decision rule: $\mathbf{x}\ 7\ !\ \ g(\mathbf{x})\ 2\ f\,!_1;\quad !_K g$
- Bayesian decision theory: the best decision is

$$g^*(\mathbf{x}) = \arg\max_k\ p(!_k|\mathbf{x}) = \arg\max_k\ \frac{\Pr(!_k)p(\mathbf{x}|!_k)}{p(\mathbf{x})}$$
$$= \arg\max_k\ \Pr(!_k)\ \ p(\mathbf{x}|!_k)$$

  a.k.a. the **maximum a posterior (MAP) rule** or Bayes decision rule.

- why is this optimal?

Optimality of the MAP rule (I)

Theorem 1

Assume $p(\mathbf{x}; !)$ is known, when $\mathbf{x}$ is used to predict $!$, the MAP rule leads to the lowest expected risk (using 0-1 loss).

- the 0-1 loss function: $I(!; !') = \begin{cases} 0 & \text{when } ! = !' \\ 1 & \text{otherwise} \end{cases}$
- the expected risk of any rule $\mathbf{x} \mapsto ! \; g(\mathbf{x}) \in \{!_1; \; !_K\}$:

$$R(g) = \mathrm{E}_{p(\mathbf{x};!)}\Big[I\big(!; g(\mathbf{x})\big)\Big] = \int_{\mathbf{x}} \sum_{k=1}^{K} I\big(!_k; g(\mathbf{x})\big) p(\mathbf{x}; !_k) d\mathbf{x}$$

$$= \int_{\mathbf{x}} \underbrace{\Big[\sum_{k=1}^{K} I\big(!_k; g(\mathbf{x})\big) p(!_k|\mathbf{x})\Big]}_{\sum_{!_k \neq g(\mathbf{x})} p(!_k|\mathbf{x})} p(\mathbf{x}) d\mathbf{x}$$

Optimality of the MAP rule (II)

- due to $\sum_{k=1}^{K} p(\omega_k|\mathbf{x}) = 1$, we have

$$\sum_{\omega_k \neq g(\mathbf{x})} p(\omega_k|\mathbf{x}) = 1 \quad p\big(g(\mathbf{x})|\mathbf{x}\big)$$

- we have

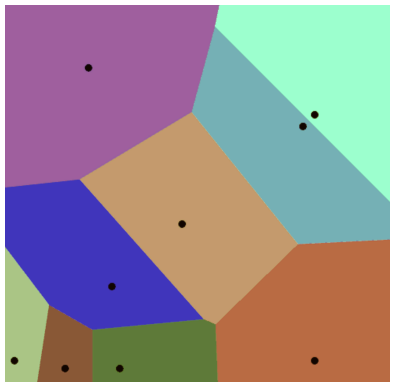
$$R(g) \# =) \quad 8\mathbf{x}; \Big[1 \quad p\big(g(\mathbf{x})|\mathbf{x}\big)\Big] \# =) \quad 8\mathbf{x}; p\big(g(\mathbf{x})|\mathbf{x}\big) \ "$$

- since $g(\mathbf{x})$ 2 $f \omega_1;$ $\omega_K g$, we choose:

$$g^*(\mathbf{x}) = \arg\max_k \ p(\omega_k|\mathbf{x})$$

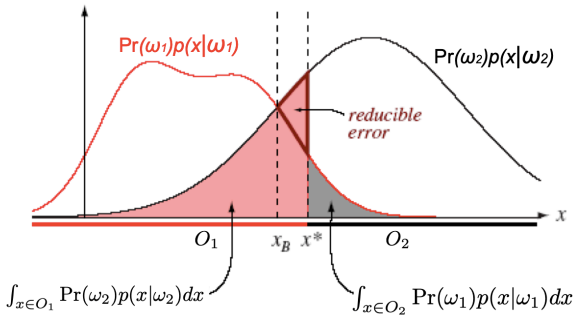
Classification Error Probability

- any rule $\mathbf{x} \nearrow g(\mathbf{x}) \in \{\omega_1, \ldots, \omega_K\}$ partitions input space into K regions, i.e. $O_1, O_2, \ldots, O_K$
  $\mathbf{x} \in O_k \Rightarrow g(\mathbf{x}) = \omega_k$



- the expected risk is the probability of classification error:

$$\begin{aligned} R(g) &= \Pr(\text{error}) = 1 - \Pr(\text{correct}) \\ &= 1 - \sum_{k=1}^{K} \Pr(\mathbf{x} \in O_k, \omega_k) \\ &= 1 - \sum_{k=1}^{K} \Pr(\omega_k) \int_{\mathbf{x} \in O_k} p(\mathbf{x}|\omega_k) d\mathbf{x} \end{aligned}$$



- the Bayes error: $R(g^*)$ of the MAP rule (the lowest possible error)

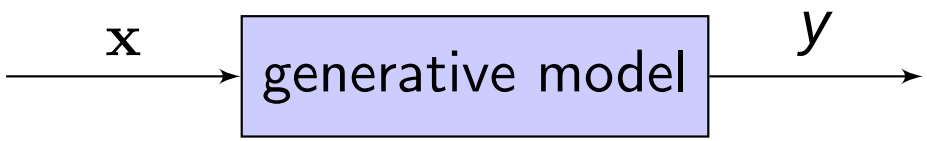
Example: the MAP rule for independent binary features

- 2-class ($!_1$ and $!_2$) classification: $\Pr(!_1)$ and $\Pr(!_2)$
- use d independent binary features $\mathbf{x} = \left[x_1; x_2; \quad ; x_d\right]^{\mathsf{T}}$, where $x_i \; 2 \; f0; 1g \quad 8i = 1; 2; \quad ; d$
- denote $\quad_i = \Pr(x_i = 1 j !_1)$ and $\quad_i = \Pr(x_i = 1 j !_2)$, we have: $p(\mathbf{x} j !_1) = \prod_{i=1}^{d} \quad_i^{x_i} (1 \quad \quad_i)^{1 \quad x_i} \quad p(\mathbf{x} j !_2) = \prod_{i=1}^{d} \quad_i^{x_i} (1 \quad \quad_i)^{1 \quad x_i}$
- the MAP rule: given $\mathbf{x}$, classify as $!_1$ if $\Pr(!_1) \quad p(\mathbf{x} j !_1)$ $\Pr(!_2) \quad p(\mathbf{x} j !_2)$, otherwise $!_2$.
- take logarithm to derive a **linear** decision boundary:

$$g(\mathbf{x}) = \sum_{i=1}^{d} \quad_i x_i + \quad_0 = \begin{cases} 0 & =) \quad !_1 \\ < 0 & =) \quad !_2 \end{cases}$$

where $\quad_i = \ln \frac{\quad_i (1 \quad \quad_i)}{\quad_i (1 \quad \quad_i)}$ and $\quad_0 = \sum_{i=1}^{d} \ln \frac{1 \quad \quad_i}{1 \quad \quad_i} + \ln \frac{\Pr(!_1)}{\Pr(!_2)}$

Generative Models for Regression



- input: n-dimensional vector $\mathbf{x} \in \mathrm{R}^n$; output: $y \in \mathrm{R}$
- the joint distribution $p(\mathbf{x}, y)$ is known
- $\mathbf{x}$ is used to predict y as $y = g(\mathbf{x})$
- what is the best decision rule $g(\mathbf{x})$ for $\mathbf{x} \nearrow y$?
- Bayesian decision theory suggests the best rule as:

$$g^*(\mathbf{x}) = \mathrm{E}(y|\mathbf{x}) = \int_y y \; p(y|\mathbf{x}) dy$$

Theorem 2

Assume $p(\mathbf{x}, y)$ is known, the conditional mean $\mathrm{E}(y|\mathbf{x})$ leads to the lowest expected risk (using mean square loss).

Optimality of Conditional Mean for Regression

Proof:

- The expected risk of any rule $\mathbf{x} \; ! \; g(\mathbf{x}) \; 2 \; \mathrm{R}$:

$$R(g) \quad = \quad \mathrm{E}_{p(\mathbf{x};y)}\Big[l\big(\,!\,;g(\mathbf{x})\big)\Big] = \int_{\mathbf{x}}\int_{y}\Big(y \quad g(\mathbf{x})\Big)^{2} p(\mathbf{x};y)d\mathbf{x}dy$$

$$= \int_{\mathbf{x}} \underbrace{\Big[\int_{y}\Big(y \quad g(\mathbf{x})\Big)^{2} p(y\mathrm{j}\mathbf{x})dy\Big]}_{Q(g|\mathbf{x})} p(\mathbf{x})d\mathbf{x}$$

- Take partial derivative w.r.t. g as:

$$\frac{@Q(g\mathrm{j}\mathbf{x})}{@g(\;)} = 0 \;\; =) \;\; \int_{y}\Big(g(\mathbf{x}) \quad y\Big)p(y\mathrm{j}\mathbf{x})dy = 0$$

$$=) \;\; g^{*}(\mathbf{x}) = \int_{y} y \;\; p(y\mathrm{j}\mathbf{x})dy = \mathrm{E}(y\mathrm{j}\mathbf{x})$$

Plug-in MAP Decision Rule for classification

- since the true distributions $\Pr(\omega_k)$ and $p(\mathbf{x}|\omega_k)$ are unknown, the optimal MAP decision rule is not feasible in practice
- given training data: $D = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_N, y_N)\}$
- choose two probabilistic models:

  $\hat{p}(\omega_k)$ to approximate $\Pr(\omega_k)$

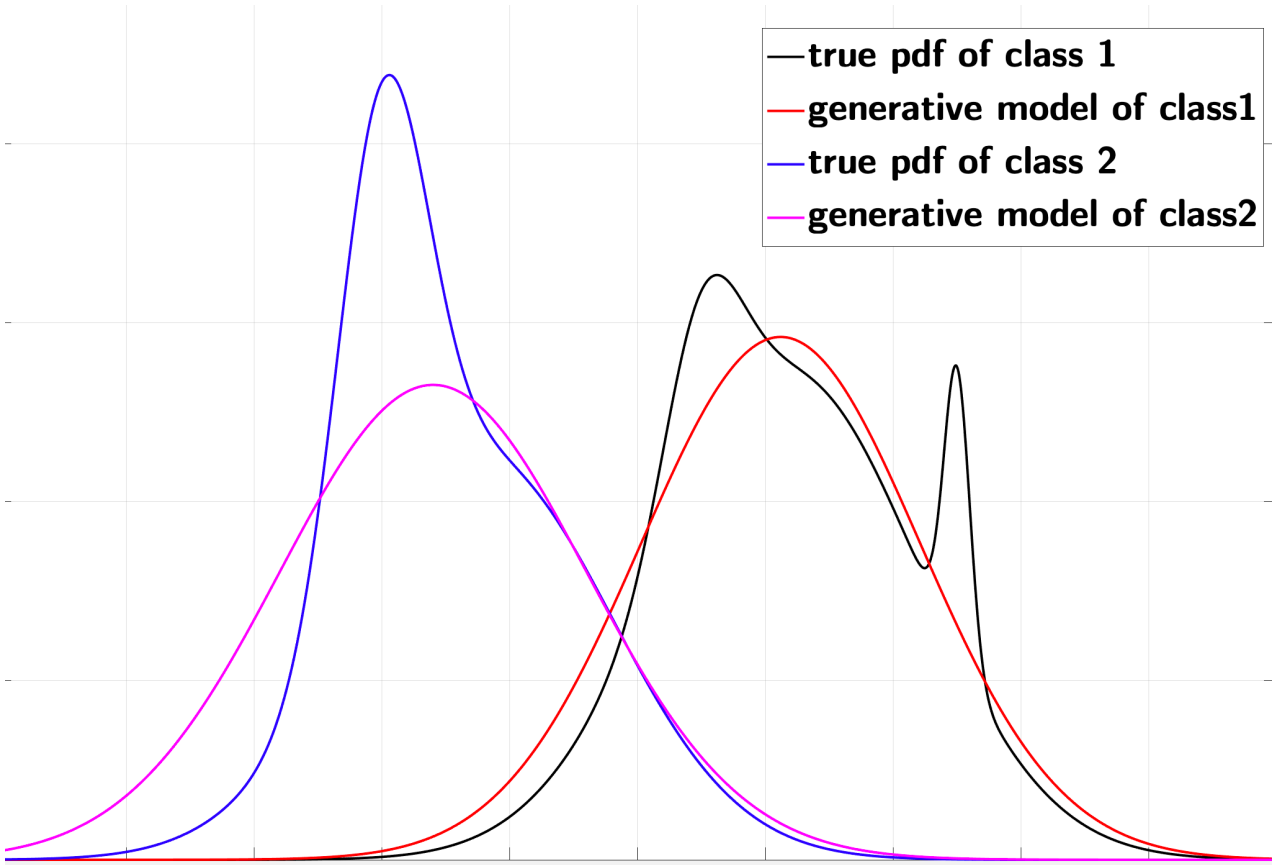
  $\hat{p}_k(\mathbf{x})$ to approximate $p(\mathbf{x}|\omega_k)$ $(\forall k = 1, 2, \ldots, K)$
- parameter estimation: estimate $\{\ldots, \theta_1, \ldots, \theta_K\}$ using D
- the optimal MAP rule in theory:

$$\omega^* = \arg\max_k \Pr(\omega_k)\ p(\mathbf{x}|\omega_k)$$

- the plug-in MAP decision rule in pratice:

$$\omega^* = \arg\max_k \hat{p}(\omega_k)\ \hat{p}_k(\mathbf{x})$$

Plug-in MAP Decision Rule



Statistical Data Modeling

Assume we have collected some training samples:

$$D = \{(\mathbf{x}_1; y_1); \quad ; (\mathbf{x}_N; y_N)\}$$

where each $(\mathbf{x}_i; y_i)$ $p(\mathbf{x}; y)$ $(8i = 1; 2; \quad ; N)$.

1. choose some probabilistic models:

$$\Pr(! _k) \quad \hat{p}_{\lambda}(! _k)$$

$$p(\mathbf{x}j ! _k) \quad \hat{p}_{\theta_k}(\mathbf{x}) \quad (8k = 1; 2; \quad ; K)$$

2. estimate the model parameters:

$$D \quad ! \quad \{\lambda; \theta_1; \quad ; \theta_K\}$$

3. apply the plug-in MAP rule:

$$\hat{g}(\mathbf{x}) = \arg\max_k \quad \hat{p}_{\lambda}(! _k) \quad \hat{p}_{\theta_k}(\mathbf{x})$$

Maximum Likelihood Estimation (I)

- generative models for classification $f\omega_1; \quad ; \omega_K g$:

  prior probabilities: $\Pr(\omega_k) \quad (k = 1; \quad ; K)$

  class-conditional distribution: $p(\mathbf{x}j\omega_k) \quad (k = 1; \quad ; K)$

- collect training data for each class: $D_k \quad p(\mathbf{x}j\omega_k)$
- **density estimation**: estimate the probability distribution from a finite number of samples
- select probabilistic models: $\hat{p}_k(\mathbf{x}) \quad p(\mathbf{x}j\omega_k)$
- **maximum likelihood estimation (MLE)**: learn $\hat{p}_k(\mathbf{x})$ to maximize the probability of observing the training data D_k

$$\overset{*}{k} = \arg\max_k \hat{p}_k(D_k) \quad (k = 1; \quad ; K)$$

- MLE: fit data best; best interpret the observed data

Maximum Likelihood Estimation (II)

- drop index k and $\hat{p}(\ ) \ ! \ p(\ )$, MLE turns to be:

$$_{\text{MLE}} = \arg\max p\ (D) = \arg\max p\ (\mathbf{x}_1; \mathbf{x}_2; \quad ; \mathbf{x}_N)$$

  where $D = f\mathbf{x}_1; \mathbf{x}_2; \quad ; \mathbf{x}_N g$

- assume all data are *i.i.d.* (independent and identically distributed), i.e., all samples are drawn independently from the same distribution:

$$p\ (\mathbf{x}_1; \mathbf{x}_2; \quad ; \mathbf{x}_N) = \prod_{i=1}^{N} p\ (\mathbf{x}_i)$$

- why called maximum *likelihood* (not probability)?
  - $p\ (\mathbf{x})$: data distribution of various $\mathbf{x}$ if is given (fixed)
  - $p\ (\mathbf{x})$: likelihood function of if $\mathbf{x}$ is given (fixed)

Maximum Likelihood Estimation (III)

- in many cases, it is more convenient to work with the logarithm of the likelihood rather than the likelihood itself
- denote the log-likelihood function $l(\ ) = \ln p\ (D)$, we have

$$_{\text{ML}} = \arg\max\ l(\ ) = \arg\max\ \sum_{i=1}^{N} \ln p\ (\mathbf{x}_i)$$

- optimization methods for ML estimation:

  differential calculus for simple models, e.g., single univariate/multivariate Gaussian, ...
  Lagrange optimization for models with constraints, e.g., multinomial, markov chain, ...
  Expectation-Maximization (EM) method for mixture models, e.g., Gaussian mixture models (GMM), hidden Markov models

MLE Example: Univariate Gaussian Models

- the training set: $D = \{x_1, x_2, \ldots, x_N\}$ $(\forall x_i \in \mathbb{R})$
- choose a univariate Gaussian approximate the unknown distribution:

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

- the log-likelihood function:

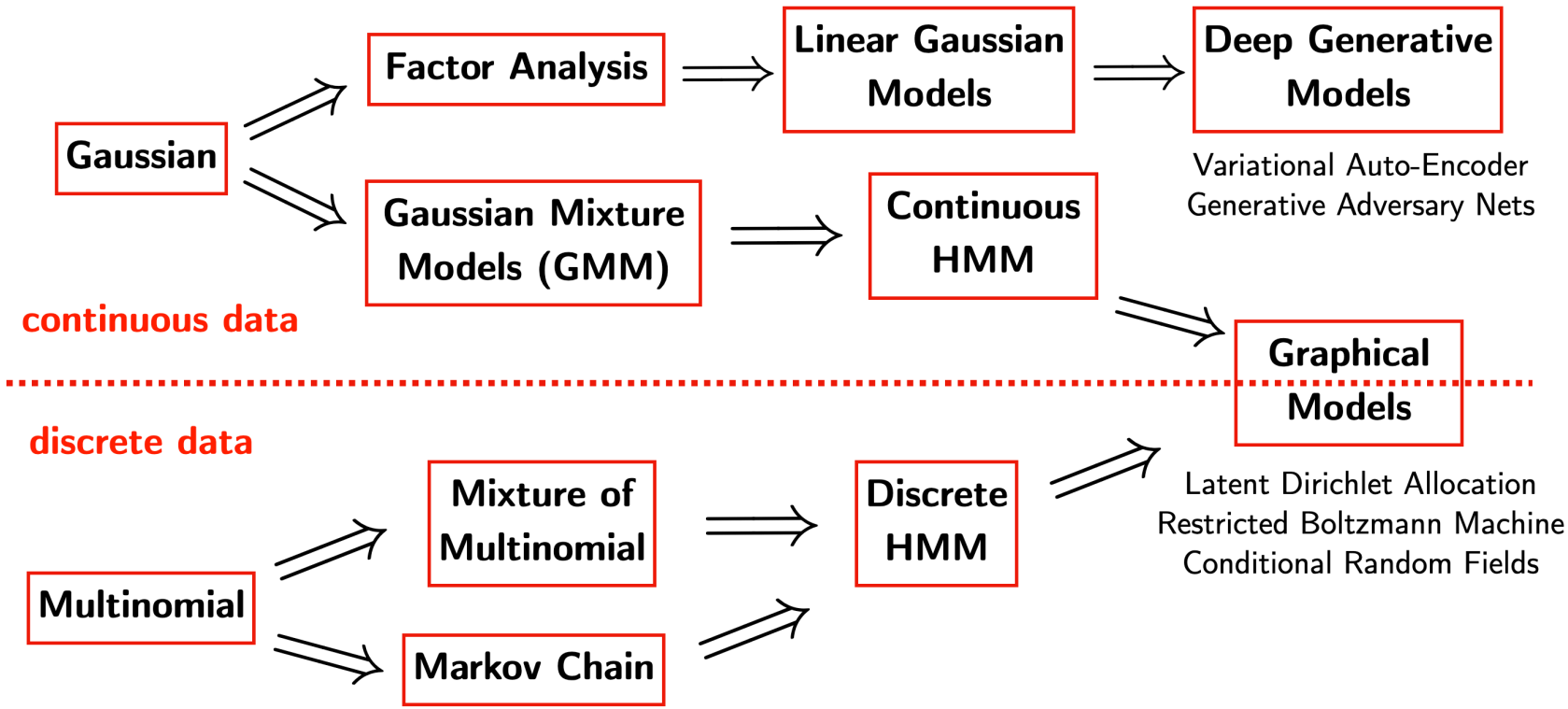
$$l(\mu, \sigma^2) = \sum_{i=1}^{N} \ln p(x_i) = \sum_{i=1}^{N} \left[-\frac{\ln(2\pi\sigma^2)}{2} - \frac{(x_i - \mu)^2}{2\sigma^2} \right]$$

- the MLE of the unknown Gaussian mean and variance:

$$\frac{\partial l(\mu, \sigma^2)}{\partial \mu} = 0 \Rightarrow \mu_{MLE} = \frac{1}{N} \sum_{i=1}^{N} x_i$$

$$\frac{\partial l(\mu, \sigma^2)}{\partial \sigma^2} = 0 \Rightarrow \sigma^2_{MLE} = \frac{1}{N} \sum_{i=1}^{N} (x_i - \mu_{MLE})^2$$

Roadmap of Generative Models



Generative Models (in a nutshell)

- Gaussian-derived generative models for continuous data
- multinomial-derived generative models for discrete data
- **unimodal models**: Gaussian, multinomial, Markov chains, generalized linear models, etc.
- **mixture models**: Gaussian mixture models, hidden Markov models, etc.
- **entangled models**: factor analysis, linear Gaussian models, deep generative models (e.g. VAE, GAN)
- **graphical models**: naive Bayes, latent Dirichlet allocation, restricted Boltzmann machine, conditional random fields, etc.
- Bayesian learning: treat model parameters as random variables