

# Introduction to Probabilistic Modeling

Volodymyr Kuleshov

Cornell Tech

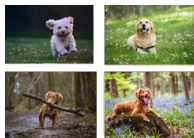
Lecture 2

# Announcements

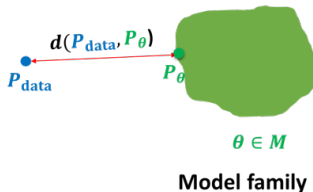
- Recorded lectures will appear in Canvas under "Zoom"
- Good luck with ICML deadline!

# Learning a generative model

- We are given a training set of examples, e.g., images of dogs



$$\mathbf{x}_i \sim P_{\text{data}} \\ i = 1, 2, \dots, n$$



- We want to learn a probability distribution  $p(x)$  over images  $x$  such that
  - **Generation:** If we sample  $x_{\text{new}} \sim p(x)$ ,  $x_{\text{new}}$  should look like a dog (*sampling*)
  - **Representation learning:** We should be able to learn what these images have in common, e.g., ears, tail, etc. (*features*)
- First step: how to represent  $p(x)$

- ① Defining Probabilistic Models of the Data
  - Examples of Probabilistic Models
  - The Curse of Dimensionality
  - Parameter-Efficient Models Through Conditional Independence
  - Bayesian Networks: An Example of Shallow Generative Models
- ② Discriminative vs. Generative Models
  - Naive Bayes vs. Logistic Regression
  - Which One to Use?
- ③ A First Glimpse of Deep Generative Models

# Probabilistic Models: Basic Discrete Distributions

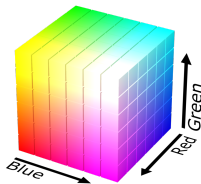
- Bernoulli distribution: (biased) coin flip
  - Domain:  $\{Heads, Tails\}$
  - Specify  $P(X = Heads) = p$ . Then  $P(X = Tails) = 1 - p$ .
  - Write:  $X \sim Ber(p)$
  - Sampling: flip a (biased) coin
- Categorical distribution: (biased)  $m$ -sided dice
  - Domain:  $\{1, \dots, m\}$
  - Specify  $P(Y = i) = p_i$ , such that  $\sum p_i = 1$
  - Write:  $Y \sim Cat(p_1, \dots, p_m)$
  - Sampling: roll a (biased) die

# Probabilistic Models: A Multi-Variate Joint Distribution

Suppose we want to define a distribution over one pixel in an image.

We use three discrete random variables:

- Red Channel  $R$ .  $\text{Val}(R) = \{0, \dots, 255\}$
- Green Channel  $G$ .  $\text{Val}(G) = \{0, \dots, 255\}$
- Blue Channel  $B$ .  $\text{Val}(B) = \{0, \dots, 255\}$

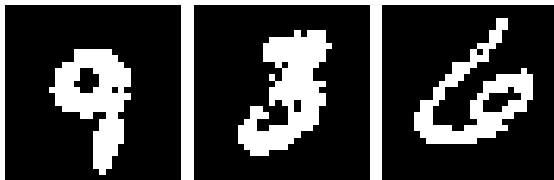


Sampling from the joint distribution  $(r, g, b) \sim p(R, G, B)$  randomly generates a color for the pixel. How many parameters do we need to specify the joint distribution  $p(R = r, G = g, B = b)$ ?

$$256 * 256 * 256 - 1$$

# The Curse of Dimensionality in Probabilistic Models

Suppose we want to model a BW image of digit with  $n = 28 \cdot 28$  pixels.



- Pixels  $X_1, \dots, X_n$  are modeled as binary (Bernoulli) random variables, i.e.,  $\text{Val}(X_i) = \{0, 1\} = \{\text{Black}, \text{White}\}$ .
- How many possible states?

$$\underbrace{2 \times 2 \times \dots \times 2}_{n \text{ times}} = 2^n$$

- Sampling from  $p(x_1, \dots, x_n)$  generates an image
- How many parameters to specify the joint distribution  $p(x_1, \dots, x_n)$  over  $n$  binary pixels?  $2^n - 1$

# Parameter-Efficient Models Through Conditional Independence

- If  $X_1, \dots, X_n$  are independent, then

$$p(x_1, \dots, x_n) = p(x_1)p(x_2) \cdots p(x_n)$$

- How many possible states?  $2^n$
- How many parameters to specify the joint distribution  $p(x_1, \dots, x_n)$ ?
  - How many to specify the marginal distribution  $p(x_1)$ ? 1
- $2^n$  **entries can be described by just  $n$  numbers** (if  $|\text{Val}(X_i)| = 2$ )!
- Independence assumption is too strong. Model not likely to be useful
  - For example, each pixel chosen independently when we sample from it.





# Key Notion: Conditional Independence

- Two events  $A, B$  are conditionally independent given event  $C$  if

$$p(A \cap B | C) = p(A | C)p(B | C)$$

- Random variables  $X, Y$  are conditionally independent given  $Z$  if for all values  $x \in \text{Val}(X)$ ,  $y \in \text{Val}(Y)$ ,  $z \in \text{Val}(Z)$

$$p(X = x \cap Y = y | Z = z) = p(X = x | Z = z)p(Y = y | Z = z)$$

- We will also write  $p(X, Y | Z) = p(X | Z)p(Y | Z)$ . Note the more compact notation.
- Equivalent definition:  $p(X | Y, Z) = p(X | Z)$ .
- We write  $X \perp Y | Z$
- Similarly for sets of random variables,  $\mathbf{X} \perp \mathbf{Y} | \mathbf{Z}$

# Two Important Rules in Probability

① **Chain rule** Let  $S_1, \dots, S_n$  be events,  $p(S_i) > 0$ .

$$p(S_1 \cap S_2 \cap \dots \cap S_n) = p(S_1)p(S_2 | S_1) \cdots p(S_n | S_1 \cap \dots \cap S_{n-1})$$

② **Bayes' rule** Let  $S_1, S_2$  be events,  $p(S_1) > 0$  and  $p(S_2) > 0$ .

$$p(S_1 | S_2) = \frac{p(S_1 \cap S_2)}{p(S_2)} = \frac{p(S_2 | S_1)p(S_1)}{p(S_2)}$$

# Structure through conditional independence

- Using Chain Rule

$$p(x_1, \dots, x_n) = p(x_1)p(x_2 \mid x_1)p(x_3 \mid x_1, x_2) \cdots p(x_n \mid x_1, \dots, x_{n-1})$$

- How many parameters?  $1 + 2 + \dots + 2^{n-1} = 2^n - 1$ 
  - $p(x_1)$  requires 1 parameter
  - $p(x_2 \mid x_1 = 0)$  requires 1 parameter,  $p(x_2 \mid x_1 = 1)$  requires 1 parameter  
Total 2 parameters.
  - ...
- $2^n - 1$  is still exponential, chain rule does not buy us anything.
- Now suppose  $X_{i+1} \perp X_1, \dots, X_{i-1} \mid X_i$ , then

$$\begin{aligned} p(x_1, \dots, x_n) &= p(x_1)p(x_2 \mid x_1)p(x_3 \mid \cancel{x_1}, x_2) \cdots p(x_n \mid \cancel{x_1, \dots, x_{i-1}}, x_{i-1}) \\ &= p(x_1)p(x_2 \mid x_1)p(x_3 \mid x_2) \cdots p(x_n \mid x_{n-1}) \end{aligned}$$

- How many parameters?  $2n - 1$ . Exponential reduction!

# Parameter-Efficient Models Through Conditional Independence

- Suppose we have 4 random variables  $X_1, \dots, X_4$
- Using Chain Rule we can **always** write

$$p(x_1, \dots, x_4) = p(x_1)p(x_2 \mid x_1)p(x_3 \mid x_1, x_2)p(x_4 \mid x_1, x_2, x_3)$$

- If  $X_4 \perp X_2 \mid \{X_1, X_3\}$ , we can simplify as

$$p(x_1, \dots, x_n) = p(x_1)p(x_2 \mid x_1)p(x_3 \mid x_1, x_2)p(x_4 \mid x_1, \cancel{x_2}, x_3)$$

- Using Chain Rule with a *different ordering* we can **always** also write

$$p(x_1, \dots, x_4) = p(x_4)p(x_3 \mid x_4)p(x_2 \mid x_3, x_4)p(x_1 \mid x_2, x_3, x_4)$$

- If  $X_1 \perp \{X_2, X_3\} \mid X_4$ , we can simplify as

$$p(x_1, \dots, x_4) = p(x_4)p(x_3 \mid x_4)p(x_2 \mid x_3, x_4)p(x_1 \mid \cancel{x_2}, \cancel{x_3}, x_4)$$

- Bayesian Networks: assume an **ordering** and a set of **conditional independencies** to get compact representation

# Bayesian Networks: General Idea

- Use conditional parameterization (instead of joint parameterization)
- For each random variable  $X_i$ , specify  $p(x_i|\mathbf{x}_{\mathbf{A}_i})$  for set  $\mathbf{X}_{\mathbf{A}_i}$  of random variables
- Then get joint parametrization as

$$p(x_1, \dots, x_n) = \prod_i p(x_i|\mathbf{x}_{\mathbf{A}_i})$$

- Need to guarantee it is a *legal* probability distribution. It has to correspond to a chain rule factorization, with factors simplified due to assumed conditional independencies

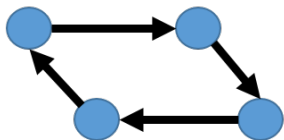
# Bayesian Networks: Formal Definition

- A **Bayesian network** is specified by a *directed acyclic* graph  $G = (V, E)$  with:
  - 1 One node  $i \in V$  for each random variable  $X_i$
  - 2 One conditional probability distribution (CPD) per node,  $p(x_i \mid \mathbf{x}_{\text{Pa}(i)})$ , specifying the variable's probability conditioned on its parents' values
- Graph  $G = (V, E)$  is called the structure of the Bayesian Network
- Defines a joint distribution:

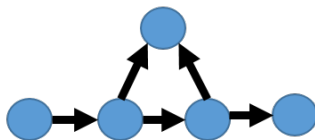
$$p(x_1, \dots, x_n) = \prod_{i \in V} p(x_i \mid \mathbf{x}_{\text{Pa}(i)})$$

- Claim:  $p(x_1, \dots, x_n)$  is a valid probability distribution
- **Economical representation:** exponential in  $|\text{Pa}(i)|$ , not  $|V|$

# What is a Directed Acyclic Graph?



Directed cycle

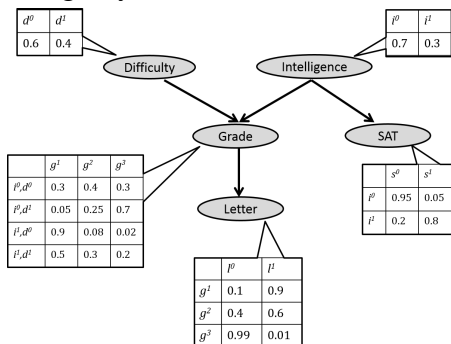


DAG

DAG stands for Directed Acyclic Graph

# Bayesian Networks: An Example

- Consider the following Bayesian network:



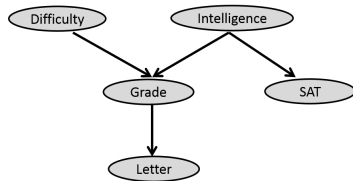
- What is its joint distribution?

$$p(x_1, \dots, x_n) = \prod_{i \in V} p(x_i \mid \mathbf{x}_{\text{Pa}(i)})$$

$$p(d, i, g, s, l) = p(d)p(i)p(g \mid i, d)p(s \mid i)p(l \mid g)$$



# Graph Structure Encodes Conditional Independencies



- The joint distribution corresponding to the above BN factors as

$$p(d, i, g, s, l) = p(d)p(i)p(g | i, d)p(s | i)p(l | g)$$

- However, by the chain rule, *any* distribution can be written as

$$p(d, i, g, s, l) = p(d)p(i | d)p(g | i, d)p(s | i, d, g)p(l | g, d, i, s)$$

- Thus, we are assuming the following extra independencies:

$$D \perp I, \quad S \perp \{D, G\} | I, \quad L \perp \{I, D, S\} | G.$$

# Summary so Far

- **Bayesian networks** given by  $(G, P)$  where  $P$  is specified as a set of local **conditional probability distributions** associated with  $G$ 's nodes
- Efficient representation using a graph-based data structure
- Computing the probability of any assignment is obtained by multiplying CPDs
- Can identify some conditional independence properties by looking at graph properties
- Next: generative vs. discriminative; functional parameterizations

## ① Defining Probabilistic Models of the Data

- Examples of Probabilistic Models
- The Curse of Dimensionality
- Parameter-Efficient Models Through Conditional Independence
- Bayesian Networks: An Example of Shallow Generative Models

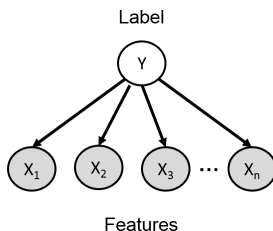
## ② Discriminative vs. Generative Models

- Naive Bayes vs. Logistic Regression
- Which One to Use?

## ③ A First Glimpse of Deep Generative Models

# Naive Bayes: A Generative Classification Algorithm

- Classify e-mails as spam ( $Y = 1$ ) or not spam ( $Y = 0$ )
  - Let  $1 : n$  index the words in our vocabulary (e.g., English)
  - $X_i = 1$  if word  $i$  appears in an e-mail, and 0 otherwise
  - E-mails are drawn according to some distribution  $p(Y, X_1, \dots, X_n)$
- Words are conditionally independent given  $Y$ :



- Then

$$p(y, x_1, \dots, x_n) = p(y) \prod_{i=1}^n p(x_i | y)$$

# Naive Bayes: A Generative Classification Algorithm

- Classify e-mails as spam ( $Y = 1$ ) or not spam ( $Y = 0$ )
  - Let  $1 : n$  index the words in our vocabulary (e.g., English)
  - $X_i = 1$  if word  $i$  appears in an e-mail, and 0 otherwise
  - E-mails are drawn according to some distribution  $p(Y, X_1, \dots, X_n)$
- Suppose that the words are conditionally independent given  $Y$ . Then,

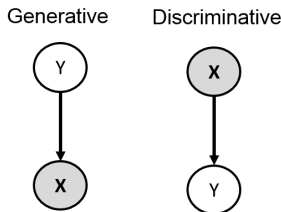
$$p(y, x_1, \dots, x_n) = p(y) \prod_{i=1}^n p(x_i | y)$$

**Estimate** parameters from data. **Predict** with Bayes rule:

$$p(Y = 1 | x_1, \dots, x_n) = \frac{p(Y = 1) \prod_{i=1}^n p(x_i | Y = 1)}{\sum_{y=\{0,1\}} p(Y = y) \prod_{i=1}^n p(x_i | Y = y)}$$

- Are the independence assumptions made here reasonable? Nearly all probabilistic models are “wrong”, but many are nonetheless useful

# Discriminative Models



- However, suppose all we need for prediction is  $p(Y | \mathbf{X})$
- In the left model, we need to specify/learn *both*  $p(Y)$  and  $p(\mathbf{X} | Y)$ , then compute  $p(Y | \mathbf{X})$  via Bayes rule
- In the right model, it suffices to estimate just the **conditional distribution**  $p(Y | \mathbf{X})$ 
  - We never need to model/learn/use  $p(\mathbf{X})$ !
  - Called a **discriminative** model because it is only useful for discriminating  $Y$ 's label when given  $\mathbf{X}$

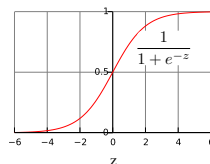
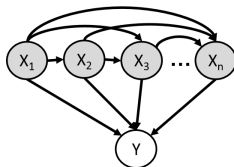
# Logistic Regression: Discriminative Classification Algorithm

- 1 For the discriminative model, assume that

$$p(Y = 1 \mid \mathbf{x}; \alpha) = f(\mathbf{x}, \alpha)$$

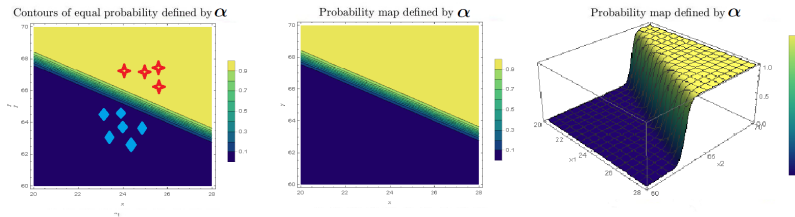
- 2 Not represented as a table anymore. It is a parameterized function of  $\mathbf{x}$  (regression)
  - Has to be between 0 and 1
  - Depend in some *simple* but reasonable way on  $x_1, \dots, x_n$
  - Completely specified by a vector  $\alpha$  of  $n + 1$  parameters (**compact representation**)

Linear dependence: let  $z(\alpha, \mathbf{x}) = \alpha_0 + \sum_{i=1}^n \alpha_i x_i$ . Then,  
 $p(Y = 1 \mid \mathbf{x}; \alpha) = \sigma(z(\alpha, \mathbf{x}))$ , where  $\sigma(z) = 1/(1 + e^{-z})$  is called the **logistic function**:



# Logistic Regression: Discriminative Classification Algorithm

Linear dependence: let  $z(\boldsymbol{\alpha}, \mathbf{x}) = \alpha_0 + \sum_{i=1}^n \alpha_i x_i$ . Then,  
 $p(Y = 1 | \mathbf{x}; \boldsymbol{\alpha}) = \sigma(z(\boldsymbol{\alpha}, \mathbf{x}))$ , where  $\sigma(z) = 1/(1 + e^{-z})$  is called the **logistic function**

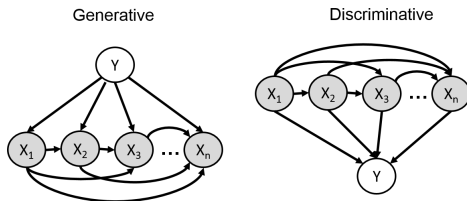


- 1 Decision boundary  $p(Y = 1 | \mathbf{x}; \boldsymbol{\alpha}) > 0.5$  is linear in  $\mathbf{x}$
- 2 Equal probability contours are straight lines
- 3 Probability rate of change has very specific form (third plot)



# Discriminative vs. Generative Models

- Since  $\mathbf{X}$  is a random vector, chain rules will give
  - $p(Y, \mathbf{X}) = p(Y)p(X_1 | Y)p(X_2 | Y, X_1) \cdots p(X_n | Y, X_1, \cdots, X_{n-1})$
  - $p(Y, \mathbf{X}) = p(X_1)p(X_2 | X_1)p(X_3 | X_1, X_2) \cdots p(Y | X_1, \cdots, X_{n-1}, X_n)$

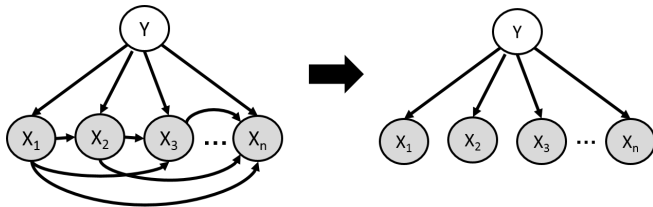


We must make the following choices:

- 1 In the generative model,  $p(Y)$  is simple, but how do we parameterize  $p(X_i | \mathbf{X}_{pa(i)}, Y)$ ?
- 2 In the discriminative model, how do we parameterize  $p(Y | \mathbf{X})$ ? Here we assume we don't care about modeling  $p(\mathbf{X})$  because  $\mathbf{X}$  is always given to us in a classification problem

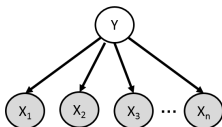
# Naive Bayes

- 1 For the generative model, assume that  $X_i \perp \mathbf{X}_{-i} \mid Y$  (**naive Bayes**)

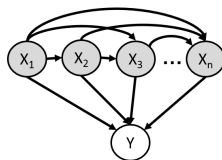


# Discriminative models are powerful

Generative (Naïve Bayes)



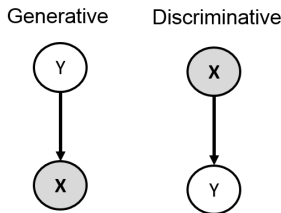
Discriminative (logistic regression)



- Logistic model does *not* assume  $X_i \perp \mathbf{X}_{-i} \mid Y$ , unlike naive Bayes
- This can make a big difference in many applications
- For example, in spam classification, let  $X_1 = 1[\text{“bank” in e-mail}]$  and  $X_2 = 1[\text{“account” in e-mail}]$
- Regardless of whether spam, these always appear together, i.e.  $X_1 = X_2$
- Learning in naive Bayes results in  $p(X_1 \mid Y) = p(X_2 \mid Y)$ . Thus, naive Bayes **double counts the evidence**
- Learning with logistic regression sets  $\alpha_1 = 0$  or  $\alpha_2 = 0$ , in effect ignoring it

# Generative models are still very useful

Using chain rule  $p(Y, \mathbf{X}) = p(\mathbf{X} | Y)p(Y) = p(Y | \mathbf{X})p(\mathbf{X})$ . Corresponding Bayesian networks:



- 1 Using a conditional model is only possible when  $\mathbf{X}$  is always observed
  - When some  $X_i$  variables are unobserved, the generative model allows us to compute  $p(Y | \mathbf{X}_{evidence})$  by marginalizing over the unseen variables

## ① Defining Probabilistic Models of the Data

- Examples of Probabilistic Models
- The Curse of Dimensionality
- Parameter-Efficient Models Through Conditional Independence
- Bayesian Networks: An Example of Shallow Generative Models

## ② Discriminative vs. Generative Models

- Naive Bayes vs. Logistic Regression
- Which One to Use?

## ③ A First Glimpse of Deep Generative Models

- ① In discriminative models, we assume that

$$p(Y = 1 \mid \mathbf{x}; \alpha) = f(\mathbf{x}, \alpha)$$

- ② **Linear** dependence:

- let  $z(\alpha, \mathbf{x}) = \alpha_0 + \sum_{i=1}^n \alpha_i x_i$ .
- $p(Y = 1 \mid \mathbf{x}; \alpha) = \sigma(z(\alpha, \mathbf{x}))$ , where  $\sigma(z) = 1/(1 + e^{-z})$  is the **logistic function**
- Dependence might be too simple

- ③ **Non-linear** dependence: let  $\mathbf{h}(A, \mathbf{b}, \mathbf{x}) = g(A\mathbf{x} + \mathbf{b})$  be a non-linear transformation of the inputs (*features*).

$$p_{\text{Neural}}(Y = 1 \mid \mathbf{x}; \alpha, A, \mathbf{b}) = \sigma(\alpha_0 + \sum_{i=1}^h \alpha_i h_i)$$

- More flexible
- More parameters:  $A, \mathbf{b}, \alpha$

# Neural Models

- ① In discriminative models, we assume that

$$p(Y = 1 \mid \mathbf{x}; \alpha) = f(\mathbf{x}, \alpha)$$

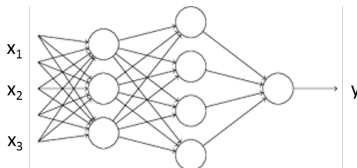
- ② **Linear** dependence: let  $z(\alpha, \mathbf{x}) = \alpha_0 + \sum_{i=1}^n \alpha_i x_i$ .  
 $p(Y = 1 \mid \mathbf{x}; \alpha) = f(z(\alpha, \mathbf{x}))$ , where  $f(z) = 1/(1 + e^{-z})$  is the **logistic function**

- Dependence might be too simple

- ③ **Non-linear** dependence: let  $\mathbf{h}(A, \mathbf{b}, \mathbf{x}) = f(A\mathbf{x} + \mathbf{b})$  be a non-linear transformation of the inputs (*features*).

$$p_{\text{Neural}}(Y = 1 \mid \mathbf{x}; \alpha, A, \mathbf{b}) = g(\alpha_0 + \sum_{i=1}^h \alpha_i h_i)$$

- More flexible
- More parameters:  $A, \mathbf{b}, \alpha$
- Can repeat multiple times to get a neural network



# Bayesian Networks vs Neural Models

- Using Chain Rule

$$p(x_1, x_2, x_3, x_4) = p(x_1)p(x_2 \mid x_1)p(x_3 \mid x_1, x_2)p(x_4 \mid x_1, x_2, x_3)$$

Fully General

- Bayes Net

$$p(x_1, x_2, x_3, x_4) \approx p(x_1)p(x_2 \mid x_1)p(x_3 \mid \cancel{x_1}, x_2)p(x_4 \mid x_1, \cancel{x_2}, \cancel{x_3})$$

Assumes conditional independencies

- Neural Models

$$p(x_1, x_2, x_3, x_4) \approx p(x_1)p(x_2 \mid x_1)p_{\text{Neural}}(x_3 \mid x_1, x_2)p_{\text{Neural}}(x_4 \mid x_1, x_2, x_3)$$

Assume specific functional form for the conditionals. A sufficiently deep neural net can approximate any function.



# Continuous Variables

- If  $X$  is a continuous random variable, we can usually represent it using its **probability density function**  $p_X : \mathbb{R} \rightarrow \mathbb{R}^+$ . However, we cannot represent this function as a table anymore. Typically consider parameterized densities:
  - Gaussian:  $X \sim \mathcal{N}(\mu, \sigma)$  if  $p_X(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/2\sigma^2}$
  - Uniform:  $X \sim \mathcal{U}(a, b)$  if  $p_X(x) = \frac{1}{b-a} 1[a \leq x \leq b]$
  - Etc.
- If  $\mathbf{X}$  is a continuous random vector, we can usually represent it using its **joint probability density function**:
  - Gaussian: if  $p_X(x) = \frac{1}{\sqrt{(2\pi)^n |\Sigma|}} \exp\left(-\frac{1}{2}(x - \mu)^T \Sigma^{-1}(x - \mu)\right)$
- Chain rule, Bayes rule, etc all still apply. For example,

$$p_{X,Y,Z}(x, y, z) = p_X(x)p_{Y|X}(y | x)p_{Z|\{X,Y\}}(z | x, y)$$

# Continuous Variables

- This means we can still use Bayesian networks with continuous (and discrete) variables. Examples:
- **Mixture of 2 Gaussians:** Network  $Z \rightarrow X$  with factorization  $p_{Z,X}(z,x) = p_Z(z)p_{X|Z}(x|z)$  and
  - $Z \sim \text{Bernoulli}(p)$
  - $X | (Z = 0) \sim \mathcal{N}(\mu_0, \sigma_0)$  ,  $X | (Z = 1) \sim \mathcal{N}(\mu_1, \sigma_1)$
  - The parameters are  $p, \mu_0, \sigma_0, \mu_1, \sigma_1$
- **Infinite Mixture of Gaussians:** Network  $Z \rightarrow X$  with factorization  $p_{Z,X}(z,x) = p_Z(z)p_{X|Z}(x|z)$ 
  - $Z \sim \mathcal{U}(a, b)$
  - $X | (Z = z) \sim \mathcal{N}(z, \sigma)$
  - The parameters are  $a, b, \sigma$
- **Neural Infinite Mixture of Gaussians:** Network  $Z \rightarrow X$  with factorization  $p_{Z,X}(z,x) = p_Z(z)p_{X|Z}(x|z)$  and
  - $Z \sim \mathcal{N}(0, 1)$
  - $X | (Z = z) \sim \mathcal{N}(\mu_\theta(z), e^{\sigma_\phi(z)})$  where  $\mu_\theta : \mathbb{R} \rightarrow \mathbb{R}$  and  $\sigma_\phi$  are neural networks with parameters (weights)  $\theta, \phi$  respectively
  - **Note:** Even if  $\mu_\theta, \sigma_\phi$  are deep nets, functional form is Gaussian