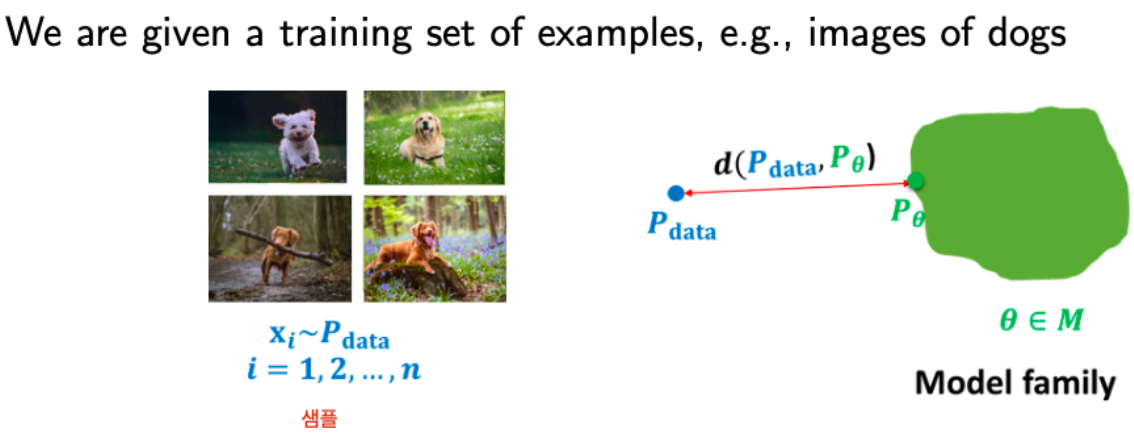
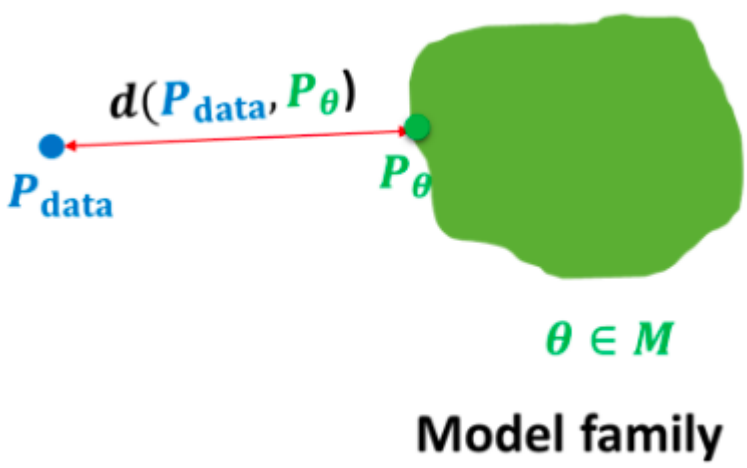
CS236n : Lecture 2

1. Learning Generative models



우리는 주어진 샘플로부터 P(x)를 모델링하게 되는데,
이렇게 잘 모델링된 P(x)는 다음 3가지에 이용이 될 수 있다.

- 생성
- 밀도 추정 (밀도 추정을 통해 P(x)값이 낮다면 해당 분포에는 잘 맞지 않는 샘플 → Anomaly Detection)
- 비지도 기반 표현 학습 (Unsupervised 기반 환경에서 이미지들간의 공통된 시각적 특징을 포착하게 해줌)



파라미터로 조정가능한 확률분포 P(x)의 파라미터에 따른 모든
가능한 경우의 집합 중(Model Family), 가장 샘플 데이터와의 간극이 적은 (교안에서는 이 간극을 d(pdata, ptheta)로 표현. 이 값이 가장 작은) 확률 분포 P(x)의 파라미터를 찾아야함

2. Example of Joint Distribution

Discrete한 확률 변수들이 구성되는 공통 확률분포에서의 파라미터들은 각 사건(state)에 대한 확률인데 MNIST IMAGE인 간단한 시나리오만 하더라도 공통확률 분포를 모델링 하기 위해 엄청난 파라미터 수를 필요로함.

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

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

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

$$2^n - 1$$

각 픽셀에 대해 가지는 값을 Xi라는 확률 변수로 두면 Val(Xi) = {0,1} = {흰, 검}

2^n-1개

(2^n개의 state가 발생할 수 있고, 각 state가 발생할 확률의 총합은 1이라는 것을 고려하면 2^n-1개의 확률 값이 파라미터임)

공통 확률분포를 정의하기 위한 Parameter가.... 너무 많아 ㅠㅠ

줄일 수 있는 방법이 없을까?

→ 확률변수간의 관계에 대한 가정을 하자!!!

**조건부 분포와 공통확률분포, 주변 확률 개념 : https://excelsior-cjh.tistory.com/193

3. Structure Through Independence

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

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

- 확률 변수들이 서로 독립이다 -> 2^n-1에서 n개로 줄일 수 있음(여전히 각 확률변수가 0 또는 1의 값만 가질 수 있다고 가정했을 때) 그러나 이 가정은 말이 안됨. 이미지의 각 픽셀들이 서로 연관성이 없다는 것을 가정하게 되므로...

두가지 Rule을 집고 넘어가자!

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

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

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

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

Chain_Rule : https://chatgpt.com/share/68888482-0874-8008-a0d2-972c0127f130

- Using Chain Rule Autoregressive Model에 사용되는 방식

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

각각의 확률 변수가 가질 수 있는 값이 {0,1} 밖에 없을 때의 상황임

- How many parameters? $1 + 2 + \cdots + 2^{n-1} = 2^n - 1$
  - $p(x_1)$ 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.

- Chain rule에서는 확률변수 간의 어떤 가정을 내린 것은 아님 (그냥 joint distribution을 conditional distribution들의 곱으로 나타낸 것임). 원래 joint parameter에서 가지는 총 state의 수로 파라미터 개수를 가져감.
- 다만 특별한 가정을 추가하여 더 단순화 할 수 있음 → 파라미터 수 감소

4. Conditional Independence를 가정(특별한 가정)해서 파라미터 수를 줄이자!

- $2^n - 1$ is still exponential, chain rule does not buy us anything.
- Now suppose $X_{i+1} \perp X_1, \ldots, X_{i-1} \mid X_i$, then

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

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

→ 조건부 독립 : 어떤 확률 변수가 주어졌을 때의 독립성을 표현. 위에서 Xi가 주어졌을 때, Xi+1은 X1~Xi-1까지의 확률 변수와 독립이다.

5. Bayesian network

공통 확률분포를 표현하려는 목표를 가지는 네트워크. 그러나 Bayesian Network는 사전 지식을 반영하여 확률변수 간의 조건부 독립성(어떤 변수 X_i는 자신의 **부모 노드(Parents)**에만 조건부로 의존하고, **부모가 주어지면 나머지 변수들과는 조건부 독립**이라고 가정합니다.)을 도입함으로써 이 공통 확률을 구성하는 파라미터의 수를 줄인다.

- **Bayesian networks** given by (G, P) where P is specified as a set of local **conditional probability distributions** associated with G's nodes

- 노드 : 확률 변수
- 노드 값 : 확률 변수에 대한 conditional distribution (given parents)
- 링크 : Direct influence from the parent to the child
  Indirect influence도 존재!

*DAG
• Directed (방향성 있음): 간선(edge)에 방향이 있어서, 노드 A에서 B로 가는 건 가능하지만 B에서 A로는 (동일 간선 기준) 갈 수 없음. →

조건부 의존성을 반영 (A → B이면 A가 B에 영향을 주는 것임)
• Acyclic (순환 없음): 어떤 노드에서 출발해서 간선을 따라가다 보면 다시 자기 자신으로 돌아오는 경로가 없음. 즉, 사이클이 없음.

베이지안 네트워크에서의 Joint Probability Factorization

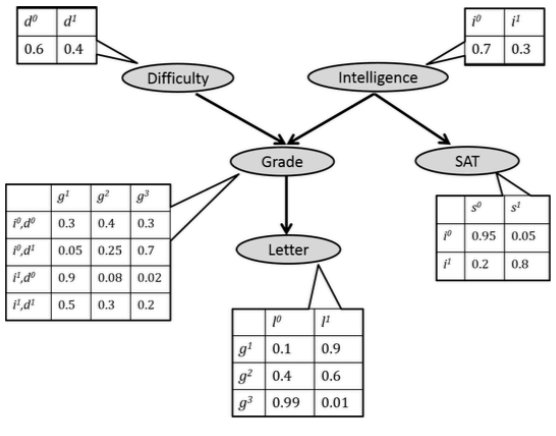
Defines a joint distribution:

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

파라미터 수 Decreasing?

- 아무 의존 관계 없이 joint를 정의하면:
  - 이진 변수 8개 → 28=2562^8 = 25628=256 조합 → 255개의 파라미터 필요
- 베이지안 네트워크에서는:
  - **부모 given 조건부 확률**만 정의하면 됨
  - 각 확률변수에 대해 부모 노드 확률 변수 이외의 확률 변수는 조건으로 반영하지 않음(조건부 독립, 부모가 given이면 다른 Variable과는 조건부 독립) → 파라미터수가 줄어듬
  - 예: `P(X6 | X3, X4)` → 4개의 값만 있으면 됨
  - ex)

- Consider the following Bayesian network:



- What is its joint distribution?

$$p(x_1, \ldots x_n) = \prod_{i \in V} p(x_i \mid \mathbf{x}_{\mathrm{Pa}(i)})$$
$$p(d, i, g, s, l) = p(d)p(i)p(g \mid i, d)p(s \mid i)p(l \mid g)$$

4. Bayesian Network에서의 샘플링 순서 (Topological Ordering)

Can sample from the joint by sampling from the CPDs according to the DAG ordering

Can identify some conditional independence properties by looking at graph properties

→ Bayesian Network로 Joint Probabiliy를 정의했을 때, Joint Probability에 대한 Sampling은 DAG 구조 ordering에 따라 sampling 하면됨.

→ Bayesian Network를 통해 Conditional Independence를 확인할 수 있음(Bayesian Ball Theory)

<Summary>

→ 베이지안 네트워크도 역시 공통 확률분포를 표현하려는 목표를 가진다. 그러나 Bayesian Network는 사전 지식을 반영하여 확률변수 간의 조건부 독립성(어떤 변수 X_i는 자신의 **부모 노드(Parents)**에만 조건부로 의존하고, **부모가 주어지면 나머지 변수들과는 조건부 독립**이라고 가정합니다.)을 도입함으로써 이 공통확률을 구성하는 파라미터의 수를 줄인다.

→ 배이지안 네트워크는 이 확률 변수들 간의 조건부 의존 관계가 DAG로 표현되어 있습니다. 이때 링크는 조건부 의존 관계 A→B로의 영향을 나타냄.(=A가 B에 영향을 준다)

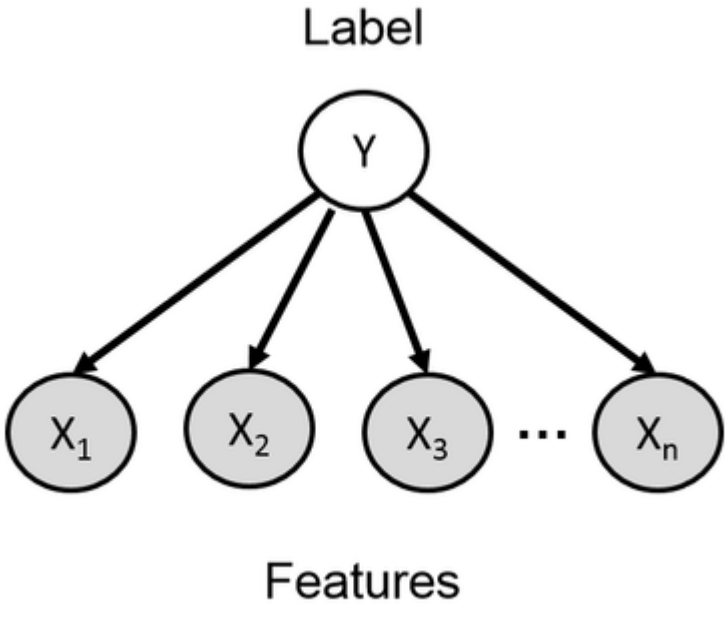
→ 그 DAG의 parent→child 순서를 따라 차례대로 확률변수를 생성하는 생성 모델이라고도 볼 수 있습니다.

→ 또한 이런 Bayesian Network의 DAG구조를 통해 우리는 확률 변수간의 조건부 독립성 또한 판단할 수 있습니다. (Bayesian Ball Theorem)

KAIST Supplementary Materials

- In the left model, we need to specify/learn *both* $p(Y)$ and $p(\mathbf{X} \mid Y)$, then compute $p(Y \mid \mathbf{X})$ via Bayes rule
- In the right model, it suffices to estimate just the **conditional distribution** $p(Y \mid \mathbf{X})$

Naive Bayes



→ Naive Bayes도 역시 공통확률분포 P(features, Label)을 모델링.

→ Naive Bayes에서 가정하는 확률변수간의 관계를 Bayesian Network로 표현하면 위와 같다.

두가지 특징

- Label이 각각의 Feature에 영향을 주는 구조
- 가정하는 것 : 조건부 독립성, Label이 주어졌을 때, 각각의 Feature들은 서로 독립임.

그럼 Joint Probability는 다음과 같음

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

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

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

뒤에 설명하겠지만 Naive Bayes 방식은 Feature와 Label의 Joint Probability를 모델링하기에 학습데이터로부터 P(Y)와 P(X|Y)의 파라미터를 학습해야한다.

그후에 Bayes Rule을 통해 P(Y|X)를 예측한다 → 입력이 X인데 Label 예측이 어떻게 돼?를 P(x,y)를 모델링하는 Generative Model에서도 수행가능

Discriminative versus generative models

우선 내가 정리한 바는 아래와 같다.

✅ **Generative Model (생성 모델)**

- **목표:** 데이터의 **joint probability** $P(X, Y)$ 를 모델링
- **방법:** $P(Y|X)$ 를 구하려면 베이즈 정리를 통해 $P(X|Y)P(Y)$ 로 계산 가능

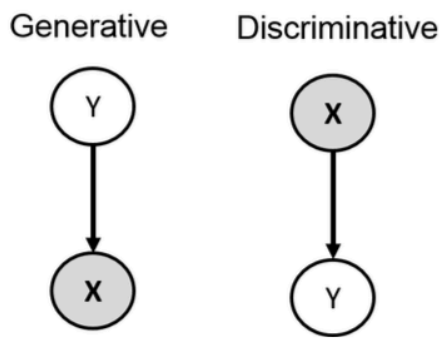
$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)}$$

- **예시:** Naive Bayes, Gaussian Mixture Model, Hidden Markov Model, Variational Autoencoder (VAE), Generative Adversarial Networks (GANs)*
  (*GAN은 explicit probability를 모델링하진 않지만 생성 모델로 분류됨)

✅ **Discriminative Model (판별 모델)**

- **목표:** 조건부 확률 $P(Y|X)$ 또는 결정 경계(direct mapping)를 모델링
- **방법:** X 가 주어졌을 때 Y 를 어떻게 잘 맞출지를 직접 학습
- **예시:** Logistic Regression, Support Vector Machine, Decision Tree, Neural Networks (대부분), Conditional Random Fields (CRFs)

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



각각의 방식에서 X, Y의 관계성의 Bayesian Network로의 표현과 Joint Distribution Factorization은 위와 같다.

However, suppose all we need for prediction is $p(Y \mid \mathbf{X})$
In the left model, we need to specify/learn *both* $p(Y)$ and $p(\mathbf{X} \mid Y)$, then compute $p(Y \mid \mathbf{X})$ via Bayes rule
In the right model, it suffices to estimate just the **conditional distribution** $p(Y \mid \mathbf{X})$

- 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 $\mathbf{X}$

일반적인 ML은 X가 주어졌을 때 Y가 뭔지를 알아내는 것이 과제다

→ P(Y|X)를 예측하는 것

이를 위해 GM에서는 P(Y), P(X|Y)를 명시 혹은 학습해야하고, DM에서는 P(Y|X)만 명시 혹은 학습하면 된다.

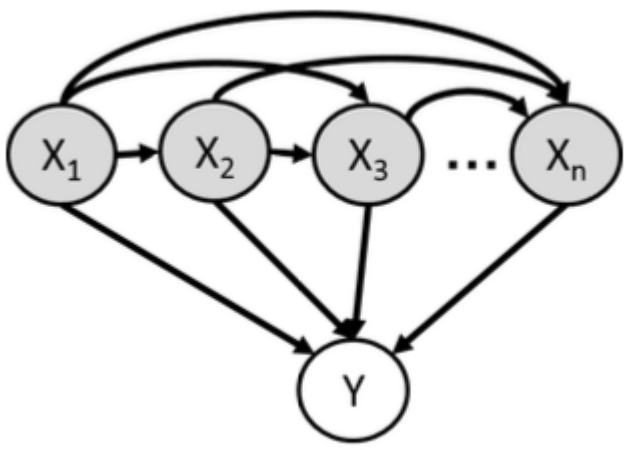
이로써, Discriminative Model에서는 Direct하게 P(Y|X)를 학습하고, Generative Model에서는 Bayes Rule을 이용해 P(Y|X)를 도출한다.

How To Train P(Y|X) in DM? : Logistic Regression

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

Logistic Regression은 다음과 같이 P(Y=1|x)를 a를 파라미터로 가지는 Function으로 모델링하여 학습.

이외의 내용은 아는 것이니 생략.



→ Logistic Regression에서의 확률 변수 간의 관계를 나타내는 Bayesian Network

→ Naive Bayes와 같이 Given Y일 때, Feature간의 독립성을 가정하지 않음, Naive Bayes와 달리 feature간의 관계를 고려한 의사결정을 할 수 있음.

→ 그러나 Discriminative Model은 X를 구성하는 모든 요소를 가지고 있어야 P(Y|X)를 구할 수 있음.

→ 반면 Generative Model은 X의 일부 X evidence(partly unseen variable)만 가지고 있어도 Marginalization을 통해 P(Y|X)를 구할 수 있음.

Advance in Discriminitive Model → Non-linearity

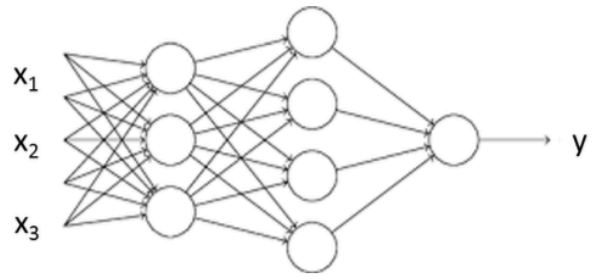
Logistic Regression은 non-linearity를 가정하지 않음. P(Y|X)를 예측하는 F(X; a)가 non-linearity를 가진다면 더욱 복잡한 관계를 capture할 수 있지 않을까?

→ 이 F(X; a)를 Neural Network로 구현하자!

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*).

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

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



Comparison Between Bayesian Network & 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

→ 일반적인 Joint Probablity를 Chain Rule을 통해 Factorization

→ 확률변수에 대한 가정은 없는 상태

→ 파라미터 수가 매우 많음, 모든 state에 대한 확률값을 가지고 있어야함

Bayes Net

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

Assumes conditional independencies

Simplifying the conditions by dropping

→ Conditional Independence를 가정함으로써 joint probability modeling에서의 파라미터 수를 줄일 수 있음

Neural Models

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

→ 맨 앞의 Chain Rule을 통한 Joint Distributon을 Neural Model을 통해 표현하겠다!