기계학습이론 중간대비

Machine Learning이란?

Computer가 **Data를 이용하여 특정 Task를 하는 방법을 배우는 과정**이다. 배움의 정도를 P를 통해 검증할 수 있다.

- **Task**는 **Function H (Hypothesis)를 배우는 것**이다.
- **Performance measure** P: Posterior이다.
- Prior와 Data를 기반으로 **가장 좋은 Hypothesis를 알아내는 방법에 MAP를 사용**한다.
    - Posterior에 Log를 취해 Likelihood와 Prior로 분리한다.
    - **Likelihood**: Hypothesis가 맞다고 가정했을 때, 해당 데이터가 나오는 지를 판단
    - **Prior**: 설계자가 사전에 갖고 있는 지식 , Regularization 등

기계학습에 필요한 Preliminaries을 배우는 이유를 생각해보자.

먼저 **기계학습**의 목표는 **Intelligence, 지능을 모방하는 것**이다.

- Grandmother cell을 생각하면 Intelligence는 굉장히 어렵고 복잡한 함수로 이루어진다고 생각할 수 있다.

하지만 복잡한 함수를 그대로 다루는 것은 불가능하다. 따라서 **우리는 Linear approximation을 하여 해결하는 방법**을 생각할 수 있다.

- $\sqrt{7}$의 값을 $f(x) = x^2 - 7$의 Root를 Netwon method로 다루는 방법과 유사하다.

이때, **Linear approximation**을 하기 위해 **Calculus**가 필요하다.

Calculus를 이용하여 **다변수 함수를 Linear approximation**하는 방법을 생각해보자. 이때, 다변수 함수의 미분결과는 하나의 숫자가 아니라, **Matrix**로 표현된다.

- 우리가 다루는 것은 **Vector**이기 때문이다!
- Linear approximation 뿐만 아니라 **Multidimensional function에 대해 Newton's method를 이용한 Optimization에서 Matrix가 사용되기도 하기 때문에 중요**하다.

이 **Matrix**는 그 자체로 **Linear transformation** (선형 변환)이며, 우리는 **Linear Algebra**의 도구들을 사용해 이 행렬을 다루고 계산한다.

Probability는 모델의 성능 및 불확실성을 표현하고, 모델의 성능 개선에 중요하게 사용된다.

Maximum likelihood estimate는 Data를 중시하며 Induction 적인 사고방식이다.

- **Piror**: Uniform distribution
  - MLE는 Prior가 **Uniform distribution인 MAP**이다.
- **Feature extractor**: Learnable
- **Inductive bias**: 약하다.
- **Hypothesis space**: 크다
- **Task가 쉽고 Dataset이 클 때** 유리하다.
  - Dataset이 크면 사실 Task가 쉽다.

Maximum a Posterior는 Prior, knowledge를 중시하며 Deduction 적인 사고 방식이다.

- **Piror**: Non-Uniform distribution
- **Feature extractor**: Fixed
- **Inductive bias**: 강하다
- **Hypothesis space**: 작다
- **Task가 어렵고 Dataset이 작을 때** 유리하다.
- **Soft prior**: Preference
- **Hard prior**: Constraint

Supervise learning에서는 우리의 **Task**는 **input x로부터 y를 output으로 내도록 하는 함수 H를 배우는 것**이다.

Task를 평가하는 **Performance mesure는 Task의 종류마다 다르다.**

- **Regression:** $E[|f(x) - h(x)|^2]$
- **Classification**: $E[1(f(x) \neq h(x))]$
- $f(x)$가 정답을 의미한다.
- Performance measure를 최소화해야 한다.

Regression에서 $y|x$**가 Gaussian을 따른다고 가정**하면 **MSE를 최소화**하는 것과 동일하던 Performance measure는 **Negative Log Likelihood와 동일**해진다.

함수 $H(x)$를 배우는 것을 **확률 관점**에서 서술하면, 결국 데이터의 분포 $P(y_i|x_i)$를 학습하는 것과 동일해진다.

- **이 관점에서 "주어진 데이터에 대해 데이터를 가장 잘 설명하는 Hypothesis를 찾는다"**라고 할 수 있다.
    - Probability distribution fitting
    - 즉, Maximum likelihood는 **주어진 데이터에 대해 데이터를 가장 잘 설명하는 Hypothesis를 찾는 과정이다.**
        - $p(E|H) = p((y, x)|H) = p(y|x, H)p(x|H) = p(y|x, H)$

데이터의 분포를 유사하게 하고자 하기 때문에 **KL-divergence** 관점에서 서술할 수 있다.

- $KL(p||q)$로 표현하고, $E_p[logp - logq]$이다.
    - p는 목표로 하는 분포, q는 추정 분포이다.
    - 실제로는 Data가 주어졌을 때, Data의 분포를 정확히 알기 쉽지 않기 때문에 **Emprical distribution (**p_s**)**을 대신 사용한다.
- 결국 KL-divergence의 정의에 의해 $E_{p_s}[logp - logq] = -H(p_s(x)) - E_{p_s}[logq(x)]$로 나타낼 수 있고 우리는 q에 대한 정의만 사용하기 때문에 $H(p_s(x))$

를 상수 처리하면 $-E_{p_s}[logq(x)]$만 남는다.

- $-E_{p_s}[logq(x)]$ = $\int p_s(x) - logq(x)dx = -\frac{1}{n}\sum_i logq(x_i) = -\frac{1}{n}\sum_i logp(y_i|x_i, H)$가 되어 **KL Divergence를 줄이는 것은 NLL과 동일함을 확인할 수 있다.**
- $\int p_s(x) - logq(x)dx = -\frac{1}{n}\sum_i logq(x)$인 이유
    - $E_{p_s}[f(x)] = \int f(x)p_s(x)dx = \int f(x)\frac{1}{n}\sum_i \delta(x - x_i)dx$ 에서 $\int \delta(x - x_i)dx = 1$ / $\int f(x_i)\delta(x - x_i)dx = f(x_i)$이기 때문에 $\int \frac{1}{n}\sum_i f(x)\delta(x - x_i)dx = \frac{1}{n}\sum_i f(x_i)$가 된다.

마지막으로 **Emprical Risk Minimization으로 생각할 수 있다.**

- Loss $l(y_i, h(x_i))$를 $-logp(y_i|x_i, H)$로 잡자.
- $L_s(h) = \hat{E}_{(x_i,y_i)\subset S)}[l(y_i, h(x_i))] = \frac{1}{|S|}\sum_{(x_i,y_i)\subset S)} l(y_i, h(x_i))$ 이다.
- **NLL은 $-logp(y_i|x_i, H)$가 Loss인 ERM이다.**

Classification 관점은 다르다.

- 위에서 **NLL ~ ERM 연결 과정까지 모두 동일**하다.
- **Likelihood:** $p(y_i|x_i, H) = \sigma(z)^{y_i}(1 - \sigma(z))^{1-y_i}$이다.
    - z = $H(x) = W^T x$로, W가 나타내는 **Boundary와 data x와의 거리**이다.
    - $\sigma()$: 거리 Z를 통해 x**가 클래스 1일 확률**을 나타낸 것이다.
- 다른 점은 $p(y_i|x_i, H)$가 Gaussian distribution이 아니라, **2-class인 경우 Bernoulli distribution**을 **n-class인 경우 categorical distribution을 따른다는 점**이 다르다.

Logistic regression에 **NLL을** 적용하면 다음과 같다.

- $L_s(W) = -\sum_i log[f_W(x_i)]_{y_i}$
- $p(y|x)$가 $Bern(y|\sigma\hat{(z)})$를 따른다면 **sigmoid를 이용한다.**
    - $z = Wx$, $f_W(x_i)$ **= Sigmoid(z)**가 된다.
- $p(y|x)$가 $Cat(y|f_W(x_i))$를 따른다면 **softmax**를 이용한다.

- $f_W(x)$는 Softmax(Wx)이다. Wx가 결과가 된다.
- W는 C*P, X는 P*1

위에서 본 Regression, Classification model은 전부 다 **Linear model**이다. W^Tx를 사용하기 때문이다.

Classification model을 예시로 보면 위 Linear model은 **XOR Probelm**을 해결하지 못한다.

이 문제를 해결하기 위해 **두 가지 방법**을 사용해볼 수 있다.

1. **Data 자체의 차원을 조정**하여 Linear하게 해결하도록 할 수 있다.
2. **Nonlinear basis function**을 이용한다.

하지만 **Nonlinear basis function**을 사용하게 되면 함수가 표현 가능한 Feature가 제한되고, 사람이 직접 정해 주어야 하기 때문에 쉽지 않다.

이로 인해 **Nonlinear basis function까지 model이 직접 배우도록 하여 표현력을 높이려는 Parameterized feature extractor** 방법을 사용한다.

- Non-linear function처럼 동작하도록 하려면 **Linear하게 계산한 후에 Activation fucntion을 통해 Non-linearity를 추가**할 수 있다.

Parameterizd feature extractor는 위처럼 Nonlinear basis function처럼 동작하도록 하는 것을 학습해야 함으로 2-Layer NNs로 구현할 수 있다.

- **첫 번째 Weight는 Activation function 함수를 통해 비선형성을 학습**하고, **두 번째 Weight는 최종 출력을 학습**하도록 한다.
  - $h = \sigma(W_1^Tx), y = W_2^Th$
  - W_1을 조정하여 비선형 특징을 조절하고 W_2를 조정하여 최종 출력을 조정한다.
- **Universal Approximation Theorem**에 의하면 2-Layer NNs로 어떠한 함수던지 근사할 수 있다.

이 과정 후에 2-layer보다 **더 깊게 쌓으며** Model의 성능을 개선한다.

- Model의 Layer가 쌓일수록 **모델이 Feature를 계층적으로 학습하기 때문에 성능이 더 좋아진다고 한다.**

위 과정에서는 **Inductive bias**가 점점 약해진다.

- **Inductive bias**: 사람의 사전 지식, Prior
    - 전체 **Function space를 줄여나가는 방법**을 찾는 것

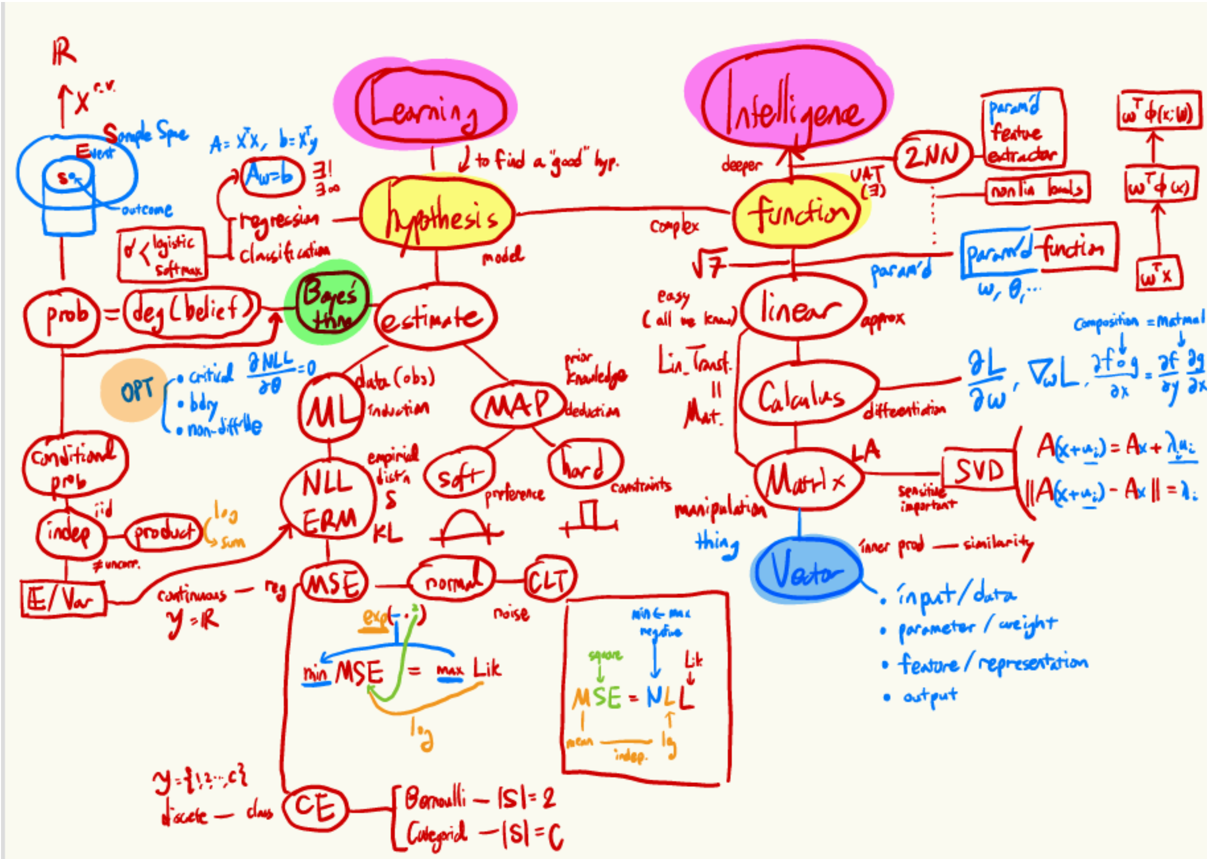
Inductive bias가 너무 약해지면 성능이 오히려 저하될 수 있기에 **Inductive bias를 오히려 추가**하기도 한다.

- Model에게 어떤 Task를 다루는 지 등을 미리 알려준다.

하지만 요즘 추세는 **Inductive bias를 다시 줄이는 추세**이다.

- Transformer의 등장 배경과 관련 있다.

	Bernoulli	**Gaussian**	**Reg'n**	**Class'n**
example	coin tossing	data fitting	Housing Price	Iris
data	$y_i \in \{\mathrm{T}, \mathrm{H}\}$	$y_i \in \mathbb{R}$	$\{(x_i, y_i)\}_{i=1}^n, y_i \in \mathbb{R}$	$\{(x_i, y_i)\}_{i=1}^n, y_i \in [C]$
target type	discrete	continuous	continuous	discrete
hypothesis	$y_i \sim_{iid} \mathrm{Bern}(\theta)$	$y_i \sim_{iid} \mathcal{N}(\mu, \sigma^2)$	$y_i \mid x_i \sim_{iid} \mathcal{N}(h(x), \sigma^2)$	$y_i \mid x_i \sim_{iid} \mathrm{Cat}(h(x))$
hyp. var.	$\theta \in [0, 1]$	$\mu \in \mathbb{R}, \sigma^2 \in \mathbb{R}^+$	$h : \mathcal{X} \to \mathbb{R}$	$h : \mathcal{X} \to [0, 1]^C$
NLL	CE	MSE	MSE	CE
equation	$-[k \log(\theta) + (n-k) \log(1-\theta)]$	$\frac{1}{2\sigma^2} \sum_{i=1}^n (y_i - \mu)^2$	$\frac{1}{2\sigma^2} \sum_{i=1}^n (y_i - h(x_i))^2$	$-\sum_{i=1}^n e_{y_i}^\top \log h(x_i)$
prior (e.g.)	$p(\theta) \propto \theta^m (1-\theta)^m$	$p(\mu) \propto \exp(-\mu^2)$	?[120]	?
prior (e.g.)	$p(\theta) \propto U(\theta; [0.4, 0.6])$	-	?[121]	?[122]
MLE	$\theta^* = \frac{k}{n}$	$\mu^* = \frac{1}{n} \sum_{i=1}^n y_i$	h^*?[123]	h^*?



- Learning은 곧 좋은 Hypothesis를 찾는 과정이다.
- Hypothesis 자체가 Function이기 때문에 Model은 Intelligence를 모방하는 어떤 것이 된다.
- Estimate는 좋은 Hypothesis를 구성하는 어떤 Parameter를 찾는 과정이다.