Deep Learning

Machine Learning Basic

Deep Learning Book
Bishop: Pattern Recognition and Machine Learning
Machine Learning: a Probabilistic Perspective
Learning from data

Kyungwoo Song

Contents

- 5.1 Learning Algorithm
- 5.2 Capacity, Overfitting and Underfitting
- 5.3 Hyper-parameters and Validation Sets
- 5.4 Estimators, Bias and Variance
- 5.5 Maximum Likelihood Estimation
- 5.6 Bayesian Statistics
- 5.9 Stochastic Gradient Descent
- 5.11 Challenges Motivating Deep Learning

Source:

Learning Algorithm

5.1 Learning Algorithms

- A machine learning algorithm is an algorithm that is able to learn from data.
- But what do we mean by learning?
  - Mitchell (1997) provides the definition "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E."

- Task
  - Learning is our means of attaining the ability to perform the task.
  - Example) If we want a robot to be able to walk, then walking is the task
  - Machine learning tasks are usually described in terms of how the machine learning system should process an example.
    - ❖Example: collection of features, $x \in R^n$ where each entry x_i of the vector is another feature

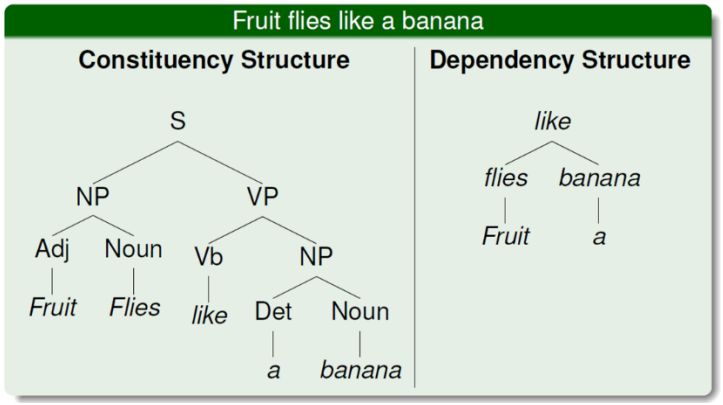
Source: Machine Learning, Tom Mitchell, McGraw Hill, 1997.

- Task
  - Classification
    - $f: R^n \rightarrow \{1, \ldots, k\}$
    - $y = f(x)$, the model assigns an input described by vector x to a category identified by numeric code y
  - Classification with missing inputs
    - the computer program is not guaranteed that every measurement in its input vector will always be provided
    - This kind of situation arises frequently in medical diagnosis
    - When some of the inputs may be missing, rather than providing a single classification function, the learning algorithm must learn a set of functions.
  - Regression
    - $f: R^n \rightarrow R$
  - Transcription
    - Observe an unstructured representation of data and transcribe it into discrete, textual form
    - OCR (optical character recognition), Speech recognition, …

Source: Machine Learning, Tom Mitchell, McGraw Hill, 1997.

5.1 Learning Algorithms

- Task
  - Machine Translation
  - Structured output
    - ❖Broad category
    - ❖The program output several values that are all tightly inter-related
    - ❖Subsumes the transcription and translation
    - ❖Example) Parsing
      - ➢ Mapping a natural language into a tree that describes its grammatical structure
    - ❖Example) pixel-wise segmentation of image

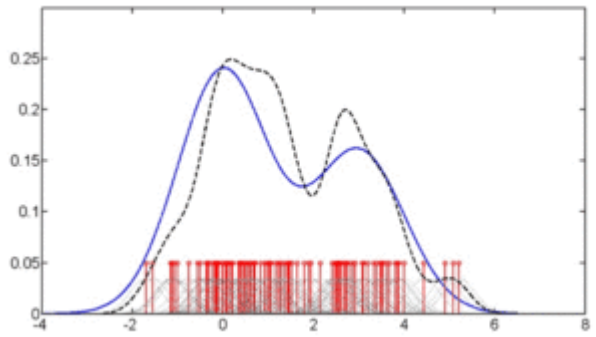


Source: https://www.cs.bgu.ac.il/~elhadad/nlp13/nlp03.html https://www.researchgate.net/figure/Example-cases-of-pixel-wise-segmentation-performed-by-SegNet-on-real-road-scenarios_fig2_304789242

- Task
  - Anomaly detection
    - ❖Sifts through a set of events or objects as being unusual or atypical
    - ❖Example) credit card
      - ➢ By modeling your purchasing habits, a credit card company can detect misuse of your cards
  - Synthesis and Sampling
    - ❖In this type of task, the machine learning algorithm is asked to generate new examples that are similar to those in the training data.
  - Imputation of missing values
    - ❖Some entries x_i of x is missing
    - ❖The algorithm provide a prediction of the values of the missing entries
  - Denoising
    - ❖Corrupted example $\tilde{x} \in R^n$ obtained by an unknown corruption process from a clean example $x \in R^n$
    - ❖Predict the clean example $p(x|\tilde{x})$

Source: https://www.cs.bgu.ac.il/~elhadad/nlp13/nlp03.html https://www.researchgate.net/figure/Example-cases-of-pixel-wise-segmentation-performed-by-SegNet-on-real-road-scenarios_fig2_304789242

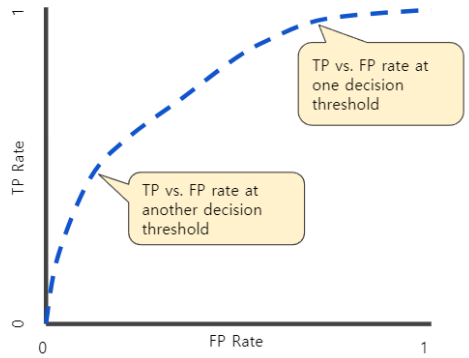
- Task
  - Density estimation or probability mass function estimation
    - The machine learning algorithm is asked to learn a function $p_{model}: R^n \rightarrow R$
    - $p_{model}(x)$: probability density function (for continuous x) or a probability mass function (for discrete x)
    - It must know where examples cluster tightly and where they are unlikely to occur
    - Example) we can use the distribution to solve the missing value imputation task
      - If a value x_i is missing and all of the other values, denoted x_{-1}, we know the distribution over it is given by $p(x_i|x_{-i})$
    - In practice, $p(x)$ are computationally intractable



Blue: True density
Red: samples generated from the true distribution
Dashed black: estimated density

Source: https://www.cs.bgu.ac.il/~elhadad/nlp13/nlp03.html https://www.researchgate.net/figure/Example-cases-of-pixel-wise-segmentation-performed-by-SegNet-on-real-road-scenarios_fig2_304789242

- Performance
  - Accuracy: proportion of examples for which the model produces the correct output
  - Error rate: proportion of incorrect output
  - Recall: $\frac{TP}{TP+FN}$, Precision: $\frac{TP}{TP+FP}$, F1: $2 \times \frac{1}{\frac{1}{Precision}+\frac{1}{Recall}}$
    - ❖Recall: 실제 positive 인 것 중에서, 우리 모델이 positive 로 잘 예측한 비율
    - ❖Precision: 우리 모델이 positive 라고 예측한 것 중에서, 실제로 positive 인 비율



		Predicted condition	
	Total population = P + N	Positive (PP)	Negative (PN)
Actual condition	Positive (P)	True positive (TP), hit	False negative (FN), type II error, miss, underestimation
	Negative (N)	False positive (FP), type I error, false alarm, overestimation	True negative (TN), correct rejection

  - ROC-AUC
    - ❖TPR: $\frac{TP}{TP+FN}$ (Recall)
    - ❖FPR: $\frac{FP}{FP+TN}$
    - ❖항상 Positive로 예측: $TPR \uparrow$, $FPR \uparrow$
    - ❖항상 Negative로 예측: $TPR \downarrow$, $FPR \downarrow$

Source: https://en.wikipedia.org/wiki/Precision_and_recall https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc?hl=ko

5.1 Learning Algorithms

Model	CIFAR10 32 × 32 NLL (↓)	CIFAR10 32 × 32 FID (↓)	CIFAR10 32 × 32 IS (↑)	CelebA 64 × 64 NLL	CelebA 64 × 64 FID	CelebA-HQ 256 × 256 FID	STL-10 48 × 48 FID	STL-10 48 × 48 IS
UDM (RVE) + ST	3.04	2.33	**10.11**	1.93	**2.78**	**7.16**	**7.71**	**13.43**
Likelihood-based Models								
CR-NVAE [26]	**2.51**	-	-	**1.86**	-	-	-	-
LSGM (FID) [21]	3.43	**2.10**	-	-	-	7.22	-	-
DenseFlow-74-10 [4]	2.98	-	-	1.99	-	-	-	-
Gamma Distribution DDIM [27]	-	-	-	-	2.92	-	-	-
VDM [8]	2.65	-	-	-	-	-	-	-
NCSN++ cont. (deep, VE) [7]	3.45	2.20	9.89	-	-	-	-	-
DDPM++ cont. (deep, sub-VP) [7]	2.99	2.41	9.57	-	-	-	-	-
ScoreFlow (cont. norm. flow) [12]	2.74	5.70	-	-	-	-	-	-
Improved DDPM (L_{simple}) [19]	3.37	2.90	-	-	-	-	-	-
Likelihood-free Models								
StyleGAN2-ADA+Tuning [28]	-	2.92	10.02	-	-	-	-	-
Styleformer [29]	-	2.82	9.94	-	3.66	-	15.17	11.01
PGGAN [30]	-	-	8.8	-	-	8.03	-	-
TransGAN [31]	-	9.26	9.02	-	5.01	-	18.28	10.43

- Performance
  - For tasks such as density estimation,
    - ❖ Log-likelihood: the average log-likelihood the model assigns to some examples
    - ❖ Inception Score
      - ➢ Fidelity, Diversity
      - ➢ $\exp E_{x\sim p_g}\left[D_{KL}\left(p(y|x)\,||\,p(y)\right)\right]$
        - ✓ Inception v3 model (2048 dim)
        - ✓ $p(y|x)$: conditional class distribution
        - ✓ $p(y) = \int_x p(y|x)p_g(x)\,dx$: marginal class distribution
      - ➢ $p(y|x)$ should be low entropy (Fidelity)
      - ➢ $p(y)$ should be high entropy (Diversity)
    - ❖ Fréchet inception distance (FID)
      - ➢ FID compares the distribution of generated images with the distribution of real images
      - ➢ 2-Wasserstein metric between two Gaussian distributions
      - ➢ For univariate normal, $\left(\mu_x - \mu_y\right)^2 + \left(\sigma_x - \sigma_y\right)^2$
      - ➢ For multivariate normal, $\left\|\mu_x - \mu_y\right\|^2 - tr(\Sigma_x + \Sigma_Y - 2\Sigma_x\Sigma_y)$
        - ✓ Inception v3 model (2048 dim)

각 class별로 동일한 image를 내보낸다면?

Source: https://arxiv.org/abs/2106.05527

5.1 Learning Algorithms

- Experience
  - Most of the learning algorithms in this book can be understood as being allowed to experience an entire dataset
    - Dataset: collection of examples
  - Unsupervised learning algorithm: learn useful properties of the structure of dataset
    - Entire probability distribution that generated a dataset, Clustering, …
    - implicitly or explicitly learn the probability distribution $p(x)$,
  - Supervised learning algorithm: each example is associated with a label or target
    - Classification, Regression, …
    - Estimate $p(y|x)$
  - It is note that unsupervised learning and supervised learning are not formally defined terms.
    - $p(x) = \prod_{i=1}^{n} p(x_i|x_1, \ldots, x_{i-1})$
    - Unsupervised problem of modeling $p(x)$ can be splitted into N supervised learning problems
  - Semi-supervised learning, multi-instance learning, reinforcement learning, self-supervised learning, …
    - multi-instance learning: an entire collection of examples is labeled as containing or not containing an example of a class, but the individual members of the collection are not labeled

Source: https://nilg.ai/blog/202105/an-introduction-to-multiple-instance-learning/

Generalization error

5.2 Capacity, Overfitting and Underfitting

- The central challenge in ML
  - we must perform well on new, previously unseen inputs, not just those on which our model was trained
  - The ability to perform well on previously unobserved inputs is called generalization.

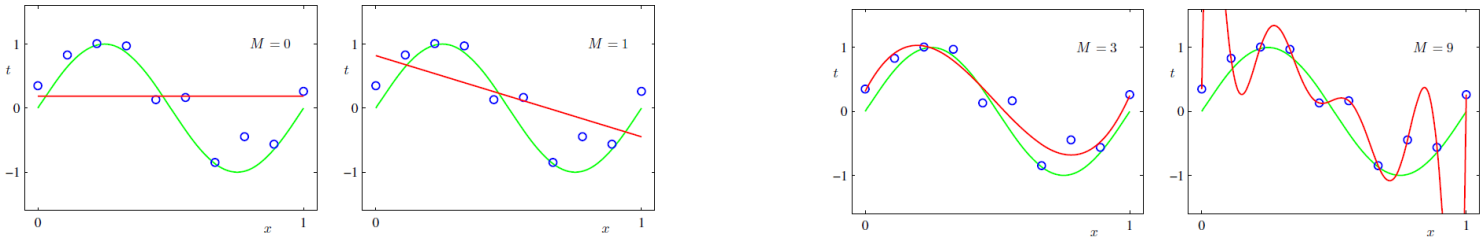
- Generalization error
  - Expected value of the error on a new input
  - We typically estimate the generalization error on a test set
  - Training error: $\frac{1}{m^{(train)}} \left\| X^{(train)} w - y^{(train)} \right\|_2^2$
  - Test error: $\frac{1}{m^{(test)}} \left\| X^{(test)} w - y^{(test)} \right\|_2^2$
  - How can we affect performance on the test set when we get to observe only the training set?
    - ❖Statistical learning theory

Source:

Generalization error

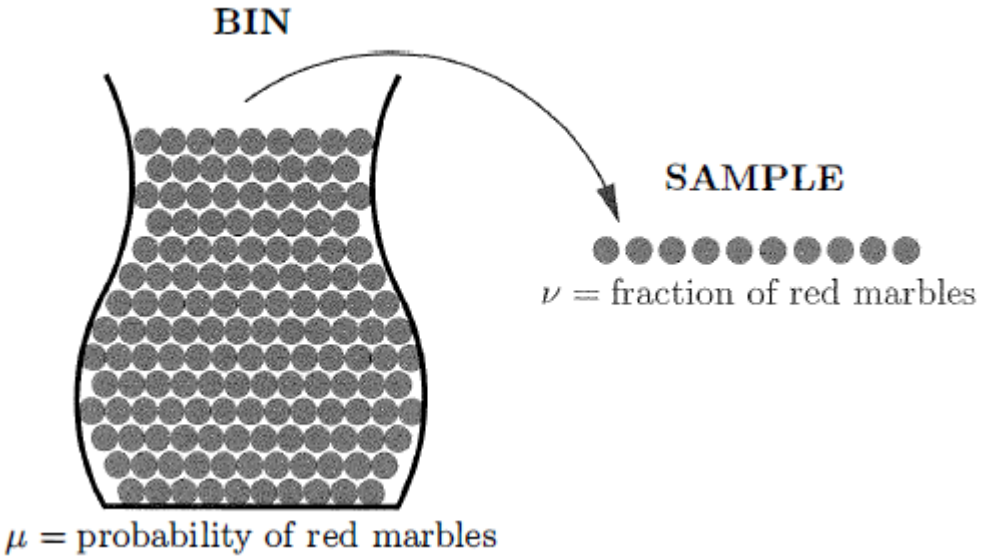
5.2 Capacity, Overfitting and Underfitting

- We sample the training set, then use it to choose the parameters to reduce training set error, then sample the test set
  - The expected test error is greater than or equal to the expected value of training error
    - ❖ The factors determining how well a machine learning algorithm will perform
      - ➢ Make the training error small
      - ➢ Make the gap between training and test error small
    - ❖ Underfitting: model is not able to obtain a sufficiently low error value on the training
    - ❖ Overfitting: gap between the training error and test error is too large
  - We can control whether a model is more likely to overfit or underfit by altering its capacity
    - ❖ Models with low capacity may struggle to fit the training set
    - ❖ Models with high capacity can overfit by memorizing properties of the training set



Source:

- Occam’s razor
  - Among competing hypotheses that explain known observations equally well, one should choose the “simplest” one.
  - How to measure the model capacity?
    - ❖Vapnik-Chervonenkis dimension (VC dimension)
    - ❖VC dimension 을 알아보기 위해, 잠시 돌아가도록 하겠습니다.



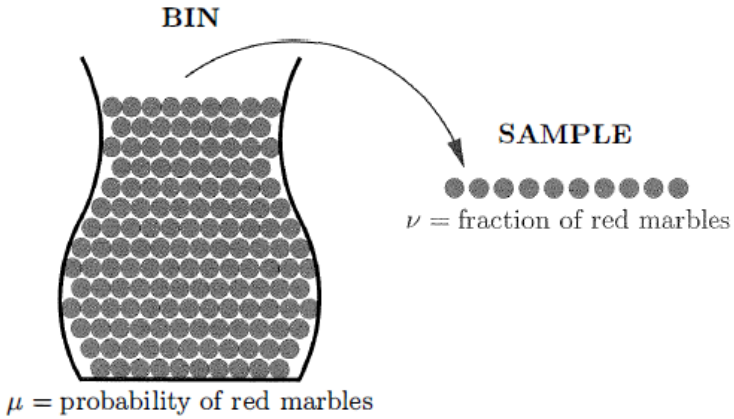
We pick N marbles independently

- ν: 무작위로 구슬을 N 개뽑았을 때, 그 구슬 중 빨간색의 비율
- μ: 전체에서,빨간색 구슬의 비율

Source:

Hoeffding's Inequality

5.2 Capacity, Overfitting and Underfitting



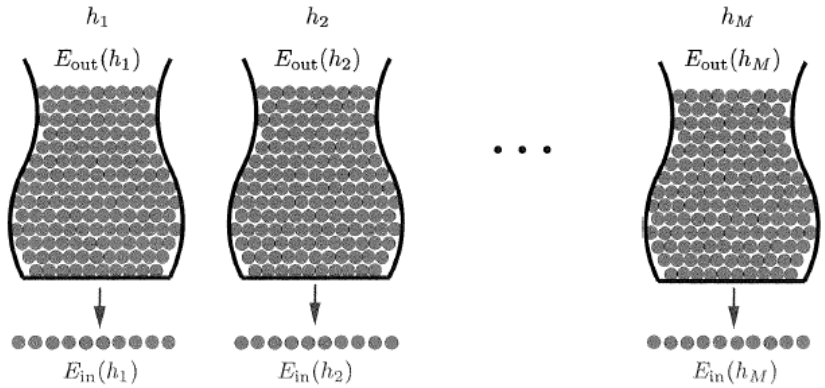
We pick N marbles independently
- v: 무작위로 구슬을 N 개뽑았을 때, 그 구슬 중 빨간색의 비율
- μ: 전체에서,빨간색 구슬의 비율

- Hoeffding's Inequality
  - Provides an upper bound on the probability that sum of bounded independent r.v. deviates from its expected value
  - Let $Y_1, \dots Y_n$ be iid observations such that $E[Y_i] = \mu$ and $a \leq Y_i \leq b$. Then, for any $\epsilon > 0,\ P(|\bar{Y} - \mu| \geq \epsilon) \leq 2e^{-2n\epsilon^2/(b-a)^2}$
  - $\Rightarrow P[|v - \mu| > \epsilon] \leq 2e^{-2\epsilon^2 N}$
- ML과 연결시킨다면, 빨간색 구슬을 뽑는것은 Hypothesis h가 잘못 분류한 데이터, 초록색은 정확하게 분류한 데이터로 생각

Source:

5.2 Capacity, Overfitting and Underfitting

- 1개의 bin으로 생각 했을 때: 1개의 hypothesis, h, 만을 고려 했을 때
  - $P[|v-\mu| > \epsilon] \leq 2e^{-2\epsilon^2 N}$
- 하지만 우리는, hypothesis set $h \in \mathcal{H}$을 고려하고자 함
  - Assumption: H has a finite number of hypotheses
  - $\mathcal{H} = \{h_1, \ldots, h_M\}$
  - $E_{in}(h_i)$: In sample error
    - ❖앞에서의 μ 에 대응
    - ❖$E_{in}(h_i) = \frac{1}{N}\sum_{n=1}^{N} I_{h(x_n)\neq f(x_n)}$
  - $E_{out}(h_i)$: Out of sample error
    - ❖앞에서의 v 에 대응
    - ❖$E_{out}(h_i) = P[h(x) \neq f(x)]$
  - g: Final hypothesis
  - $P[|E_{in}(g) - E_{out}(g)| > \epsilon]$???

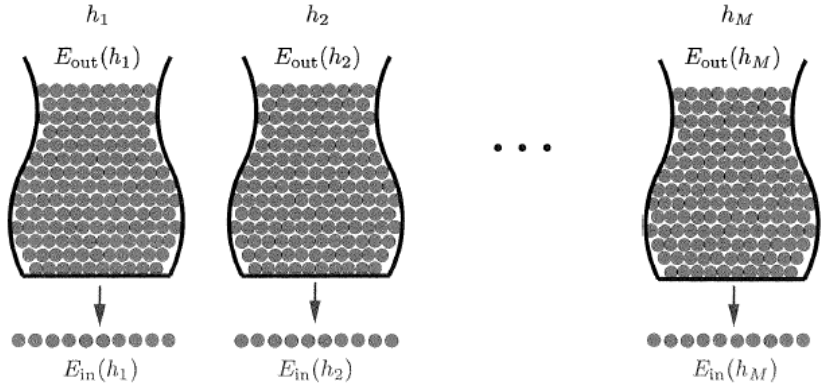


Source:

- 하지만 우리는, hypothesis set $h \in \mathcal{H}$을 고려하고자 함
  - g: Final hypothesis
  - $P[|E_{in}(g) - E_{out}(g)| > \epsilon]$???

Bad event

- $|E_{in}(g) - E_{out}(g)| > \epsilon$
  - $\Rightarrow |E_{in}(h_1) - E_{out}(h_1)| > \epsilon$
  - or $|E_{in}(h_2) - E_{out}(h_2)| > \epsilon$
  - …
  - or $|E_{in}(h_M) - E_{out}(h_M)| > \epsilon$



- $P(|E_{in}(g) - E_{out}(g)| > \epsilon) \leq P(|E_{in}(h_1) - E_{out}(h_1)| > \epsilon$ or $|E_{in}(h_2) - E_{out}(h_2)| > \epsilon$ … or $|E_{in}(h_M) - E_{out}(h_M)| > \epsilon)$
  $\leq \sum_{m=1}^{M} P(|E_{in}(h_m) - E_{out}(h_m)| > \epsilon)$

- $P[|E_{in}(g) - E_{out}(g)| > \epsilon] \leq 2Me^{-2\epsilon^2 N} \Rightarrow E_{out}(g) \leq E_{in}(g) + \sqrt{\frac{1}{2N}\ln\frac{2M}{\delta}}$ with probability $\geq 1 - \delta$
  - ❖where $\delta = 2Me^{2N\epsilon^2}$
  - ❖만약 $M \gg 0$라면, $2Me^{-2\epsilon^2 N}$는 1보다 큰 값이 되고, 그럼 유의미하지 않음
  - ❖단순 sum (disjoint) 으로 loose bound 를 구한 것

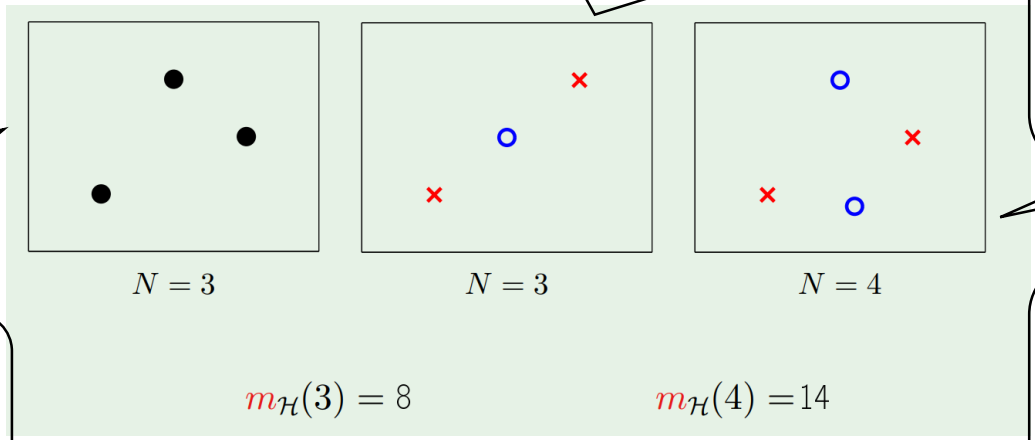
Source:

- $P[|E_{in}(g) - E_{out}(g)| > \epsilon] \leq 2Me^{-2\epsilon^2 N} \Rightarrow E_{out}(g) \leq E_{in}(g) + \sqrt{\frac{1}{2N}\ln\frac{2M}{\delta}}$ with probability $\geq 1 - \delta$
  - 만약 $M \gg 0$라면, $2Me^{-2\epsilon^2 N}$는 1보다 큰 값이 되고, 그럼 유의미하지 않음
- 좋은데, M (The number of hypothesis)에 depend 하네.
  - 만약 $\mathcal{H}$가 infinite 하다면?
  - Unfortunately, almost all interesting learning models have infinite $\mathcal{H}$
    - ❖h끼리 비슷한것들이 많지 않을까? m개로 grouping 할 수 있다면?
- We would like to replace M
  - 만약 N에 polynomial 로 bound를 할 수 있다면, 유의미할텐데…
- Dichotomy and growth function (For given N points) (데이터 관점에서 보자)
  - Definition) Let $x_1, \ldots, x_N \in \mathcal{X}$. The dichotomies generated by $\mathcal{H}$ on these points are defined by $\mathcal{H}(x_1, \ldots, x_N) = \{h(x_1), \ldots h(x_N) | h \in H\}$
  - Definition) growth function $m_H(N)$: maximum number of dichotomies
  - For binary case, $m_H(N) \leq 2^N$
    - ❖$\mathcal{H}(x_1, \ldots, x_N) \subset \{-1, +1\}^N$

Source:

- Dichotomy and growth function (For given N points)
  - Definition) Let $x_1, \dots, x_N \in \mathcal{X}$. The dichotomies generated by $\mathcal{H}$ on these points are defined by $\mathcal{H}(x_1, \dots, x_N) = \{h(x_1), \dots h(x_N) | h \in H\}$
  - Definition) growth function $m_H(N)$: maximum number of dichotomies
  - For binary case, $m_H(N) \leq 2^N$
    - $\mathcal{H}(x_1, \dots, x_N) \subset \{-1, +1\}^N$

6개의 dichotomy



8개의 dichotomy

3개에 대해서 2차원라인으로 Shatter
$\mathcal{H}$ shatter 3 points

$N = 3$

$m_{\mathcal{H}}(3) = 8$

최대 8

$N = 3$

$N = 4$

$m_{\mathcal{H}}(4) = 14$

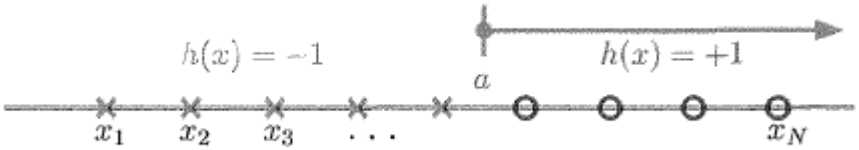
4개에 대해서, 2차원 라인으로는 16개의 dichotomy 찾을 수 없다
⇒ shatter 할 수 없다

$\mathcal{H}$가 Shattering 을 못하는 거라면, Learning이 가능 (현실적인 bound)

Break point for shattering: 4
Maximum non-break point (VC dim): 3

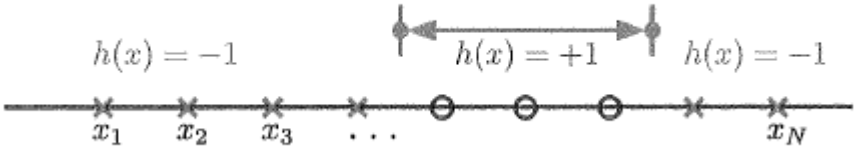
Source:

- Positive rays
  - $\mathcal{H}$ consists of all hypotheses $h: R \rightarrow \{-1, +1\}$ of the form $h(x) = sign(x - a)$
  - $m_{\mathcal{H}}(N) = N + 1$

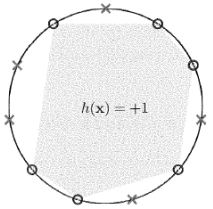


N+1 구간에서 2개의 점
+ 동일한 점 2개 선택

- Positive Intervals
  - $\mathcal{H}$: one dimension that return +1 within some interval
  - $m_{\mathcal{H}}(N) = \binom{N+1}{2} + 1$



- Convex sets
  - Convex set: A set is convex if the line segment connecting any two points in the set lies entirely within the set
  - $\mathcal{H}$: set of $h: R^2 \rightarrow \{-1, +1\}$ that are positive inside some convex set
  - $m_{\mathcal{H}}(N) = 2^N$



Source:

- $P[|E_{in}(g) - E_{out}(g)| > \epsilon] \leq 2Me^{-2\epsilon^2 N}$
- $E_{out}(g) \leq E_{in}(g) + \sqrt{\frac{1}{2N}\ln\frac{2M}{\delta}}$ with probability $\geq 1 - \delta$
  - 만약 M을 $m_H(N)$으로 바꿔끼운다고 생각하면, $m_H(N)$이 polynomial 일 때, 유의미
  - 다만, 이전 장에서 확인할 수 있었듯이, $m_H(N)$ 자체를 구하기가 쉽지 않음
  - $\Rightarrow m_H(N)$ 의 bound를 구하자
  - Break point 가 존재하면, $m_H(N)$ 은 $k-1$ 차의 polynomial로 bound가 된다.
    - ❖Minimum break point: k (더 이상 shattering이 안되는 지점)
    - ❖VC dimension: $k-1$

Theorem 2.4. If $m_{\mathcal{H}}(k) < 2^k$ for some value k, then

$$m_{\mathcal{H}}(N) \leq \sum_{i=0}^{k-1} \binom{N}{i}$$

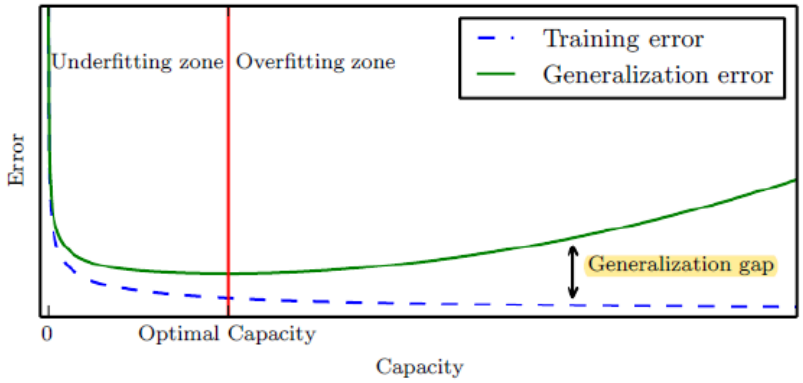
for all N. The RHS is polynomial in N of degree $k-1$.

VC generalization bound

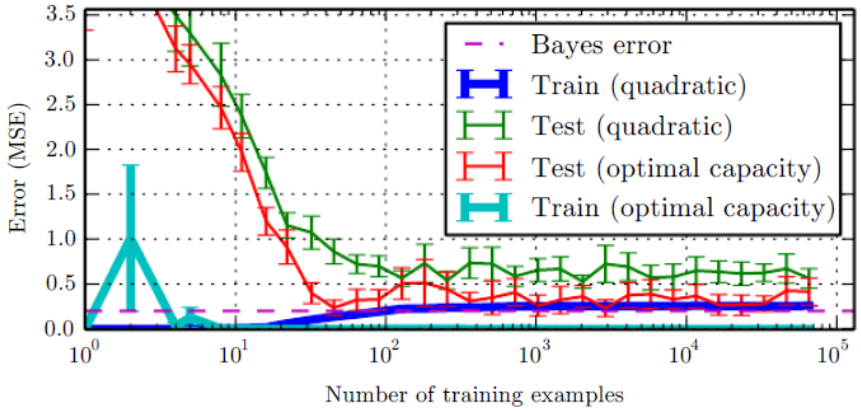
$$E_{out}(g) \leq E_{in}(g) + \sqrt{\frac{8}{N}\ln\frac{4m_H(2N)}{\delta}} \text{ with probability } \geq 1 - \delta$$

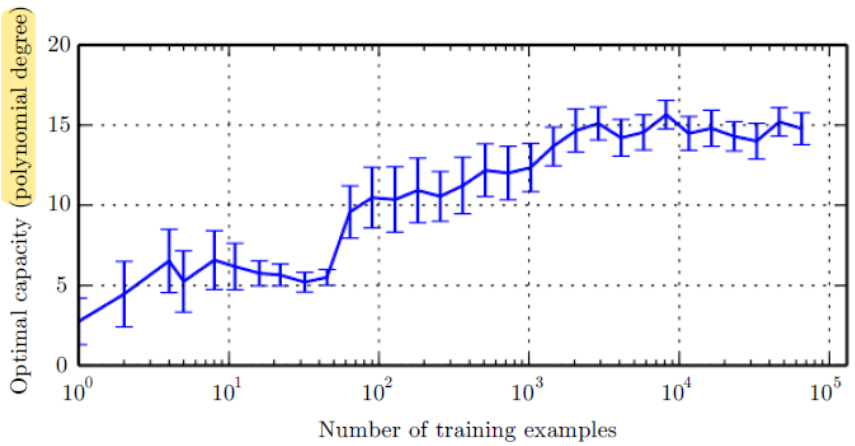
Source:

5.2 Capacity, Overfitting and Underfitting



Bayes error: The error incurred by an oracle making predictions from the true distribution $p(x, y)$



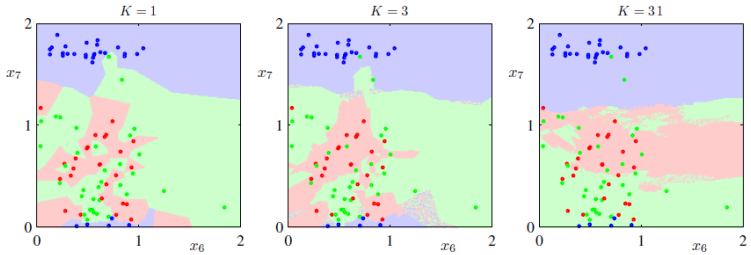


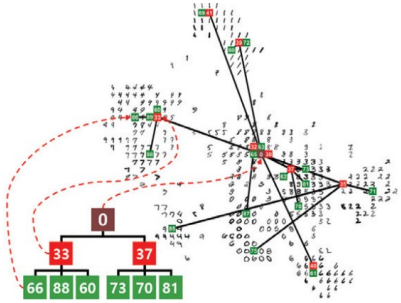
Source:

5.2 Capacity, Overfitting and Underfitting

- We must remember that while simpler functions are more likely to generalize (to have a small gap between training and test error) we must still choose a sufficiently complex hypothesis to achieve low training error

- To reach the most extreme case of arbitrarily high capacity, we introduce non-parametric model
  - Parametric model: learn a function described by a parameter vector whose size is finite and fixed
  - non-parametric models have no such limitation
- Example) nearest neighbor regression, nested Chinese Restaurant Processes (nCRP)





Source: Hierarchically Clustered Representation Learning

Hyperparameters

5.3 Hyperparameters and Validation Sets

- Most machine learning algorithms have several settings that we can use to control the behavior of the learning algorithm
  - Hyperparameters
  - Example) In the polynomial regression, the degree of the polynomial is a hyperparameter
  - Example) The λ value used to control the strength of weight decay

- 만약 training error 를 minimize 하는 것으로 hyper-parameter 잡는다면?
  - We can always fit the training set better with a higher degree polynomial and a weight decay setting of $\lambda = 0$
  - To solve the problem, we need a validation set

Source:

AIC and BIC

5.3 Hyperparameters and Validation Sets

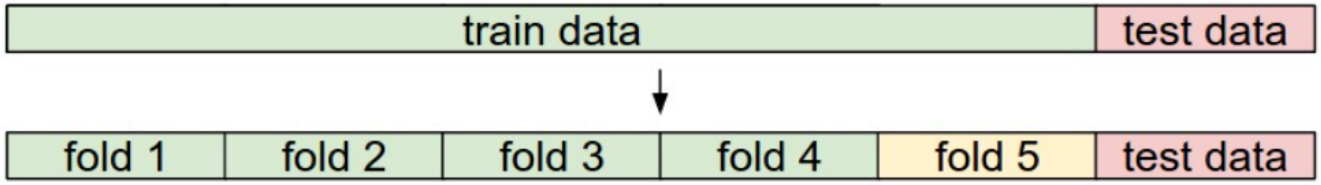
- Another model selection method
  - Information criteria: correct for the bias of maximum likelihood by the addition of a penalty term to compensate for the over-fitting or more complex models
  - AIC: Akaike information criterion
    - $\ln p(D|\theta_{MLE}) - M$
    - M: the number of adjustable parameters
    - $\ln p(D|\theta_{MLE})$: best-fit log likelihood
  - BIC: Bayesian information criterion
    - $\ln p(D|\theta_{MAP}) - \frac{1}{2} M \ln N$
      - M: the number of adjustable parameters
      - N: the number of data points
      - $\ln p(D) \approx \ln p(D|\theta_{MAP}) - \frac{1}{2} M \ln N$

Source: https://cs231n.github.io/classification/ PRML Chapter 4

K fold Cross-Validation

5.3 Hyperparameters and Validation Sets

- K fold Cross-Validation
  - For example, in 5-fold cross-validation, we would split the training data into 5 equal folds, use 4 of them for training, and 1 for validation
  - We would then iterate over which fold is the validation fold, evaluate the performance, and finally average the performance across the different folds.
  - $K = N$: leave-one-out cross validation
    - ❖We always train on $N - 1$ items and validate on the remaining one



Common data splits. A training and test set is given. The training set is split into folds (for example 5 folds here). The folds 1-4 become the training set. One fold (e.g. fold 5 here in yellow) is denoted as the Validation fold and is used to tune the hyperparameters. Cross-validation goes a step further and iterates over the choice of which fold is the validation fold, separately from 1-5. This would be referred to as 5-fold cross-validation. In the very end once the model is trained and all the best hyperparameters were determined, the model is evaluated a single time on the test data (red).

Source: https://cs231n.github.io/classification/

Definition

Let $\hat{\theta}$ be a point estimator for a parameter θ. Then $\hat{\theta}$ is an unbiased estimator if $E(\hat{\theta}) = \theta$. If $E(\hat{\theta}) \neq \theta, \hat{\theta}$ is said to be biased

Example

- If X_i is a Bernoulli r.v. with parameter p, then: $\hat{p} = \frac{1}{n}\sum_{i=1}^{n} X_i$ is an unbiased estimator of p

- Why? $E(\hat{p}) = E\left(\frac{1}{n}\sum_{i=1}^{n} X_i\right) = \frac{1}{n}\sum_{i=1}^{n} E(X_i) = \frac{1}{n}\sum_{i=1}^{n} p = p$
- NOTE: $\hat{p} = \frac{1}{n}\sum_{i=1}^{n} X_i$ is the maximum likelihood estimator (MLE) of p

Source:

Definition

The bias of a point estimator $\hat{\theta}$ is given by $B(\hat{\theta}) = E(\hat{\theta}) - \theta$

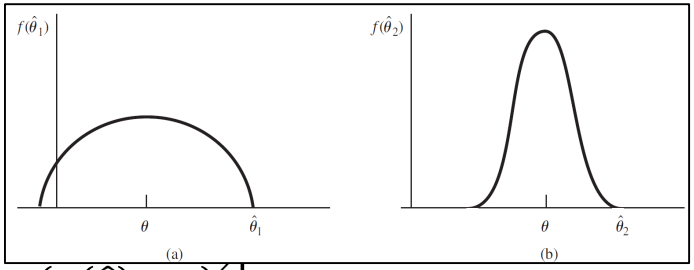
- $B(\hat{\theta}) = E(\hat{\theta}) - \theta > 0$: Positively biased point estimator

Average of the square of the distance

Unbias, low variance (ex. MVUE)

Definition

The mean square error of a point estimator $\hat{\theta}$ is MSE($\hat{\theta}$)= $E[(\hat{\theta} - \theta)^2]$



- NOTE: MSE($\hat{\theta}$)= $E\left[(\hat{\theta} - \theta)^2\right] = V(\hat{\theta}) + B(\hat{\theta})^2$
- Why? $E\left[(\hat{\theta} - \theta)^2\right] = E\left[(\hat{\theta} - E(\hat{\theta}) + E(\hat{\theta}) - \theta)^2\right]$

$$= E\left[\left(\hat{\theta} - E(\hat{\theta})\right)^2 + 2\left(\hat{\theta} - E(\hat{\theta})\right)\left(E(\hat{\theta}) - \theta\right) + \left(E(\hat{\theta}) - \theta\right)^2\right]$$

$$= V(\hat{\theta}) + E\left[2\left(\hat{\theta} - E(\hat{\theta})\right)\left(E(\hat{\theta}) - \theta\right)\right] + B(\hat{\theta})^2$$

$$= V(\hat{\theta}) + B(\hat{\theta})^2$$

$E(\hat{\theta}) - \theta$: constant

Source:

Example

Let $Y_1, Y_2, \dots, Y_n$ be a random sample with $E(Y_i) = \mu$ and $V(Y_i) = \sigma^2$.

Show that $S'^2 = \frac{1}{n-1}\sum_{i=1}^{n}(Y_i - \bar{Y})^2$ is a unbiased estimator for σ^2

Solution

$$E\left[\frac{1}{n-1}\sum_{i=1}^{n}(Y_i - \bar{Y})^2\right] = \frac{1}{n-1}E\left[\sum_{i=1}^{n}(Y_i - \bar{Y})^2\right]$$

$$= \frac{1}{n-1}\left(\sum_{i=1}^{n}E\left(Y_i^2\right) - nE(\bar{Y}^2)\right)$$

$$= \frac{1}{n-1}\left(\sum_{i=1}^{n}(\sigma^2 + \mu^2) - n\left(\frac{\sigma^2}{n} + \mu^2\right)\right)$$

$$= \sigma^2$$

Source:

Trade off

5.4 Estimators, Bias and Variance

- The field of statistics gives us many tools that can be used to achieve the machine learning goal of solving a task not only on the training set but also to generalize
- Bias and variance are useful to formally characterize notions of generalization, underfitting and overfitting
- Trade of f between bias and variance to minimize MSE

The mean square error of a point estimator $\hat{\theta}$ is MSE($\hat{\theta}$)= $E[(\hat{\theta}-\theta)^2]$

- NOTE: MSE($\hat{\theta}$)= $E\left[(\hat{\theta}-\theta)^2\right] = V(\hat{\theta}) + B(\hat{\theta})^2$
- Why? $E\left[(\hat{\theta}-\theta)^2\right] = E\left[(\hat{\theta} - E(\hat{\theta}) + E(\hat{\theta}) - \theta)^2\right]$

$$= E\left[\left(\hat{\theta}-E(\hat{\theta})\right)^2 + 2\left(\hat{\theta}-E(\hat{\theta})\right)\left(E(\hat{\theta})-\theta\right) + \left(E(\hat{\theta})-\theta\right)^2\right]$$

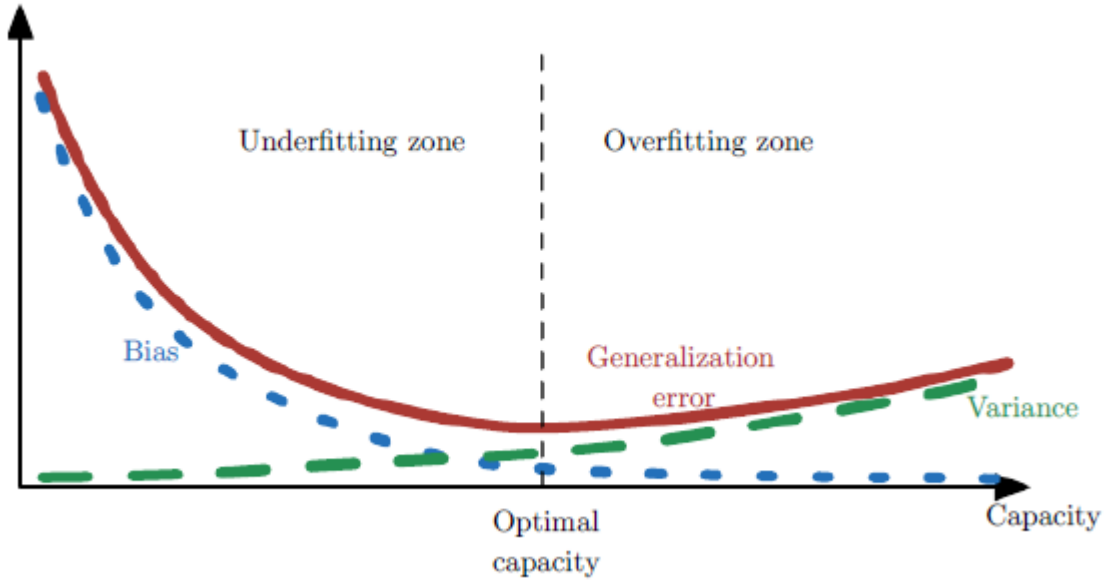
$$= V(\hat{\theta}) + E\left[2\left(\hat{\theta}-E(\hat{\theta})\right)\left(E(\hat{\theta})-\theta\right)\right] + B(\hat{\theta})^2$$

$$= V(\hat{\theta}) + B(\hat{\theta})^2$$

Source:

Bias and Variance Tradeoff

5.4 Estimators, Bias and Variance

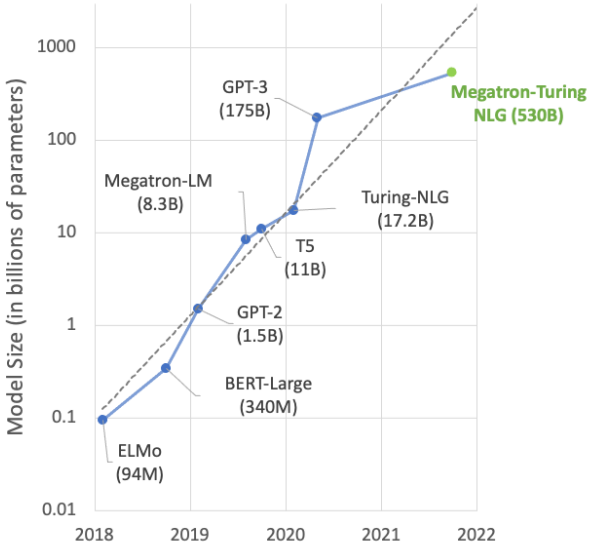


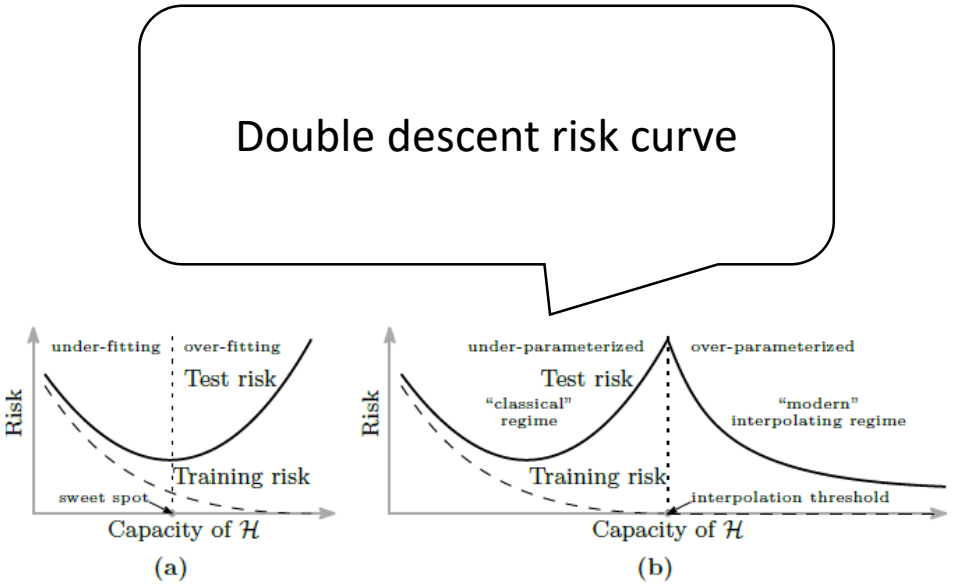
- As capacity increase: bias tends to decrease and variance tends to increase
- U-shaped curve for generalization error (bold red curve)
- => There is an optimal capacity

Source:

Bias and Variance Tradeoff ?

5.4 Estimators, Bias and Variance



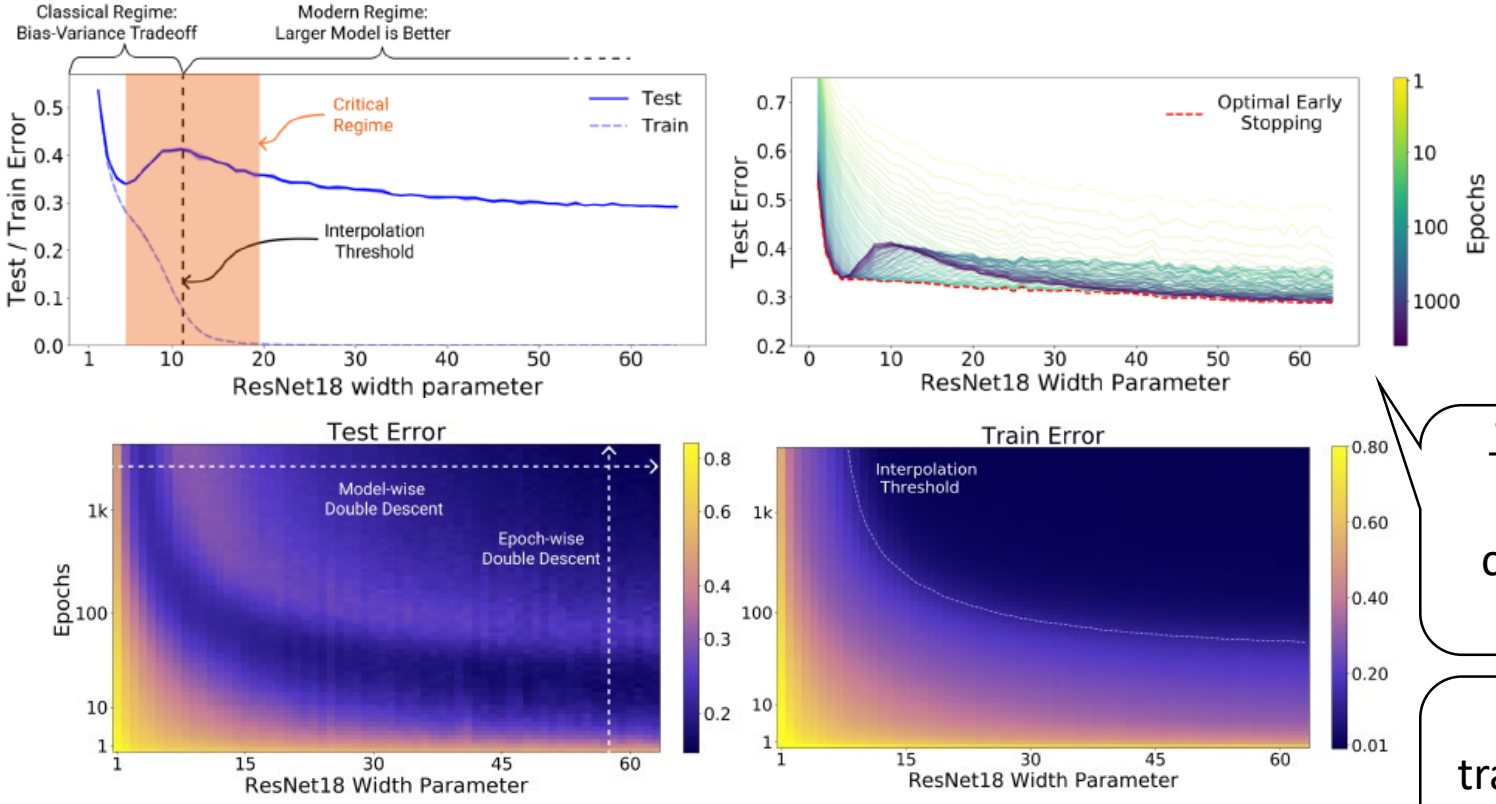


- Conventional wisdom
  - If $\mathcal{H}$ is too small, all predictors in $\mathcal{H}$ may under-fit the training data
  - If $\mathcal{H}$ is too large, the empirical risk minimizer may over-fit spurious pattern
- The classical thinking is concerned with finding the “sweet spot”
  - Early Stopping
- Increasing the function class capacity beyond this point leads to decreasing risk, typically going below the risk achieved at the sweet spot in the “classical" regime

Source: https://developer.nvidia.com/blog/using-deepspeed-and-megatron-to-train-megatron-turing-nlg-530b-the-worlds-largest-and-most-powerful-generative-language-model/ https://arxiv.org/pdf/1810.00736.pdf https://arxiv.org/abs/1812.11118

Bias and Variance Tradeoff ?

5.4 Estimators, Bias and Variance



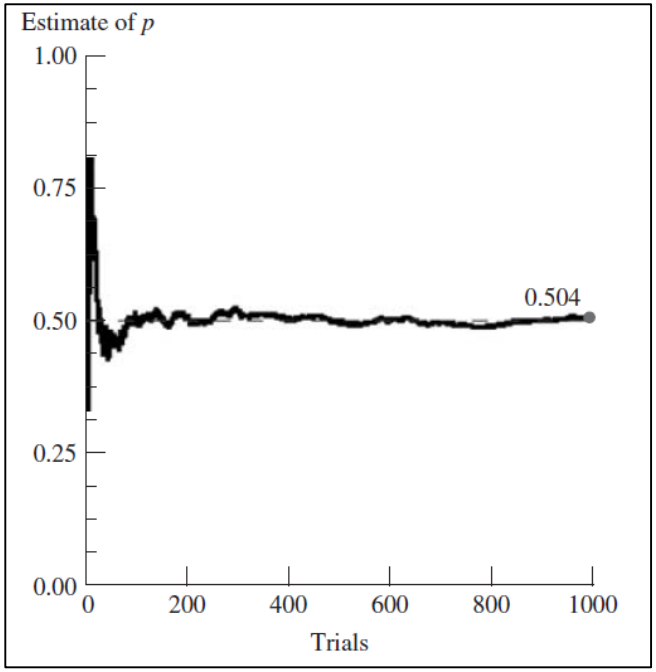
그럼 어떻게 해야 Model complexity를 측정하지?

Average 0 training error를 달성할 수 있는 최대 sample 수

- Effective Model Complexity (EMC)
  - $EMC_{D,\epsilon}(T) = \max\{n | E_{S\sim D^n}[Error_S(T(S))] \leq \epsilon\}$
    - $EMC_{D,\epsilon}(T) \ll n$: any perturbation T that increases its EMC will decrease the test error
    - $EMC_{D,\epsilon}(T) \gg n$: any perturbation T that increases its EMC will decrease the test error
    - $EMC_{D,\epsilon}(T) \approx n$: any perturbation T that increases might decrease or increase the test error

Source: https://developer.nvidia.com/blog/using-deepspeed-and-megatron-to-train-megatron-turing-nlg-530b-the-worlds-largest-and-most-powerful-generative-language-model/ https://arxiv.org/pdf/1810.00736.pdf https://arxiv.org/abs/1812.11118

- Coin toss
  - Toss n times (probability p resulting in heads)
  - Sample proportion (the estimator of p) is $\frac{Y}{n}$
  - We believe that as n gets larger, $\frac{Y}{n}$ should get closer to the true value of p



$P(\left|\frac{Y}{n} - p\right| \leq \epsilon)$ should be close to 1 for large n

$\Rightarrow \frac{Y}{n}$ converges in probability to p

Source:

Definition 9.2

The estimator $\hat{\theta}_n$ is said to be a consistent estimator of θ if, for any positive number ϵ, $\lim_{n\to\infty} P\left(\left|\hat{\theta}_n - \theta\right| \le \epsilon\right) = 1$ or equivalently, $\lim_{n\to\infty} P\left(\left|\hat{\theta}_n - \theta\right| > \epsilon\right) = 0$

Unbiasedness and Consistency

- Unbiasedness: $E\left(\hat{\theta}\right) = \theta$
- Consistency: $\lim_{n\to\infty} P\left(\left|\hat{\theta}_n - \theta\right| \le \epsilon\right) = 1$

Example

- $X_1, \ldots, X_n \sim N(\mu, \sigma^2)$
  - Unbiased but not consistent
    - X_1 ($E[X_1] = \mu$, but X_1 is not consistent)
  - Consistent but not unbiased
    - $\hat{\sigma}^2 = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \bar{Y})^2$

Source:

Maximum Likelihood Estimation

5.5 Maximum Likelihood Estimation

- The **likelihood** of the sample $L(y_1, \dots, y_n|\theta)$, is defined to be the joint probability (density) of $y_1, \dots, y_n$. $L(y_1, \dots, y_n|\theta) = f(y_1, \dots, y_n|\theta) = f(y_1|\theta) \times \cdots \times f(y_n|\theta)$
- Consider a set of m examples $X = \{x^{(1)}, \dots, x^{(m)}\}$ drawn independently from the true but unknown data generating distribution $p_{data}(x)$
- $p_{model}(x; \theta)$: parametric family of probability distributions over the sample space
  - It maps any configuration x to a real number estimating $p_{data}(x)$
- Maximum likelihood estimator for θ and KL divergence
  - $\theta_{ML} = argmax_\theta p_{model}(X; \theta) = argmax_\theta \prod_{i=1}^{m} p_{model}(x^{(i)}; \theta)$
  - $\theta_{ML} = argmax_\theta \sum_{i=1} \log p_{model}(x^{(i)}; \theta)$
  - $\theta_{ML} = argmax_\theta E_{x \sim \hat{p}_{data}(x)}[\log p_{model}(x; \theta)]$
  - $D_{KL}(\hat{p}_{data}(x) \,||\, p_{model}) = E_{x \sim \hat{p}_{data}(x)}[\log \hat{p}_{data}(x) - \log p_{model}(x; \theta)]$
  - Any loss consisting of a negative log-likelihood: cross-entropy (MSE: cross entropy between the empirical distribution and a Gaussian model)

$\hat{p}_{data}(x)$: empirical distribution

Source:

Maximum Likelihood Estimation

5.5 Maximum Likelihood Estimation

- The **likelihood** of the sample $L(y_1, \dots, y_n|\theta)$, is defined to be the joint probability (density) of $y_1, \dots, y_n$. $L(y_1, \dots, y_n|\theta) = f(y_1, \dots, y_n|\theta) = f(y_1|\theta) \times \cdots \times f(y_n|\theta)$
- Maximum likelihood estimator for θ and KL divergence
  - $\theta_{ML} = argmax_\theta E_{x\sim\hat{p}_{data}(x)}[\log p_{model}(x;\theta)]$
  - $D_{KL}(\hat{p}_{data}(x) \,||\, p_{model}) = E_{x\sim\hat{p}_{data}(x)}[\log \hat{p}_{data}(x) - \log p_{model}(x;\theta)]$
- MLE has the property of consistency under these conditions
  - The true distribution p_{data} must lie within the model family$p_{model}(\cdot;\theta)$
  - The true distribution p_{data} must correspond to exactly one value of θ
    - ❖Maximum likelihood can recover the correct p_{data}
- as the number of examples $m \to \infty$, MLE is the best estimator asymptotically

Source:

5.6 Bayesian Statistics

- So far we have discussed frequentist statistics
  - Single value of θ
- Another approach: consider all possible values of θ
  - Bayesian statistics
- We represent our knowledge of θ using the prior $p(\theta)$
  - $p\left(\theta \middle| x^{(1)}, \dots, x^{(m)}\right) = \frac{p\left(x^{(1)}, \dots, x^{(m)} \middle| \theta\right) p(\theta)}{p\left(x^{(1)}, \dots, x^{(m)}\right)}$
    - ❖Belief about θ
- Bayesian estimation offers two important differences
  - 1) make prediction using a full distribution over θ
    - ❖$p\left(x^{(m+1)} \middle| x^{(1)}, \dots, x^{(m)}\right) = \int p\left(x^{(m+1)} \middle| \theta\right) p\left(\theta \middle| x^{(1)}, \dots, x^{(m)}\right) d\theta$
  - 2) prior has an influence by shifting probability mass density towards regions of the parameter space that are preferred a priori
    - ❖The prior often expresses a preference for models that are simpler or more smooth
- However, typically it suffer from high computational cost

Source:

5.6 Bayesian Statistics

- $\hat{y}^{(train)} = X^{(train)}w$
  - $p\left(y^{(train)} \middle| X^{(train)}, w\right) = N\left(y^{(train)}; X^{(train)}w, I\right)$

  $\propto \exp\left(-\frac{1}{2}\left(y^{(train)} - X^{(train)}w\right)^T\left(y^{(train)} - X^{(train)}w\right)\right)$ (MSE formulation)
- We need to specify the prior distribution
  - $p(w) = N(w; \mu_0, \Lambda_0) \propto \exp\left(-\frac{1}{2}(w-\mu_0)^T\Lambda_0^{-1}(w-\mu_0)\right)$
    - ❖Gaussian prior
- $p(w|X,y) \propto p(y|X,w)p(w)$
- $\propto \exp\left(-\frac{1}{2}\left(y^{(train)} - X^{(train)}w\right)^T\left(y^{(train)} - X^{(train)}w\right)\right) \times \exp\left(-\frac{1}{2}(w-\mu_0)^T\Lambda_0^{-1}(w-\mu_0)\right)$
- $\propto \exp\left(-\frac{1}{2}(w-\mu_m)^T\Lambda_m^{-1}(w-\mu_m)\right)$, Gaussian posterior
  - $\Lambda_m = \left(X^TX + \Lambda_0^{-1}\right)^{-1}$, $\mu_m = \left(X^Ty + \Lambda_0^{-1}\mu_0\right)^{-1}$
- Note) Maximum A Posteriori (MAP) Estimation: point estimation
  - $\theta_{MAP} = argmax_\theta p(\theta|x) = argmax_\theta \log p(x|\theta) + \log p(\theta)$ (θ가 만약 Gaussian이면 어떻게 될까요?)

Source:

- The negative conditional log-likelihood of the training
  - $J(\theta) = E_{x,y\sim\hat{p}_{data}} L(x, y, \theta) = \frac{1}{m}\sum_{i=1}^{m} L(x^{(i)}, y^{(i)}, \theta)$
    - $L(x, y, \theta) = -\log p(y|x; \theta)$
- $\nabla_\theta J(\theta) = \frac{1}{m}\sum_{i=1}^{m} \nabla_\theta L(x^{(i)}, y^{(i)}, \theta)$
  - The computational cost of this operation is $O(m)$
    - As the training set size grows to billions of examples …
- Minibatch
  - $B = \{x^{(1)}, \ldots, x^{(m')}\}$ drawn uniformly from the training set
  - The estimate of the gradient
    - $g = \frac{1}{m'}\nabla_\theta \sum_{i=1}^{m'} L(x^{(i)}, y^{(i)}, \theta)$
    - $\theta \leftarrow \theta - \epsilon g$ where ϵ is the learning rate

SGD generalizes so well when used on large over-parameterized models
⇒ Implicit regularization
⇒ Implicit bias of SGD seems to yield a well-generalizing ERM
만약 K개의 동일한 data가 있다고 하면,
SGD는 GD 대비 K배 빠르게 학습 가능

Source: https://openreview.net/pdf?id=jqjsLUrB8F

The curse of Dimensionality

5.11 Challenges Motivating Deep Learning

- Many machine learning problems become exceedingly difficult when the number of dimensions in the data is high
  - The curse of dimensionality
  - The number of possible distinct configurations of a set of variables increases exponentially as the number of variables increases.

- In spaces of high dimensionality, most of the volume of a sphere is concentrated in a thin shell near the surface
  - $V_D(r) = K_D r^D$, D: dimensions, r: radius of sphere, K_D: constant
  - $\frac{V_D(1) - V_D(1-\epsilon)}{V_D(1)} = 1 - (1-\epsilon)^D$

Source: