CS60010: Deep Learning

Sudeshna Sarkar

Spring 2018

11 Jan 2018

ML BASICS

Bayes Error

- The ideal model is an oracle that simply knows the true probability distribution that generates the data.
- Even an ideal model will incur error
  - The mapping may be inherently stochastic, or y may be a deterministic function that involves other variables besides those included in x.
- The error incurred by an oracle making predictions from the true distribution $p(x, y)$ is called the *Bayes error*.

Generalization error

- Expected generalization error does not increase as the number of training examples increases.

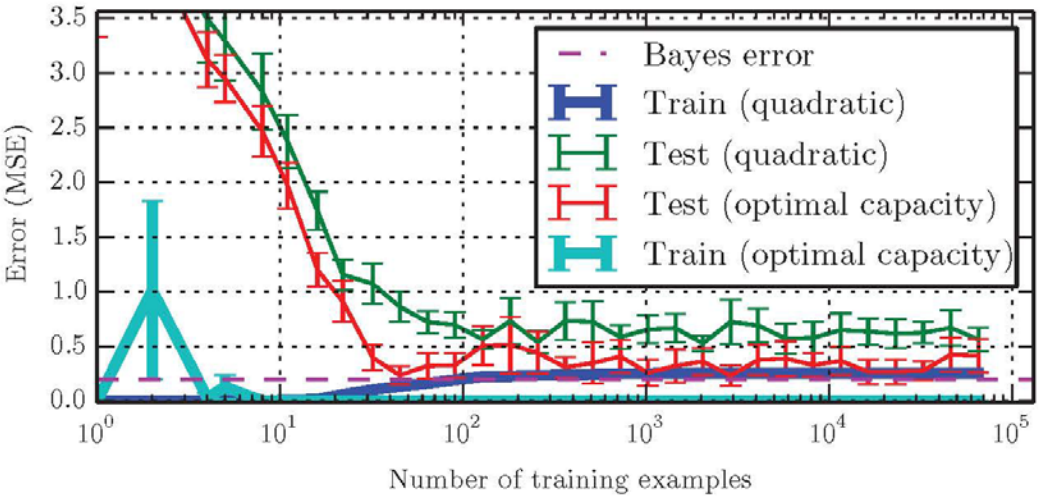
- Non-parametric models
  - more data yields better generalization until the best possible error is achieved.

- Any fixed parametric model with less than optimal capacity will asymptote to an error value that exceeds the Bayes error.

The No Free Lunch Theorem

- Inductive reasoning, or inferring general rules from a limited set of examples, is not logically valid.
- To logically infer a rule describing every member of a set, one must have information about every member of that set
- ML avoids this problem by offering only probabilistic rules, rather than the entirely certain rules used in purely logical reasoning.
- The no free lunch theorem (Wolpert, 1996) states that, averaged over all possible data generating distributions, every classification algorithm has the same error rate when classifying previously unobserved points.
  - understand what kinds of distributions are relevant to the "real world"
  - what kinds of ML algorithms perform well on data drawn from distributions we care about

Training Set Size



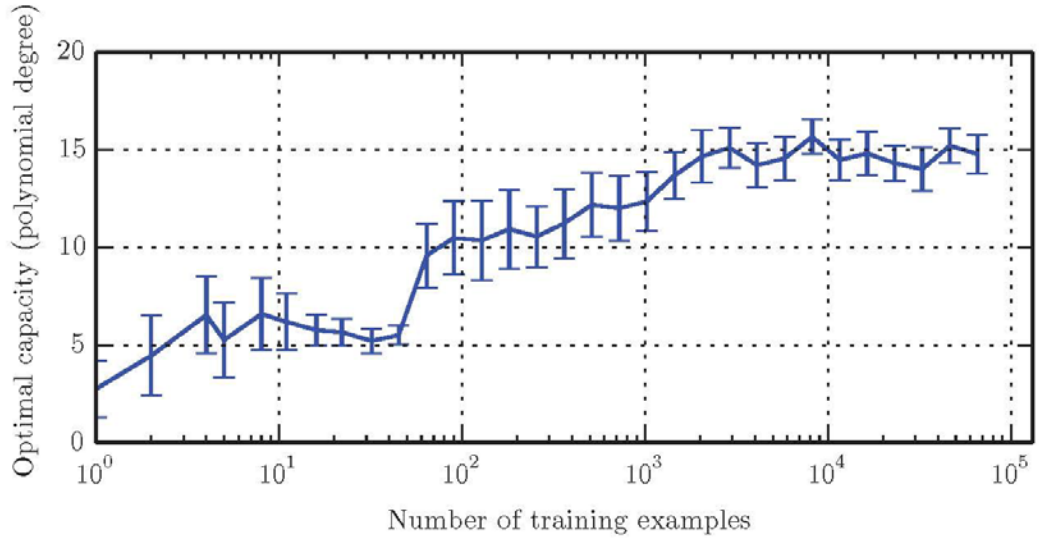


Figure 5.4

The effect of the training dataset size on the train and test error, as well as on the optimal model capacity.

A synthetic regression problem based on adding noise to a degree-5 polynomial, For each size, generated 40 different training sets to plot error bars showing 95 percent confidence intervals.

(Top) The MSE on the training and test set for: a quadratic model, and a model with degree chosen to minimize the test error.

For the quadratic model, the training error increases as the size of the training set increases because larger datasets are harder to fit. Simultaneously, the test error decreases, because fewer incorrect hypotheses are consistent with the training data. The quadratic model does not have enough capacity to solve the task.

The test error at optimal capacity asymptotes to the Bayes error. The training error can fall below the Bayes error, due to the ability of the training algorithm to memorize specific instances of the training set. As the training size increases to infinity, the training error of any fixed-capacity model (here, the quadratic model) must rise to at least the Bayes error.

(Bottom) As the training set size increases, the optimal capacity (shown here as the degree of the optimal polynomial regressor) increases. The optimal capacity plateaus after reaching sufficient complexity to solve the task

Regularization

- No free lunch theorem implies we must design ML algorithms to perform well on a specific task. We do so by building a set of preferences into the learning algorithm.
- The behaviour of our algorithm is strongly affected not just by how large we make the set of functions allowed in its hypothesis space, but by the specific identity of those functions
- We can also give a learning algorithm a preference for one solution in its hypothesis space to another
- For example, we can modify the training criterion for linear regression to include weight decay

$$J(w) = MSE_{train} + \lambda w^T w$$

- λ controls the strength of preference for smaller weights

Weight Decay

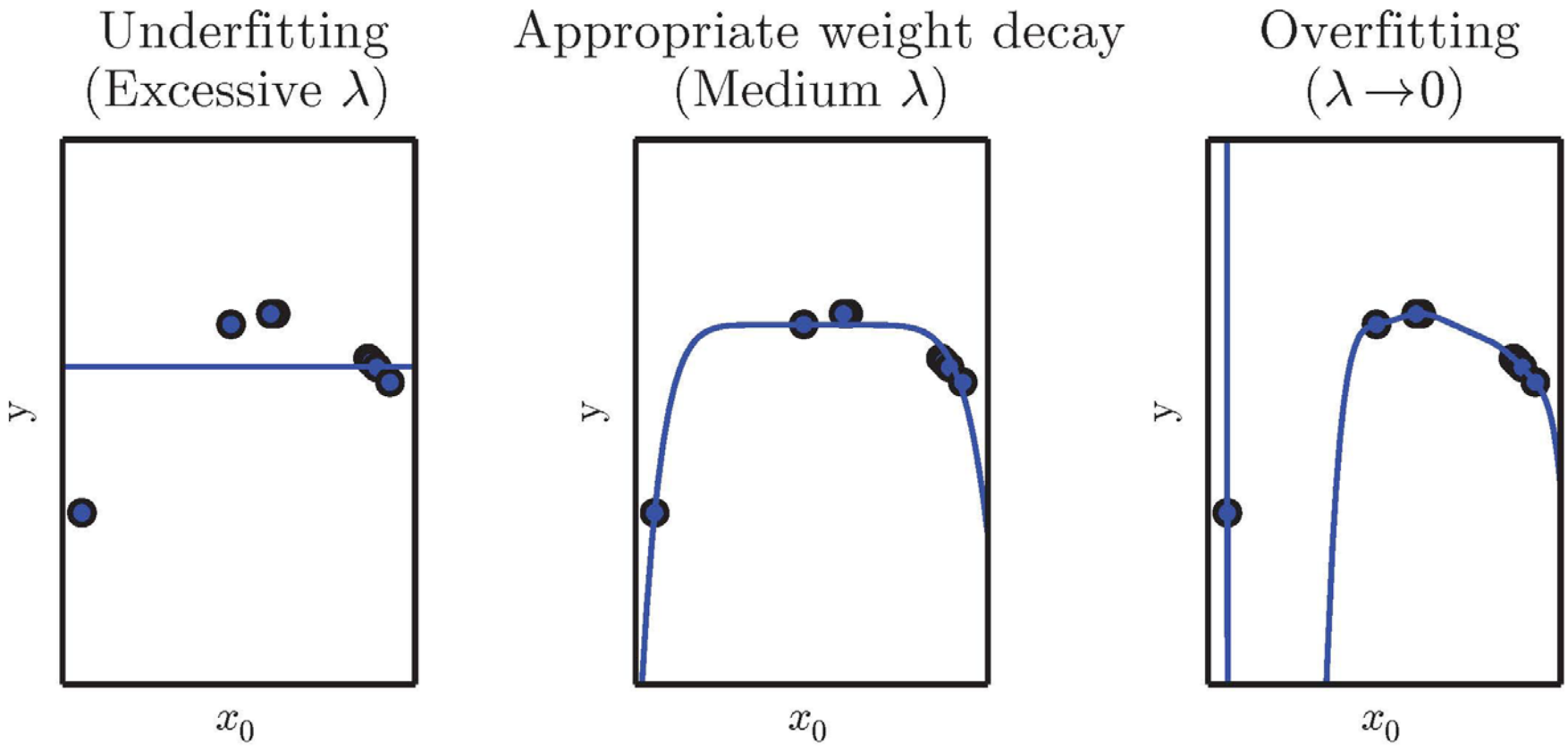


Figure 5.5

- We fit a high-degree polynomial regression model to our example training set.
- The true function is quadratic, we use models with degree 9.
- We vary the amount of weight decay to prevent these high-degree models from overfitting.
- With very large λ, we can force the model to learn a function with no slope at all. This underfits because it can only represent a constant function.
- (Cente)eWith a medium value of λ, the learning algorithm recovers a curve with the right general shape. weight decay has encouraged it to use a simpler function described by smaller coefficients.
- (Right)With weight decay approaching zero thedegree-9 polynomial overfits significantly

Regularization

- More generally, we can regularize a model that learns a function $f(x;\theta)$ by adding a penalty called a regularizer to the cost function.
- In the case of weight decay, the regularizer is $\Omega(w) = w^T w$

- Many other regularizers are possible
- Regularization is any modification we make to a learning algorithm that is intended to reduce its generalization error but not its training error.

Hyperparameters and Validation Sets

- Most ML algs have several settings that we can use to control the behaviour - hyperparameters.
- Poly regression problem –
  - degree of the poly is the hyperparameter
  - Value of lambda (weight decay)
- Sometimes a setting is chosen to be a hyper-parameter that the learning algorithm does not learn because it is difficult to optimize.
- Hyper-parameters that control model capacity cannot be learned on training set
- To solve this problem, we need a validation set

Cross-Validation

Estimators, Bias and Variance

- Function Estimation (or function approximation): predict a variable y given an input vector x. We may assume

$$y = f(x) + \epsilon$$

- Bias of an estimator:

$$bias(\widehat{\theta_m}) = \mathrm{E}[\hat{\theta}_m] - \theta$$

- Variance and Standard Error
  - how much we expect it to vary as a function of the data sample.

$$Var(\hat{\theta})$$

  - the square root of thevariance is called the standard error, denoted SE(ˆθ).
  - a measure of howwe would expect the estimate we compute from data to vary as we independentlyresample the dataset from the underlying data generating process.

Trading off Bias and Variance to Minimize Mean Squared Error

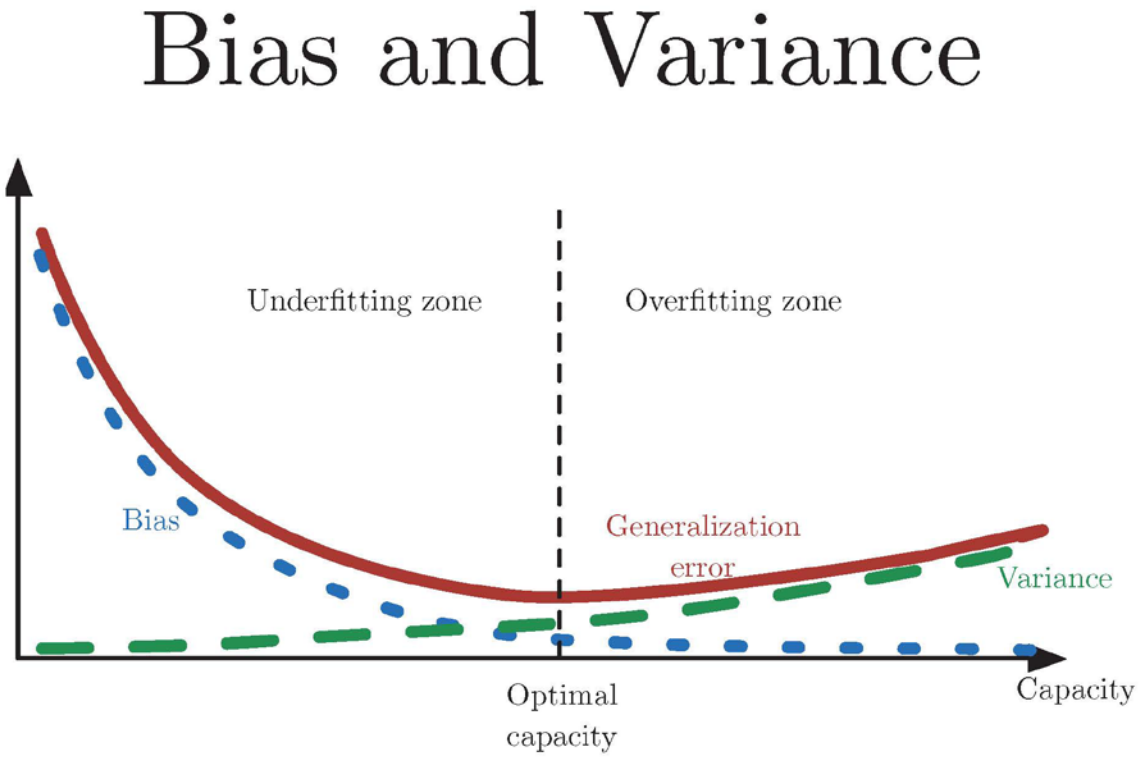


Figure 5.6

(Goodfellow 2016)

Maximum Likelihood Estimation

- principle from which we can derive specific functions that are good estimators for different models.
- $X = \{x^{(1)}, x^{(2)}, \dots, x^{(m)}\}$ drawn independently from $p_{data}(x)$
- Let $p_{model}(x; \theta)$ be a parametric family of probability distributions over the same space indexed by θ - maps any configuration x to a real number estimating the true probability $p_{data}(x)$.
- MLE of x

$$\theta_{ML} = \underset{\theta}{argmax}\, p_{model}(X; \theta)$$

$$= \underset{\theta}{argmax} \prod_{i=1}^{m} p_{model(x^{(i)};\theta)}$$

$$\theta_{ML} = \underset{\theta}{argmax} \sum_{i=1}^{m} \log p_{model(x^{(i)};\theta)}$$

$$\theta_{ML} = \underset{\theta}{argmax}\, \mathbb{E}_{x \sim \hat{p}_{data}} \log p_{model}(x; \theta)$$

MLE

- One way to interpret maximum likelihood estimation is to view it as minimizing the dissimilarity between the empirical distribution $\hat{p}_{data}$ defined by the training set and the model distribution.
- Measured by KL-divergence

$$D_{KL}(\hat{p}_{data}||p_{model}) = \mathbb{E}_{x\sim\hat{p}_{data}}[\log \hat{p}_{data}(x) - \log p_{model}(x)]$$

- The term on the left is a function only of the data-generating process, not the model. So we only need to minimize

$$-\mathbb{E}_{x\sim\hat{p}_{data}}[\log p_{model}(x)]$$

- Minimizing this KL divergence corresponds exactly to minimizing the cross-entropy between the distributions.

Conditional Log-Likelihood

- ML estimator can be generalized to estimate a conditional probability $P(y|x;\theta)$

$$\theta_{ML} = \underset{\theta}{argmax}\, P(Y|X;\theta)$$

For i.i.d. examples,

$$\theta_{ML} = \underset{\theta}{argmax} \sum_{i=1}^{m} \log P(y^{(i)}|x^{(i)};\theta)$$