Introduction to Statistical Machine Learning

Xiang Zhou

School of Data Science
Department of Mathematics
City University of Hong Kong

Mathematical Description of Data

Notations

- input - output relation
  - x: inputs, feature vectors, predictors, independent variables.
  - y: output, response, dependent variable.
- $\mathcal{X}$ and $\mathcal{Y}$ denote the spaces of the generic x and y variables, respectively.
  - Generally $\mathcal{X} = \mathbb{R}^p$; qualitative features are coded using, for example, dummy variables (such as 0,1, -1, etc).
  - Typically $\mathcal{Y} \in \mathbb{R}^1$ is a scalar, or takes a finite number of values as a subset of $\mathbb{N}$; it can be a vector in some scenarios.
- The random variable (X, Y) has the joint distribution $p(x, y)$ on the sample space $\mathcal{X} \times \mathcal{Y}$.

Ground truth

- It is usually assumed that the ground truth for the relation between from input to the output is a deterministic input-output mapping from $x \in \mathcal{X}$ to $y_{\text{true}} \in \mathcal{Y}$:

$$y_{\text{true}} = f^{\star}(x)$$

  where the ground truth f^* is an unknown function and has to be approximated by learning from the training dataset.

Data as iid r.v.

- In **supervised learning**, the data (observations) are given as the collection of the pairs [1]

$$\mathtt{D} = \left\{(x^{(i)}, y^{(i)}) : 1 \le i \le N\right\} \subset (\mathcal{X} \times \mathcal{Y})^N$$

  which is assumed iid samples of the r.v. (X, Y) with an unknown joint distribution $p(x, y)$ on the product space $\mathcal{X} \times \mathcal{Y}$.
  - Regression: $\mathcal{Y}$ is continuous/quantitiative, e.g., $\mathbb{R}^d$ or its subset.
  - Classification: $\mathcal{Y}$ is discrete and finite (categorical variable), encoded by a finite number $\{1, \dots, K\}$. In this case, "y" is usually called "label".
- In **unsupervised learning**, the observations only have $\left\{x^{(i)}\right\}$, the information $y^{(i)}$ is zero or there is no definition of y variable. The purpose is to identify the pattern of $\left\{x^{(i)}\right\}$ itself, such as model reduction.

[1] sometimes it is denoted as $\mathtt{D} = \{(x_i, y_i) : 1 \le i \le n\}$ if the subindex has no ambiguity.

Output as perturbed truth

- The observed $x^{(i)} \in \mathcal{X}$ are samples from the marginal distribution p_X, i.e., $x^{(i)} \sim X$; in some cases, they are deterministic and assigned by a procedure of experiment design.
- The corresponding observed $y^{(i)}$ are assumed to be the *perturbed* truth $f^\star(x^{(i)})$ with additional measurement error $\varepsilon^{(i)}$ which are assumed to be iid and independent from X.

  $$y^{(i)} = f^\star(x^{(i)}) + \varepsilon^{(i)}.$$

  $\{\varepsilon^{(i)}\}$ are assumed iid and distributed as a generic r.v. ε.
- The effect of the noise ε can never be eliminated by any statistical learning algorithms (irreducible error).

- So, the joint distribution $p(x,y)$ of (X,Y) is completely determined by the triplet:

$$(p_X, f^\star, p_\varepsilon)$$

    - p_X: the distribution of the input
    - $f^\star$: the input-output function,
    - p_ε: the distribution of error.
- The joint distribution $p(x,y)$ manifests by the available dataset `D`. Given X,Y random variables $\sim p$, how to identify $f^\star$?

Statistics and machine learning

Different views and terminologies:

Machine Learning	**Statistics**
Supervised learning	Classification/regression
Unsupervised learning	Clustering
Semisupervised learning	Classification/regression with missing responses
Features/outcomes	Covariates/responses
Training set/testing set	Sample/population
Learner	Statistical model
Generalization error	Misclassification error/prediction error

Suggestion

Learn machine learning like a statistician or an applied mathematician, not a software engineer

- Start from stating a problem, not show an algorithm first: Many times, the students think the methods/algorithms/procedures as the problem itself.
- Pay close attentions to the "modelling" process: how to turn the data problem into the statistical model. In particularly the underlying principle which applies very general.
- Try to rigorously (and in most general context) understand (like a math theorem) the heuristic arguments used in practice, even for some toy examples.
- Draw strict boundaries between general principles and specific methods. In between, computational method survives and thrives.
- Diagnose and rationalize your results of numerical experiments. Test different dataset/parameters/methods.

Learning Theory: An Approximation Theory Viewpoint

Reference:

Learning Theory: An Approximation Theory Viewpoint; by F. Cucker and D.X. Zhou, Cambridge University Press 2007

Given r.v.s X and Y, find a function $f : \mathcal{X} \to \mathcal{Y}$ so that $f(X)$ can explain Y best in certain sense.

Conditional Expectation as Optimal Prediction

The best L^2 approximation of a function f of the r.v. X to a r.v. Y is achieved by the conditional probability. The **(generalized) squared error**[1]

$$\mathcal{E}(f) := \mathbb{E}\,|Y - f(X)|^2 \tag{1}$$

has a minimum at

$$\boxed{f^*(x) = \mathbb{E}(Y|X = x)}.$$

i.e.,

$$\mathbb{E}\,|Y - f^*(X)|^2 = \min_{f\text{: a Borel function}} \mathbb{E}(|Y - f(X)|^2)$$

f in fact should satisfy the condition that the RHS L_2 norm exists.

[1]also named as generalization risk, mean square error, L_2 error, etc.

Proof.

- We show first that $\mathbb{E}[(Y - f^*(X))h(X)] = 0$ [a] is true for any function h. Using the double expectation theorem [b], we have

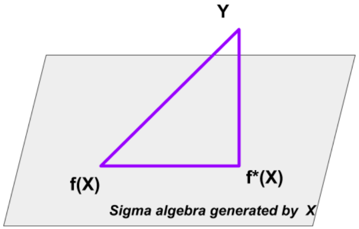
$$\mathbb{E}[(Y - f^*(X))h(X)] = \mathbb{E}\left[\mathbb{E}[Y - f^*(X))|X]h(X)\right]$$
$$= \mathbb{E}\left[\mathbb{E}(Yh(X)|X) - f^*(X)h(X)\right] = 0.$$

- Note that
$(y - f(x))^2 = (y - f^*(x))^2 + (f(x) - f^*(x))^2 - 2(y - f^*(x))h(x)$
where $h(x) = f(x) - f^*(x)$, then for any f

$$\boxed{\mathbb{E}(|f(X) - Y|^2) = \mathbb{E}(|f^*(X) - Y|^2) + \mathbb{E}\left[|f(X) - f^*(X)|^2\right]} \quad (2)$$

□

[a] sometimes it is denoted $Y - f^*(X) \perp h(X)$, the perpendicular property in L_2 space.
[b] $\mathbb{E}[\mathbb{E}(Y|X)] = \mathbb{E}Y$



---

reference for elementary math: Understanding Conditional Expectation via Vector Projection

The following exercise is to use the viewpoint of variational calculus: minimizing functions in the function space

Exercise

Use the method of Calculus of Variation to solve [a]

$$\inf_f \iint (f(x) - y)^2 p_{X,Y}(x, y)dxdy$$

where $p_{X,Y}$ *is the joint pdf of the r.v.s* (X, Y). *The optimal* f^* *satisfies*

$$\int (f^*(x) - y)p_{X,Y}(x, y)dy = 0, \qquad \forall x$$

i.e., $\mathbb{E}\, Y = \mathbb{E}\, f^*(X)$

[a] Rigorously, f is in the p-weighted L_2 space

What if generalized the L_2 norm to L_p norm ?

Exercise (Conditional distribution of multivariate Gaussian r.v.)

Suppose that $X = (X_1, X_2)$ is a two dimensional Gaussian random variable with mean $\mu = (\mu_1, \mu_2)$ and the covariance matrix $\Sigma = \begin{pmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 \\ \rho\sigma_1\sigma_2 & \sigma_2^2 \end{pmatrix}$. What is the conditional pdf $p(x_1|x_2)$ of X_1 given $X_2 = x_2$? What is the expectation of X_1 given $X_2 = x_2$? What is the general result in n dimension ?

$$X_1|_{X_2=x_2} \sim \mathcal{N}(\mu_1 + \rho\frac{\sigma_1}{\sigma_2}(x_2 - \mu_2), \sigma_1^2(1-\rho^2))$$

- The objective function $\mathcal{E}(f)$ is also called loss function, risk, etc.
- $\mathbb{E}\, f^*(X) = \mathbb{E}\, Y$: $f^*(X)$ is an unbiased estimate of Y;
- The variance of the difference between Y and the optimal prediction $f^*(X)$ is

$$\sigma_*^2(x) := \mathbb{E}\left[(Y - f^*(X))^2 | X = x\right]$$

- Take average of $\sigma^2(x)$ over x, then the averaged uncertainty is

$$\sigma_*^2 := \mathbb{E}_X\, \sigma^2(X) = \mathbb{E}\left[|Y - f^*(X)|^2\right] = \mathcal{E}(f^*)$$

  is the variance of the measurement error $Y - f^*(X)$: irreducible error

Notations

We have shown that (2) for any two r.v. X, Y:

$$\mathcal{E}(f) = \underbrace{\mathbb{E}(|f(X) - Y|^2)}_{\textbf{Mean Square Error}} = \underbrace{\mathbb{E}_{X,Y}(|f^*(X) - Y|^2)}_{\mathcal{E}(f^*),\textbf{Bayes error}} + \underbrace{\mathbb{E}_X\left[|f(X) - f^*(X)|^2\right]}_{\textbf{model error}} \tag{3}$$

where $f^*(x) = \mathbb{E}(Y|X = x)$ is called **Bayes rule**.

- Bayes error: irreducible error;
- Model error: the distance from f to f^*.

Application to classification with 0-1 loss

- Assume $Y \in \{1, \dots, K\}$ and $X \in \mathbb{R}^p$.
- $f : \mathcal{X} \to \mathcal{Y}$ is a piece-wise constant function.
- A Loss function $\ell(Y, f(Y))$ for penalizing errors in misclassification.
- Most Common choice is the 0-1 loss

$$\ell(Y, f(X)) = I(Y \neq f(X)) = \begin{cases} 1 & \text{if } Y \neq f(X) \\ 0 & \text{if } Y = f(X) \end{cases}.$$

- The expected prediction error, or the generalization error, is

$$\mathcal{E}(f) = \mathbb{E}\, I(Y \neq f(X)) = \mathbb{P}(Y \neq f(X)) = 1 - \mathbb{P}(Y = f(X)).$$

- The Bayes rule minimizing $\mathcal{E}(f)$ is the one maximizing $\mathbb{P}(Y = f(X)) = \int_{\mathcal{X}} \mathbb{P}(Y = f(x)|X = x) p_X(x) dx$, which is

$$f^*(x) = \operatorname*{argmax}_k \mathbb{P}(Y = k|X = x).$$

since $\mathbb{P}(Y = k|X = x) \leq \mathbb{P}(Y = f^*(x)|X = x), \forall k$

How do we estimate f^* ?

- direction method by definition of conditional expectation;
- optimization method by minimization of generalization error;

Nearest neighbors method to approximate condition expectation

A natural way to approximate the function $\mathbb{E}(Y|X=x)$ is to replace the expectation by the average:

$$f^*(x) = \mathbb{E}(Y|X=x) \approx \text{AVE}\left\{y_i : \text{ where } \; x_i = x\right\}, \quad \forall x \in \mathcal{X}$$

or

$$f^*(x) = \int_{\mathcal{Y}} y\, p_{Y|X}(y;x)dy$$

where $p_{Y|X}(y;x) = \frac{p_{X,Y}(x,y)}{p_X(x)}$ by estimating the density $p_{X,Y}(x,y)$ first?
Question: Would these work well in practice ? in high dimension ?
(1) this formula is defined point-wisely; (2) one does not have an accurate estimation of the expectation or the joint distribution $p(x,y)$, particularly in high dimension.

k-NN

- Use windows with size Δ

$$\mathbb{E}(Y|X=x) \approx \text{AVE}\left\{y_i : \text{ where } \ |x_i - x| \leq \Delta\right\}$$

- Use k -nearest neighbors: have a look at its neighbors, and take a vote:

$$\mathbb{E}(Y|X=x) \approx \text{AVE}\left\{y_i : x_i \in \mathcal{N}_k(x)\right\}$$

where $\mathcal{N}_k(x)$ is a neighborhood of x that contains exactly k neighbors (k-nearest neighbors).

Curse of dimensionality

k-NN can fail in high dimensions, because it becomes difficult to gather k observations close to a target point x_0.

- Neighborhoods tend to be spatially large, and estimates are biased.
- Reducing the spatial size of the neighborhood means reducing k, and the variance of the estimate increases.

In general (see Figure 2.6),

- Most points are at the boundary, and points tend to be equidistant (Hall et al., JRSS-B, 2005).
- Sampling density is proportional to $n^{1/p}$: the number of sample size increases exponentially in dimension p: If 100 points are sufficient to estimate a function in $\mathbb{R}^1$, 100^{10} are needed to achieve similar accuracy in $\mathbb{R}^{10}$.

This is similar to the curse of dimensionality in using the mesh grid based method to solve high dimensional PDE.

Minimizing the generalization error: learning as minimization

- Loss function $\ell(y,\hat{y}) : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}^+ \cup \{0\}$
    - L_2, L_1 norm
    - 0-1 loss: $\ell(y,\hat{y}) = I(y \neq \hat{y})$.
    - ...
- Population loss(Generalization error): $\mathcal{E}(f) = \mathbb{E}\,\ell(Y, f(X))$

The direct approach of minimizing the risk

$$\min_f \mathcal{E}(f) = \mathbb{E}\,\ell(Y, f(X))$$

where f can be in a very general class of functions, such as continuous function on $\mathcal{X}$, or even piecewise continuous function.
It is important to specify a hypothesis space $\mathcal{H}$ to restrict the search space.

1. the function space of f to search: hypothesis space(model class) $\mathcal{H}$.
2. approximate the expectation $\mathbb{E}$ in $\mathcal{E}$ with the use of data $\mathtt{D} = \{x_i, y_i\}$.

Notations

1. The minimizer of
$$\min_{f\in\mathcal{H}} \mathcal{E}(f)$$
is denoted by $f_{\mathcal{H}}$.
2. Population risk is approximated by empirical risk associated with $\mathtt{D} = \{(x_i, y_i)\} : i = 1, \ldots, N$:
$$\mathcal{E}(f) = \mathbb{E}\left[|Y - f(X)|^2\right] \approx \mathcal{E}_N(f) = \frac{1}{N}\sum_i \ell(y_i, f(x_i))$$
The learning algorithm produces the learnt function $\hat{f}_{\mathtt{D}}$ (regression function) as the solution to
$$\mathcal{E}_N(\hat{f}_{\mathtt{D}}) = \min_{f\in\mathcal{H}} \mathcal{E}_N(f) = \min_{f\in\mathcal{H}} \frac{1}{N} \sum_{(x_i,y_i)\in\mathtt{D}} \ell(y_i, f(x_i))$$
   - The **training error** is $\mathcal{E}_N(\hat{f}_{\mathtt{D}})$.
   - The **test error** is $\mathcal{E}(\hat{f}_{\mathtt{D}})$.

- $\hat{f}_{\mathtt{D}}$ depends on *three* elements: the hypothesis space $\mathcal{H}$, the (training) data `D` (random !) and the loss function ℓ. So, $\hat{f}_{\mathtt{D}}$ is a random function to approximate the ground truth f.
- For the fixed $\mathcal{H}$ and ℓ, $\hat{f}_{\mathtt{D}}$ is essentially a mapping from `D` to the function space $\mathcal{H}$. As $N \to \infty$, $\hat{f}_{\mathtt{D}} \to f_{\mathcal{H}}$ by the law of large number.

Hypothesis space: representation of the function f

- parametric approach: e.g. $f(x) = \theta \cdot x$ with a set of finite and (usually fixed) number of parameters θ
- non-parametric approach: functional analysis viewpoint, f is only in some function space. (such as in the traditional computational math for representing the solution to some PDE)
- there is no rigorous boundary between parametric/non-parametric: eventually the computer represents a function f only by a finite number of freedoms.
- need to be restricted to certain model class in practice. Examples: linear model, polynomial, spline, kernel machine, neural network, etc.

some examples of $\mathcal{H}$

- Linear function (chapter 3 , ESL)
- Basis functions and dictionary methods (chapter 5, ESL)

$$f_\theta(x) = \sum_{m=1}^{M} \theta_m h_m(x),$$

  where h_m's are pre-specified basis functions.
- Kernel methods and local regression (chapter 6, ESL)

$$RSS(f, x_0) = \sum_{i=1}^{n} K(x_0, x_i)(y_i - f(x_i))^2,$$

  where K is a kernel measuring the closeness between points.
- Roughness penalty (chapter 5, ESL)

$$\text{Penalized RSS}(f; \lambda) = \text{RSS}(f) + \lambda J(f),$$

  where $J(f)$ is a regularization term on the model complexity.

The generalization error is then decomposed into

$$\mathcal{E}(f^*) = \underbrace{\mathcal{E}(f^*) - \mathcal{E}(f_{\mathcal{H}})}_{\textbf{approximation error}} + \underbrace{\mathcal{E}(f_{\mathcal{H}}) - \mathcal{E}_N(\hat{f}_{\mathtt{D}})}_{\textbf{sampling error}} + \underbrace{\mathcal{E}_N(\hat{f}_{\mathtt{D}})}_{\textbf{training error}}$$

- Approximation error: characterized how large (complex) the hypothesis space $\mathcal{H}$ is. analogy to the interpolation inequality in finite element method. A larger space of $\mathcal{H}$ indicates
  - A larger effective dimension of this space;
  - A smaller approximation error; lower bias
  - More difficult for numerical optimization;
- Sampling error: vanishes if the number of data points N goes to infinity. This error may be analyzed by probability inequality (e.g. Chebychev); analysis method used for Monte Carlo simulation. A larger sample size for training data indicates:
  - Smaller sampling error; lower variance;
  - more challenges for data collection and Big Data techniques;
- Training error:
  - design of optimization method, numerical convergence rate,
  - the time and memory cost ...
  - the output is a further approximation to $\hat{f}_{\mathtt{D}}$: the optimization algorithm stops with a time cost T: $\hat{f}_{\mathtt{D}}^{(T)}$

Very Important Remarks

- Although different fields focus on each of three errors separately, to address the problems for machine learning needs trans-disciplinary efforts:
    - approximation theory
    - statistical learning
    - computation

  A joint strategy is very important to have a complete picture of the machine learning problem.
- These three topics are not completely separate. For example, the necessary number of sample size N may depend on the complexity of hypothesis space, which further depends on the dimension of $\mathcal{X}, \mathcal{Y}$ and the specific representation form of functions in $\mathcal{H}$.

Generalization Gap

The optimal function in theory is f^*; the optimal function in practical computation is $\hat{f}_{\mathtt{D}}$. The difference in risks of these two:

$$\mathcal{E}(\hat{f}_{\mathtt{D}}) - \mathcal{E}(f^*)$$

is called **generalization gap**: it measures to which extent the performance of the algorithm learnt from a given training data set $\mathtt{D}$ is valid for the whole distribution $p(x, y)$.

First inequality for Generalization Gap

Exercise

Prove the upper bound of the generalization gap:

$$\boxed{\mathcal{E}(\hat{f}_{\mathtt{D}}) - \mathcal{E}(f^*) \leq 2 \sup_{f \in \mathcal{H}} |\mathcal{E}_N(f) - \mathcal{E}(f)|}$$

$$\approx 2 \sup_{f \in \mathcal{H}} \frac{1}{\sqrt{N}} \mathbb{V}(\ell(Y, f(X)))$$

The first RHS depends on the hypothesis space $\mathcal{H}$ and the dataset $\mathtt{D}$ of size N in $\mathcal{E}_N$; and the second depends on $\mathcal{H}$ and the joint distribution $p(x, y)$ of (X, Y).
The supremum in the above inequality can describe the **complexity** of hypothesis space $\mathcal{H}$, which increases when $\mathcal{H}$ is "larger". WARNING: this upper bound is not sharp. A good lower bound is extremely difficult to obtain.

Statistical Learning: Viewpoint of Bias-Variance Trade-off

Bias-Variance Decomposition / Trade-off

Let f be a given deterministic function $\mathcal{X} \to \mathcal{Y}$.
Assume that the response r.v. Y is defined by

$$Y = f(X) + \varepsilon$$

where the r.v. ε is independent of X and has mean 0 and variance σ_ε^2.
Then $f^*(x) = \mathbb{E}[Y|X = x] = f(x)$ is the optimal approximation in the square error sense.

test error decomposition

Based on a training dataset $\mathtt{D} = \{x_i, y_i\}$ where $y_i = f(x_i) + \varepsilon_i$ and $\varepsilon_i \sim \varepsilon$, one learns a regression function, denoted by $\hat{f}_{\mathtt{D}}(\cdot)$, then the **test error** [1] at a new input x_0 (with a new independent measurement error $\varepsilon_0 \sim \varepsilon$) is

$$\mathbb{E}[(Y - \hat{f}_{\mathtt{D}}(X))^2 | X = x_0] = \mathbb{E}_{\varepsilon_0, \mathtt{D}}[(f(x_0) + \varepsilon_0 - \hat{f}_{\mathtt{D}}(x_0))^2]$$

$$= \sigma_\varepsilon^2 + \mathbb{E}_{\mathtt{D}}[\Big(f(x_0) - \hat{f}_{\mathtt{D}}(x_0)\Big)^2] \quad \because \mathbb{E}(\varepsilon_0) = 0, \text{ and } \mathtt{D} \perp \varepsilon_0$$

$$= \sigma_\varepsilon^2 + \mathbb{E}_{\mathtt{D}}[\Big(f(x_0) - \mathbb{E}\,\hat{f}_{\mathtt{D}}(x_0) + \mathbb{E}\,\hat{f}_{\mathtt{D}}(x_0) - \hat{f}_{\mathtt{D}}(x_0)\Big)^2]$$

$$= \boxed{\sigma_\varepsilon^2 + \underbrace{(f(x_0) - \mathbb{E}\,\hat{f}_{\mathtt{D}}(x_0))^2}_{\text{Bias}^2} + \underbrace{\mathbb{V}(\hat{f}_{\mathtt{D}}(x_0))}_{\text{Variance}}}$$

Sometimes, this quantity should be taken expectation w.r.t. $x_0 \sim p_X$.

[1] also called **Prediction Error**. In some cases, this definition does not contain ε_0 contribution.

- low bias: large model space, low training error, overfitting, bad generalization ability (high variance);
- low variance: rigid model space, insensitive to the perturbation of the dataset used in fitting; good generalization for the new data from the same distribution.
- BAD news: it is almost impossible to decrease the bias and variance terms simultaneously!
- Criteria for model assessment or variable selection: **good trade-off between the bias and variance**
  - Analyse the bias or variance or model complexity to have analytical results (very limited cases)
  - A practical method is to estimate $\mathbb{E}[(Y-\hat{f}_{\mathtt{D}}(X))^2] \approx \frac{1}{n}\sum_j (Y'_j - \hat{f}_{\mathtt{D}}(X'_j))$ where X'_j are not from $\mathtt{D}$ but an independent dataset $\mathtt{D}'$.
  - Where is the "new" dataset $\mathtt{D}'$? cross-validation (split 1 cent into 1/2 cents) and bootstrap (1 cent used twice) .

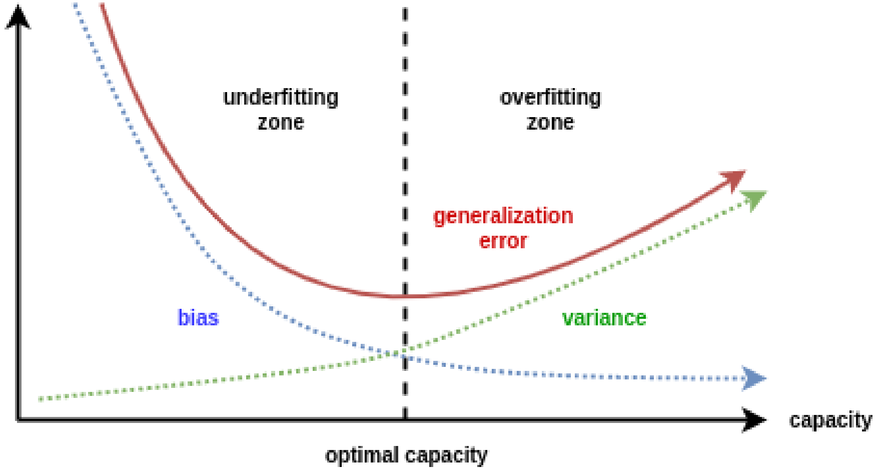


Figure: Bias-Variance Tradeoff as a Function of Model Capacity/Complexity

Model assessment

Typical objectives:

- Choose a value of a tuning parameter (hyperparameter) used in the model (such as k in k-NN)
- Estimate the prediction performance (test error) of a given model

Remarks:

- For both objectives, the best approach is to run the procedure on an independent test set, if one is available.
- If possible, one should use different test data for (1) and (2) above: a *validation set* for (1) and a *test set* for (2).
- Often there is insufficient data to create a separate validation or test set. In this case, *Cross-Validation* is useful.

K-fold cross validation

Denote the hyper-parameter by λ. K-fold cross validation is the most popular method for estimating a tuning parameter λ.

Divide the dataset (of size N) into K subsets: $\mathcal{A}_1, \ldots, \mathcal{A}_K$ ($K = 2, 5, 10$ or N)

- For each $k = 1, \ldots, K$, fit the model with parameter λ to $\{\mathcal{A}_1, \ldots, \mathcal{A}_{k-1}, \mathcal{A}_{k+1}, \ldots, \mathcal{A}_K\}$ giving $f_\lambda^{-k}(\cdot)$, and compute its prediction error on $\mathcal{A}_k$:

$$E_k(\lambda) = \sum_{x_i \in \mathcal{A}_k} \ell(y_i, f_\lambda^{-k}(x_i)).$$

- The average of these K values $E_k(\lambda)$ give the cross-validation error (per sample)

$$CV(\lambda) := \frac{1}{N} \sum_{k=1}^{K} E_k(\lambda).$$

- Choose the optimal λ^* yielding the smallest $CV(\lambda)$.

K-fold cross validation

- Cross-validation is often abbreviated as CV.
- In the subset selection procedure, λ is the subset size
- $f^{-k}(\lambda)$ is the best model of size λ, found from the training set that leaves out the k-th part of the data
- $E_k(\lambda)$ is its estimated test error on the k-th part.
- Using K-fold CV, the K test error estimates are averaged to give the final CV estimated test error.
- The output is the model associated with λ^*, typically, computed by using all N data.

Bootstrap

- Bootstrap works by sampling N times with replacement from the training set to form a "bootstrap" data set. Then model is estimated on the bootstrap data set, and predictions are made on the original training set.
- This process is repeated many times and the results are averaged.
- *Bootstrap is most useful for estimating standard errors of predictions.*
- Can also use modified versions of the bootstrap to estimate prediction error. Sometimes produces better estimates than CV (still an open question!)