Machine Learning

Lecture 05: The Bias-Variance decomposition

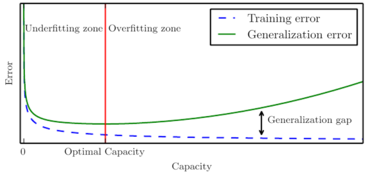
Nevin L. Zhang
lzhang@cse.ust.hk

Department of Computer Science and Engineering
The Hong Kong University of Science and Technology

This set of notes is based on internet resources and
Andrew Ng. Lecture Notes on Machine Learning. Stanford.

Outline

Introduction

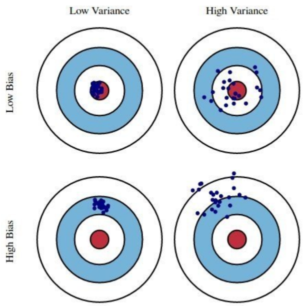


- Earlier, we have learned that
  - Training error always decreases with model capacity, while
  - Generalization error decreases with model capacity model initially, and increases with it after a certain point.
- Model selection: Choose a model of appropriate capacity so as to minimize the generalization error.

Introduction

- Objective of this lecture:
  - Point out that generalization error has two sources: **bias** and **variance**.
  - Use the decomposition to explain the dependence of generalization error on model capacity.
  - Model selection: Trade-off between bias and variance.
- The bias-variance decomposition will be derived in the context of regression, but the bias-variance trade-off applies to classification also.

Bias and Variance: The Concept



True Distributions

$$\begin{tikzcd}
P_1(\mathbf{x}, y) \arrow[d] & P_2(\mathbf{x}, y) \arrow[d] & \cdots\cdots & P_k(\mathbf{x}, y) \arrow[d, "\boxed{\text{Data Generation}}"] \\
D1 \arrow[d] & D2 \arrow[d] & \cdots\cdots & Dk \arrow[d, "\boxed{\text{Learning Algorithm}}"] \\
Q_1(y|\mathbf{x}) & Q_2(y|\mathbf{x}) & & Q_k(y|\mathbf{x})
\end{tikzcd}$$

- An algorithm is to be applied on different occasions.
- **High bias**: Poor performances on most occasions.
  - Cause: Erroneous assumptions in the learning algorithm.
- **High variance**: Different performances on different occasions.
  - Cause: Fluctuations in the training set.

Outline

Regression Problem Restated

The notations used in this lecture will be different from previous lectures so as to be consistence with the relevant literature.

- Previous statement:
  - Given: A **training set** $\mathcal{D} = \{\mathbf{x}_i, y_i\}_{i=1}^{N}$, where $y_i \in \mathbb{R}$,
  - Task: Determine the weights $\mathbf{w}$:

$$y = f(\mathbf{x}) = \mathbf{w}^\top \phi(\mathbf{x})$$

- New statement
  - Given: A training set $S = \{(\mathbf{x}_i, y_i)\}_{i=1}^{m}$ where $y_i \in \mathbb{R}$, and a **hypothesis class** $\mathcal{H}$ of regression functions, e.g,

$$\mathcal{H} = \{h(\mathbf{x}) = \mathbf{w}^\top \phi(\mathbf{x}) | \mathbf{w}\}$$

  - Task: Choose one **hypothesis** h from $\mathcal{H}$.

Training and Training Error

- The training/empirical error of a hypothesis h is calculated on the training set $S = \{(\mathbf{x}_i, y_i)\}_{i=1}^{m}$

$$\hat{\epsilon}(h) = \frac{1}{m}\sum_{i=1}^{m}(y_i - h(\mathbf{x}_i))^2$$

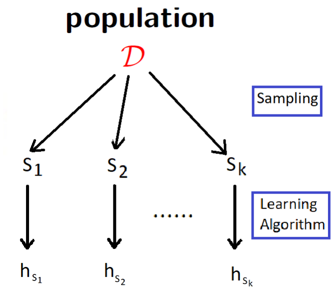
- **Training**: Obtain a optimal hypothesis $\hat{h}$ by minimizing the training error:

$$\hat{h} = \arg\min_{h \in \mathcal{H}} \hat{\epsilon}(h),$$

- The **training error** is: $\hat{\epsilon}(\hat{h})$

Random Fluctuations in Training Set

- We assume that the training set consist of i.i.d samples from a **population** (i.e., true distribution) $\mathcal{D}$.
- Obviously, the learned function $\hat{h}$ depends on the particular training set used. So, we denote it as h_S.
- The learning algorithm is to be applied in the future. There are multiple ways in which the sampling can turn out. In other words, the training set we will get is only one of many possible training sets.



The Generalization Error

- The **generalization error** of the learned function h_S is

$$\epsilon(h_S) = E_{(\mathbf{x},y)\sim\mathcal{D}}[(y - h_S(\mathbf{x}))^2]$$

- The difference between the generalization error and the training error is called the **generalization gap**:

$$\epsilon(h_S) - \hat{\epsilon}(h_S)$$

- The generalization gap depends on randomness in the training set S.
- We should care about the overall performance of an algorithm over all possible training sets, rather than its performance on a particular training set. So, ideally we want to minimize the expected generalization error

$$\epsilon = E_S[\epsilon(h_S)] = E_S[E_{(\mathbf{x},y)\sim\mathcal{D}}[(y - h_S(\mathbf{x}))^2]]$$

The Bias-Variance Decomposition

$$
\begin{aligned}
\epsilon &= E_S E_{(\mathbf{x},y)}[(y - h_S(\mathbf{x}))^2] \\
&= E_S E_{(\mathbf{x},y)}[(y - h_S)^2] \quad \text{Dropping“ } (\mathbf{x})\text{” for readability} \\
&= E_S E_{(\mathbf{x},y)}[(y - E_S(h_S) + E_S(h_S) - h_S)^2] \\
&= E_S E_{(\mathbf{x},y)}[(y - E_S(h_S))^2] + E_S E_{(\mathbf{x},y)}[(E_S(h_S) - h_S)^2] \\
&\quad +2 E_S E_{(\mathbf{x},y)}[(y - E_S(h_S))(E_S(h_S) - h_S)] \\
&= E_S E_{(\mathbf{x},y)}[(y - E_S(h_S))^2] + E_S E_{(\mathbf{x},y)}[(E_S(h_S) - h_S)^2] \\
&\quad +2 E_{(\mathbf{x},y)}[(y - E_S(h_S))(E_S(E_S(h_S)) - E_S(h_S))] \\
&= E_S E_{(\mathbf{x},y)}[(y - E_S(h_S))^2] + E_S E_{(\mathbf{x},y)}[(E_S(h_S) - h_S)^2] \\
&= E_{(\mathbf{x},y)}[(y - E_S(h_S(\mathbf{x})))^2] + E_S E_{(\mathbf{x})}[(E_S(h_S(\mathbf{x})) - h_S(\mathbf{x}))^2]
\end{aligned}
$$

Bias-Variance Decomposition

- $E_S E_{(\mathbf{x})}[(h_S(\mathbf{x}) - E_S(h_S(\mathbf{x})))^2]$:
    - This term is due to randomness in the choice of training set S.
    - It is called the **variance**.
- $E_{(\mathbf{x},y)}[(y - E_s(h_s(\mathbf{x})))^2]$:
    - This term is due to the choice of the hypothesis class $\mathcal{H}$.
    - It is called the **bias**^{2}
- Error decomposition:

$$\epsilon = E_{(\mathbf{x},y)}[(y - E_S(h_S(\mathbf{x})))^2] + E_S E_{(\mathbf{x})}[(E_S(h_S(\mathbf{x})) - h_S(\mathbf{x}))^2]$$

$$\text{Expected Generalization Error} = \text{Bias}^2 + \text{Variance}$$

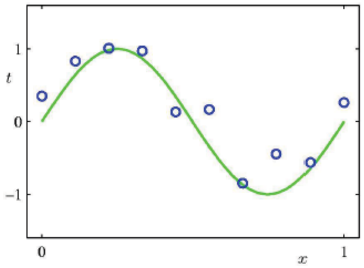
Bias-Variance Decomposition

$$\text{Expected Generalization Error} = \text{Bias}^2 + \text{Variance}$$

- The bias is an error from **erroneous assumptions** in the learning algorithm. High bias can cause an algorithm to miss the relevant relations between features and target outputs (underfitting).
- The variance is an error from **sensitivity to small fluctuations in the training set**. High variance can cause an algorithm to model the random noise in the training data, rather than the intended outputs (overfitting).

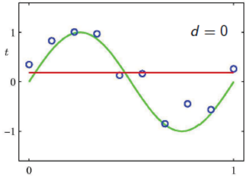
Outline

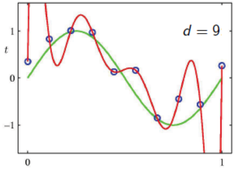
Bias-Variance Decomposition: Illustration



- Suppose the green curve is the true function.
- We randomly sample 10 training points (blue) from the function.
- Consider learning a polynomial function $y = h(x)$ of order d from the data.
- We the above multiple times.

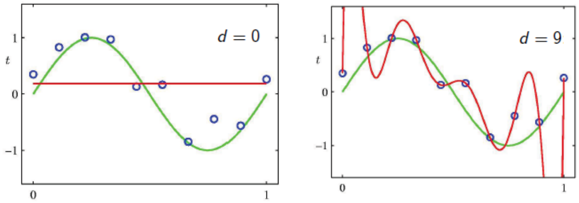
Bias-Variance Tradeoff: Illustration





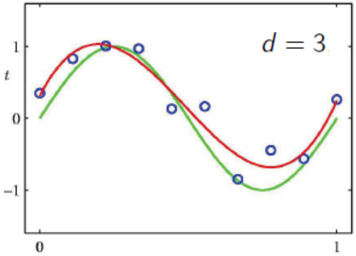
- If we choose $d = 0$, then we have
  - **Low variance**: If there is another training set sampled from the true function (blue) and we run the learning algorithm on it, we will get roughly the same function.
  - **High bias**: While the hypothesis is linear, the true function is not. If we sample a large number of training sets from the true function and learn a function from each of them, the average will still be very different from the true function.
  - In this case, the generalization would be high. And it is due to **underfitting**: hypothesis function too rigid to fit the data points.

Bias-Variance Tradeoff: Illustration



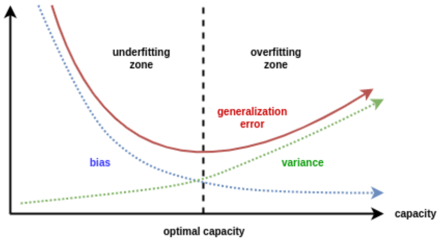
- If we choose $d = 9$, then we
  - **High variance**: If there is another training set sampled from the true function and we run the learning algorithm on it, we are likely to get a very different function.
  - **Low bias**: If we sample a large number of training sets from the true function and learn a function from each of them, the average will still approximate the true function well.
  - In this case, the generalization would be high. It is due to **overfitting**: hypothesis too soft, fit the data points too much.

Bias-Variance Tradeoff: Illustration



- If we choose $d = 3$, we get low generalization error
  - not too much variance and not too much bias
  - the hypothesis fit the data just right

Bias-Variance Tradeoff



- Usually, the bias decreases with the complexity of the hypothesis class $\mathcal{H}$ (model capacity),while the variance increases with it.
- To minimize the expected generalized error, one needs to make proper tradeoff between bias and variance by choosing a model that is neither too simple nor too complex.

Bias-Variance Tradeoff

- Cross validation and regularization are methods for doing so.
- Ridge regression:

$$J(\mathbf{w}, w_0) = \frac{1}{m}\sum_{i=1}^{m}(y_i - (w_0 + \mathbf{w}^\top \phi(\mathbf{x}_i)))^2 + \lambda||\mathbf{w}||_2^2$$

- LASSO:

$$J(\mathbf{w}, w_0) = \frac{1}{m}\sum_{i=1}^{m}(y_i - (w_0 + \mathbf{w}^\top \phi(\mathbf{x}_i)))^2 + \lambda||\mathbf{w}||_1$$

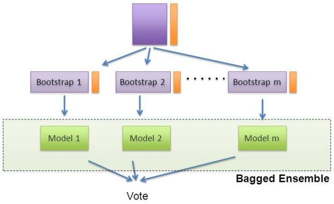
- Regularization reduces the variance by forcing the solution to be simple. Sometimes, it increases the bias.

Bias-Variance Decomposition for Classification

- The bias-variance decomposition was originally formulated for least-squares regression.
- For the case of classification under the 0-1 loss, it's possible to find a similar decomposition.
- If the classification problem is phrased as probabilistic classification, then the expected squared error of the predicted probabilities with respect to the true probabilities can be decomposed in a similar fashion.

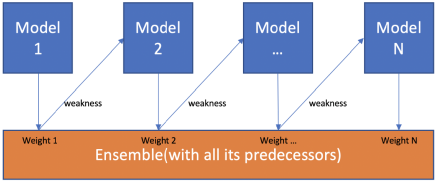
Outline

Bagging (Bootstrap Aggregation) for Variance Reduction



- Train several different models separately on different randomly sampled (with replacement) subsets of data called **bootstrap samples**,
- Classification: Have all of the models vote on the output for test examples.
- Analogy:
  - Estimate average income of HKer by randomly interviewing 10 people.
  - Variance is reduced if do it multiple times and take average.
- In HW2, we will analyze this mathematically.

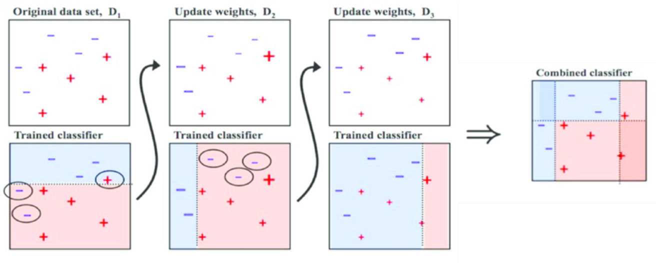
Boosting for Bias Reduction



- Assign equal weights to all the training examples and choose a base algorithm.
- At each step of iteration, we apply the base algorithm to the training set and increase the weights of the incorrectly classified examples.
- We iterate n times, each time applying base learner on the training set with updated weights.
- The final model is the weighted sum of the n learners.

Boosting for Bias Reduction

Illustration of Boosting



Boosting techniques:

- AdaBoost
- Gradient Boosting
- XGBoost

Outline

Classification Problem Restated

- Given: A training set $S = \{\mathbf{x}_i, y_i\}_{i=1}^{m}$ where $y_i \in \{-1, 1\}$, and a **hypothesis class** $\mathcal{H}$ of classifiers, e.g,

$$\mathcal{H} = \{h(\mathbf{x}, \mathbf{w}, b) = sign(\mathbf{w}^\top \phi(\mathbf{x}) + b) | \mathbf{w}, b\}$$

- Choose one **hypothesis** h from $\mathcal{H}$.

Errors for Classification

- The **training/empirical error** is is calculated on a training set $S = \{\mathbf{x}_i, y_i\}_{i=1}^{m}$:

$$\hat{\epsilon}(h) = \frac{1}{m}\sum_{i=1}^{m}\mathbf{1}(y_i \neq h(\mathbf{x}_i))$$

- We assume that both the training set and test set are iid samples from a **population** $\mathcal{D}$, which includes unseen examples, some of which we will encounter in the future. The **generalization error** is:

$$\epsilon(h) = E_{(\mathbf{x},y)\sim\mathcal{D}}\mathbf{1}(y \neq h(\mathbf{x})]$$

Key Issues in Learning Theory

- Learning algorithms obtain a hypothesis h by minimizing the training error:

  $$\hat{h} = \arg\min_{h \in \mathcal{H}} \hat{\epsilon}(h)$$

  Note that in practice we usually minimize a **surrogate loss function** instead of the training error itself. Nonetheless, the results derived on training error still apply.

- Define the **optimal hypothesis** to be the one that minimizing the generalization error:

  $$h^* = \arg\min_{h \in \mathcal{H}} \epsilon(h)$$

  It is ideally what we want. However, we cannot get it directly.

Key Issues in Learning Theory

1. The classifier $\hat{h}$ is obtained by minimizing the training error. Why should it have low generalization error of $\epsilon(\hat{h})$?
2. What can we do to make the generalization error of the learned classifier $\hat{h}$ as low as possible?
3. How much data do we need if we want to guarantee the generalization error of $\hat{h}$ be close to the optimal?

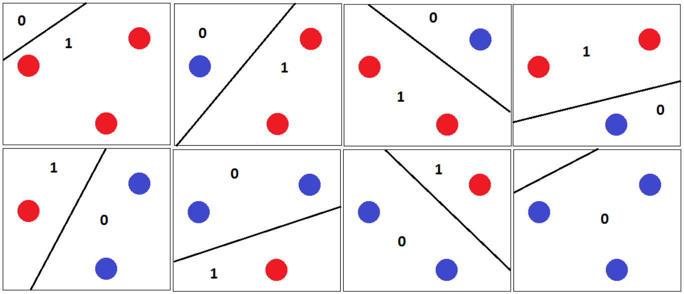
To give the answers, we first need to introduce the concept of **VC dimension**.

Shattering

- We say that a hypothesis class $\mathcal{H}$ of binary classifiers **shatters** a set of points if
  - No matter how we label the points (with 0 or 1), there is a classifier $h \in \mathcal{H}$ that separates the two classes with no errors.

Shattering: Example

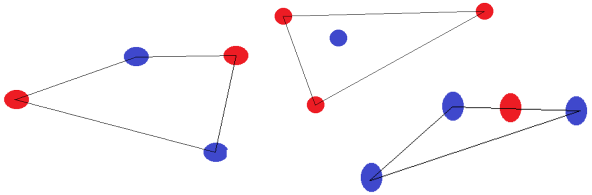
- Consider the hypothesis class $\mathcal{H}$ of all linear classifers with x_1 and x_2 as inputs. Here is a set of three points that $\mathcal{H}$ shatters:



- Note that there are sets of three points that are not shattered by linear classifiers. (Figure out an example yourself.)

Shattering: Example

- However, linear classifiers do not shatter any set of points on a plane
    - If the four points form a figure with 4 sides, we can label opposite corners with the same label. Then no line can separate the two classes.
    - If they do not form a figure with 4 sides, we can still label the points in such a way such that no line can separate the two classes.



VC Dimension

- The **Vapnik-Chervonenkis (VC) dimension** $VC(\mathcal{H})$ of a hypothesis class $\mathcal{H}$ of binary classifiers is:
  - The size of the largest set that is shattered by $\mathcal{H}$.
  - If $\mathcal{H}$ can shatter arbitrarily large sets, then $VC(\mathcal{H}) = \infty$.
- Example: $\mathcal{H} = \{$Linear classifiers with two inputs x_1 and $x_2\}$
  - $\mathcal{H}$ shatters at least one 3-point set.
  - $\mathcal{H}$ cannot shatter any 4-point sets.
  - So, $VC(\mathcal{H}) = 3$.
- It turns out that, for "most" hypothesis classes, the VC dimension (assuming a "reasonable" parameterization) is also roughly linear in the number of parameters.

Vapnik-Chervonenkis Theorem [1]

Theorem

Given a hypothesis class $\mathcal{H}$, let $d = VC(\mathcal{H})$. Then,

1. *With probability at least $1-\delta$, we have that all for $h \in \mathcal{H}$,*

$$|\epsilon(h) - \hat{\epsilon}(h)| \leq O(\sqrt{\frac{d}{m}\log\frac{m}{d} + \frac{1}{m}\log\frac{1}{\delta}}) \tag{1}$$

2. *With probability at least $1-\delta$, we have*

$$\epsilon(\hat{h}) \leq \epsilon(h^*) + O(\sqrt{\frac{d}{m}\log\frac{m}{d} + \frac{1}{m}\log\frac{1}{\delta}}) \tag{2}$$

[1] This is a simplified version of the theorem taken from Andrew Ng's notes.

Answers to the Questions

- The classifier $\hat{h}$ is obtained by minimizing the training error. Why should it have low generalization error of $\epsilon(\hat{h})$?
  - According to Part 1 of the VC Theorem, the training error $\hat{\epsilon}(h)$ of any hypothesis h is related to its generalization error $\epsilon(h)$. Hence, minimizing $\hat{\epsilon}(h)$ can also brings down $\epsilon(h)$.
  - The larger the sample size m, the more closely $\epsilon(h)$ and $\hat{\epsilon}(h)$ are related. Hence, in general, more data implies better performance of learning algorithms.
- The following corollary of the VC theorem tells us how much data we need in order for an algorithm to learn "well".

Corollary

For $|\epsilon(h) - \hat{\epsilon}(h)| \leq \gamma$ for hold for all $h \in \mathcal{H}$ with probability $1 - \delta$, it suffices that $m = O_{\gamma,\delta}(d)$.

This is a **sample complexity** result.

Answers to the Questions

- What can we do to make the generalization error of the learned classifier $\hat{h}$ as low as possible?
  - Part 2 of the VC Theorem provides an answer.
  - Note that $h^* = \arg\min_{h \in \mathcal{H}} \epsilon(h)$. Consider the impact of enlarging $\mathcal{H}$ on the two terms on the RHS of (2)
    - The first term would decrease,
    - The second term would increase
  - Reducing the first term is to reduce "bias", and while reducing the second term is to reduce "variance".
  - The second "variance" terms decreases with sample size m. So, large sample size helps to avoid overfitting.
  - Validation and regularization can be used to make a proper tradeoff between the two terms.

Answers to the Questions

- How much data do we need if we want to guarantee the generalization error of $\hat{h}$ close to the optimal?
  - The following corollary of the VC theorem answers this question

Corollary

For $\epsilon(\hat{h}) \leq \epsilon(h^*) + \gamma$ with probability $1 - \delta$, it suffices that $m = O_{\gamma,\delta}(d)$.