Learning Theory & Regularization

Shan-Hung Wu
shwu@cs.nthu.edu.tw

Department of Computer Science,
National Tsing Hua University, Taiwan

Machine Learning

Outline

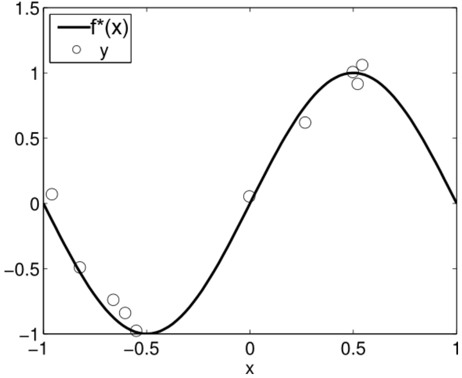
1. **Learning Theory**
2. **Point Estimation: Bias and Variance**
   - Consistency*
3. **Decomposing Generalization Error**
4. **Regularization**
   - Weight Decay
   - Validation

Outline

1. **Learning Theory**
2. Point Estimation: Bias and Variance
   - Consistency*
3. Decomposing Generalization Error
4. Regularization
   - Weight Decay
   - Validation

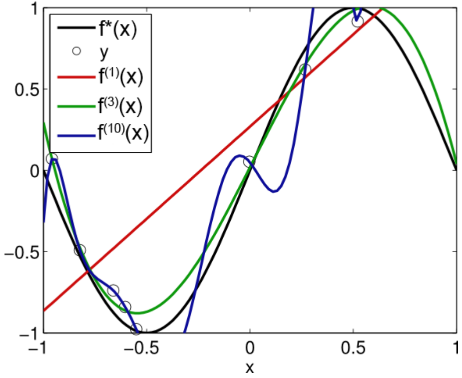
Which Polynomial Degree Is Better? I

- Given a training set $\mathbb{X} = \{(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)})\}_{i=1}^{N}$ i.i.d. sampled from of $\mathrm{P}(\mathrm{x}, \mathrm{y})$
- Assume $\mathrm{P}(\mathrm{x}, \mathrm{y}) = \mathrm{P}(\mathrm{y} \,|\, \mathrm{x})\mathrm{P}(\mathrm{x})$, where
  - $\mathrm{P}(\mathrm{x}) \sim \mathrm{Uniform}(-1, 1)$
  - $\mathrm{y} = \sin(\pi \mathrm{x}) + \varepsilon$, $\varepsilon \sim \mathcal{N}(0, \sigma^2)$



Which Polynomial Degree Is Better? II

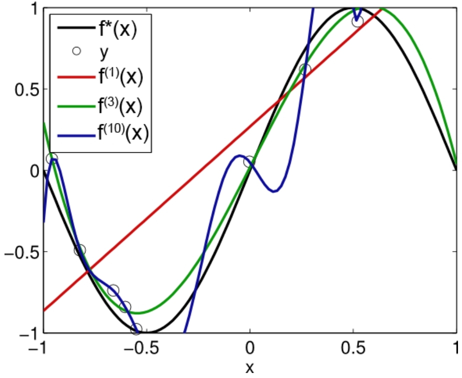
- Consider 3 unregularized polynomial regressors of degrees $P = 1$, 3, and 10



- Which one would you pick?

Which Polynomial Degree Is Better? II

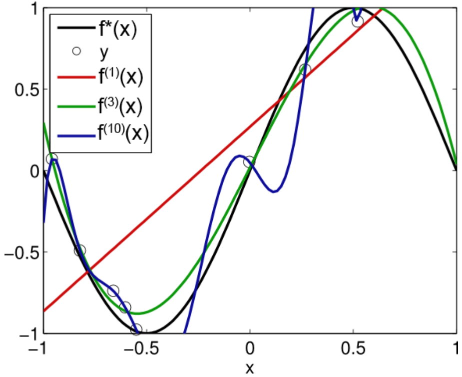
- Consider 3 unregularized polynomial regressors of degrees $P = 1$, 3, and 10



- Which one would you pick? Probably not $P = 1$ nor $P = 10$

Which Polynomial Degree Is Better? II

- Consider 3 unregularized polynomial regressors of degrees $P = 1$, 3, and 10



- Which one would you pick? Probably not $P = 1$ nor $P = 10$
- Note that $P = 10$ has ***zero*** training error
    - Any N points can be perfectly fitted by a polynomial of degree $N - 1$

Empirical Error vs. Generalization Error

- In ML, we usually "learn" a function by minimizing the ***empirical error/risk*** defined over a training set of size N:

$$C_N(\boldsymbol{w}) \text{ or } C_N[f] = \frac{1}{N}\sum_{i=1}^{N}\text{loss}\left(f(\boldsymbol{x}^{(i)};\boldsymbol{w}),\boldsymbol{y}^{(i)}\right)$$

  - E.g., $C_N(\boldsymbol{w}) = \frac{1}{2}\sum_{i=1}^{N}\left(y^{(i)} - \boldsymbol{w}^\top\boldsymbol{x}^{(i)}\right)^2$ in linear regression

Empirical Error vs. Generalization Error

- In ML, we usually "learn" a function by minimizing the ***empirical error/risk*** defined over a training set of size N:

$$C_N(\boldsymbol{w}) \text{ or } C_N[f] = \frac{1}{N}\sum_{i=1}^{N} \text{loss}\left(f(\boldsymbol{x}^{(i)};\boldsymbol{w}),\boldsymbol{y}^{(i)}\right)$$

  - E.g., $C_N(\boldsymbol{w}) = \frac{1}{2}\sum_{i=1}^{N}\left(y^{(i)} - \boldsymbol{w}^\top \boldsymbol{x}^{(i)}\right)^2$ in linear regression

- But our goal is to have a low ***generalization error/risk*** defined over the underlying data distribution:

$$C(\boldsymbol{w}) \text{ or } C[f] = \int \text{loss}\left(f(\boldsymbol{x};\boldsymbol{w}),y\right) d\mathrm{P}(\boldsymbol{x},\boldsymbol{y})$$

  - Can be estimated by the ***testing error*** $C_{N'}(\boldsymbol{w}) = \frac{1}{N'}\sum_{i=1}^{N'} \text{loss}\left(f(\boldsymbol{x}'^{(i)};\boldsymbol{w}),\boldsymbol{y}'^{(i)}\right)$ defined over the testing set $\mathbb{X}' = \{(\boldsymbol{x}'^{(i)},\boldsymbol{y}'^{(i)})\}_{i=1}^{N'}$

Empirical Error vs. Generalization Error

- In ML, we usually "learn" a function by minimizing the ***empirical error/risk*** defined over a training set of size N:

$$C_N(\boldsymbol{w}) \text{ or } C_N[f] = \frac{1}{N}\sum_{i=1}^{N} \text{loss}\left(f(\boldsymbol{x}^{(i)};\boldsymbol{w}), \boldsymbol{y}^{(i)}\right)$$

  - E.g., $C_N(\boldsymbol{w}) = \frac{1}{2}\sum_{i=1}^{N}\left(y^{(i)} - \boldsymbol{w}^\top \boldsymbol{x}^{(i)}\right)^2$ in linear regression
- But our goal is to have a low ***generalization error/risk*** defined over the underlying data distribution:

$$C(\boldsymbol{w}) \text{ or } C[f] = \int \text{loss}(f(\boldsymbol{x};\boldsymbol{w}), y)\, d\mathrm{P}(\boldsymbol{x}, \boldsymbol{y})$$

  - Can be estimated by the ***testing error*** $C_{N'}(\boldsymbol{w}) = \frac{1}{N'}\sum_{i=1}^{N'} \text{loss}\left(f(\boldsymbol{x}'^{(i)};\boldsymbol{w}), \boldsymbol{y}'^{(i)}\right)$ defined over the testing set $\mathbb{X}' = \{(\boldsymbol{x}'^{(i)}, \boldsymbol{y}'^{(i)})\}_{i=1}^{N'}$
- Does a low $C_N[f]$ implies low $C[f]$?

Empirical Error vs. Generalization Error

- In ML, we usually "learn" a function by minimizing the ***empirical error/risk*** defined over a training set of size N:

$$C_N(\boldsymbol{w}) \text{ or } C_N[f] = \frac{1}{N}\sum_{i=1}^{N} \text{loss}\left(f(\boldsymbol{x}^{(i)};\boldsymbol{w}),\boldsymbol{y}^{(i)}\right)$$

  - E.g., $C_N(\boldsymbol{w}) = \frac{1}{2}\sum_{i=1}^{N}\left(y^{(i)} - \boldsymbol{w}^\top \boldsymbol{x}^{(i)}\right)^2$ in linear regression
- But our goal is to have a low ***generalization error/risk*** defined over the underlying data distribution:

$$C(\boldsymbol{w}) \text{ or } C[f] = \int \text{loss}\left(f(\boldsymbol{x};\boldsymbol{w}),y\right) d\mathrm{P}(\boldsymbol{x},\boldsymbol{y})$$

  - Can be estimated by the ***testing error*** $C_{N'}(\boldsymbol{w}) = \frac{1}{N'}\sum_{i=1}^{N'} \text{loss}\left(f(\boldsymbol{x}'^{(i)};\boldsymbol{w}),\boldsymbol{y}'^{(i)}\right)$ defined over the testing set $\mathbb{X}' = \{(\boldsymbol{x}'^{(i)},\boldsymbol{y}'^{(i)})\}_{i=1}^{N'}$
- Does a low $C_N[f]$ implies low $C[f]$? No, as $P = 10$ indicates

No-Free-Lunch Theorem

- Why $C[f]$ is defined over a ***particular*** data generating distribution P?

No-Free-Lunch Theorem

- Why $C[f]$ is defined over a ***particular*** data generating distribution P?

Theorem (No-Free-Lunch Theorem [4])

Averaged over all possible data generating distributions, every classification algorithm has the same error rate when classifying unseen points.

No-Free-Lunch Theorem

- Why $C[f]$ is defined over a ***particular*** data generating distribution P?

Theorem (No-Free-Lunch Theorem [4])

Averaged over all possible data generating distributions, every classification algorithm has the same error rate when classifying unseen points.

- No machine learning algorithm is better than any other universally

No-Free-Lunch Theorem

- Why $C[f]$ is defined over a ***particular*** data generating distribution P?

Theorem (No-Free-Lunch Theorem [4])

Averaged over all possible data generating distributions, every classification algorithm has the same error rate when classifying unseen points.

- No machine learning algorithm is better than any other universally
- The goal of ML is ***not*** to seek a universally good learning algorithm
- Instead, a good algorithm that performs well on data drawn from a ***particular*** P we care about

Learning Theory

- Let $f^* = \arg\min_f C[f]$ be the best possible function we can get

Learning Theory

- Let $f^* = \arg\min_f C[f]$ be the best possible function we can get
- Since we are seeking a prediction function in a model (hypothesis space) $\mathbb{F}$, this is what can have at best: $f^*_{\mathbb{F}} = \arg\min_{f \in \mathbb{F}} C[f]$

Learning Theory

- Let $f^* = \arg\min_f C[f]$ be the best possible function we can get
- Since we are seeking a prediction function in a model (hypothesis space) $\mathbb{F}$, this is what can have at best: $f^*_{\mathbb{F}} = \arg\min_{f \in \mathbb{F}} C[f]$
- But we only minimizes empirical errors on limited examples of size N, this is what we actually have $f_N = \arg\min_{f \in \mathbb{F}} C_N[f]$
  - Ignoring numerical errors (due to, e.g., numerical optimization)

Learning Theory

- Let $f^* = \arg\min_f C[f]$ be the best possible function we can get
- Since we are seeking a prediction function in a model (hypothesis space) $\mathbb{F}$, this is what can have at best: $f^*_{\mathbb{F}} = \arg\min_{f\in\mathbb{F}} C[f]$
- But we only minimizes empirical errors on limited examples of size N, this is what we actually have $f_N = \arg\min_{f\in\mathbb{F}} C_N[f]$
  - Ignoring numerical errors (due to, e.g., numerical optimization)
- ***Learning theory***: how to characterize

$$C[f_N] = \int \text{loss}(f_N(\boldsymbol{x};\boldsymbol{w}), y)\, d\text{P}(\boldsymbol{x}, y)?$$

  - Not to confuse $C[f_N]$ with $C_N[f]$

Learning Theory

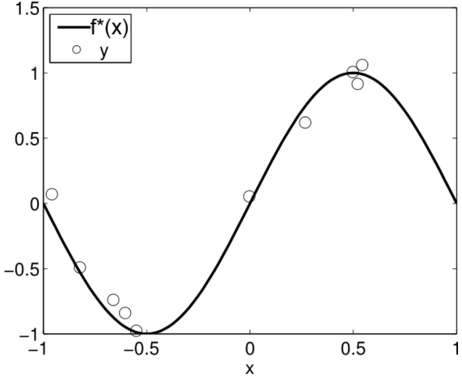
- Let $f^* = \arg\min_f C[f]$ be the best possible function we can get
- Since we are seeking a prediction function in a model (hypothesis space) $\mathbb{F}$, this is what can have at best: $f^*_{\mathbb{F}} = \arg\min_{f\in\mathbb{F}} C[f]$
- But we only minimizes empirical errors on limited examples of size N, this is what we actually have $f_N = \arg\min_{f\in\mathbb{F}} C_N[f]$
  - Ignoring numerical errors (due to, e.g., numerical optimization)
- ***Learning theory***: how to characterize

$$C[f_N] = \int \text{loss}(f_N(\boldsymbol{x};\boldsymbol{w}), y) d\text{P}(\boldsymbol{x}, y)?$$

  - Not to confuse $C[f_N]$ with $C_N[f]$
- Bounding methods
- Decomposition methods

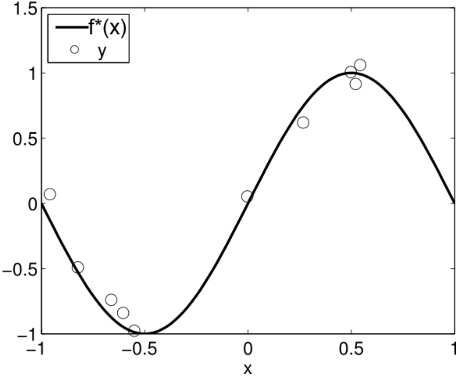
Bounding Methods I

- $\min_f C[f] = C[f^*]$ is called the ***Bayes error***
  - Larger than 0 when there is randomness in $\mathrm{P}(\mathrm{y}\,|\,\mathrm{x})$
  - E.g., in our regression problem: $y = f^*(\boldsymbol{x};\boldsymbol{w}) + \varepsilon$, $\varepsilon \sim \mathcal{N}(0, \sigma^2)$



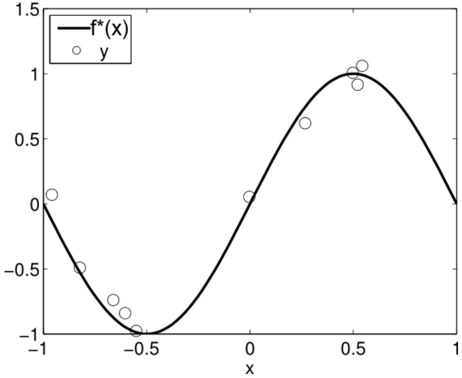
Bounding Methods I

- $\min_f C[f] = C[f^*]$ is called the ***Bayes error***
  - Larger than 0 when there is randomness in $\mathrm{P}(\mathrm{y}\,|\,\mathrm{x})$
  - E.g., in our regression problem: $y = f^*(\boldsymbol{x};\boldsymbol{w}) + \varepsilon$, $\varepsilon \sim \mathcal{N}(0, \sigma^2)$
- ***Cannot be avoided*** even we know $\mathrm{P}(\mathrm{x}, \mathrm{y})$ in the ground truth



Bounding Methods I

- $\min_f C[f] = C[f^*]$ is called the ***Bayes error***
  - Larger than 0 when there is randomness in $\mathrm{P}(\mathrm{y}\,|\,\mathrm{x})$
  - E.g., in our regression problem: $y = f^*(\boldsymbol{x};\boldsymbol{w}) + \varepsilon$, $\varepsilon \sim \mathcal{N}(0,\sigma^2)$
- ***Cannot be avoided*** even we know $\mathrm{P}(\mathrm{x},\mathrm{y})$ in the ground truth



- So, our target is to make $C[f_N]$ as close to $C[f^*]$ as possible

Bounding Methods II

- Let $\mathscr{E} = C[f_N] - C[f^*]$ be the ***excess error***
- We have
$$\mathscr{E} = \underbrace{C[f^*_{\mathbb{F}}] - C[f^*]}_{\mathscr{E}_{\text{app}}} + \underbrace{C[f_N] - C[f^*_{\mathbb{F}}]}_{\mathscr{E}_{\text{est}}}$$

Bounding Methods II

- Let $\mathscr{E} = C[f_N] - C[f^*]$ be the ***excess error***
- We have

$$\mathscr{E} = \underbrace{C[f_{\mathbb{F}}^*] - C[f^*]}_{\mathscr{E}_{\text{app}}} + \underbrace{C[f_N] - C[f_{\mathbb{F}}^*]}_{\mathscr{E}_{\text{est}}}$$

- $\mathscr{E}_{\text{app}}$ is called the ***approximation error***
- $\mathscr{E}_{\text{est}}$ is called the ***estimation error***

Bounding Methods II

- Let $\mathscr{E} = C[f_N] - C[f^*]$ be the ***excess error***
- We have

$$\mathscr{E} = \underbrace{C[f^*_{\mathbb{F}}] - C[f^*]}_{\mathscr{E}_{\text{app}}} + \underbrace{C[f_N] - C[f^*_{\mathbb{F}}]}_{\mathscr{E}_{\text{est}}}$$

- $\mathscr{E}_{\text{app}}$ is called the ***approximation error***
- $\mathscr{E}_{\text{est}}$ is called the ***estimation error***
- How to reduce these errors?

Bounding Methods II

- Let $\mathscr{E} = C[f_N] - C[f^*]$ be the ***excess error***
- We have
$$\mathscr{E} = \underbrace{C[f^*_{\mathbb{F}}] - C[f^*]}_{\mathscr{E}_{\text{app}}} + \underbrace{C[f_N] - C[f^*_{\mathbb{F}}]}_{\mathscr{E}_{\text{est}}}$$
- $\mathscr{E}_{\text{app}}$ is called the ***approximation error***
- $\mathscr{E}_{\text{est}}$ is called the ***estimation error***
- How to reduce these errors?
- We can reduce $\mathscr{E}_{\text{app}}$ by choosing a ***more complex*** $\mathbb{F}$
  - A complex $\mathbb{F}$ has a larger capacity
  - E.g., larger polynomial degree P in polynomial regression

Bounding Methods II

- Let $\mathscr{E} = C[f_N] - C[f^*]$ be the ***excess error***
- We have

$$\mathscr{E} = \underbrace{C[f_{\mathbb{F}}^*] - C[f^*]}_{\mathscr{E}_{\text{app}}} + \underbrace{C[f_N] - C[f_{\mathbb{F}}^*]}_{\mathscr{E}_{\text{est}}}$$

- $\mathscr{E}_{\text{app}}$ is called the ***approximation error***
- $\mathscr{E}_{\text{est}}$ is called the ***estimation error***
- How to reduce these errors?
- We can reduce $\mathscr{E}_{\text{app}}$ by choosing a ***more complex*** $\mathbb{F}$
  - A complex $\mathbb{F}$ has a larger capacity
  - E.g., larger polynomial degree P in polynomial regression
- How to reduce $\mathscr{E}_{\text{est}}$?

Bounding Methods III

- Bounds of $\mathscr{E}_{\text{est}}$ for, e.g., binary classifiers [1, 2, 3]:

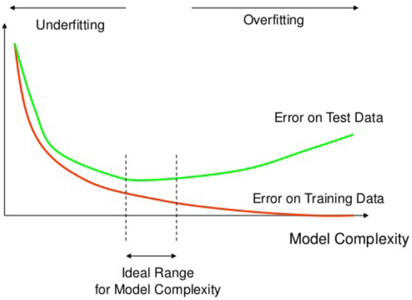
$$\mathscr{E}_{\text{est}} = O\left[\left(\frac{\text{Complexity}(\mathbb{F})\log N}{N}\right)^{\alpha}\right], \alpha \in \left[\frac{1}{2}, 1\right], \text{ with high probability}$$

- So, to reduce $\mathscr{E}_{\text{est}}$, we should either have
  - ***Simpler model*** (e.g., smaller polynomial degree P), or
  - Larger training set

Model Complexity, Overfit, and Underfit

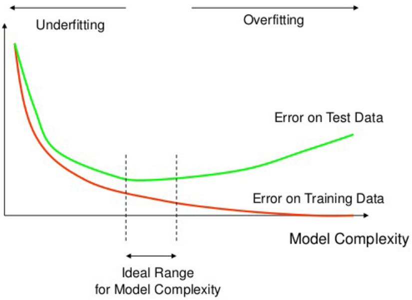
- Too simple a model leads to high $\mathscr{E}_{\text{app}}$

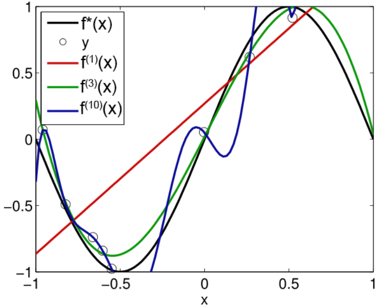
- Too complex a model leads to high $\mathscr{E}_{\text{est}}$



Model Complexity, Overfit, and Underfit

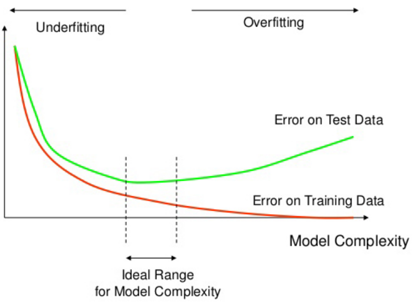
- Too simple a model leads to high $\mathscr{E}_{\text{app}}$ due to ***underfitting***
  - f_N fails to capture the shape of f^*
- Too complex a model leads to high $\mathscr{E}_{\text{est}}$

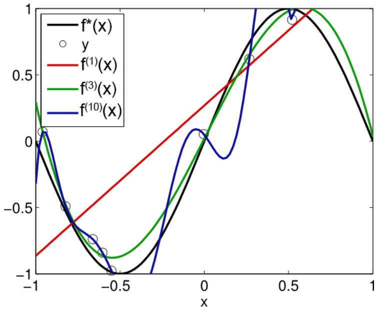




Model Complexity, Overfit, and Underfit

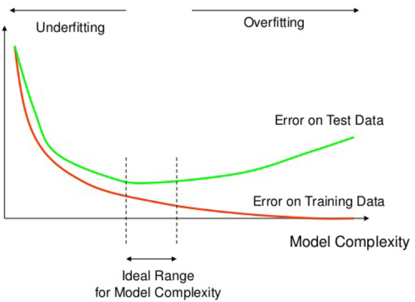
- Too simple a model leads to high $\mathscr{E}_{\text{app}}$ due to ***underfitting***
  - f_N fails to capture the shape of f^*
  - ***High*** training error; ***high*** testing error (given a sufficiently large N)
- Too complex a model leads to high $\mathscr{E}_{\text{est}}$

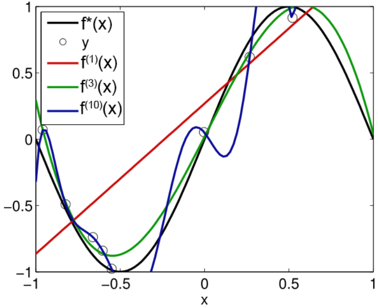




Model Complexity, Overfit, and Underfit

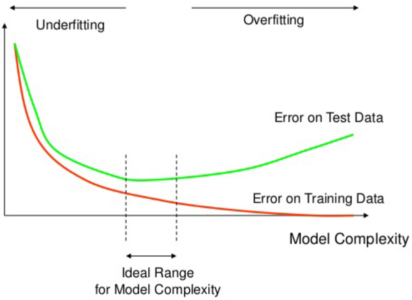
- Too simple a model leads to high $\mathscr{E}_{\text{app}}$ due to ***underfitting***
  - f_N fails to capture the shape of f^*
  - ***High*** training error; ***high*** testing error (given a sufficiently large N)
- Too complex a model leads to high $\mathscr{E}_{\text{est}}$ due to ***overfitting***
  - f_N captures not only the shape of f^* but also some spurious patterns (e.g., noise) local to a particular training set

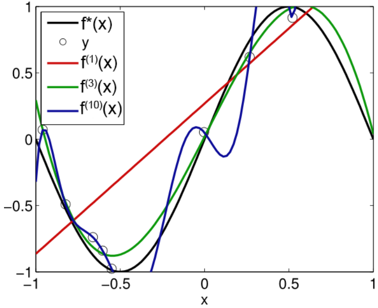




Model Complexity, Overfit, and Underfit

- Too simple a model leads to high $\mathscr{E}_{\text{app}}$ due to ***underfitting***
  - f_N fails to capture the shape of f^*
  - ***High*** training error; ***high*** testing error (given a sufficiently large N)
- Too complex a model leads to high $\mathscr{E}_{\text{est}}$ due to ***overfitting***
  - f_N captures not only the shape of f^* but also some spurious patterns (e.g., noise) local to a particular training set
  - ***Low*** training error; ***high*** testing error





Sample Complexity and Learning Curves

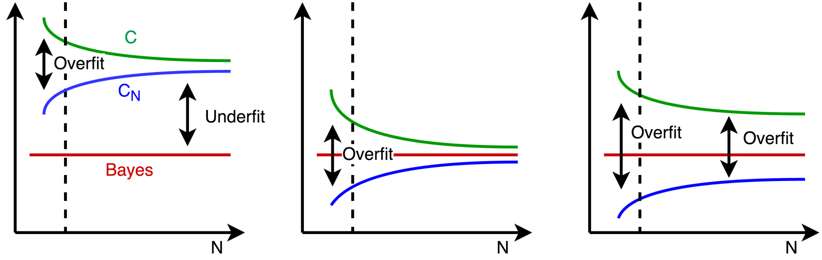
- How many training examples (N) are sufficient?

Sample Complexity and Learning Curves

- How many training examples (N) are sufficient?
- Different models/algorithms may have different ***sample complexity***
  - I.e., the N required to learn a target function with specified generalizability

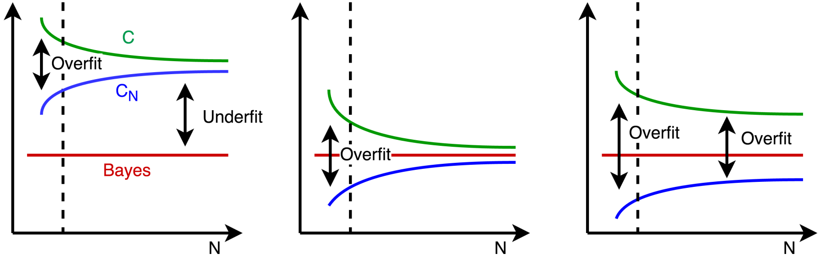
Sample Complexity and Learning Curves

- How many training examples (N) are sufficient?
- Different models/algorithms may have different ***sample complexity***
  - I.e., the N required to learn a target function with specified generalizability
- Can be visualized using the ***learning curves***



Sample Complexity and Learning Curves

- How many training examples (N) are sufficient?
- Different models/algorithms may have different ***sample complexity***
  - I.e., the N required to learn a target function with specified generalizability
- Can be visualized using the ***learning curves***
- Too small N results in overfit regardless of model complexity



Decomposition Methods

- Bounding methods analyze $C[f_N]$ qualitatively
  - General, as no (or weak) assumption on data distribution is made

Decomposition Methods

- Bounding methods analyze $C[f_N]$ qualitatively
  - General, as no (or weak) assumption on data distribution is made
- However, in practice, these bounds are too loose to quantify $C[f_N]$

Decomposition Methods

- Bounding methods analyze $C[f_N]$ qualitatively
  - General, as no (or weak) assumption on data distribution is made
- However, in practice, these bounds are too loose to quantify $C[f_N]$
- In some particular situations, we can ***decompose*** $C[f_N]$ into multiple meaningful terms

Decomposition Methods

- Bounding methods analyze $C[f_N]$ qualitatively
  - General, as no (or weak) assumption on data distribution is made
- However, in practice, these bounds are too loose to quantify $C[f_N]$
- In some particular situations, we can ***decompose*** $C[f_N]$ into multiple meaningful terms
- Assume particular
  - Loss function $\mathrm{loss}(\cdot)$, and
  - Data generating distribution $\mathrm{P}(\mathrm{x}, \mathrm{y})$

Decomposition Methods

- Bounding methods analyze $C[f_N]$ qualitatively
  - General, as no (or weak) assumption on data distribution is made
- However, in practice, these bounds are too loose to quantify $C[f_N]$
- In some particular situations, we can ***decompose*** $C[f_N]$ into multiple meaningful terms
- Assume particular
  - Loss function $\mathrm{loss}(\cdot)$, and
  - Data generating distribution $\mathrm{P}(\mathrm{x},\mathrm{y})$
- Require knowledge about the ***point estimation***

Outline

1. Learning Theory
2. Point Estimation: Bias and Variance
   - Consistency*
3. Decomposing Generalization Error
4. Regularization
   - Weight Decay
   - Validation

Sample Mean and Variance

- ***Point estimation*** is the attempt to estimate some fixed but unknown quantity θ of a random variable by using sample data

Sample Mean and Variance

- ***Point estimation*** is the attempt to estimate some fixed but unknown quantity θ of a random variable by using sample data
- Let $\mathbb{X} = \{x^{(1)}, \cdots, x^{(n)}\}$ be a set of n i.i.d. samples of a random variable x, a ***point estimator*** or ***statistic*** is a function of the data:

$$\hat{\theta}_n = g(x^{(1)}, \cdots, x^{(n)})$$

  - The value $\hat{\theta}_n$ is called the ***estimate*** of θ

Sample Mean and Variance

- ***Point estimation*** is the attempt to estimate some fixed but unknown quantity θ of a random variable by using sample data
- Let $\mathbb{X} = \{x^{(1)}, \cdots, x^{(n)}\}$ be a set of n i.i.d. samples of a random variable x, a ***point estimator*** or ***statistic*** is a function of the data:

$$\hat{\theta}_n = g(x^{(1)}, \cdots, x^{(n)})$$

  - The value $\hat{\theta}_n$ is called the ***estimate*** of θ
- Sample mean: $\hat{\mu}_{\mathrm{x}} = \frac{1}{n}\sum_i x^{(i)}$
- Sample variance: $\hat{\sigma}_{\mathrm{x}} = \frac{1}{n}\sum_i (x^{(i)} - \hat{\mu}_{\mathrm{x}})^2$

Sample Mean and Variance

- ***Point estimation*** is the attempt to estimate some fixed but unknown quantity θ of a random variable by using sample data
- Let $\mathbb{X} = \{x^{(1)}, \cdots, x^{(n)}\}$ be a set of n i.i.d. samples of a random variable x, a ***point estimator*** or ***statistic*** is a function of the data:

$$\hat{\theta}_n = g(x^{(1)}, \cdots, x^{(n)})$$

  - The value $\hat{\theta}_n$ is called the ***estimate*** of θ
- Sample mean: $\hat{\mu}_{\mathrm{x}} = \frac{1}{n}\sum_i x^{(i)}$
- Sample variance: $\hat{\sigma}_{\mathrm{x}} = \frac{1}{n}\sum_i (x^{(i)} - \hat{\mu}_{\mathrm{x}})^2$
- How good are these estimators?

Bias & Variance

- ***Bias*** of an estimator:

$$\text{bias}(\hat{\theta}_n) = \text{E}_{\mathbb{X}}(\hat{\theta}_n) - \theta$$

  - Here, the expectation is defined over ***all possible $\mathbb{X}$'s of size n***, i.e., $\text{E}_{\mathbb{X}}(\hat{\theta}_n) = \int \hat{\theta}_n d\text{P}(\mathbb{X})$
- We call a statistic ***unbiased estimator*** iff it has zero bias

Bias & Variance

- ***Bias*** of an estimator:

$$\text{bias}(\hat{\theta}_n) = \text{E}_{\mathbb{X}}(\hat{\theta}_n) - \theta$$

  - Here, the expectation is defined over ***all possible $\mathbb{X}$'s of size n***, i.e., $\text{E}_{\mathbb{X}}(\hat{\theta}_n) = \int \hat{\theta}_n d\text{P}(\mathbb{X})$
- We call a statistic ***unbiased estimator*** iff it has zero bias
- ***Variance*** of an estimator:

$$\text{Var}_{\mathbb{X}}(\hat{\theta}_n) = \text{E}_{\mathbb{X}}\left[\left(\hat{\theta}_n - \text{E}_{\mathbb{X}}[\hat{\theta}_n]\right)^2\right]$$

Bias & Variance

- ***Bias*** of an estimator:

$$\text{bias}(\hat{\theta}_n) = \mathrm{E}_{\mathbb{X}}(\hat{\theta}_n) - \theta$$

  - Here, the expectation is defined over ***all possible $\mathbb{X}$'s of size n***, i.e., $\mathrm{E}_{\mathbb{X}}(\hat{\theta}_n) = \int \hat{\theta}_n d\mathrm{P}(\mathbb{X})$
- We call a statistic ***unbiased estimator*** iff it has zero bias
- ***Variance*** of an estimator:

$$\mathrm{Var}_{\mathbb{X}}(\hat{\theta}_n) = \mathrm{E}_{\mathbb{X}}\left[\left(\hat{\theta}_n - \mathrm{E}_{\mathbb{X}}[\hat{\theta}_n]\right)^2\right]$$

- Is $\hat{\mu}_{\mathrm{x}} = \frac{1}{n}\sum_i x^{(i)}$ an unbiased estimator of μ_{x}?

Bias & Variance

- ***Bias*** of an estimator:

$$\text{bias}(\hat{\theta}_n) = \mathrm{E}_{\mathbb{X}}(\hat{\theta}_n) - \theta$$

  - Here, the expectation is defined over ***all possible $\mathbb{X}$'s of size n***, i.e., $\mathrm{E}_{\mathbb{X}}(\hat{\theta}_n) = \int \hat{\theta}_n d\mathrm{P}(\mathbb{X})$
- We call a statistic ***unbiased estimator*** iff it has zero bias
- ***Variance*** of an estimator:

$$\mathrm{Var}_{\mathbb{X}}(\hat{\theta}_n) = \mathrm{E}_{\mathbb{X}}\left[\left(\hat{\theta}_n - \mathrm{E}_{\mathbb{X}}[\hat{\theta}_n]\right)^2\right]$$

- Is $\hat{\mu}_{\mathrm{x}} = \frac{1}{n}\sum_i x^{(i)}$ an unbiased estimator of μ_{x}? Yes [Homework]

Bias & Variance

- ***Bias*** of an estimator:

$$\text{bias}(\hat{\theta}_n) = \mathrm{E}_{\mathbb{X}}(\hat{\theta}_n) - \theta$$

  - Here, the expectation is defined over ***all possible*** $\mathbb{X}$***'s of size*** n, i.e., $\mathrm{E}_{\mathbb{X}}(\hat{\theta}_n) = \int \hat{\theta}_n d\mathrm{P}(\mathbb{X})$
- We call a statistic ***unbiased estimator*** iff it has zero bias
- ***Variance*** of an estimator:

$$\mathrm{Var}_{\mathbb{X}}(\hat{\theta}_n) = \mathrm{E}_{\mathbb{X}}\left[\left(\hat{\theta}_n - \mathrm{E}_{\mathbb{X}}[\hat{\theta}_n]\right)^2\right]$$

- Is $\hat{\mu}_{\mathrm{x}} = \frac{1}{n}\sum_i x^{(i)}$ an unbiased estimator of μ_{x}? Yes [Homework]
- What much is $\mathrm{Var}_{\mathbb{X}}(\hat{\mu}_{\mathrm{x}})$?

Variance of $\hat{\mu}_{\mathrm{X}}$

$$\mathrm{Var}_{\mathbb{X}}(\hat{\mu}) \quad = \mathrm{E}_{\mathbb{X}}[(\hat{\mu} - \mathrm{E}_{\mathbb{X}}[\hat{\mu}])^2] = \mathrm{E}[\hat{\mu}^2 - 2\hat{\mu}\mu + \mu^2] = \mathrm{E}[\hat{\mu}^2] - \mu^2$$

Variance of $\hat{\mu}_{\mathrm{X}}$

$$\begin{aligned}\mathrm{Var}_{\mathbb{X}}(\hat{\mu}) &= \mathrm{E}_{\mathbb{X}}[(\hat{\mu} - \mathrm{E}_{\mathbb{X}}[\hat{\mu}])^2] = \mathrm{E}[\hat{\mu}^2 - 2\hat{\mu}\mu + \mu^2] = \mathrm{E}[\hat{\mu}^2] - \mu^2 \\ &= \mathrm{E}[\frac{1}{n^2}\sum_{i,j} x^{(i)}x^{(j)}] - \mu^2 = \frac{1}{n^2}\sum_{i,j}\mathrm{E}[x^{(i)}x^{(j)}] - \mu^2\end{aligned}$$

Variance of $\hat{\mu}_{\mathrm{X}}$

$$
\begin{aligned}
\mathrm{Var}_{\mathbb{X}}(\hat{\mu}) &= \mathrm{E}_{\mathbb{X}}[(\hat{\mu} - \mathrm{E}_{\mathbb{X}}[\hat{\mu}])^2] = \mathrm{E}[\hat{\mu}^2 - 2\hat{\mu}\mu + \mu^2] = \mathrm{E}[\hat{\mu}^2] - \mu^2 \\
&= \mathrm{E}[\tfrac{1}{n^2}\textstyle\sum_{i,j} x^{(i)}x^{(j)}] - \mu^2 = \tfrac{1}{n^2}\textstyle\sum_{i,j}\mathrm{E}[x^{(i)}x^{(j)}] - \mu^2 \\
&= \tfrac{1}{n^2}\left(\textstyle\sum_{i=j}\mathrm{E}[x^{(i)}x^{(j)}] + \textstyle\sum_{i\neq j}\mathrm{E}[x^{(i)}x^{(j)}]\right) - \mu^2
\end{aligned}
$$

Variance of $\hat{\mu}_{\mathrm{X}}$

$$
\begin{aligned}
\mathrm{Var}_{\mathbb{X}}(\hat{\mu}) &= \mathrm{E}_{\mathbb{X}}[(\hat{\mu} - \mathrm{E}_{\mathbb{X}}[\hat{\mu}])^2] = \mathrm{E}[\hat{\mu}^2 - 2\hat{\mu}\mu + \mu^2] = \mathrm{E}[\hat{\mu}^2] - \mu^2 \\
&= \mathrm{E}[\frac{1}{n^2}\sum_{i,j} x^{(i)}x^{(j)}] - \mu^2 = \frac{1}{n^2}\sum_{i,j}\mathrm{E}[x^{(i)}x^{(j)}] - \mu^2 \\
&= \frac{1}{n^2}\left(\sum_{i=j}\mathrm{E}[x^{(i)}x^{(j)}] + \sum_{i\neq j}\mathrm{E}[x^{(i)}x^{(j)}]\right) - \mu^2 \\
&= \frac{1}{n^2}\left(\sum_{i}\mathrm{E}[x^{(i)2}] + n(n-1)\mathrm{E}[x^{(i)}]\mathrm{E}[x^{(j)}]\right) - \mu^2
\end{aligned}
$$

Variance of $\hat{\mu}_{\mathrm{x}}$

$$
\begin{aligned}
\mathrm{Var}_{\mathbb{X}}(\hat{\mu}) &= \mathrm{E}_{\mathbb{X}}[(\hat{\mu}-\mathrm{E}_{\mathbb{X}}[\hat{\mu}])^2] = \mathrm{E}[\hat{\mu}^2 - 2\hat{\mu}\mu + \mu^2] = \mathrm{E}[\hat{\mu}^2] - \mu^2 \\
&= \mathrm{E}[\tfrac{1}{n^2}\textstyle\sum_{i,j} x^{(i)}x^{(j)}] - \mu^2 = \tfrac{1}{n^2}\textstyle\sum_{i,j}\mathrm{E}[x^{(i)}x^{(j)}] - \mu^2 \\
&= \tfrac{1}{n^2}\left(\textstyle\sum_{i=j}\mathrm{E}[x^{(i)}x^{(j)}] + \textstyle\sum_{i\neq j}\mathrm{E}[x^{(i)}x^{(j)}]\right) - \mu^2 \\
&= \tfrac{1}{n^2}\left(\textstyle\sum_{i}\mathrm{E}[x^{(i)2}] + n(n-1)\mathrm{E}[x^{(i)}]\mathrm{E}[x^{(j)}]\right) - \mu^2 \\
&= \tfrac{1}{n}\mathrm{E}[\mathrm{x}^2] + \tfrac{(n-1)}{n}\mu^2 - \mu^2 = \tfrac{1}{n}\left(\mathrm{E}[\mathrm{x}^2]-\mu^2\right) = \tfrac{1}{n}\sigma_{\mathrm{x}}^2
\end{aligned}
$$

Variance of $\hat{\mu}_{\mathrm{x}}$

$$
\begin{aligned}
\mathrm{Var}_{\mathbb{X}}(\hat{\mu}) &= \mathrm{E}_{\mathbb{X}}[(\hat{\mu} - \mathrm{E}_{\mathbb{X}}[\hat{\mu}])^2] = \mathrm{E}[\hat{\mu}^2 - 2\hat{\mu}\mu + \mu^2] = \mathrm{E}[\hat{\mu}^2] - \mu^2 \\
&= \mathrm{E}[\tfrac{1}{n^2}\textstyle\sum_{i,j} x^{(i)}x^{(j)}] - \mu^2 = \tfrac{1}{n^2}\textstyle\sum_{i,j}\mathrm{E}[x^{(i)}x^{(j)}] - \mu^2 \\
&= \tfrac{1}{n^2}\left(\textstyle\sum_{i=j}\mathrm{E}[x^{(i)}x^{(j)}] + \textstyle\sum_{i\neq j}\mathrm{E}[x^{(i)}x^{(j)}]\right) - \mu^2 \\
&= \tfrac{1}{n^2}\left(\textstyle\sum_{i}\mathrm{E}[x^{(i)2}] + n(n-1)\mathrm{E}[x^{(i)}]\mathrm{E}[x^{(j)}]\right) - \mu^2 \\
&= \tfrac{1}{n}\mathrm{E}[\mathrm{x}^2] + \tfrac{(n-1)}{n}\mu^2 - \mu^2 = \tfrac{1}{n}\left(\mathrm{E}[\mathrm{x}^2] - \mu^2\right) = \tfrac{1}{n}\sigma_{\mathrm{x}}^2
\end{aligned}
$$

- The variance of $\hat{\mu}_{\mathrm{x}}$ diminishes as $n \to \infty$

Unbiased Estimator of $\sigma_{\mathbf{x}}$

- Is $\hat{\sigma}_{\mathrm{x}} = \frac{1}{n}\sum_i (x^{(i)} - \hat{\mu}_{\mathrm{x}})^2$ and an unbiased estimator of σ_{x}?

Unbiased Estimator of σ_{x}

- Is $\hat{\sigma}_{\mathrm{x}} = \frac{1}{n}\sum_i (x^{(i)} - \hat{\mu}_{\mathrm{x}})^2$ and an unbiased estimator of σ_{x}? ***No***

  $$\mathrm{E}_{\mathbb{X}}[\hat{\sigma}] \quad = \mathrm{E}[\frac{1}{n}\sum_i (x^{(i)} - \hat{\mu})^2] = \mathrm{E}[\frac{1}{n}(\sum_i x^{(i)2} - 2\sum_i x^{(i)}\hat{\mu} + \sum_i \hat{\mu}^2)]$$

Unbiased Estimator of $\sigma_{\mathbf{x}}$

- Is $\hat{\sigma}_{\mathrm{x}} = \frac{1}{n}\sum_i (x^{(i)} - \hat{\mu}_{\mathrm{x}})^2$ and an unbiased estimator of σ_{x}? ***No***

$$
\begin{aligned}
\mathrm{E}_{\mathbb{X}}[\hat{\sigma}] &= \mathrm{E}[\tfrac{1}{n}\sum_i (x^{(i)} - \hat{\mu})^2] = \mathrm{E}[\tfrac{1}{n}(\sum_i x^{(i)2} - 2\sum_i x^{(i)}\hat{\mu} + \sum_i \hat{\mu}^2)] \\
&= \mathrm{E}[\tfrac{1}{n}(\sum_i x^{(i)2} - n\hat{\mu}^2)] = \tfrac{1}{n}(\sum_i \mathrm{E}[x^{(i)2}] - n\mathrm{E}[\hat{\mu}^2])
\end{aligned}
$$

Unbiased Estimator of $\sigma_{\mathbf{x}}$

- Is $\hat{\sigma}_{\mathrm{x}} = \frac{1}{n}\sum_i (x^{(i)} - \hat{\mu}_{\mathrm{x}})^2$ and an unbiased estimator of σ_{x}? ***No***

$$\begin{aligned}
\mathrm{E}_{\mathbb{X}}[\hat{\sigma}] &= \mathrm{E}[\tfrac{1}{n}\sum_i (x^{(i)} - \hat{\mu})^2] = \mathrm{E}[\tfrac{1}{n}(\sum_i x^{(i)2} - 2\sum_i x^{(i)}\hat{\mu} + \sum_i \hat{\mu}^2)] \\
&= \mathrm{E}[\tfrac{1}{n}(\sum_i x^{(i)2} - n\hat{\mu}^2)] = \tfrac{1}{n}(\sum_i \mathrm{E}[x^{(i)2}] - n\mathrm{E}[\hat{\mu}^2]) \\
&= \mathrm{E}[\mathrm{x}^2] - \mathrm{E}[\hat{\mu}^2] = \mathrm{E}[(\mathrm{x} - \mu)^2 + 2\mathrm{x}\mu - \mu^2] - \mathrm{E}[\hat{\mu}^2] \\
&= (\sigma^2 + \mu^2) - (\mathrm{Var}[\hat{\mu}] + \mathrm{E}[\hat{\mu}]^2)
\end{aligned}$$

Unbiased Estimator of σ_{x}

- Is $\hat{\sigma}_{\mathrm{x}} = \frac{1}{n}\sum_i (x^{(i)} - \hat{\mu}_{\mathrm{x}})^2$ and an unbiased estimator of σ_{x}? ***No***

$$
\begin{aligned}
\mathrm{E}_{\mathbb{X}}[\hat{\sigma}] &= \mathrm{E}[\tfrac{1}{n}\sum_i (x^{(i)} - \hat{\mu})^2] = \mathrm{E}[\tfrac{1}{n}(\sum_i x^{(i)2} - 2\sum_i x^{(i)}\hat{\mu} + \sum_i \hat{\mu}^2)] \\
&= \mathrm{E}[\tfrac{1}{n}(\sum_i x^{(i)2} - n\hat{\mu}^2)] = \tfrac{1}{n}(\sum_i \mathrm{E}[x^{(i)2}] - n\mathrm{E}[\hat{\mu}^2]) \\
&= \mathrm{E}[\mathrm{x}^2] - \mathrm{E}[\hat{\mu}^2] = \mathrm{E}[(\mathrm{x} - \mu)^2 + 2\mathrm{x}\mu - \mu^2] - \mathrm{E}[\hat{\mu}^2] \\
&= (\sigma^2 + \mu^2) - (\mathrm{Var}[\hat{\mu}] + \mathrm{E}[\hat{\mu}]^2) \\
&= \sigma^2 + \mu^2 - \tfrac{1}{n}\sigma^2 - \mu^2 = \tfrac{n-1}{n}\sigma^2 \neq \sigma^2
\end{aligned}
$$

Unbiased Estimator of σ_{x}

- Is $\hat{\sigma}_{\mathrm{x}} = \frac{1}{n}\sum_i (x^{(i)} - \hat{\mu}_{\mathrm{x}})^2$ and an unbiased estimator of σ_{x}? ***No***

$$
\begin{aligned}
\mathrm{E}_{\mathbb{X}}[\hat{\sigma}] &= \mathrm{E}[\tfrac{1}{n}\sum_i (x^{(i)} - \hat{\mu})^2] = \mathrm{E}[\tfrac{1}{n}(\sum_i x^{(i)2} - 2\sum_i x^{(i)}\hat{\mu} + \sum_i \hat{\mu}^2)] \\
&= \mathrm{E}[\tfrac{1}{n}(\sum_i x^{(i)2} - n\hat{\mu}^2)] = \tfrac{1}{n}(\sum_i \mathrm{E}[x^{(i)2}] - n\mathrm{E}[\hat{\mu}^2]) \\
&= \mathrm{E}[\mathrm{x}^2] - \mathrm{E}[\hat{\mu}^2] = \mathrm{E}[(\mathrm{x} - \mu)^2 + 2\mathrm{x}\mu - \mu^2] - \mathrm{E}[\hat{\mu}^2] \\
&= (\sigma^2 + \mu^2) - (\mathrm{Var}[\hat{\mu}] + \mathrm{E}[\hat{\mu}]^2) \\
&= \sigma^2 + \mu^2 - \tfrac{1}{n}\sigma^2 - \mu^2 = \tfrac{n-1}{n}\sigma^2 \neq \sigma^2
\end{aligned}
$$

- What's the unbiased estimator of σ_{x}?

Unbiased Estimator of σ_{x}

- Is $\hat{\sigma}_{\mathrm{x}} = \frac{1}{n}\sum_i (x^{(i)} - \hat{\mu}_{\mathrm{x}})^2$ and an unbiased estimator of σ_{x}? ***No***

$$
\begin{aligned}
\mathrm{E}_{\mathbb{X}}[\hat{\sigma}] &= \mathrm{E}[\tfrac{1}{n}\sum_i (x^{(i)} - \hat{\mu})^2] = \mathrm{E}[\tfrac{1}{n}(\sum_i x^{(i)2} - 2\sum_i x^{(i)}\hat{\mu} + \sum_i \hat{\mu}^2)] \\
&= \mathrm{E}[\tfrac{1}{n}(\sum_i x^{(i)2} - n\hat{\mu}^2)] = \tfrac{1}{n}(\sum_i \mathrm{E}[x^{(i)2}] - n\mathrm{E}[\hat{\mu}^2]) \\
&= \mathrm{E}[\mathrm{x}^2] - \mathrm{E}[\hat{\mu}^2] = \mathrm{E}[(\mathrm{x}-\mu)^2 + 2\mathrm{x}\mu - \mu^2] - \mathrm{E}[\hat{\mu}^2] \\
&= (\sigma^2 + \mu^2) - (\mathrm{Var}[\hat{\mu}] + \mathrm{E}[\hat{\mu}]^2) \\
&= \sigma^2 + \mu^2 - \tfrac{1}{n}\sigma^2 - \mu^2 = \tfrac{n-1}{n}\sigma^2 \neq \sigma^2
\end{aligned}
$$

- What's the unbiased estimator of σ_{x}?

$$
\hat{\sigma}_{\mathrm{x}} = \frac{n}{n-1}\left(\frac{1}{n}\sum_i (x^{(i)} - \hat{\mu}_{\mathrm{x}})^2\right) = \frac{1}{n-1}\sum_i (x^{(i)} - \hat{\mu}_{\mathrm{x}})^2
$$

Mean Square Error

- ***Mean square error*** of an estimator:

$$\mathrm{MSE}(\hat{\theta}_n) = \mathrm{E}_{\mathbb{X}}\left[(\hat{\theta}_n - \theta)^2\right]$$

Mean Square Error

- ***Mean square error*** of an estimator:

$$\mathrm{MSE}(\hat{\theta}_n) = \mathrm{E}_{\mathbb{X}}\left[(\hat{\theta}_n - \theta)^2\right]$$

- Can be decomposed into the bias and variance:

$$\mathrm{E}_{\mathbb{X}}\left[(\hat{\theta}_n - \theta)^2\right] = \mathrm{E}\left[(\hat{\theta}_n - \mathrm{E}[\hat{\theta}_n] - \mathrm{E}[\hat{\theta}_n] + \theta)^2\right]$$

Mean Square Error

- *Mean square error* of an estimator:

$$\mathrm{MSE}(\hat{\theta}_n) = \mathrm{E}_{\mathbb{X}}\left[(\hat{\theta}_n - \theta)^2\right]$$

- Can be decomposed into the bias and variance:

$$\begin{aligned}\mathrm{E}_{\mathbb{X}}\left[(\hat{\theta}_n - \theta)^2\right] &= \mathrm{E}\left[(\hat{\theta}_n - \mathrm{E}[\hat{\theta}_n] - \mathrm{E}[\hat{\theta}_n] + \theta)^2\right] \\ &= \mathrm{E}\left[(\hat{\theta}_n - \mathrm{E}[\hat{\theta}_n])^2 + (\mathrm{E}[\hat{\theta}_n] - \theta)^2 + 2(\hat{\theta}_n - \mathrm{E}[\hat{\theta}_n])(\mathrm{E}[\hat{\theta}_n] - \theta)\right]\end{aligned}$$

Mean Square Error

- *Mean square error* of an estimator:

$$\mathrm{MSE}(\hat{\theta}_n) = \mathrm{E}_{\mathbb{X}}\left[(\hat{\theta}_n - \theta)^2\right]$$

- Can be decomposed into the bias and variance:

$$\begin{aligned}\mathrm{E}_{\mathbb{X}}\left[(\hat{\theta}_n - \theta)^2\right] &= \mathrm{E}\left[(\hat{\theta}_n - \mathrm{E}[\hat{\theta}_n] - \mathrm{E}[\hat{\theta}_n] + \theta)^2\right] \\ &= \mathrm{E}\left[(\hat{\theta}_n - \mathrm{E}[\hat{\theta}_n])^2 + (\mathrm{E}[\hat{\theta}_n] - \theta)^2 + 2(\hat{\theta}_n - \mathrm{E}[\hat{\theta}_n])(\mathrm{E}[\hat{\theta}_n] - \theta)\right] \\ &= \mathrm{E}\left[(\hat{\theta}_n - \mathrm{E}[\hat{\theta}_n])^2\right] + \mathrm{E}\left[(\mathrm{E}[\hat{\theta}_n] - \theta)^2\right] + 2\mathrm{E}\left(\hat{\theta}_n - \mathrm{E}[\hat{\theta}_n]\right)(\mathrm{E}[\hat{\theta}_n] - \theta)\end{aligned}$$

Mean Square Error

- *Mean square error* of an estimator:

$$\text{MSE}(\hat{\theta}_n) = \text{E}_{\mathbb{X}}\left[(\hat{\theta}_n - \theta)^2\right]$$

- Can be decomposed into the bias and variance:

$$\begin{aligned}
\text{E}_{\mathbb{X}}\left[(\hat{\theta}_n - \theta)^2\right] &= \text{E}\left[(\hat{\theta}_n - \text{E}[\hat{\theta}_n] - \text{E}[\hat{\theta}_n] + \theta)^2\right] \\
&= \text{E}\left[(\hat{\theta}_n - \text{E}[\hat{\theta}_n])^2 + (\text{E}[\hat{\theta}_n] - \theta)^2 + 2(\hat{\theta}_n - \text{E}[\hat{\theta}_n])(\text{E}[\hat{\theta}_n] - \theta)\right] \\
&= \text{E}\left[(\hat{\theta}_n - \text{E}[\hat{\theta}_n])^2\right] + \text{E}\left[(\text{E}[\hat{\theta}_n] - \theta)^2\right] + 2\text{E}\left(\hat{\theta}_n - \text{E}[\hat{\theta}_n]\right)(\text{E}[\hat{\theta}_n] - \theta) \\
&= \text{E}\left[(\hat{\theta}_n - \text{E}[\hat{\theta}_n])^2\right] + \left(\text{E}[\hat{\theta}_n] - \theta\right)^2 + 2 \cdot 0 \cdot (\text{E}[\hat{\theta}_n] - \theta)
\end{aligned}$$

Mean Square Error

- *Mean square error* of an estimator:

$$\mathrm{MSE}(\hat{\theta}_n) = \mathrm{E}_{\mathbb{X}}\left[(\hat{\theta}_n - \theta)^2\right]$$

- Can be decomposed into the bias and variance:

$$\begin{aligned}
\mathrm{E}_{\mathbb{X}}\left[(\hat{\theta}_n - \theta)^2\right] &= \mathrm{E}\left[(\hat{\theta}_n - \mathrm{E}[\hat{\theta}_n] - \mathrm{E}[\hat{\theta}_n] + \theta)^2\right] \\
&= \mathrm{E}\left[(\hat{\theta}_n - \mathrm{E}[\hat{\theta}_n])^2 + (\mathrm{E}[\hat{\theta}_n] - \theta)^2 + 2(\hat{\theta}_n - \mathrm{E}[\hat{\theta}_n])(\mathrm{E}[\hat{\theta}_n] - \theta)\right] \\
&= \mathrm{E}\left[(\hat{\theta}_n - \mathrm{E}[\hat{\theta}_n])^2\right] + \mathrm{E}\left[(\mathrm{E}[\hat{\theta}_n] - \theta)^2\right] + 2\mathrm{E}\left(\hat{\theta}_n - \mathrm{E}[\hat{\theta}_n]\right)(\mathrm{E}[\hat{\theta}_n] - \theta) \\
&= \mathrm{E}\left[(\hat{\theta}_n - \mathrm{E}[\hat{\theta}_n])^2\right] + \left(\mathrm{E}[\hat{\theta}_n] - \theta\right)^2 + 2 \cdot 0 \cdot (\mathrm{E}[\hat{\theta}_n] - \theta) \\
&= \mathrm{Var}_{\mathbb{X}}(\hat{\theta}_n) + \mathrm{bias}(\hat{\theta}_n)^2
\end{aligned}$$

- MSE of an unbiased estimator is its variance

Outline

1. Learning Theory
2. Point Estimation: Bias and Variance
   - Consistency*
3. Decomposing Generalization Error
4. Regularization
   - Weight Decay
   - Validation

Consistency

- So far, we discussed the "goodness" of an estimator based on samples of fixed size

Consistency

- So far, we discussed the "goodness" of an estimator based on samples of fixed size
- If we have more samples, will the estimate become more accurate?

Consistency

- So far, we discussed the "goodness" of an estimator based on samples of fixed size
- If we have more samples, will the estimate become more accurate?
- An estimator is (weak) ***consistent*** iff:

$$\lim_{n\to\infty} \hat{\theta}_n \xrightarrow{\Pr} \theta,$$

where $\xrightarrow{\Pr}$ means "converge in probability"

Consistency

- So far, we discussed the "goodness" of an estimator based on samples of fixed size
- If we have more samples, will the estimate become more accurate?
- An estimator is (weak) ***consistent*** iff:

$$\lim_{n\to\infty} \hat{\theta}_n \xrightarrow{\Pr} \theta,$$

  where $\xrightarrow{\Pr}$ means "converge in probability"
- Strong consistent iff "converge almost surely"

Law of Large Numbers

Theorem (Weak Law of Large Numbers)

The sample mean $\hat{\mu}_{\mathrm{x}} = \frac{1}{n}\sum_i x^{(i)}$ *is a consistent estimator of* μ_{x}*, i.e.,* $\lim_{n\to\infty} \Pr(|\hat{\mu}_{\mathrm{x},n} - \mu_{\mathrm{x}}| < \varepsilon) = 1$ *for any* $\varepsilon > 0$*.*

Law of Large Numbers

Theorem (Weak Law of Large Numbers)

The sample mean $\hat{\mu}_{\mathrm{x}} = \frac{1}{n}\sum_i x^{(i)}$ *is a consistent estimator of* μ_{x}*, i.e.,* $\lim_{n\to\infty} \Pr(|\hat{\mu}_{\mathrm{x},n} - \mu_{\mathrm{x}}| < \varepsilon) = 1$ *for any* $\varepsilon > 0$*.*

Theorem (Strong Law of Large Numbers)

In addition, $\hat{\mu}_{\mathrm{x}}$ *is a strong consistent estimator:* $\Pr(\lim_{n\to\infty} \hat{\mu}_{\mathrm{x},n} = \mu_{\mathrm{x}}) = 1$*.*

Outline

1 Learning Theory

2 Point Estimation: Bias and Variance
- Consistency*

3 Decomposing Generalization Error

4 Regularization
- Weight Decay
- Validation

Expected Generalization Error

- In ML, we get $f_N = \arg\min_{f\in\mathbb{F}} C_N[f]$ by minimizing the empirical error over a training set of size N
- How to decompose the generalization error $C[f_N]$?

Expected Generalization Error

- In ML, we get $f_N = \arg\min_{f \in \mathbb{F}} C_N[f]$ by minimizing the empirical error over a training set of size N
- How to decompose the generalization error $C[f_N]$?
- Regard $f_N(\boldsymbol{x})$ as an estimate of true label y given $\boldsymbol{x}$
  - f_N an estimator mapped from i.i.d. samples in the training set $\mathbb{X}$
- To evaluate the estimator f_N, we consider the expected generalization error:

$$\mathrm{E}_{\mathbb{X}}\left(C[f_N]\right) = \mathrm{E}_{\mathbb{X}}\left[\int \mathrm{loss}(f_N(\boldsymbol{x}) - y) d\mathrm{P}(\boldsymbol{x}, y)\right]$$

Expected Generalization Error

- In ML, we get $f_N = \arg\min_{f \in \mathbb{F}} C_N[f]$ by minimizing the empirical error over a training set of size N
- How to decompose the generalization error $C[f_N]$?
- Regard $f_N(\boldsymbol{x})$ as an estimate of true label y given $\boldsymbol{x}$
  - f_N an estimator mapped from i.i.d. samples in the training set $\mathbb{X}$
- To evaluate the estimator f_N, we consider the expected generalization error:

$$\begin{aligned} \mathrm{E}_{\mathbb{X}}\left(C[f_N]\right) &= \mathrm{E}_{\mathbb{X}}\left[\int \mathrm{loss}(f_N(\boldsymbol{x}) - y)d\mathrm{P}(\boldsymbol{x}, y)\right] \\ &= \mathrm{E}_{\mathbb{X},\mathbf{x},\mathrm{y}}\left[\mathrm{loss}(f_N(\mathbf{x}) - \mathrm{y})\right] \end{aligned}$$

Expected Generalization Error

- In ML, we get $f_N = \arg\min_{f \in \mathbb{F}} C_N[f]$ by minimizing the empirical error over a training set of size N
- How to decompose the generalization error $C[f_N]$?
- Regard $f_N(\boldsymbol{x})$ as an estimate of true label y given $\boldsymbol{x}$
  - f_N an estimator mapped from i.i.d. samples in the training set $\mathbb{X}$
- To evaluate the estimator f_N, we consider the expected generalization error:

$$\begin{aligned} \mathrm{E}_{\mathbb{X}}\left(C[f_N]\right) &= \mathrm{E}_{\mathbb{X}}\left[\int \mathrm{loss}(f_N(\boldsymbol{x}) - y) d\mathrm{P}(\boldsymbol{x}, y)\right] \\ &= \mathrm{E}_{\mathbb{X},\mathbf{x},\mathrm{y}}\left[\mathrm{loss}(f_N(\mathbf{x}) - \mathrm{y})\right] \\ &= \mathrm{E}_{\mathbf{x}}\left(\mathrm{E}_{\mathbb{X},\mathrm{y}}\left[\mathrm{loss}(f_N(\boldsymbol{x}) - \mathrm{y}) | \mathbf{x} = \boldsymbol{x}\right]\right) \end{aligned}$$

Expected Generalization Error

- In ML, we get $f_N = \arg\min_{f \in \mathbb{F}} C_N[f]$ by minimizing the empirical error over a training set of size N
- How to decompose the generalization error $C[f_N]$?
- Regard $f_N(\boldsymbol{x})$ as an estimate of true label y given $\boldsymbol{x}$
  - f_N an estimator mapped from i.i.d. samples in the training set $\mathbb{X}$
- To evaluate the estimator f_N, we consider the expected generalization error:

$$\begin{aligned} \mathrm{E}_{\mathbb{X}}\left(C[f_N]\right) &= \mathrm{E}_{\mathbb{X}}\left[\int \mathrm{loss}(f_N(\boldsymbol{x}) - y) d\mathrm{P}(\boldsymbol{x}, y)\right] \\ &= \mathrm{E}_{\mathbb{X},\mathbf{x},\mathrm{y}}\left[\mathrm{loss}(f_N(\mathbf{x}) - \mathrm{y})\right] \\ &= \mathrm{E}_{\mathbf{x}}\left(\mathrm{E}_{\mathbb{X},\mathrm{y}}\left[\mathrm{loss}(f_N(\boldsymbol{x}) - \mathrm{y}) | \mathbf{x} = \boldsymbol{x}\right]\right) \end{aligned}$$

- There's a simple decomposition of $\mathrm{E}_{\mathbb{X},\mathrm{y}}\left[\mathrm{loss}(f_N(\boldsymbol{x}) - \mathrm{y}) | \boldsymbol{x}\right]$ for linear/polynomial regression

Example: Linear/Polynomial Regression

- In linear/polynomial regression, we have
  - $\text{loss}(\cdot) = (\cdot)^2$ a squared loss
  - $y = f^*(\boldsymbol{x}) + \varepsilon$, $\varepsilon \sim \mathcal{N}(0, \sigma^2)$, thus $\mathrm{E}_{\mathrm{y}}[\mathrm{y}|\boldsymbol{x}] = f^*(\boldsymbol{x})$ and $\mathrm{Var}_{\mathrm{y}}[\mathrm{y}|\boldsymbol{x}] = \sigma^2$

Example: Linear/Polynomial Regression

- In linear/polynomial regression, we have
  - $\mathrm{loss}(\cdot) = (\cdot)^2$ a squared loss
  - $y = f^*(\boldsymbol{x}) + \varepsilon$, $\varepsilon \sim \mathcal{N}(0, \sigma^2)$, thus $\mathrm{E}_{\mathrm{y}}[\mathrm{y}|\boldsymbol{x}] = f^*(\boldsymbol{x})$ and $\mathrm{Var}_{\mathrm{y}}[\mathrm{y}|\boldsymbol{x}] = \sigma^2$
- We can decompose the mean square error:

$$\begin{aligned} &\mathrm{E}_{\mathbb{X},\mathrm{y}}[\mathrm{loss}(f_N(\boldsymbol{x}) - \mathrm{y})|\boldsymbol{x}] = \mathrm{E}_{\mathbb{X},\mathrm{y}}[(f_N(\boldsymbol{x}) - \mathrm{y})^2|\boldsymbol{x}] \\ &\quad = \mathrm{E}_{\mathbb{X},\mathrm{y}}[\mathrm{y}^2 + f_N(\boldsymbol{x})^2 - 2f_N(\boldsymbol{x})\mathrm{y}|\boldsymbol{x}] \end{aligned}$$

Example: Linear/Polynomial Regression

- In linear/polynomial regression, we have
  - $\text{loss}(\cdot) = (\cdot)^2$ a squared loss
  - $y = f^*(\boldsymbol{x}) + \varepsilon$, $\varepsilon \sim \mathcal{N}(0, \sigma^2)$, thus $\mathrm{E}_{\mathrm{y}}[\mathrm{y}|\boldsymbol{x}] = f^*(\boldsymbol{x})$ and $\mathrm{Var}_{\mathrm{y}}[\mathrm{y}|\boldsymbol{x}] = \sigma^2$
- We can decompose the mean square error:

$$
\begin{aligned}
&\mathrm{E}_{\mathbb{X},\mathrm{y}}[\text{loss}(f_N(\boldsymbol{x}) - \mathrm{y})|\boldsymbol{x}] = \mathrm{E}_{\mathbb{X},\mathrm{y}}[(f_N(\boldsymbol{x}) - \mathrm{y})^2|\boldsymbol{x}] \\
&\quad = \mathrm{E}_{\mathbb{X},\mathrm{y}}[\mathrm{y}^2 + f_N(\boldsymbol{x})^2 - 2f_N(\boldsymbol{x})\mathrm{y}|\boldsymbol{x}] \\
&\quad = \mathrm{E}_{\mathrm{y}}[\mathrm{y}^2|\boldsymbol{x}] + \mathrm{E}_{\mathbb{X}}[f_N(\boldsymbol{x})^2|\boldsymbol{x}] - 2\mathrm{E}_{\mathbb{X},\mathrm{y}}[f_N(\boldsymbol{x})\mathrm{y}|\boldsymbol{x}]
\end{aligned}
$$

Example: Linear/Polynomial Regression

- In linear/polynomial regression, we have
  - $\text{loss}(\cdot) = (\cdot)^2$ a squared loss
  - $y = f^*(\boldsymbol{x}) + \varepsilon$, $\varepsilon \sim \mathcal{N}(0, \sigma^2)$, thus $\mathrm{E}_{\mathrm{y}}[\mathrm{y}|\boldsymbol{x}] = f^*(\boldsymbol{x})$ and $\mathrm{Var}_{\mathrm{y}}[\mathrm{y}|\boldsymbol{x}] = \sigma^2$
- We can decompose the mean square error:

$$
\begin{aligned}
&\mathrm{E}_{\mathbb{X},\mathrm{y}}[\text{loss}(f_N(\boldsymbol{x}) - \mathrm{y})|\boldsymbol{x}] = \mathrm{E}_{\mathbb{X},\mathrm{y}}[(f_N(\boldsymbol{x}) - \mathrm{y})^2|\boldsymbol{x}] \\
&\quad = \mathrm{E}_{\mathbb{X},\mathrm{y}}[\mathrm{y}^2 + f_N(\boldsymbol{x})^2 - 2f_N(\boldsymbol{x})\mathrm{y}|\boldsymbol{x}] \\
&\quad = \mathrm{E}_{\mathrm{y}}[\mathrm{y}^2|\boldsymbol{x}] + \mathrm{E}_{\mathbb{X}}[f_N(\boldsymbol{x})^2|\boldsymbol{x}] - 2\mathrm{E}_{\mathbb{X},\mathrm{y}}[f_N(\boldsymbol{x})\mathrm{y}|\boldsymbol{x}] \\
&\quad = (\mathrm{Var}_{\mathrm{y}}[\mathrm{y}|\boldsymbol{x}] + \mathrm{E}_{\mathrm{y}}[\mathrm{y}|\boldsymbol{x}]^2) + (\mathrm{Var}_{\mathbb{X}}[f_N(\boldsymbol{x})|\boldsymbol{x}] + \mathrm{E}_{\mathbb{X}}[f_N(\boldsymbol{x})|\boldsymbol{x}]^2) \\
&\qquad\quad -2\mathrm{E}_{\mathrm{y}}[\mathrm{y}|\boldsymbol{x}]\mathrm{E}_{\mathbb{X}}[f_N(\boldsymbol{x})|\boldsymbol{x}]
\end{aligned}
$$

Example: Linear/Polynomial Regression

- In linear/polynomial regression, we have
  - $\text{loss}(\cdot) = (\cdot)^2$ a squared loss
  - $y = f^*(\boldsymbol{x}) + \varepsilon$, $\varepsilon \sim \mathcal{N}(0, \sigma^2)$, thus $\mathrm{E}_{\mathrm{y}}[\mathrm{y}|\boldsymbol{x}] = f^*(\boldsymbol{x})$ and $\mathrm{Var}_{\mathrm{y}}[\mathrm{y}|\boldsymbol{x}] = \sigma^2$
- We can decompose the mean square error:

$$
\begin{aligned}
&\mathrm{E}_{\mathbb{X},\mathrm{y}}[\text{loss}(f_N(\boldsymbol{x}) - \mathrm{y})|\boldsymbol{x}] = \mathrm{E}_{\mathbb{X},\mathrm{y}}[(f_N(\boldsymbol{x}) - \mathrm{y})^2|\boldsymbol{x}] \\
&\quad = \mathrm{E}_{\mathbb{X},\mathrm{y}}[\mathrm{y}^2 + f_N(\boldsymbol{x})^2 - 2f_N(\boldsymbol{x})\mathrm{y}|\boldsymbol{x}] \\
&\quad = \mathrm{E}_{\mathrm{y}}[\mathrm{y}^2|\boldsymbol{x}] + \mathrm{E}_{\mathbb{X}}[f_N(\boldsymbol{x})^2|\boldsymbol{x}] - 2\mathrm{E}_{\mathbb{X},\mathrm{y}}[f_N(\boldsymbol{x})\mathrm{y}|\boldsymbol{x}] \\
&\quad = (\mathrm{Var}_{\mathrm{y}}[\mathrm{y}|\boldsymbol{x}] + \mathrm{E}_{\mathrm{y}}[\mathrm{y}|\boldsymbol{x}]^2) + (\mathrm{Var}_{\mathbb{X}}[f_N(\boldsymbol{x})|\boldsymbol{x}] + \mathrm{E}_{\mathbb{X}}[f_N(\boldsymbol{x})|\boldsymbol{x}]^2) \\
&\qquad\quad - 2\mathrm{E}_{\mathrm{y}}[\mathrm{y}|\boldsymbol{x}]\mathrm{E}_{\mathbb{X}}[f_N(\boldsymbol{x})|\boldsymbol{x}] \\
&\quad = \mathrm{Var}_{\mathrm{y}}[\mathrm{y}|\boldsymbol{x}] + \mathrm{Var}_{\mathbb{X}}[f_N(\boldsymbol{x})|\boldsymbol{x}] + (\mathrm{E}_{\mathbb{X}}[f_N(\boldsymbol{x})|\boldsymbol{x}] - \mathrm{E}_{\mathrm{y}}[\mathrm{y}|\boldsymbol{x}])^2
\end{aligned}
$$

Example: Linear/Polynomial Regression

- In linear/polynomial regression, we have
  - $\text{loss}(\cdot) = (\cdot)^2$ a squared loss
  - $y = f^*(\boldsymbol{x}) + \varepsilon$, $\varepsilon \sim \mathcal{N}(0, \sigma^2)$, thus $\mathrm{E}_{\mathrm{y}}[\mathrm{y}|\boldsymbol{x}] = f^*(\boldsymbol{x})$ and $\mathrm{Var}_{\mathrm{y}}[\mathrm{y}|\boldsymbol{x}] = \sigma^2$
- We can decompose the mean square error:

$$
\begin{aligned}
\mathrm{E}_{\mathbb{X},\mathrm{y}}\left[\text{loss}(f_N(\boldsymbol{x}) - \mathrm{y})|\boldsymbol{x}\right] &= \mathrm{E}_{\mathbb{X},\mathrm{y}}[(f_N(\boldsymbol{x}) - \mathrm{y})^2|\boldsymbol{x}] \\
&= \mathrm{E}_{\mathbb{X},\mathrm{y}}[\mathrm{y}^2 + f_N(\boldsymbol{x})^2 - 2f_N(\boldsymbol{x})\mathrm{y}|\boldsymbol{x}] \\
&= \mathrm{E}_{\mathrm{y}}[\mathrm{y}^2|\boldsymbol{x}] + \mathrm{E}_{\mathbb{X}}[f_N(\boldsymbol{x})^2|\boldsymbol{x}] - 2\mathrm{E}_{\mathbb{X},\mathrm{y}}[f_N(\boldsymbol{x})\mathrm{y}|\boldsymbol{x}] \\
&= (\mathrm{Var}_{\mathrm{y}}[\mathrm{y}|\boldsymbol{x}] + \mathrm{E}_{\mathrm{y}}[\mathrm{y}|\boldsymbol{x}]^2) + (\mathrm{Var}_{\mathbb{X}}[f_N(\boldsymbol{x})|\boldsymbol{x}] + \mathrm{E}_{\mathbb{X}}[f_N(\boldsymbol{x})|\boldsymbol{x}]^2) \\
&\quad - 2\mathrm{E}_{\mathrm{y}}[\mathrm{y}|\boldsymbol{x}]\mathrm{E}_{\mathbb{X}}[f_N(\boldsymbol{x})|\boldsymbol{x}] \\
&= \mathrm{Var}_{\mathrm{y}}[\mathrm{y}|\boldsymbol{x}] + \mathrm{Var}_{\mathbb{X}}[f_N(\boldsymbol{x})|\boldsymbol{x}] + (\mathrm{E}_{\mathbb{X}}[f_N(\boldsymbol{x})|\boldsymbol{x}] - \mathrm{E}_{\mathrm{y}}[\mathrm{y}|\boldsymbol{x}])^2 \\
&= \mathrm{Var}_{\mathrm{y}}[\mathrm{y}|\boldsymbol{x}] + \mathrm{Var}_{\mathbb{X}}[f_N(\boldsymbol{x})|\boldsymbol{x}] + \mathrm{E}_{\mathbb{X}}[f_N(\boldsymbol{x}) - f^*(\boldsymbol{x})|\boldsymbol{x}]^2
\end{aligned}
$$

Example: Linear/Polynomial Regression

- In linear/polynomial regression, we have
  - $\text{loss}(\cdot) = (\cdot)^2$ a squared loss
  - $y = f^*(\boldsymbol{x}) + \varepsilon$, $\varepsilon \sim \mathcal{N}(0, \sigma^2)$, thus $\mathrm{E}_{\mathrm{y}}[\mathrm{y}|\boldsymbol{x}] = f^*(\boldsymbol{x})$ and $\mathrm{Var}_{\mathrm{y}}[\mathrm{y}|\boldsymbol{x}] = \sigma^2$
- We can decompose the mean square error:

$$
\begin{aligned}
&\mathrm{E}_{\mathbb{X},\mathrm{y}}[\text{loss}(f_N(\boldsymbol{x}) - \mathrm{y})|\boldsymbol{x}] = \mathrm{E}_{\mathbb{X},\mathrm{y}}[(f_N(\boldsymbol{x}) - \mathrm{y})^2|\boldsymbol{x}] \\
&\quad = \mathrm{E}_{\mathbb{X},\mathrm{y}}[\mathrm{y}^2 + f_N(\boldsymbol{x})^2 - 2f_N(\boldsymbol{x})\mathrm{y}|\boldsymbol{x}] \\
&\quad = \mathrm{E}_{\mathrm{y}}[\mathrm{y}^2|\boldsymbol{x}] + \mathrm{E}_{\mathbb{X}}[f_N(\boldsymbol{x})^2|\boldsymbol{x}] - 2\mathrm{E}_{\mathbb{X},\mathrm{y}}[f_N(\boldsymbol{x})\mathrm{y}|\boldsymbol{x}] \\
&\quad = (\mathrm{Var}_{\mathrm{y}}[\mathrm{y}|\boldsymbol{x}] + \mathrm{E}_{\mathrm{y}}[\mathrm{y}|\boldsymbol{x}]^2) + (\mathrm{Var}_{\mathbb{X}}[f_N(\boldsymbol{x})|\boldsymbol{x}] + \mathrm{E}_{\mathbb{X}}[f_N(\boldsymbol{x})|\boldsymbol{x}]^2) \\
&\qquad\quad - 2\mathrm{E}_{\mathrm{y}}[\mathrm{y}|\boldsymbol{x}]\mathrm{E}_{\mathbb{X}}[f_N(\boldsymbol{x})|\boldsymbol{x}] \\
&\quad = \mathrm{Var}_{\mathrm{y}}[\mathrm{y}|\boldsymbol{x}] + \mathrm{Var}_{\mathbb{X}}[f_N(\boldsymbol{x})|\boldsymbol{x}] + (\mathrm{E}_{\mathbb{X}}[f_N(\boldsymbol{x})|\boldsymbol{x}] - \mathrm{E}_{\mathrm{y}}[\mathrm{y}|\boldsymbol{x}])^2 \\
&\quad = \mathrm{Var}_{\mathrm{y}}[\mathrm{y}|\boldsymbol{x}] + \mathrm{Var}_{\mathbb{X}}[f_N(\boldsymbol{x})|\boldsymbol{x}] + \mathrm{E}_{\mathbb{X}}[f_N(\boldsymbol{x}) - f^*(\boldsymbol{x})|\boldsymbol{x}]^2 \\
&\quad = \sigma^2 + \mathrm{Var}_{\mathbb{X}}[f_N(\boldsymbol{x})|\boldsymbol{x}] + \text{bias}[f_N(\boldsymbol{x})|\boldsymbol{x}]^2
\end{aligned}
$$

Bias-Variance Tradeoff I

$$
\begin{aligned}
\mathrm{E}_{\mathbb{X}}\left(C[f_N]\right) &= \mathrm{E}_{\mathbf{x}}\left(\mathrm{E}_{\mathbb{X},\mathrm{y}}\left[\mathrm{loss}(f_N(\boldsymbol{x})-\mathrm{y})|\boldsymbol{x}\right]\right) \\
&= \mathrm{E}_{\mathbf{x}}\left(\sigma^2+\mathrm{Var}_{\mathbb{X}}[f_N(\boldsymbol{x})|\boldsymbol{x}]+\mathrm{bias}[f_N(\boldsymbol{x})|\boldsymbol{x}]^2\right)
\end{aligned}
$$

Bias-Variance Tradeoff I

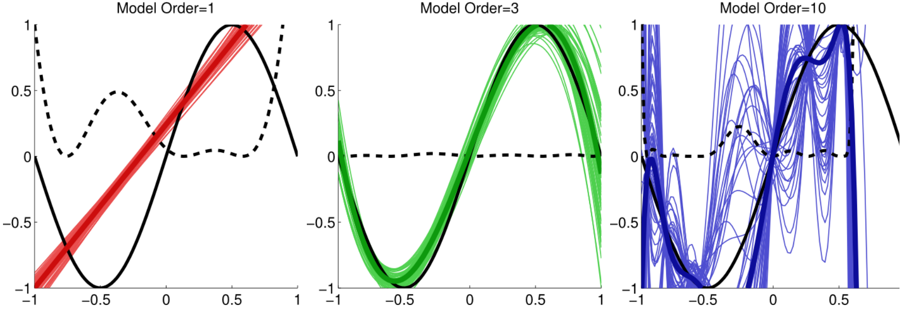
$$\begin{aligned}\mathrm{E}_{\mathbb{X}}\left(C[f_N]\right) &= \mathrm{E}_{\mathbf{x}}\left(\mathrm{E}_{\mathbb{X},\mathrm{y}}\left[\mathrm{loss}(f_N(\boldsymbol{x})-\mathrm{y})|\boldsymbol{x}\right]\right)\\ &= \mathrm{E}_{\mathbf{x}}\left(\sigma^2+\mathrm{Var}_{\mathbb{X}}[f_N(\boldsymbol{x})|\boldsymbol{x}]+\mathrm{bias}[f_N(\boldsymbol{x})|\boldsymbol{x}]^2\right)\end{aligned}$$

- The first term cannot be avoided when $\mathrm{P}(\mathrm{y}|\mathbf{x})$ is stochastic

Bias-Variance Tradeoff I

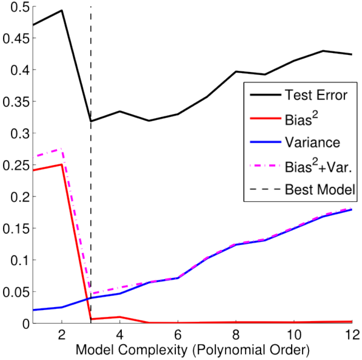
$$\begin{aligned}\mathrm{E}_{\mathbb{X}}\left(C[f_N]\right) &= \mathrm{E}_{\mathbf{x}}\left(\mathrm{E}_{\mathbb{X},\mathrm{y}}\left[\mathrm{loss}(f_N(\boldsymbol{x})-\mathrm{y})|\boldsymbol{x}\right]\right)\\ &= \mathrm{E}_{\mathbf{x}}\left(\sigma^2+\mathrm{Var}_{\mathbb{X}}[f_N(\boldsymbol{x})|\boldsymbol{x}]+\mathrm{bias}[f_N(\boldsymbol{x})|\boldsymbol{x}]^2\right)\end{aligned}$$

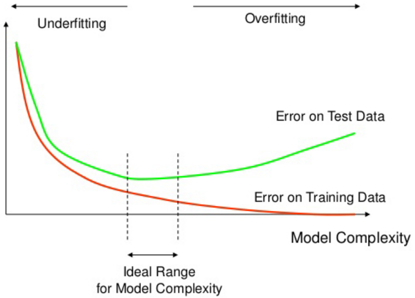
- The first term cannot be avoided when $\mathrm{P}(\mathrm{y}|\mathbf{x})$ is stochastic
- ***Model complexity*** controls the tradeoff between variance and bias
- E.g., polynomial regressors (dotted line = average training error):



Bias-Variance Tradeoff II

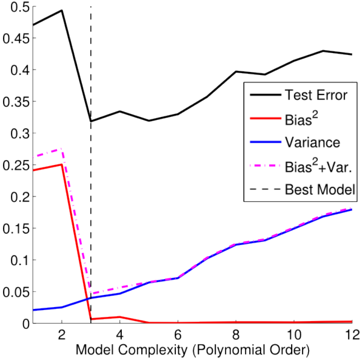
- Provides another way to understand the generalization/testing error

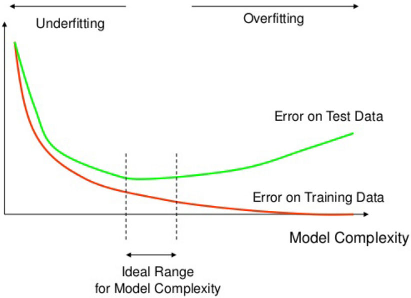




Bias-Variance Tradeoff II

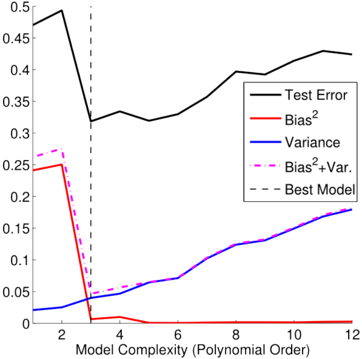
- Provides another way to understand the generalization/testing error
- Too simple a model leads to high bias or underfitting
  - ***High*** training error; ***high*** testing error (given a sufficiently large N)

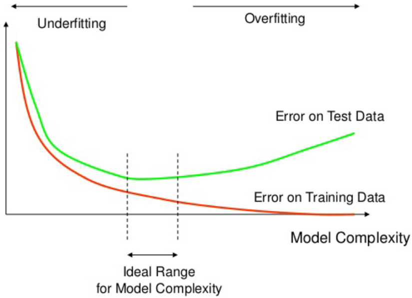




Bias-Variance Tradeoff II

- Provides another way to understand the generalization/testing error
- Too simple a model leads to high bias or underfitting
  - ***High*** training error; ***high*** testing error (given a sufficiently large N)
- Too complex a model leads to high variance or overfitting
  - ***Low*** training error; ***high*** testing error





Outline

1 Learning Theory

2 Point Estimation: Bias and Variance
- Consistency*

3 Decomposing Generalization Error

4 Regularization
- Weight Decay
- Validation

Regularization

- We get $f_N = \arg\min_{f \in \mathbb{F}} C_N[f]$ by minimizing the empirical error

Regularization

- We get $f_N = \arg\min_{f \in \mathbb{F}} C_N[f]$ by minimizing the empirical error
- But what we really care about is the generalization error $C[f_N]$

Regularization

- We get $f_N = \arg\min_{f \in \mathbb{F}} C_N[f]$ by minimizing the empirical error
- But what we really care about is the generalization error $C[f_N]$
- ***Regularization*** refers to any technique designed to improve the generalizability of f_N
- Any idea inspired by the learning theory?

Regularization

- We get $f_N = \arg\min_{f \in \mathbb{F}} C_N[f]$ by minimizing the empirical error
- But what we really care about is the generalization error $C[f_N]$
- ***Regularization*** refers to any technique designed to improve the generalizability of f_N
- Any idea inspired by the learning theory?
- Regularization in the cost function: ***weight decay***
- Regularization during the training process: ***validation***

Outline

1 Learning Theory

2 Point Estimation: Bias and Variance
- Consistency*

3 Decomposing Generalization Error

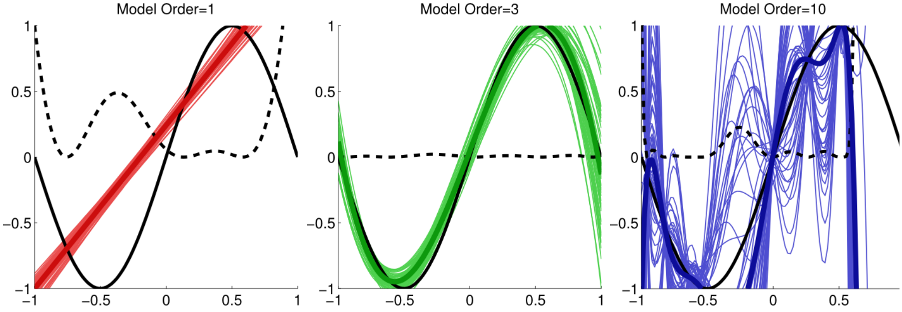
4 Regularization
- Weight Decay
- Validation

Panelizing Complex Functions

- ***Occam's razor***: among equal-performing models, the simplest one should be selected

Panelizing Complex Functions

- ***Occam's razor***: among equal-performing models, the simplest one should be selected
- Idea: to add a term in the cost function that panelizes complex functions
- So, with sufficiently complex $\mathbb{F}$:
  - Minimizing the empirical error term reduces bias
  - Minimizing the penalty term reduces variance



What to Panelize?

- What impacts $\text{Complexity}(\mathbb{F})$ in a model?

What to Panelize?

- What impacts $\text{Complexity}(\mathbb{F})$ in a model?
- Some constants in the model $\mathbb{F}$
  - E.g., degree P in polynomial regression
- Restricts the capacity of $\mathbb{F}$

What to Panelize?

- What impacts $\mathrm{Complexity}(\mathbb{F})$ in a model?
- Some constants in the model $\mathbb{F}$
  - E.g., degree P in polynomial regression
- Restricts the capacity of $\mathbb{F}$
- However, cannot be penalized in a cost fucntion since fixed

What to Panelize?

- What impacts $\text{Complexity}(\mathbb{F})$ in a model?
- Some constants in the model $\mathbb{F}$
  - E.g., degree P in polynomial regression
- Restricts the capacity of $\mathbb{F}$
- However, cannot be penalized in a cost fucntion since fixed
- Alternatively, ***function parameters***
  - E.g., the parameter $\boldsymbol{w}$ of a function $f(\cdot;\boldsymbol{w}) \in \mathbb{F}$
- Also restricts the capacity of $\mathbb{F}$
- Can be penalized

What to Panelize?

- What impacts $\text{Complexity}(\mathbb{F})$ in a model?
- Some constants in the model $\mathbb{F}$
  - E.g., degree P in polynomial regression
- Restricts the capacity of $\mathbb{F}$
- However, cannot be penalized in a cost fucntion since fixed
- Alternatively, ***function parameters***
  - E.g., the parameter $\boldsymbol{w}$ of a function $f(\cdot;\boldsymbol{w}) \in \mathbb{F}$
- Also restricts the capacity of $\mathbb{F}$
- Can be penalized
- But which $\boldsymbol{w}$ implies a complex model?

Weight Decay

- In practice, $\boldsymbol{w} = \mathbf{0}$ is usually the "simplest" function
  - E.g, in binary classification for labels $\{-1, 1\}$, a perceptron with $\boldsymbol{w} = \mathbf{0}$ means random guessing

Weight Decay

- In practice, $\boldsymbol{w} = \mathbf{0}$ is usually the "simplest" function
  - E.g, in binary classification for labels $\{-1, 1\}$, a perceptron with $\boldsymbol{w} = \mathbf{0}$ means random guessing
- ***Weight decay***: to penalize the ***norm*** of $\boldsymbol{w}$, which is nonnegative and equals to 0 when $\boldsymbol{w} = \mathbf{0}$

Weight Decay

- In practice, $\boldsymbol{w} = \mathbf{0}$ is usually the "simplest" function
  - E.g, in binary classification for labels $\{-1, 1\}$, a perceptron with $\boldsymbol{w} = \mathbf{0}$ means random guessing
- ***Weight decay***: to penalize the ***norm*** of $\boldsymbol{w}$, which is nonnegative and equals to 0 when $\boldsymbol{w} = \mathbf{0}$
- E.g., the ***Ridge regression***:

$$\arg\min_{\boldsymbol{w},b} \frac{1}{2}\|\boldsymbol{y} - (\boldsymbol{X}\boldsymbol{w} - b\mathbf{1})\|^2 \text{ subject to } \|\boldsymbol{w}\|^2 \leq T$$

for some constant $T > 0$

Weight Decay

- In practice, $\boldsymbol{w} = \boldsymbol{0}$ is usually the "simplest" function
  - E.g, in binary classification for labels $\{-1, 1\}$, a perceptron with $\boldsymbol{w} = \boldsymbol{0}$ means random guessing
- ***Weight decay***: to penalize the ***norm*** of $\boldsymbol{w}$, which is nonnegative and equals to 0 when $\boldsymbol{w} = \boldsymbol{0}$
- E.g., the ***Ridge regression***:

$$\arg\min_{\boldsymbol{w},b} \frac{1}{2}\|\boldsymbol{y} - (\boldsymbol{X}\boldsymbol{w} - b\boldsymbol{1})\|^2 \text{ subject to } \|\boldsymbol{w}\|^2 \leq T$$

  for some constant $T > 0$
- In practice, we usually solve a simpler problem:

$$\arg\min_{\boldsymbol{w},b} \frac{1}{2N}\|\boldsymbol{y} - (\boldsymbol{X}\boldsymbol{w} - b\boldsymbol{1})\|^2 + \frac{\alpha}{2}\|\boldsymbol{w}\|^2$$

  where $\alpha > 0$ is a constant representing both T and the KKT multiplier

Weight Decay

- In practice, $\boldsymbol{w} = \boldsymbol{0}$ is usually the "simplest" function
  - E.g, in binary classification for labels $\{-1, 1\}$, a perceptron with $\boldsymbol{w} = \boldsymbol{0}$ means random guessing
- ***Weight decay***: to penalize the ***norm*** of $\boldsymbol{w}$, which is nonnegative and equals to 0 when $\boldsymbol{w} = \boldsymbol{0}$
- E.g., the ***Ridge regression***:

$$\arg\min_{\boldsymbol{w},b} \frac{1}{2} \|\boldsymbol{y} - (\boldsymbol{X}\boldsymbol{w} - b\boldsymbol{1})\|^2 \text{ subject to } \|\boldsymbol{w}\|^2 \leq T$$

  for some constant $T > 0$
- In practice, we usually solve a simpler problem:

$$\arg\min_{\boldsymbol{w},b} \frac{1}{2N} \|\boldsymbol{y} - (\boldsymbol{X}\boldsymbol{w} - b\boldsymbol{1})\|^2 + \frac{\alpha}{2} \|\boldsymbol{w}\|^2$$

  where $\alpha > 0$ is a constant representing both T and the KKT multiplier
- What does a larger α means?

Weight Decay

- In practice, $\boldsymbol{w} = \boldsymbol{0}$ is usually the "simplest" function
  - E.g, in binary classification for labels $\{-1, 1\}$, a perceptron with $\boldsymbol{w} = \boldsymbol{0}$ means random guessing
- ***Weight decay***: to penalize the ***norm*** of $\boldsymbol{w}$, which is nonnegative and equals to 0 when $\boldsymbol{w} = \boldsymbol{0}$
- E.g., the ***Ridge regression***:

$$\arg\min_{\boldsymbol{w},b} \frac{1}{2}\|\boldsymbol{y} - (\boldsymbol{X}\boldsymbol{w} - b\boldsymbol{1})\|^2 \text{ subject to } \|\boldsymbol{w}\|^2 \leq T$$

  for some constant $T > 0$

- In practice, we usually solve a simpler problem:

$$\arg\min_{\boldsymbol{w},b} \frac{1}{2N}\|\boldsymbol{y} - (\boldsymbol{X}\boldsymbol{w} - b\boldsymbol{1})\|^2 + \frac{\alpha}{2}\|\boldsymbol{w}\|^2$$

  where $\alpha > 0$ is a constant representing both T and the KKT multiplier

- What does a larger α means? We prefer a more simple function

Flat Regressors

$$\arg\min_{\boldsymbol{w},b} \frac{1}{2}\left(\|\boldsymbol{y} - (\boldsymbol{X}\boldsymbol{w} - b\mathbf{1})\|^2 + \alpha\|\boldsymbol{w}\|^2\right)$$

- The bias b is ***not*** regularized, why?

Flat Regressors

$$\arg\min_{\boldsymbol{w},b} \frac{1}{2}\left(\|\boldsymbol{y} - (\boldsymbol{X}\boldsymbol{w} - b\mathbf{1})\|^2 + \alpha\|\boldsymbol{w}\|^2\right)$$

- The bias b is ***not*** regularized, why?
- We want the simplest function with $\boldsymbol{w} = \mathbf{0}$ means "a dummy regressor by averaging"
  - Remember R^2 (coefficient of determination)?
- However, the label y's may not be standardized to have zero mean

Flat Regressors

$$\arg\min_{\boldsymbol{w},b} \frac{1}{2}\left(\|\boldsymbol{y}-(\boldsymbol{X}\boldsymbol{w}-b\mathbf{1})\|^2+\alpha\|\boldsymbol{w}\|^2\right)$$

- The bias b is ***not*** regularized, why?
- We want the simplest function with $\boldsymbol{w}=\mathbf{0}$ means "a dummy regressor by averaging"
  - Remember R^2 (coefficient of determination)?
- However, the label y's may not be standardized to have zero mean
- This explains why we prefer a "flat" hyperplane in the previous lecture
- We have discussed how to solve the Ridge regression problem

Sparse Weight Decay

- Alternatively we can minimizes the ***L^1-norm*** in weight decay
- E.g., ***LASSO*** (least absolute shrinkage and selection operator):

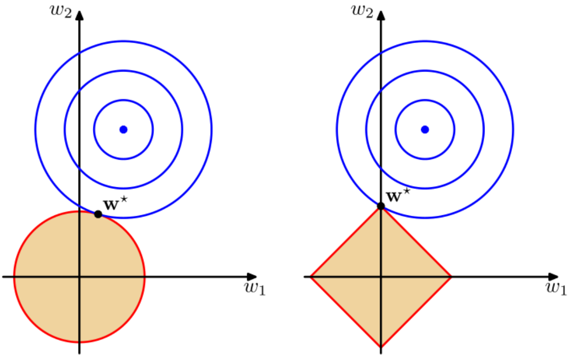
$$\arg\min_{\boldsymbol{w},b} \frac{1}{2N} \|\boldsymbol{y} - (\boldsymbol{X}\boldsymbol{w} - b\mathbf{1})\|^2 + \alpha \|\boldsymbol{w}\|_1$$

  for some constant $\alpha > 0$
- Usually results in ***sparse*** $\boldsymbol{w}$ that has many zero attributes
- Why?

Sparsity

$$\arg\min_{\boldsymbol{w},b} \frac{1}{2N} \|\boldsymbol{y} - (\boldsymbol{X}\boldsymbol{w} - b\mathbf{1})\|^2 + \alpha\|\boldsymbol{w}\|_1$$

- The surface of the cost function is the sum of SSE (blue contours) and 1-norm (red contours)
- Optimal point locates on some axes



Elastic Net**

- LASSO can be used as a feature selection technique
  - The sparse $\boldsymbol{w}$ selects explanatory variables that are most correlated to the target variable

Elastic Net**

- LASSO can be used as a feature selection technique
  - The sparse $\boldsymbol{w}$ selects explanatory variables that are most correlated to the target variable
- Limitations:
  1. Selects at most N variables if $D > N$
  2. No ***group selection***
     - Important in some applications, e.g., gene selection problems

Elastic Net**

- LASSO can be used as a feature selection technique
  - The sparse $\boldsymbol{w}$ selects explanatory variables that are most correlated to the target variable
- Limitations:
  1. Selects at most N variables if $D > N$
  2. No ***group selection***
     - Important in some applications, e.g., gene selection problems
- ***Elastic net*** combines Ridge and LASSO:

$$\arg\min_{\boldsymbol{w},b} \frac{1}{2N}\|\boldsymbol{y}-(\boldsymbol{X}\boldsymbol{w}-b\mathbf{1})\|^2+\alpha\left(\beta\|\boldsymbol{w}\|_1+\frac{1-\beta}{2}\|\boldsymbol{w}\|^2\right)$$

for some constant $\beta \in (0,1)$

Elastic Net**

- LASSO can be used as a feature selection technique
  - The sparse $\boldsymbol{w}$ selects explanatory variables that are most correlated to the target variable
- Limitations:
  1. Selects at most N variables if $D > N$
  2. No ***group selection***
     - Important in some applications, e.g., gene selection problems
- ***Elastic net*** combines Ridge and LASSO:

$$\arg\min_{\boldsymbol{w},b} \frac{1}{2N}\|\boldsymbol{y} - (\boldsymbol{X}\boldsymbol{w} - b\boldsymbol{1})\|^2 + \alpha\left(\beta\|\boldsymbol{w}\|_1 + \frac{1-\beta}{2}\|\boldsymbol{w}\|^2\right)$$

  for some constant $\beta \in (0,1)$
- Still gives a sparse $\boldsymbol{w}$
- Highly correlated variables will have similar values in $\boldsymbol{w}$

Outline

1 Learning Theory

2 Point Estimation: Bias and Variance
- Consistency*

3 Decomposing Generalization Error

4 Regularization
- Weight Decay
- Validation

Tuning Hyperparameters

- In ML, we call the constants that are fixed in a model the ***hyperparameters***
  - Degree P in polynomial regression
  - Coefficient α of the weight decay term in the cost function of Ridge and LASSO, etc.

Tuning Hyperparameters

- In ML, we call the constants that are fixed in a model the ***hyperparameters***
  - Degree P in polynomial regression
  - Coefficient α of the weight decay term in the cost function of Ridge and LASSO, etc.
- Usually reflect some assumptions about the model
- Changing their values changes model complexity
  - And therefore generalization performance

Tuning Hyperparameters

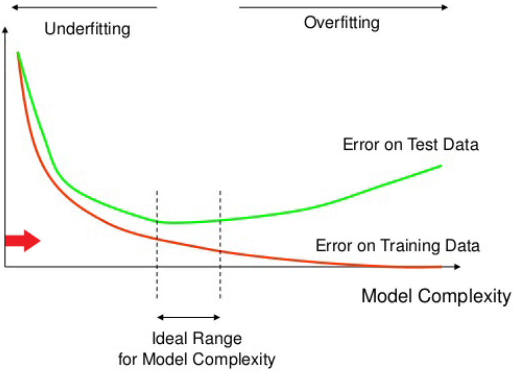
- In ML, we call the constants that are fixed in a model the ***hyperparameters***
  - Degree P in polynomial regression
  - Coefficient α of the weight decay term in the cost function of Ridge and LASSO, etc.
- Usually reflect some assumptions about the model
- Changing their values changes model complexity
  - And therefore generalization performance
- How to set appropriate values?

Tuning Hyperparameters

- In ML, we call the constants that are fixed in a model the ***hyperparameters***
  - Degree P in polynomial regression
  - Coefficient α of the weight decay term in the cost function of Ridge and LASSO, etc.
- Usually reflect some assumptions about the model
- Changing their values changes model complexity
  - And therefore generalization performance
- How to set appropriate values?
- Train a model many times with different hyperparameters, and choose the function with best generalizability
- Very time consuming, can we have heuristics to speed up the process?

Structured Risk Minimization

- Consider again the Occam's razor
- ***Structured risk minimization***: start from the simplest model, gradually increase its complexity, and stop when overfitting



Validation Set

- Pitfall:

Validation Set

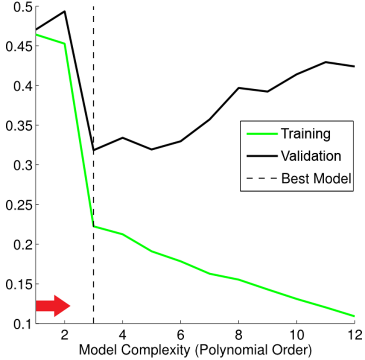
- Pitfall: we peep the testing set during the training process
  - The final function will overfit the testing set
  - Optimistic testing error

Validation Set

- Pitfall: we peep the testing set during the training process
  - The final function will overfit the testing set
  - Optimistic testing error
- Fix?

Validation Set

- Pitfall: we peep the testing set during the training process
  - The final function will overfit the testing set
  - Optimistic testing error
- Fix? Split a ***validation set*** from the training set and use it for hyperparameter selection



Reference I

[1] Olivier Bousquet.
Concentration inequalities and empirical processes theory applied to the analysis of learning algorithms.
Ph.D. thesis, Ecole Polytechnique, Palaiseau, France, 2002.

[2] Pascal Massart.
Some applications of concentration inequalities to statistics.
In *Annales de la Faculté des sciences de Toulouse: Mathématiques*, volume 9, pages 245–303, 2000.

[3] Vladimir N Vapnik and A Ya Chervonenkis.
On the uniform convergence of relative frequencies of events to their probabilities.
In *Measures of Complexity*, pages 11–30. Springer, 2015.

Reference II

[4] David H Wolpert.
The lack of a priori distinctions between learning algorithms.
Neural computation, 8(7):1341–1390, 1996.