# Notes of Machine Learning

**Xiangli Chen**

Computer Science Department
University of Illinois at Chicago

June 11, 2017

## Outline

## What is machine learning

**Definition**

A computer is said to learn from experience $E$ with respect to some class of tasks $T$ and performance measure $P$, if its performance at tasks in $T$, as measured by $P$, improves with experience $E$.

## Outline

## Probability and Distribution

## Some Special distributions

## Asymptotic Theory

### Definition

Converge in probability:
Let $\{X_n\}$ be a sequence of random variables and let $X$ be a random variable defined on a sample space. We say that $X_n$ converges in probability to $X$ if, for all $\epsilon > 0$,

$$\lim_{n\to\infty} p((x_n - x) \geq \epsilon) = 0$$

or equivalently,

$$\lim_{n\to\infty} p((x_n - x) < \epsilon) = 1.$$

If so, we write

$$X_n \xrightarrow{p} X.$$

## Asymptotic Theory

**Definition**

Consistency:
Let $X$ be a random variable with cdf $F(x, \theta)$, $\theta \in \Theta$. Let $X_1, \cdots, X_n$ be a sample from the distribution of $X$ and let $T_n$ denote a statistic. We say $T_n$ is a consistent estimator of $\theta$ if

$$T_n \xrightarrow{p} \theta.$$

## Asymptotic Theory

### Definition

Converge in distribution:
Let $\{X_n\}$ be a sequence of random variables and let $X$ be a random variable defined on a sample space. Let $F_{X_n}$ and $F_X$ be, respectively, the cdfs of $X_n$ and $X$. Let $C(F_X)$ denote the set of all points where $F_X$ is continuous. We say that $X_n$ converges in distribution to $X$ if

$$\lim_{n\to\infty} F_{X_n}(x) = F_X(x), \text{ for all } x \in C(F_X).$$

We denote this convergence by If so, we write

$$X_n \xrightarrow{D} X.$$

## Asymptotic Theory

**Definition**

Bounded in probability (Stochastically bounded):
We say that the sequence of random variables $\{X_n\}$ is bounded in probability if, for all $\epsilon > 0$, there exists a constant $C > 0$ and an integer $N_\epsilon$ such that for all $n \geq N_\epsilon$

$$p(|X_n| \leq C) \geq 1 - \epsilon.$$

## Asymptotic Theory-Converge Rate

Consider random sequences $\{X_n\}_{n=1}^{\infty}$ and $\{Y_n\}_{n=1}^{\infty}$.
The $o_p$ notation.
We say that $X_n = o_p(Y_n)$ if

$$\frac{X_n}{Y_n} \to_p 0.$$

The $O_p$ notation.
We say $X_n = O_p(Y_n)$ ($X_n$ is of order no larger than $Y_n$) if

$$\frac{X_n}{Y_n} = O_p(1).$$

If $X_n = O_p(1)$, we say that $X_n$ is bounded in probability.

## Maximum Likelihood Estimation (MLE)

MLE: Choose $\theta$ that maximizes the probability of observed data $\mathcal{D}$

$$\hat{\theta}_{MLE} = \underset{\theta}{\operatorname{argmax}}\, p(\mathcal{D}|\theta)$$

In general, we assume $\mathcal{D} : x_1, \cdots, x_n$ independent draw and identically distributed (iid) of $p(x|\theta)$.

$$\hat{\theta}_{MLE} = \underset{\theta}{\operatorname{argmax}} \prod_{i=1}^{n} p(x_i|\theta)$$

Let $J(\theta) = \prod_{i=1}^{n} p(x_i|\theta)$, often,

$$\partial J(\theta)/\partial\theta|_{\theta=\hat{\theta}_{MLE}} = 0$$

In pratice, for computation benefit, we consider the log form, $L(\theta)$

## MLE-examples

## Maximum a Posteriori Estimation (MAP)

A Bayesian approach: $p(\theta|\mathcal{D}) \propto p(\mathcal{D}|\theta)p(\theta)$

- prior($p(\theta)$): represents expert knowledge
- conjugate priors: $p(\theta)$ and $p(\theta|\mathcal{D})$ have the same form

Maximum a posteriori (MAP) estimation: choose value that is most probable given observed data and prior belief

$$\begin{aligned}\hat{\theta}_{MAP} &= \underset{\theta}{\operatorname{argmax}}\, p(\theta|\mathcal{D}) \\ &= \underset{\theta}{\operatorname{argmax}}\, p(\mathcal{D}|\theta)p(\theta)\end{aligned}$$

If $p(\theta) \sim \mathit{Uniform}$ (a constant), MAP and MLE are the same.
Typically, when $|\mathcal{D}|$ increases, MAP $\rightarrow$ MLE that the data dominate the posteriori distribution.

## MAP-examples

## Outline

## Naive Bayes Formulation

Training data: $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$ and $x = (x^{(1)}, \cdots, x^{(d)})$
Classifer (Bayesian approach):

$$\begin{aligned} h(x) &= \underset{y}{\operatorname{argmax}}\, p(y|x) \\ &= \underset{y}{\operatorname{argmax}}\, p(x|y)p(y) \end{aligned}$$

Suppose $x^{(j)}$ : $M$ different values; $y$ : $K$ different class lables
$p(x|y)$ requires estimate $(M^d - 1)K$ values
Naive Bayesian assumption (conditional independence):

$$p(x^{(1)}, \cdots, x^{(d)}|y) = \prod_{j=1}^{d} p(x^{(j)}|y)$$

$p(x|y)$ requires estimate only $(M-1)dK$ values

## Naive Bayes - Discrete Features

Estimation via MLE of joint distribution (Generative approach)

$$(\hat{\theta}, \hat{\pi}) = \underset{\theta,\pi}{\text{argmax}}\, p(\mathcal{D}|\theta, \pi) = \underset{\theta,\pi}{\text{argmax}} \prod_{i=1}^{N} \prod_{j=1}^{d} p(x_i^{(j)}|y_i)p(y_i)$$

For class prior $\hat{\pi}_k = \hat{p}(y_k) = \dfrac{\#\mathcal{D}\{Y = y_k\}}{\#\mathcal{D}}$

For likelihood $\hat{\theta}_{mk}^{(j)} = \hat{p}(x_m^{(j)}|y_k) = \dfrac{\#\mathcal{D}\{X^{(j)} = x_m^{(j)} \wedge Y = y_k\}}{\#\mathcal{D}\{Y = y_k\}}$

Naive Bayesian classifier:

$$h_{NB}(x) = \underset{y}{\text{argmax}}\, \hat{p}(y) \prod_{j=1}^{d} \hat{p}(x^{(j)}|y)$$

## Naive Bayes - Smoothing

Issue: insufficient training data
Never see a training data where $X^{(j)} = a$ when $Y = b$

$$p(X^{(j)} = a | Y = b) = 0$$

Smoothing estimation method

$$\hat{\theta}_{mk}^{(j)} = \frac{\#\mathcal{D}\{X^{(j)} = x_m^{(j)} \wedge Y = y_k\} + l_{mk}^{(j)}}{\#\mathcal{D}\{Y = y_k\} + \sum_{m=1}^{M} l_m^{(j)}}$$

Similarly,

$$\hat{\pi}_k = \hat{p}(y_k) = \frac{\#\mathcal{D}\{Y = y_k\} + l_k}{\#\mathcal{D} + \sum_{k=1}^{K} l_k}$$

## Naive Bayes - Smoothing - Interpretation

Two ways to interpret smoothing:

- add $\sum_{m=1}^{M} l_{mk}^{(j)}$ "virtual" examples for estimating $\hat{\theta}_{mk}^{(j)}$.
- MAP estimate for $\hat{\theta}_{mk}^{(j)}$ assuming a Dirichlet prior distribution over $\hat{\theta}_{mk}^{(j)}$. If $l_{mk}^{(j)} = 1$, known as Laplace smoothing.

## Gaussian Naive Bayes - Continuous Features

Gaussian Naive Bayes (GNB):

$$p(X^{(j)} = x|Y = y_k) = \frac{1}{\sigma_k^{(j)}\sqrt{2\pi}} e^{\frac{-(x-\mu_k^{(j)})^2}{2(\sigma_k^{(j)})^2}}$$

There are $2dK$ unknown parameters: $\mu_k^{(j)}$ and $\sigma_k^{(j)}$.

## Gaussian Naive Bayes - MLE

**MLE for Gaussian distribution $X \sim \mathcal{N}(\mu, \sigma^2)$**

$$\hat{\mu}_{MLE} = \frac{1}{N}\sum_{i=1}^{N} x_i \qquad \hat{\sigma}^2_{unbiased} = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \hat{\mu})^2$$

MLE estimate for GNB

$$\hat{\mu}_k^{(j)} = \frac{\sum_{i=1}^{N} x_i^{(j)} \delta(Y_i = y_k)}{\sum_{i=1}^{N} \delta(Y_i = y_k)} \quad (\hat{\sigma}_k^{(j)})^2 = \frac{\sum_{i=1}^{N}(x_i^{(j)} - \hat{\mu}_k^{(j)})^2 \delta(Y_i = y_k)}{\sum_{i=1}^{N} \delta(Y_i = y_k) - 1}$$

GNB classifier:

$$h_{GNB}(x) = \underset{y_k}{\operatorname{argmax}}\, \hat{p}(y_k) \prod_{j=1}^{d} \mathcal{N}(\hat{\mu}_k^{(j)}, \hat{\sigma}_k^{(j)})$$

## Gaussian Naive Bayes - Decision Boundary

Decision boundary of a binary Naive Bayes classifier ($x$'s):

$$\frac{\prod_{i=1}^{d} p(x_i|y=0)p(y=0)}{\prod_{i=1}^{d} p(x_i|y=1)p(y=1)} = 1$$

Decision boundary of a binary GNB classifier with class independent variance is linear (i.e. $\sigma_{i,0}^2 = \sigma_{i,1}^2$) (log form)

$$\log \frac{\prod_{i=1}^{d} p(x_i|y=0)p(y=0)}{\prod_{i=1}^{d} p(x_i|y=1)p(y=1)} = \log \frac{p(y=0)}{p(y=1)} + \sum_{i=1}^{d} \log \frac{p(x_i|y=0)}{p(x_i|y=1)}$$

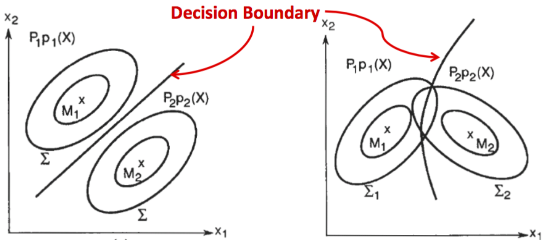
$$= \log \frac{p(y=0)}{p(y=1)} + \sum_{i=1}^{d} \frac{\mu_{i,1}^2 - \mu_{i,0}^2}{2\sigma_i^2} + \sum_{i=1}^{d} \frac{\mu_{i,0} - \mu_{i,1}}{\sigma_i^2} x_i = \underbrace{\omega_0 + \sum_{i=1}^{d} \omega_i x_i}_{\text{a linear classifier}} = 0$$

## Gaussian Naive Bayes - Decision Boundary

Decision boundary of binary GNB classifiers

$$X = (x_1, x_2) \quad P_1 = p(y = 0) \quad P_2 = p(y = 1)$$

$$p_1(X) = p(x|y = 1) \sim \mathcal{N}(M_1, \Sigma_1) \quad p_2(X) = p(x|y = 2) \sim \mathcal{N}(M_2, \Sigma_2)$$



- $\Sigma_1 = \Sigma_2 = \Sigma$ (class independent variance) - linear
- $\Sigma_1 \neq \Sigma_2$ - non-linear

## Logistic Function

Logistic function

$$f(z) = \frac{1}{1 + \exp(-z)}$$

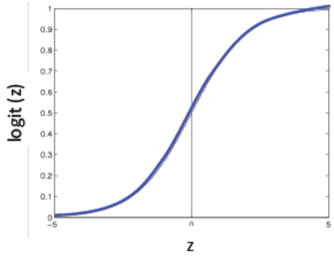


Figure 1: Logistic function

Logistic regression for binary class

$$p(y = 0|x) = \frac{1}{1 + \exp(\omega_0 + \sum_j \omega_j x_j)}$$

$$p(y = 1|x) = \frac{\exp(\omega_0 + \sum_j \omega_j x_j)}{1 + \exp(\omega_0 + \sum_j \omega_j x_j)}$$

## Decision Boundary

Decision boundary is linear

$$\frac{p(y=1|x)}{p(y=0|x)} = \exp\left(\omega_0 + \sum_j \omega_j x_j\right) = 1$$

Log form

$$\omega_0 + \sum_j \omega_j x_j = 0$$

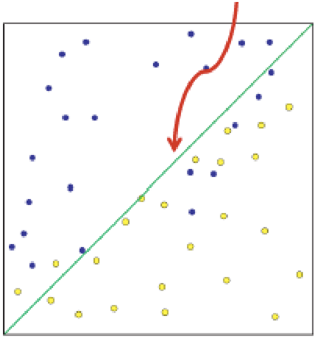


Figure 2: Decision boundary

## Training Logistic Regression Model

Data

$$\{(x_i, y_i)_{i=1}^{n}\} \qquad x_i = (x_{i1}, \cdots, x_{id})$$

Maximum conditional log likelihood (discriminative approach)

$$\hat{\omega}_{MLE} = \arg\max_{\omega} \log \prod_{i=1}^{n} p(y_i | x_i, \omega)$$

## Optimization Problem

Conditional log likelihood

$$
\begin{aligned}
l(\omega) &= \sum_{i=1}^{n} \log p(y_i | x_i, \omega) \\
&= \sum_i y_i \log p(y_i = 1 | x_i, \omega) + (1 - y_i) \log p(y_i = 0 | x_i, \omega) \\
&= \sum_i y_i (\omega_0 + \sum_j \omega_j x_{ij}) - \log(1 + \exp(\omega_0 + \sum_j \omega_j x_{ij}))
\end{aligned}
$$

- no closed form solution
- $l(\omega)$ is a concave function of $\omega$ ($-l(\omega)$ is the convex function)

## Gradient Descent Method

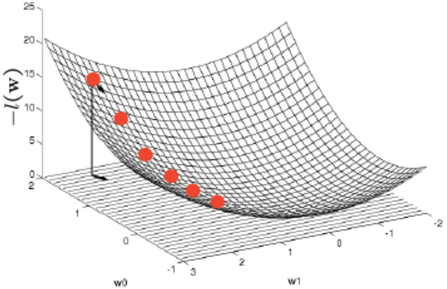
Gradient descent ($\eta > 0$)

$$\omega^{(t+1)} \leftarrow \omega^{(t)} - \triangle\omega$$

$$\triangle\omega = \eta \nabla_\omega(-l(\omega)) = -\eta \nabla_\omega(l(\omega))$$

$$\omega^{(t+1)} \leftarrow \omega^{(t)} + \eta \nabla_\omega(l(\omega))$$

$$\nabla_\omega l(\omega) = \left[\frac{\partial l(\omega)}{\partial \omega_0} \cdots \frac{\partial l(\omega)}{\partial \omega_d}\right]^T$$



$$\frac{\partial l(\omega)}{\partial \omega_0} = \sum_i (y_i - \hat{p}(y_i = 1 | x_i, \omega))$$

$$\frac{\partial l(\omega)}{\partial \omega_j} = \sum_i x_{ij}(y_i - \hat{p}(y_i = 1 | x_i, \omega))$$

# Multi-Class Logistic Regression

Logistic regression of multi-class - $Y \in \{y_1, \cdots, y_K\}$
For $k < K$

$$p(y = y_k|x) = \frac{\exp(\omega_{0k} + \sum_{j=1}^{d} \omega_{jk} x_j)}{1 + \sum_{m=1}^{K-1} \exp(\omega_{0m} + \sum_{j=1}^{d} \omega_{jm} x_j)}$$

For $k = K$

$$p(y = y_K|x) = \frac{1}{1 + \sum_{m=1}^{K-1} \exp(\omega_{0m} + \sum_{j=1}^{d} \omega_{jm} x_j)}$$

Classifier

$$f_{MCLR}(x) = \arg\max_{y_k} p(y = y_k|x)$$

## Decision Boundary of Multi-Class

linear

## Training Multi-Class Logistic Regression

Conditional log likelihood

$$
\begin{aligned}
l(\omega) &= \sum_{i=1}^{n} \log p(y_i|x_i,\omega) = \sum_{i=1}^{n}\sum_{k=1}^{K} 1_{(y_i=y_k)} \log p(y_i = y_k|x_i,\omega) \\
&= \sum_{i=1}^{n}\sum_{k=1}^{K-1} 1_{(y_i=y_k)} \left(\omega_{0k} + \sum_{j=1}^{d} \omega_{jk} x_{ij}\right) \\
&\qquad - \log\left(1 + \sum_{m=1}^{K-1} \exp\left(\omega_{0m} + \sum_{j=1}^{d} \omega_{jm} x_{ij}\right)\right)
\end{aligned}
$$

Gradient ($(K-1)\times(d+1)$ matrix, $\omega_{(0 \text{ or } j)K}$ doesn't exist)

$$\frac{\partial l(\omega)}{\partial \omega_{0k}} = \sum_i (1_{(y_i=y_k)} - \hat{p}(y_i = y_k|x_i,\omega))$$

$$\frac{\partial l(\omega)}{\partial \omega_{jk}} = \sum_i x_{ij}(1_{(y_i=y_k)} - \hat{p}(y_i = y_k|x_i,\omega))$$

## Outline

## Loss

**Definition**

Loss function:

$$L(y, f(x)) \text{ where } f : \mathcal{X} \to \mathbb{R} \quad L : \mathcal{X} \times \mathcal{Y} \times \mathbb{R} \to [0, \infty]$$

measures how good we are on a particular $(x, y)$ pair.

Loss examples

- $0-1$ loss: $L(y, f(x)) = \begin{cases} 1 & y \neq f(x) \\ 0 & y = f(x) \end{cases} = 1_{\{f(x) \neq y\}}$
- $L_1$, $L_2$ loss: $|y - f(x)|, (y - f(x))^2$
- Hinge loss: $\max(0, 1 - yf(x)) = |1 - yf(x)|_+$
- $\epsilon-$insensitive loss: $\max(0, |y - f(x)| - \epsilon)$

## Risk

**Definition**

Risk of $f$ function (The expected loss):

$$R_{L,p}(f) = \int_{\mathcal{X}\times\mathcal{Y}} L(y, f(x)) d_{p(x,y)} = \mathbb{E}[L(Y, f(X))]$$

Risk examples

- Risk of $0-1$ loss: $\mathbb{E}[1_{\{f(X)\neq Y\}}] = p(f(x) \neq y)$
- Risk of $L_2$ loss: $\mathbb{E}[(Y - f(X))^2]$

**Definition**

Bayes Risk (minimum):

$$R^*_{L,P} = \inf_{f:\mathcal{X}\to\mathbb{R}} R_{L,p}(f) = \inf_{f:\mathcal{X}\to\mathbb{R}} \int_{\mathcal{X}\times\mathcal{Y}} L(y, f(x)) d_{p(x,y)}$$

## Consistency of learning

In practive, $p(x, y)$ is unknow. Instead, training data $\mathcal{D}$ sampled from $p(x, y)$ is given.
Goal of learning:
Learn a function $f_{\mathcal{D}}$ whose risk is close to the Bayes risk $R^*_{L,p}$.
$\mathcal{D}$ is a random variable, so as $R_{L,p}(f_{\mathcal{D}})$.

**Definition**

Univerally Consistency:
A learning model is univerally consistent if for all distributions $p(x, y)$, the risk converges to the Bayes risk as the sample size $N$ increases.

$$R_{L,p}(f_{\mathcal{D}_N}) \xrightarrow{p} R^*_{L,p} \text{ as } N \to \infty$$

That $\forall \epsilon > 0 \quad \lim_{N\to\infty} p(|R_{L,p}(f_{\mathcal{D}_N}) - R^*_{L,p}| \leq \epsilon) = 0$

## Consistency of learning

Stone's theorem 1977:
Many classification (KNN, SVM), regression algorithms are universally consistent for certain loss functions under certain conditions.
Devroy 1982 (No Free Lunch):
For every consistent learning method and for every fixed convergence rate $h_n$, $\exists p(x, y)$ such that the convergence rate of this learning method on $p(x, y)$ distributed data is slower than $h_n$.

## Emipirical risk

The distribution $p(x, y)$ of data is unknown, instead we are given training data $\mathcal{D} = \{x_i, y_i\}_{i=1}^N$.

**Definition**

Empirical Risk:

$$\hat{R}_N(f) = \frac{1}{N} \sum_{i=1}^{N} L(y_i, f(x_i))$$

Empirical risk converges to risk:

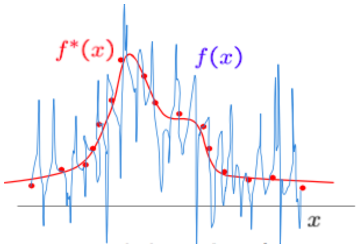
$$\lim_{N\to\infty} \frac{1}{N} \sum_{i=1}^{N} L(y_i, f(x_i)) = R_{L,p}(f) \quad \text{(Law of large numbers).}$$

## Overfitting

Optimize over all possible predictor $f : \mathcal{X} \to \mathbb{R}$ that

$$f_N^* = \arg \inf_{f:\mathcal{X}\to\mathbb{R}} \hat{R}_N(f) \text{ leads to overfitting.}$$

For example, overfitting in regression with ERM.



The empirical risk of $f(x)$ is zero, but the true risk of $f(x)$ is very high (large generalization error).
Prediction performance is very poor on new random test point.

## Overfitting solution

Restrict all possible predictors $f : \mathcal{X} \to \mathbb{R}$ to a function set $\mathcal{F}$.
ERM over the function set $\mathcal{F}$,

$$f^*_{N,\mathcal{F}} = \arg \inf_{f \in \mathcal{F}} \hat{R}_N(f).$$

Let

$$\hat{R}^*_{n,\mathcal{F}} = \inf_{f \in \mathcal{F}} \hat{R}_N(f) = \inf_{f \in \mathcal{F}} \frac{1}{N} \sum_{i=1}^{N} L(y_i, f(x_i)).$$

## Approximation and Estimation issues

1st issue

$$R^*_{\mathcal{F}} - R^* \geq 0$$

needs to be smalll. Solution: structural risk minimization (SRM)
Let $\mathcal{F}_n$ increases with the sample size $n\,(\mathcal{F}_{n+1} \supset \mathcal{F}_n)$, and let $\mathcal{F}_{n+1}$ contains more complex functions than $\mathcal{F}_n$.

2nd issue

$$\inf_{f \in \mathcal{F}} \hat{R}_N(f)$$

might be difficult to be optimized in $f$ (may be non-convex).
Solution: use loss function $L$ such that $\hat{R}_N(f)$ is convex in $f$
e.g. use convex hinge loss or $L_2$ loss to approximate $0 - 1$ non-convex loss.

## Approximation and Estimation Errors

Risk of a classifier $f$

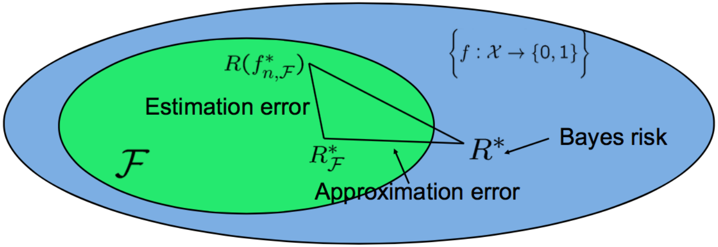
$$R(f) - R(f^*) = \underbrace{R(f) - \inf_{f \in \mathcal{F}} R(f)}_{\text{Estimation error}} + \underbrace{\inf_{f \in \mathcal{F}} R(f) - R(f^*)}_{\text{Approximation error}}$$

## Approximation and Estimation Errors

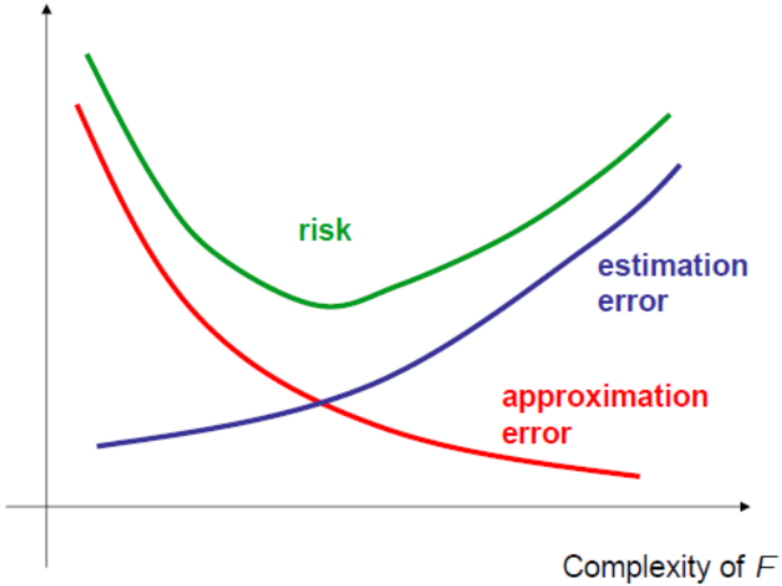
Ultimate goal: $R(f^*_{n,\mathcal{F}}) - R^* = 0$
Risk of $f^*_{n,\mathcal{F}}$

$$R(f^*_{n,\mathcal{F}}) - R^* = \underbrace{R(f^*_{n,\mathcal{F}}) - R^*_{\mathcal{F}}}_{\text{Estimation error}} + \underbrace{R^*_{\mathcal{F}} - R^*}_{\text{Approximation error}}$$

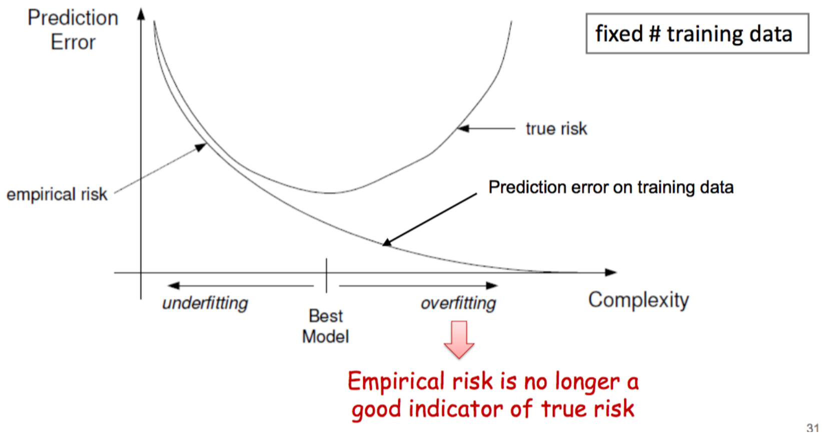
Big picture-$\mathcal{F}$ controls the model complexity



## Effect of Model Complexity



## Effect of Model Complexity



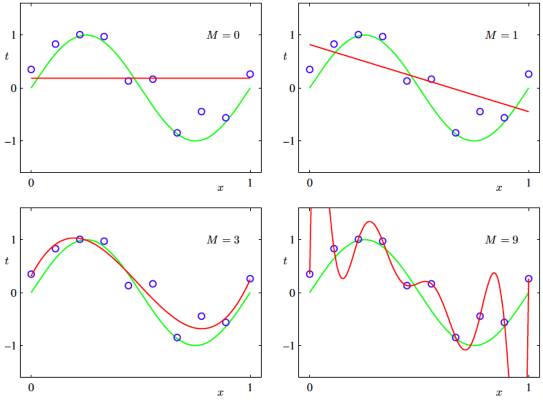
Empirical risk is no longer a good indicator of true risk

## Overfitting and Underfitting-Regerssion

Target value $\{t_n\}$ are obtained via:

$$t(x) = f^*(x) + \mathcal{N}(0, 0.3^2)$$

where $f^*(x) = sin(2\pi x)$.
Assume polynomail predictor for learning: $f_{\mathcal{F}}(x) = \sum_{j=0}^{M} \omega_j x^j$.



## Risk Bound Analysis

Come in future

## Loss Function

Risk of $f(x)$ under $L_2$ (square) loss:

$$\mathbb{E}[(Y - f(X))^2]$$

The function minimize the risk is:

$$f^*(x) = \arg\inf_f \int (y - f(x))^2 d_{p(x,y)}$$

Using calculus of variation to give

$$\frac{\partial \mathbb{E}}{\partial f(x)} = 2 \int (y - f(x)) p(x,y) d_y = 0$$

Solving for $f(x)$, we obtain

$$f^*(x) = \frac{\int y p(x,y) dy}{p(x)} = \mathbb{E}[Y|x]$$

## Expected Loss

We can expand the square loss as:

$$\begin{aligned}(f(x)-y)^2 &= (f(x)-f^*(x)+f^*(x)-y)^2 \\ &= (f(x)-f^*(x))^2 + 2(f(x)-f^*(x))(f^*(x)-y) \\ &\qquad + (f^*(x)-y)^2\end{aligned}$$

Substituting into the loss function and performing the integral over $y$, we obtain (the cross-term vanishes):

$$R(f) = \mathbb{E}[L] = \mathbb{E}[(f(X)-f^*(X))^2] + \underbrace{\mathbb{E}[(f^*(X)-Y)^2]}_{R^*}$$

The Bayes risk $R^*$ is independent of $f(x)$ which represents the irreducible minimum value of the loss function.

## Biase and Variance

Let $\hat{f}_{\mathcal{D},\mathcal{F}}$ (denoted by $\hat{f}$) learned from training data $\mathcal{D}$ over class $\mathcal{F}$. $\mathcal{D}$ is a random variable so as $\hat{f}_{\mathcal{D},\mathcal{F}}$

$$(\hat{f}(x) - f^*(x))^2 = (\hat{f}(x) - \mathbb{E}_{\mathcal{D}}[\hat{f}(x)] + \mathbb{E}_{\mathcal{D}}[\hat{f}(x)] - f^*(x))^2$$

Take the expectation with respect to $\mathcal{D}$ gives the bias and variance of a single input $x$

$$\mathbb{E}_{\mathcal{D}}[(\hat{f}(x) - f^*(x))^2] = \underbrace{(\mathbb{E}_{\mathcal{D}}[\hat{f}(x)] - f^*(x))^2}_{\text{(bias)}^2} + \underbrace{\mathbb{E}_{\mathcal{D}}[(\hat{f}(x) - \mathbb{E}_{\mathcal{D}}[\hat{f}(x)])^2]}_{\text{variance}}$$

## Biase and Variance

$$\text{Expected square loss} = (\text{bias})^2 + \text{variance} + \text{noise}$$

$$\begin{aligned}
R(\hat{f}_{\mathcal{D},\mathcal{F}}) &= \mathbb{E}_{X,Y,\mathcal{D}}[(\hat{f}_{\mathcal{D},\mathcal{F}}(X) - Y)^2] \\
&= \mathbb{E}_{X,\mathcal{D}}[(\hat{f}_{\mathcal{D},\mathcal{F}}(X) - f^*(X))^2] + \underbrace{\mathbb{E}_{X,Y}[(f^*(X) - Y)^2]}_{\textit{noise}} \\
&= \underbrace{\mathbb{E}_X[\mathbb{E}_{\mathcal{D}}[(\hat{f}_{\mathcal{D},\mathcal{F}}(X)] - f^*(X))^2]}_{(\text{bias})^2} \\
&\quad + \underbrace{\mathbb{E}_{X,\mathcal{D}}[(\hat{f}_{\mathcal{D},\mathcal{F}}(X) - \mathbb{E}_{\mathcal{D}}[(\hat{f}_{\mathcal{D},\mathcal{F}}(X)])^2]}_{\textit{variance}} + \text{noise}
\end{aligned}$$

- Large bias, small variance-poor approximation but stable
- Small bias, large variance-good approximation but instable

## Biase and Variance

Regularized least squares with Gaussian basis functions

$$\hat{R}_{N,\mathcal{F}} = \sum_{i=1}^{N} (y_i - \omega^T \Phi(x_i))^2 + \frac{\lambda}{2} \omega^T \omega \text{ where } \Phi_i(x) = e^{-\frac{(x-\mu_i)^2}{2s^2}}$$

## Illustration

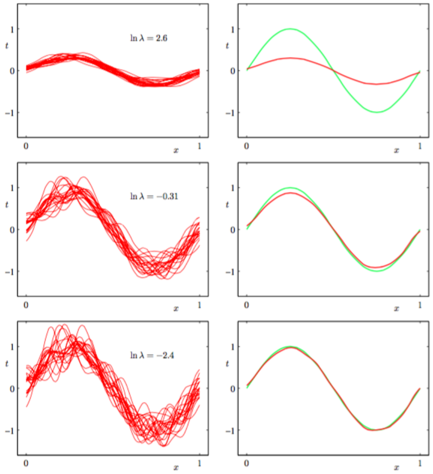
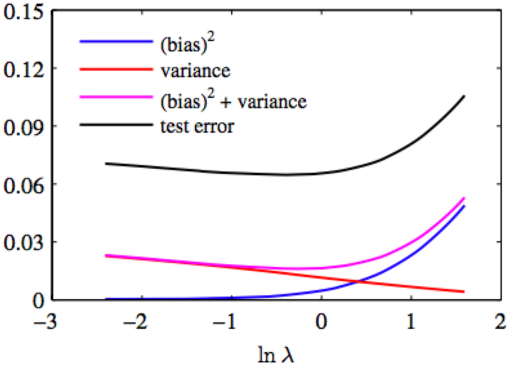


**Figure 3.5** Illustration of the dependence of bias and variance on model complexity, governed by a regularization parameter $\lambda$, using the sinusoidal data set from Chapter 1. There are $L = 100$ data sets, each having $N = 25$ data points, and there are 24 Gaussian basis functions in the model so that the total number of parameters is $M = 25$ including the bias parameter. The left column shows the result of fitting the model to the data sets for various values of $\ln\lambda$ (for clarity, only 20 of the 100 fits are shown). The right column shows the corresponding average of the 100 fits (red) along with the sinusoidal function from which the data sets were generated (green).

## Illustration

Plot of squared bias and variance, together with their sum, corresponding to the results shown in Figure 3.5. Also shown is the average test set error for a test data set size of 1000 points. The minimum value of $(\text{bias})^2 + \text{variance}$ occurs around $\ln \lambda = -0.31$, which is close to the value that gives the minimum error on the test data.



## Examples of Model Spaces

Model spaces with increasing complexity:

- Nearest neighbor classifiers with varying neighborhood sizes $k = 1, 2, 3, \cdots$

  Small neighborhood $\rightarrow$ Higher complexity

- Decision trees with depth $k$ or with $k$ leaves

  Higher depth/more leaves $\rightarrow$ Higher complexity

- Regression with polynomials of order $k = 0, 1, 2, \cdots$

  Higher degree $\rightarrow$ Higher complexity

- Kernel regression with bandwidth h

  Small bandwidth $\rightarrow$ Higher complexity

## Model Selection

Setup:
Model classes $\{\mathcal{F}_\lambda\}_{\lambda \in \Lambda}$ of increasing complexity $\mathcal{F}_1 \subset \mathcal{F}_2 \subset \cdots$
Select:
Right complexity model in a data-driven/adaptive way:

- Hold-out
- Cross-validation
- Information criteria-AIC, BIC, minimum description length (MDL)

## Hold-out

Hold-out procedure:
Suppose we are given dataset $\mathcal{D}$

1. Split dataset $\mathcal{D}$ into two sets Training set $\mathcal{D}_T$ and Validation set $\mathcal{D}_v$
2. Use $\mathcal{D}_T$ for training a predictor from each model class
$$\hat{f}_\lambda = \arg\min_{f \in \mathcal{F}_\lambda} \hat{R}_T(f) \quad \lambda \in \Lambda$$
3. Use $\mathcal{D}_v$ to select the model class giving the smallest empirical error
$$\hat{\lambda} = \arg\min_{\lambda \in \Lambda} \hat{R}_V(\hat{f}_\lambda)$$
4. Hold-out predictor
$$\hat{f} = \hat{f}_{\hat{\lambda}}$$

## Hold-out Drawbacks

Drawbacks:

- Data is not sufficient to afford setting one subset aside for getting a sense of generalization abilities
- Validation error may be misleading if we get an "unforunate" split

Limitations of hold-out can be overcome by a family of random sub-sampling methods at the expense of more computation

## Cross Validation

K-fold cross validation

- Create K-fold partition of the dataset
- Form K hold-out predictors, each time using one partition as validation and the rest K-1 as training datasets
- Final predictor is average/majority vote over the K hold-out estimates

Leave-one-out (LOO) cross-validation (special case of K-fold with $k = n$ partitions)

- Train on $n - 1$ samples and validate on only one sample per run for n runs

## Random subsampling

Random sbusampling

- Randomly subsample a fixed fraction $\alpha n (0 < \alpha < 1)$ of the dataset for validation.
- Form hold-out predictor with remaining data as training data.
- Repeat K times-final predictor is average/majority vote over the K hold-out estimate.