Bias and Variance of the Estimator

PRML 3.2
Ethem Chp. 4
In previous lectures we showed how to build classifiers when the underlying densities are known. Bayesian Decision Theory introduced the general formulation.

In most situations, however, the true distributions are unknown and must be estimated from data.

- **Parameter Estimation** (we saw the Maximum Likelihood Method)
  - Assume a particular form for the density (e.g., Gaussian), so only the parameters (e.g., mean and variance) need to be estimated
  - Maximum Likelihood
  - Bayesian Estimation

- **Non-parametric Density Estimation** (not covered)
  - Assume NO knowledge about the density
  - Kernel Density Estimation
  - Nearest Neighbor Rule
Bias and variance (1)

- How good are these estimates? Two measures of “goodness” are used for statistical estimates
  - **BIAS**: how close is the estimate to the true value?
  - **VARIANCE**: how much does the estimate change for different runs (e.g. different datasets)?

- The bias-variance tradeoff
  - In most cases, you can only decrease one of them at the expense of the other
How Good is an Estimator

- Assume our dataset $X$ is sampled from a population specified up to the parameter $\theta$; how good is an estimator $d(X)$ as an estimate for $\theta$?

- Notice that the estimate depends on sample set $X$

- If we take an expectation of the difference over different datasets $X$, $E_X[(d(X)-\theta)^2]$, and expand using the simpler notation of $E[d]= E[d(X)]$, we get:

$$E[(d(X)-\theta)^2] = E[(d(X)-E[d])^2] + (E[d] - \theta)^2$$

Using a simpler notation (dropping the dependence on $X$ from the notation – but knowing it exists):

$$E[(d-\theta)^2] = E[(d-E[d])^2] + (E[d] - \theta)^2$$
Properties of $\mu_{ML}$ and $\sigma^2_{ML}$

$$\mathbb{E}[\mu_{ML}] = \mu \quad \rightarrow \quad \mu_{ML} \text{ is an unbiased estimator}$$

$$\mathbb{E}[\sigma^2_{ML}] = \left( \frac{N - 1}{N} \right) \sigma^2 \quad \rightarrow \quad \sigma_{ML} \text{ is biased}$$

Use instead:

$$\bar{\sigma}^2 = \frac{N}{N - 1} \sigma^2_{ML}$$

$$= \frac{1}{N - 1} \sum_{n=1}^{N} (x_n - \mu_{ML})^2$$

(a)  

(b)  

(c)
Bias Variance Decomposition
Recall the expected squared loss,

\[ \mathbb{E}[L] = \int \{y(x) - \mathbb{E}[t|x]\}^2 p(x) \, dx + \int \text{var} \, [t|x] \, p(x) \, dx \]

Let's denote, for simplicity:

\[ h(x) = \mathbb{E}[t|x] = \int t \, p(t|x) \, dt. \]

We said that the second term corresponds to the noise inherent in the random variable \( t \).

What about the first term?
The Bias-Variance Decomposition (2)

- Suppose we were given multiple data sets, each of size N.
- Any particular data set, $D$, will give a particular function $y(x; D)$.
- Consider the error in the estimation:

$$\left\{ y(x; D) - h(x) \right\}^2$$

$$= \left\{ y(x; D) - \mathbb{E}_D [y(x; D)] + \mathbb{E}_D [y(x; D)] - h(x) \right\}^2$$

$$= \left\{ y(x; D) - \mathbb{E}_D [y(x; D)] \right\}^2 + \left\{ \mathbb{E}_D [y(x; D)] - h(x) \right\}^2$$

$$+ 2\left\{ y(x; D) - \mathbb{E}_D [y(x; D)] \right\}\left\{ \mathbb{E}_D [y(x; D)] - h(x) \right\}.$$
The Bias-Variance Decomposition (3)

\[
\{y(x; \mathcal{D}) - h(x)\}^2
= \{y(x; \mathcal{D}) - \mathbb{E}_\mathcal{D}[y(x; \mathcal{D})] + \mathbb{E}_\mathcal{D}[y(x; \mathcal{D})] - h(x)\}^2
= \{y(x; \mathcal{D}) - \mathbb{E}_\mathcal{D}[y(x; \mathcal{D})]\}^2 + \{\mathbb{E}_\mathcal{D}[y(x; \mathcal{D})] - h(x)\}^2
+ 2\{y(x; \mathcal{D}) - \mathbb{E}_\mathcal{D}[y(x; \mathcal{D})]\}\{\mathbb{E}_\mathcal{D}[y(x; \mathcal{D})] - h(x)\}.
\]

- Taking the expectation over \(\mathcal{D}\) yields:

\[
\mathbb{E}_\mathcal{D} \left[\{y(x; \mathcal{D}) - h(x)\}^2\right]
= \left\{\mathbb{E}_\mathcal{D}[y(x; \mathcal{D})] - h(x)\right\}^2 + \mathbb{E}_\mathcal{D} \left[\{y(x; \mathcal{D}) - \mathbb{E}_\mathcal{D}[y(x; \mathcal{D})]\}^2\right].
\]

\(\text{(bias)}^2 \quad \text{variance}\)
The Bias-Variance Decomposition (4)

- Thus we can write

\[ \text{expected loss} = (\text{bias})^2 + \text{variance} + \text{noise} \]

- where

\[
(bias)^2 = \int \{E_D[y(x; D)] - h(x)\}^2 p(x) \, dx
\]

\[
\text{variance} = \int E_D \{y(x; D) - E_D[y(x; D)]\}^2 \, p(x) \, dx
\]

\[
\text{noise} = \iint \{h(x) - t\}^2 p(x, t) \, dx \, dt
\]
- **Bias** measures how much the prediction (averaged over all data sets) differs from the desired regression function.

- **Variance** measures how much the predictions for individual data sets vary around their average.

There is a trade-off between bias and variance
- As we increase **model complexity**,
  - bias decreases (a better fit to data) and
  - variance increases (fit varies more with data)
(a) Function and data

(b) Order 1

(c) Order 3

(d) Order 5

bias

variance
Model Selection Procedures

1. **Regularization** (Breiman 1998): Penalize the augmented error:
   1. error on data + $\lambda$.model complexity
   1. If $\lambda$ is too large, we risk introducing bias
   2. Use cross validation to optimize for $\lambda$

2. **Structural Risk Minimization** (Vapnik 1995):
   1. Use a set of models ordered in terms of their complexities
      1. Number of free parameters
      2. VC dimension,…
   2. Find the best model w.r.t empirical error and model complexity.

3. **Minimum Description Length Principle**
4. **Bayesian Model Selection**: If we have some prior knowledge about the approximating function, it can be incorporated into the Bayesian approach in the form of $p(\text{model})$. 
Reminder: Introduction to Overfitting
PRML 1.1

Concepts: Polynomial curve fitting, overfitting, regularization, training set size vs model complexity
Polynomial Curve Fitting

\[ y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M = \sum_{j=0}^{M} w_j x^j \]
Sum-of-Squares Error Function

\[ E(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} \left( y(x_n, \mathbf{w}) - t_n \right)^2 \]
0th Order Polynomial

\[ M = 0 \]
1st Order Polynomial
3rd Order Polynomial

$M = 3$
9th Order Polynomial
Over-fitting

Root-Mean-Square (RMS) Error: \( E_{\text{RMS}} = \sqrt{2E(w^*)/N} \)
### Polynomial Coefficients

<table>
<thead>
<tr>
<th></th>
<th>$M = 0$</th>
<th>$M = 1$</th>
<th>$M = 3$</th>
<th>$M = 9$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$w_0^*$</td>
<td>0.19</td>
<td>0.82</td>
<td>0.31</td>
<td>0.35</td>
</tr>
<tr>
<td>$w_1^*$</td>
<td></td>
<td>-1.27</td>
<td>7.99</td>
<td>232.37</td>
</tr>
<tr>
<td>$w_2^*$</td>
<td></td>
<td>-25.43</td>
<td></td>
<td>-5321.83</td>
</tr>
<tr>
<td>$w_3^*$</td>
<td></td>
<td></td>
<td>17.37</td>
<td>48568.31</td>
</tr>
<tr>
<td>$w_4^*$</td>
<td></td>
<td></td>
<td></td>
<td>-231639.30</td>
</tr>
<tr>
<td>$w_5^*$</td>
<td></td>
<td></td>
<td></td>
<td>640042.26</td>
</tr>
<tr>
<td>$w_6^*$</td>
<td></td>
<td></td>
<td></td>
<td>-1061800.52</td>
</tr>
<tr>
<td>$w_7^*$</td>
<td></td>
<td></td>
<td></td>
<td>1042400.18</td>
</tr>
<tr>
<td>$w_8^*$</td>
<td></td>
<td></td>
<td></td>
<td>-557682.99</td>
</tr>
<tr>
<td>$w_9^*$</td>
<td></td>
<td></td>
<td></td>
<td>125201.43</td>
</tr>
</tbody>
</table>
One solution to control complexity is to penalize complex models -> regularization.
Regularization

- Use complex models, but penalize large coefficient values:

\[
\tilde{E}(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} \{y(x_n, \mathbf{w}) - t_n\}^2 + \frac{\lambda}{2} \| \mathbf{w} \|^2
\]
Regularization on 9th Order Polynomial

\[ \ln \lambda = -\infty \]

Too small \( \lambda \) – no regularization effect
Regularization on 9th degree polynomial:

$$\ln \lambda = -18$$

Right $\lambda$ – good fit
Regularization:

\[ \ln \lambda = 0 \]

Large \( \lambda \) –regularization dominates
Regularization: $E_{RMS}$ vs. $\ln \lambda$
### Polynomial Coefficients

<table>
<thead>
<tr>
<th>$w_i^*$</th>
<th>$\ln \lambda = -\infty$</th>
<th>$\ln \lambda = -18$</th>
<th>$\ln \lambda = 0$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$w_0^*$</td>
<td>0.35</td>
<td>0.35</td>
<td>0.13</td>
</tr>
<tr>
<td>$w_1^*$</td>
<td>232.37</td>
<td>4.74</td>
<td>-0.05</td>
</tr>
<tr>
<td>$w_2^*$</td>
<td>-5321.83</td>
<td>-0.77</td>
<td>-0.06</td>
</tr>
<tr>
<td>$w_3^*$</td>
<td>48568.31</td>
<td>-31.97</td>
<td>-0.05</td>
</tr>
<tr>
<td>$w_4^*$</td>
<td>-231639.30</td>
<td>-3.89</td>
<td>-0.03</td>
</tr>
<tr>
<td>$w_5^*$</td>
<td>640042.26</td>
<td>55.28</td>
<td>-0.02</td>
</tr>
<tr>
<td>$w_6^*$</td>
<td>-1061800.52</td>
<td>41.32</td>
<td>-0.01</td>
</tr>
<tr>
<td>$w_7^*$</td>
<td>1042400.18</td>
<td>-45.95</td>
<td>-0.00</td>
</tr>
<tr>
<td>$w_8^*$</td>
<td>-557682.99</td>
<td>-91.53</td>
<td>0.00</td>
</tr>
<tr>
<td>$w_9^*$</td>
<td>125201.43</td>
<td>72.68</td>
<td>0.01</td>
</tr>
</tbody>
</table>
The Bias-Variance Decomposition (5)

- Example: 100 data sets, each with 25 data points from the sinusoidal function $h(x) = \sin(2\pi x)$, varying the degree of regularization, $\lambda$. 

![Graphs showing the effect of regularization on the sinusoidal function.](image)
The Bias-Variance Decomposition (6)

- Regularization constant $\lambda = \exp\{-0.31\}$.
Regularization constant $\lambda = \exp\{-2.4\}$.
From these plots, we note that:

- an over-regularized model (large $\lambda$) will have a high bias
- while an under-regularized model (small $\lambda$) will have a high variance.

Minimum value of $\text{bias}^2 + \text{variance}$ is around $\lambda = -0.31$
This is close to the value that gives the minimum error on the test data.
Model Selection Procedures

**Cross validation**: Measure the total error, rather than bias/variance, on a validation set.

- Train/Validation sets
- K-fold cross validation
- Leave-One-Out
- No prior assumption about the models