Bias-Variance in Machine Learning
Bias-Variance: Outline

- Underfitting/overfitting:
  - Why are complex hypotheses bad?
- Simple example of bias/variance
- Error as bias+variance for regression
  - brief comments on how it extends to classification
- Measuring bias, variance and error
- Bagging - a way to reduce variance
- Bias-variance for classification
Bias/Variance is a Way to Understand Overfitting and Underfitting

Error/Loss on training set $D_{\text{train}}$

Error/Loss on an unseen test set $D_{\text{test}}$

“too simple”

“too complex”

high error

simple classifier

complex classifier
Bias-Variance: An Example
Example

\[ y = x + 2 \sin(1.5x) + N(0,0.2) \]
Example

Tom Dietterich, Oregon St

\[ y = x + 2 \sin(1.5x) + N(0,0.2) \]

Same experiment, repeated: with 50 samples of 20 points each
The true function $f$ can’t be fit perfectly with hypotheses from our class $H$ (lines) $\Rightarrow$ Error$_1$

Fix: *more* expressive set of hypotheses $H$

We don’t get the best hypothesis from $H$ because of noise/small sample size $\Rightarrow$ Error$_2$

Fix: *less* expressive set of hypotheses $H$
Bias-Variance Decomposition: Regression
Bias and variance for regression

• For regression, we can easily **decompose** the error of the learned model into two parts: bias (error 1) and variance (error 2)
  – Bias: the class of models can’t fit the data.
    • **Fix**: a *more expressive* model class.
  – Variance: the class of models *could* fit the data, but doesn’t because it’s hard to fit.
    • **Fix**: a *less expressive* model class.
Bias – Variance decomposition of error

\[ E_{D,\epsilon} \left\{ (f(x) + \epsilon - h_D(x))^2 \right\} \]

Fix test case \( x \), then do this experiment:

1. Draw size \( n \) sample \( D = (x_1, y_1), \ldots, (x_n, y_n) \)
2. Train linear regressor \( h_D \) using \( D \)
3. Draw one test example \( (x, f(x) + \epsilon) \)
4. Measure squared error of \( h_D \) on that example \( x \)

What’s the expected error?
Bias – Variance decomposition of error

Notation - to simplify this

\[ f \equiv f(x) + \varepsilon \]

\[ \hat{y} = \hat{y}_D \equiv h_D(x) \]

\[ E_{D,\varepsilon} \left\{ (f(x) + \varepsilon - h_D(x))^2 \right\} \]

- dataset and noise
- true function
- learned from D noise

\[ h \equiv E_D \{ h_D(x) \} \]

long-term expectation of learner’s prediction on this x averaged over many data sets D
Bias – Variance decomposition of error

$$E_{D,\varepsilon} \left\{ (f - \hat{y})^2 \right\}$$

$$= E \left\{ ([f - h] + [h - \hat{y}])^2 \right\}$$

$$= E \left\{ [f - h]^2 + [h - \hat{y}]^2 + 2[f - h][h - \hat{y}] \right\}$$

$$= E \left\{ [f - h]^2 + [h - \hat{y}]^2 + 2[fh - f\hat{y} - h^2 + h\hat{y}] \right\}$$

$$= E[(f - h)^2] + E[(h - \hat{y})^2] + 2\left(E[fh] - E[f\hat{y}] - E[h^2] + E[h\hat{y}]\right)$$

$$E_{D,\varepsilon} \left\{ (f(x) + \varepsilon) * E_D \{h_D(x)\} \right\}$$

$$= E_{D,\varepsilon} \left\{ (f(x) + \varepsilon) * h_D(x) \right\}$$

$$h \equiv E_D \{h_D(x)\}$$

$$\hat{y} = \hat{y}_D \equiv h_D(x)$$

$$f \equiv f(x) + \varepsilon$$

$$E_{D,\varepsilon} \left\{ E_D \{h_D(x)\} * E_D \{h_D(x)\} \right\}$$

$$= E_{D,\varepsilon} \left\{ E_D \{h_D(x)\} * h_D(x) \right\}$$
Bias – Variance decomposition of error

\[ E_{D,\epsilon} \left\{ (f - \hat{y})^2 \right\} \]
\[ = E \left\{ ([f - h] + [h - \hat{y}])^2 \right\} \]
\[ = E \left\{ [f - h]^2 + [h - \hat{y}]^2 + 2[f - h][h - \hat{y}] \right\} \]
\[ = E[(f - h)^2] + E[(h - \hat{y})^2] \]

\[ h \equiv E_D \{h_D(x)\} \]
\[ \hat{y} = \hat{y}_D \equiv h_D(x) \]
\[ f \equiv f(x) + \epsilon \]

Squared difference between best possible prediction for x, \( f(x) \), and our “long-term” expectation for what the learner will do if we averaged over many datasets D, \( E_D[h_D(x)] \).

Squared difference between our long-term expectation for the learners performance, \( E_D[h_D(x)] \), and what we expect in a representative run on a dataset D (\( \hat{y} \)).
bias

variance

x=5
Bias-variance decomposition

• This is something real that you can (approximately) measure experimentally
  – if you have synthetic data
• Different learners and model classes have different tradeoffs
  – large bias/small variance: few features, highly regularized, highly pruned decision trees, large-k k-NN…
  – small bias/high variance: many features, less regularization, unpruned trees, small-k k-NN…
Bias-Variance Decomposition: Classification
A generalization of bias-variance decomposition to other loss functions

• “Arbitrary” real-valued loss \( L(y,y') \)
  \( L(y,y') = L'(y,y), \ L(y,y) = 0, \) and \( L(y,y') = 0 \) if \( y = y' \)

• Define “optimal prediction”:
  \( y^* = \text{argmin}_y L(t,y') \)

• Define “main prediction of learner”
  \( y_m = y_{m,D} = \text{argmin}_y E_D(L(y,y')) \)

• Define “bias of learner”:
  \( \text{Bias}(x) = L(y^*, y_m) \)

• Define “variance of learner”
  \( \text{Var}(x) = E_D[L(y_m,y)] \)

• Define “noise for \( x \)”:
  \( N(x) = E_t[L(t,y^*)] \)

Claim:
\[
E_D,t[L(t,y) = c_1N(x) + \text{Bias}(x) + c_2\text{Var}(x) \]
where
\[
c_1 = Pr_D[y=y^*] - 1 \\
c_2 = 1 \text{ if } y_m = y^*, \ -1 \text{ else }
\]

For 0/1 loss, the main prediction is the most common class predicted by \( h_D(x) \), weighting \( h's \) by \( Pr(D) \)

Domingos, A Unified Bias-Variance Decomposition and its Applications, ICML 2000
Bias and variance

- For **classification**, we can also decompose the error of a learned classifier into two terms: bias and variance
  - **Bias**: the class of models *can’t* fit the data.
  - **Fix**: a *more expressive* model class.
  - **Variance**: the class of models *could* fit the data, but doesn’t because it’s hard to fit.
  - **Fix**: a *less expressive* model class.
Bias-Variance Decomposition: Measuring
Bias-variance decomposition

• This is something real that you can (approximately) measure experimentally
  – if you have synthetic data
  – …or if you’re clever

  – You need to somehow approximate $E_D\{h_D(x)\}$
  – I.e., construct many variants of the dataset $D$
Background: “Bootstrap” sampling

- **Input**: dataset $D$
- **Output**: many variants of $D$: $D_1, \ldots, D_T$
- For $t=1, \ldots, T$:
  - $D_t = \{ \}$
  - For $i=1 \ldots |D|$
    - Pick $(x,y)$ uniformly at random from $D$ (i.e., with replacement) and add it to $D_t$
    - Some examples never get picked ($\sim 37\%$)
    - Some are picked 2x, 3x, …
Measuring Bias-Variance with “Bootstrap” sampling

- Create B bootstrap variants of D (approximate many draws of D)
- For each bootstrap dataset
  - $T_b$ is the dataset; $U_b$ are the “out of bag” examples
  - Train a hypothesis $h_b$ on $T_b$
  - Test $h_b$ on each $x$ in $U_b$
- Now for each $(x,y)$ example we have many predictions $h_1(x), h_2(x), \ldots$ so we can estimate (ignoring noise)
  - **variance**: ordinary variance of $h_1(x), \ldots, h_n(x)$
  - **bias**: $\text{average}(h_1(x), \ldots, h_n(x)) - y$
Applying Bias-Variance Analysis

- By measuring the bias and variance on a problem, we can determine how to improve our model
  - If bias is high, we need to allow our model to be more complex
  - If variance is high, we need to reduce the complexity of the model

- Bias-variance analysis also suggests a way to reduce variance: bagging (later)
Bagging
Bootstrap Aggregation (Bagging)

- Use the bootstrap to create $B$ variants of $D$
- Learn a classifier from each variant
- Vote the learned classifiers to predict on a test example
Bagging (bootstrap aggregation)

• Breaking it down:
  – input: dataset $D$ and YFCL
  – output: a classifier $h_{D-BAG}$
  – use bootstrap to construct variants $D_1,\ldots,D_T$
  – for $t=1,\ldots,T$: train YFCL on $D_t$ to get $h_t$
  – to classify $x$ with $h_{D-BAG}$
    • classify $x$ with $h_1,\ldots,h_T$ and predict the most frequently predicted class for $x$ (majority vote)

Note that you can use any learner you like!

You can also test $h_t$ on the “out of bag” examples
Experiments

Freund and Schapire
Bagged, minimally pruned decision trees
Learning Curve of Californian Housing Data

Accuracy vs Sample Size

- Decision Tree
- Logistic Regression
Generally, bagged decision trees outperform the linear classifier eventually if the data is large enough and clean enough.

<table>
<thead>
<tr>
<th>Data set</th>
<th>Winner AUR</th>
<th>Winner Acc</th>
<th>Max-AUR</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nurse</td>
<td>none</td>
<td>none</td>
<td>1</td>
<td>Indistinguishable</td>
</tr>
<tr>
<td>Mushrooms</td>
<td>none</td>
<td>none</td>
<td>1</td>
<td>Indistinguishable</td>
</tr>
<tr>
<td>Optdigit</td>
<td>none</td>
<td>none</td>
<td>0.99</td>
<td>Indistinguishable</td>
</tr>
<tr>
<td>Letter–V</td>
<td>C4</td>
<td>C4</td>
<td>0.99</td>
<td>C4 dominates</td>
</tr>
<tr>
<td>Letter–A</td>
<td>C4</td>
<td>C4</td>
<td>0.99</td>
<td>C4 dominates</td>
</tr>
<tr>
<td>Intrusion</td>
<td>C4</td>
<td>C4</td>
<td>0.99</td>
<td>C4 dominates</td>
</tr>
<tr>
<td>DNA</td>
<td>C4</td>
<td>C4</td>
<td>0.99</td>
<td>C4 dominates</td>
</tr>
<tr>
<td>Covertype</td>
<td>C4</td>
<td>C4</td>
<td>0.99</td>
<td>C4 dominates</td>
</tr>
<tr>
<td>Telecom</td>
<td>C4</td>
<td>C4</td>
<td>0.98</td>
<td>C4 dominates</td>
</tr>
<tr>
<td>Pendigit</td>
<td>C4</td>
<td>C4</td>
<td>0.98</td>
<td>C4 dominates</td>
</tr>
<tr>
<td>Pageblock</td>
<td>C4</td>
<td>C4</td>
<td>0.98</td>
<td>C4 crosses</td>
</tr>
<tr>
<td>CarEval</td>
<td>none</td>
<td>C4</td>
<td>0.98</td>
<td>C4 crosses</td>
</tr>
<tr>
<td>Spam</td>
<td>C4</td>
<td>C4</td>
<td>0.97</td>
<td>C4 dominates</td>
</tr>
<tr>
<td>Chess</td>
<td>C4</td>
<td>C4</td>
<td>0.95</td>
<td>C4 dominates</td>
</tr>
<tr>
<td>CalHous</td>
<td>C4</td>
<td>C4</td>
<td>0.95</td>
<td>C4 dominates</td>
</tr>
<tr>
<td>Ailerons</td>
<td>none</td>
<td>C4</td>
<td>0.95</td>
<td>C4 crosses</td>
</tr>
<tr>
<td>Firm</td>
<td>LR</td>
<td>LR</td>
<td>0.93</td>
<td>C4 dominates</td>
</tr>
<tr>
<td>Credit</td>
<td>C4</td>
<td>C4</td>
<td>0.93</td>
<td>C4 dominates</td>
</tr>
<tr>
<td>Adult</td>
<td>LR</td>
<td>C4</td>
<td>0.9</td>
<td>Mixed</td>
</tr>
<tr>
<td>Connects</td>
<td>C4</td>
<td>none</td>
<td>0.87</td>
<td>C4 crosses</td>
</tr>
<tr>
<td>Move</td>
<td>C4</td>
<td>C4</td>
<td>0.85</td>
<td>C4 dominates</td>
</tr>
<tr>
<td>Downsize</td>
<td>C4</td>
<td>C4</td>
<td>0.85</td>
<td>C4 crosses</td>
</tr>
<tr>
<td>Coding</td>
<td>C4</td>
<td>C4</td>
<td>0.85</td>
<td>C4 crosses</td>
</tr>
<tr>
<td>German</td>
<td>LR</td>
<td>LR</td>
<td>0.8</td>
<td>LR dominates</td>
</tr>
<tr>
<td>Diabetes</td>
<td>LR</td>
<td>LR</td>
<td>0.8</td>
<td>LR dominates</td>
</tr>
<tr>
<td>Bookbinder</td>
<td>LR</td>
<td>LR</td>
<td>0.8</td>
<td>LR crosses</td>
</tr>
<tr>
<td>Bacteria</td>
<td>none</td>
<td>C4</td>
<td>0.79</td>
<td>C4 crosses</td>
</tr>
<tr>
<td>Yeast</td>
<td>none</td>
<td>none</td>
<td>0.78</td>
<td>Indistinguishable</td>
</tr>
<tr>
<td>Patent</td>
<td>C4</td>
<td>C4</td>
<td>0.75</td>
<td>Indistinguishable</td>
</tr>
<tr>
<td>Contra</td>
<td>none</td>
<td>none</td>
<td>0.73</td>
<td>Indistinguishable</td>
</tr>
<tr>
<td>IntShop</td>
<td>LR</td>
<td>LR</td>
<td>0.7</td>
<td>Indistinguishable</td>
</tr>
<tr>
<td>IntCensor</td>
<td>LR</td>
<td>LR</td>
<td>0.7</td>
<td>LR crosses</td>
</tr>
<tr>
<td>Insurance</td>
<td>none</td>
<td>none</td>
<td>0.7</td>
<td>Indistinguishable</td>
</tr>
<tr>
<td>IntPriv</td>
<td>LR</td>
<td>none</td>
<td>0.66</td>
<td>LR crosses</td>
</tr>
<tr>
<td>Mailing</td>
<td>LR</td>
<td>none</td>
<td>0.61</td>
<td>LR dominates</td>
</tr>
<tr>
<td>Abalone</td>
<td>LR</td>
<td>LR</td>
<td>0.56</td>
<td>LR dominates</td>
</tr>
</tbody>
</table>
Bagging (bootstrap aggregation)

• Experimentally:
  – especially with minimal pruning: decision trees have low bias but high variance.
  – bagging usually improves performance for decision trees and similar methods
  – It reduces variance without increasing the bias (much).
More detail on bias-variance and bagging for classification

Thanks Tom Dietterich MLSS 2014
A generalization of bias-variance decomposition to other loss functions

• “Arbitrary” real-valued loss $L(y, y')$
  But $L(y, y') = L(y', y)$, $L(y, y) = 0$, and $L(y, y')! = 0$ if $y = y'$

• Define “optimal prediction”:
  $y^* = \text{argmin}_y L(t, y')$

• Define “main prediction of learner”
  $y_m = y_{m,D} = \text{argmin}_y E_D\{L(y, y')\}$

• Define “bias of learner”:
  $\text{Bias}(x) = L(y^*, y_m)$

• Define “variance of learner”
  $\text{Var}(x) = E_D[L(y_m, y)]$

• Define “noise for x”:
  $N(x) = E_t[L(t, y^*)]$

Claim:
$E_D,t[L(t, y) = c_1 N(x) + \text{Bias}(x) + c_2 \text{Var}(x)$
where
$c_1 = Pr_D[y = y^*] - 1$
$c_2 = 1$ if $y_m = y^*$, -1 else

For 0/1 loss, the main prediction is the most common class predicted by $h_D(x)$, weighting $h$’s by $Pr(D)$

Domingos, A Unified Bias-Variance Decomposition and its Applications, ICML 2000
More detail on Domingos’ s model

- **Noisy channel:** \( y_i = \text{noise}(f(x_i)) \)
  - \( f(x_i) \) is true label of \( x_i \)
  - Noise \( \text{noise(.)} \) may change \( y \rightarrow y' \)
- \( h=h_D \) is learned hypothesis
  - from \( D=\{(x_1,y_1), \ldots, (x_m,y_m)\} \)
- for test case \( (x^*,y^*) \), and predicted label \( h(x^*) \), loss is \( L(h(x^*),y^*) \)
  - For instance, \( L(h(x^*),y^*) = 1 \) if error, else 0
More detail on Domingos’ s model

• We want to decompose $E_{D,P}\{L(h(x^*),y^*)\}$ where $m$ is size of $D$, $(x^*,y^*) \sim P$

• Main prediction of learner is $y_m(x^*)$
  $\quad - y_m(x^*) = \arg\min_y E_{D,P}\{L(h(x^*),y')\}$
  $\quad - y_m(x^*) = “most common” h_D(x^*)$ among all possible $D$’s, weighted by $\Pr(D)$

• Bias is $B(x^*) = L(y_m(x^*), f(x*))$

• Variance is $V(x^*) = E_{D,P}\{L(h_D(x^*), y_m(x^*))\}$

• Noise is $N(x^*) = L(y^*, f(x*))$
More detail on Domingos’ s model

• We want to decompose $E_{D,P}\{L(h(x^*),y^*)\}$
• Main prediction of learner is $y_m(x^*)$
  – “most common” $h_D(x^*)$ over $D$ ’s for 0/1 loss
• Bias is $B(x^*) = L(y_m(x^*) , f(x^*))$
  – main prediction vs true label
• Variance is $V(x^*) = E_{D,P}\{L(h_D(x^*) , y_m(x^*)) \}$
  – this hypothesis vs main prediction
• Noise is $N(x^*)= L(y^*, f(x^*))$
  – true label vs observed label
More detail on Domingos’ s model

• We will decompose $E_{D,P}\{L(h(x^*), y^*)\}$ into
  – *Bias* is $B(x^*) = L(y_m(x^*) , f(x^*))$
    • main prediction vs true label
    • this is 0/1, *not* a random variable
  – *Variance* is $V(x^*) = E_{D,P}\{L(h_D(x^*) , y_m(x^*) )\}$
    • this hypothesis vs main prediction
  – *Noise* is $N(x^*)= L(y^* , f(x^*))$
    • true label vs observed label
Case analysis of error

\[ f(x^*) = y^m? \]

- If yes, then:
  - \( y^m = h(x^*)? \)
    - If yes, then:
      - \( y^* = f(x^*)? \)
        - If yes, then: correct
        - If no, then: error [noise]
    - If no, then: error [variance]
  - If no, then: correct [noise cancels variance]

- If no, then:
  - \( y^m = h(x^*)? \)
    - If yes, then:
      - \( y^* = f(x^*)? \)
        - If yes, then: correct
        - If no, then: error [bias]
    - If no, then: correct [noise cancels bias]

- If no, then:
  - \( y^* = f(x^*)? \)
    - If yes, then: correct
    - If no, then: error [noise cancels variance cancels bias]
Analysis of error: unbiased case

- Let $P(y^* \neq f(x^*)) = N(x^*) = \tau$
- Let $P(y^m \neq h(x^*)) = V(x^*) = \sigma$
- If $(f(x^*) = y^m)$, then we suffer a loss if exactly one of these events occurs:
  
  $L(h(x^*), y^*) = \tau(1-\sigma) + \sigma(1-\tau)$
  
  $= \tau + \sigma - 2\tau\sigma$
  
  $= N(x^*) + V(x^*) - 2N(x^*)V(x^*)$
Analysis of error: biased case

- Let $P(y^* \neq f(x^*)) = N(x^*) = \tau$
- Let $P(y^m \neq h(x^*)) = V(x^*) = \sigma$
- If $f(x^*) \neq y^m$, then we suffer a loss if either both or neither of these events occurs:

$$L(h(x^*), y^*) = \tau \sigma + (1-\sigma)(1-\tau)$$
$$= 1 - (\tau + \sigma - 2\tau\sigma)$$
$$= B(x^*) - [N(x^*) + V(x^*) - 2 N(x^*) V(x^*)]$$

No noise, no variance

Main prediction is wrong

Noise and variance
Analysis of error: overall

\[ E[ L(h(x^*), y^*) ] = \]
if \( B(x^*) = 1 \):
\[ B(x^*) - [N(x^*) + V(x^*) - 2 N(x^*) V(x^*)] \]
if \( B(x^*) = 0 \):
\[ B(x^*) + [N(x^*) + V(x^*) - 2 N(x^*) V(x^*)] \]

Hopefully we’ll be in this case more often, if we’ve chosen a good classifier

Interaction terms are usually small
Analysis of error: without noise which is hard to estimate anyway

\[ E[ L(h(x^*), y^*) ] = \]
\[ \text{if } B(x^*) = 1: \ B(x^*) - V(x^*) \]
\[ \text{if } B(x^*) = 0: \ B(x^*) + V(x^*) \]

As with regression, we can experimentally approximately measure bias and variance with bootstrap replicates.

Typically break variance down into biased variance, \( V_b \), and unbiased variance, \( V_u \).
K-NN Experiments

- Chess (left): Increasing K primarily reduces Vu
- Audiology (right): Increasing K primarily increases B.
Glass (left), Primary tumor (right): deeper trees have lower B, higher Vu
Tree “stump” experiments (depth 2)

Bias is reduced (!)
Large tree experiments (depth 10)

Bias is not changed much.

Variance is reduced.