1. How do we pick \( d \) in polynomial regression?

One Idea:

\[
\begin{align*}
\hat{y}(0) &= w_0 + \epsilon \in \mathbb{R}^1 \\
\hat{y}(1) &= w_0 + w_1 x' \\
& \vdots \\
\hat{y}(d) &= w_0 + w_1 x' + \cdots + w_d x^d
\end{align*}
\]

\[ \hat{\theta}(d) = \arg\min_{\theta} \sum_{i=1}^{n} (y_i - \hat{y}(d))^2 \]

\[ d = \arg\min_{d=0,1,\ldots,100} \sum_{i=1}^{n} (y_i - \hat{y}(d))^2 \]

Won't work too well. Model classes are needed. \( d=100 \) will always give lowest training error.

All \( 10^\text{th} \) order polynomials

This is called the problem of "Model Selection".

\( \rightarrow \) How do I pick a model class to search in?
2. Types of Error

- Larger Model Classes will always do better on training error, but that's not what we care about.

What we really care about → Expected loss/Error

\[ X, Y \sim P(x, y) \text{ [unknown]} \]

\[ E_{P(x,y)} \left[ L(y, g(x; w)) \right] \]

all parameters to hyperparameters

What we ideally want to do

\[ \min_w \int \int L(y, g(x; w)) p(x, y) \, dx \, dy \]

Two problems

→ Integral hard to compute
→ \( p(x, y) \) unknown

So let's approximate integral with sample \((x_i, y_i) \sim P(x, y)\)

\[ E_{\text{approx}}(w) = \frac{1}{N} \sum_{i=1}^{N} L(y_i, g(x_i; w)) \]
So here's how we optimize various quantities:

---

<table>
<thead>
<tr>
<th>ALL DATA</th>
<th>TRAIN</th>
<th>VAL</th>
<th>TEST</th>
</tr>
</thead>
<tbody>
<tr>
<td>~60%</td>
<td>~20%</td>
<td>~20%</td>
<td></td>
</tr>
</tbody>
</table>

\( E_{\text{Train}} \) Used to fit model parameters

\( \hat{\theta} = \arg \min_{\theta} E_{\text{Train}} \)

\( \hat{\theta} \) Used to choose model classes

\( \hat{\theta} = \arg \min_{\theta} E_{\text{Eval}} \) (say \( d \))

Used to estimate

Expected Error/LOO,

No tweaking, No

booming on this,

otherwise becomes a biased estimate.

If not enough data to do: \((\text{train}, \text{val})\) split

we do cross-validation.

---

3. Overfitting vs Underfitting

\( E_{\text{Train}} \rightarrow \text{Model Complexity} \)

\( E_{\text{Eval}} \)

\( E_{\text{Train}} \rightarrow \text{high} \) \( \Rightarrow \) Underfitting

\( E_{\text{Train}} \rightarrow \text{low} \) \( \Rightarrow \) Overfitting
Overfitting $\Rightarrow$ Model class too large, too expressive
Need more assumptions, need smaller model class

Underfitting $\Rightarrow$ Model class too small, too simple
Need fewer assumptions

4. Bias - Variance

Error is high for both overfitting & underfitting
Can we differentiate?

Yes

Bias \quad Variance

Back to basics: Coin Toss

$D = \{x_1, \ldots, x_n\} \sim \text{Ber}(\theta^*)$ (say $\theta^* = 0.5$)

$$\text{IID}$$

<table>
<thead>
<tr>
<th>$D$</th>
<th>$\hat{\theta}_{MLE} = \frac{1}{N} \sum x_i$</th>
<th>$\hat{\theta}_{Bayes} = x_i$</th>
<th>$\hat{\theta}_{Silly} = 1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>${H, H, T}$</td>
<td>$2/3$</td>
<td>$1$</td>
<td>$1$</td>
</tr>
<tr>
<td>${T, T, H}$</td>
<td>$1/3$</td>
<td>$0$</td>
<td>$1$</td>
</tr>
<tr>
<td>${H, T, H}$</td>
<td>$2/3$</td>
<td>$1$</td>
<td>$1$</td>
</tr>
</tbody>
</table>

$$\mathbb{E}[\hat{\theta}_{MLE}] = \frac{1}{N} \sum \mathbb{E}[x_i] = \frac{1}{N} \sum_{i=1}^{N} [1 \cdot \theta^* + 0.6 \cdot (1 - \theta^*)]$$

$$= \frac{1}{N} \cdot N \cdot \theta^* = \theta^*$$
bias = \hat{E}[\hat{\theta} - \theta^*]

Difference between what your estimator aspode on average to the Truth.

\hat{\theta}_{MLE} is UNBIASED \Rightarrow E[\hat{\theta}_{MLE}] = \theta^*

What about others?

\[ E[\hat{\theta}_{easy}] = E[X] = \theta^* \] (Also Unbiased)

\[ E[\hat{\theta}_{silly}] = E[1] = 1 \] (Biased if \theta^* \neq 1)

\[ \text{Variance} \Rightarrow \text{No Variance} \]

\[ \text{Variance} = E[(\hat{\theta} - E[\hat{\theta}])^2] \]

P(\theta)

Low Bias Low Variance (This what we want)

Low Bias High Variance (Easy)

High Bias Low Variance (Silly)

High bias

High Variance

0

\theta^*

1
5. **Bias-Variance Decomposition for Squared Loss**

Say we want to estimate \( \theta^* \)

Let \( \hat{\theta} = E[\hat{\theta}] \)

Now \( E[\text{loss}] = E[(\hat{\theta} - \theta^*)^2] \)

\[ = E[(\hat{\theta} - \theta + \theta - \theta^*)^2] \]

\[ = E[(\hat{\theta} - \theta)^2 + (\theta - \theta^*)^2 - 2(\hat{\theta} - \theta)(\theta - \theta^*)] \]

\[ = E[(\hat{\theta} - \theta)^2] + E[(\theta - \theta^*)^2] - 2(\hat{\theta} - \theta)E[\theta - \theta^*] \]

\[ = \text{Var}(\hat{\theta}) + \text{Var}(\theta - \theta^*) - 2(\hat{\theta} - \theta)(E[\theta] - \theta^*) \]

\[ = \text{Var}(\hat{\theta}) + \text{bias}^2 \]

So \( E[\text{loss}] = \text{bias}^2 + \text{Var}(\hat{\theta}) \)

2 predictors (one with high bias, one with high loss can have the same \( L_2 \) loss)

6. **General Error Decomposition**

Expected Error = "Approximation Error" (\( \text{bias} \))

- you approximated reality with your model

+ "Estimation Error" (\( \text{variance} \))

- you estimated with finite data

+ "Optimization Error" (\( \text{longer bound on error} \))

+ "Bregman Error"
7. Model Selection via Regularization
   (it's put in a preference for simpler models)

\[
\min_{\mathbf{w}} \sum_{i=1}^{N} (y_i - \hat{y}(x_i; \mathbf{w}))^2 + \lambda \|\mathbf{w}\|^2
\]

Can also be viewed as a Gaussian prior on \(\mathbf{w}\) (MAP estimator)

8. Effect of Data / Learning Curves

Simple Model Class
   \(\Rightarrow\) More dat won't help

Too Expressive Model Class
   \(\Rightarrow\) More data will help, but very slowly

\[\text{Train} \quad \text{Eval} \quad \text{Data} \quad \text{Eval} \quad \text{Data} \quad \text{Eval} \quad \text{Data} \]