Supervised Learning

- Task = learn a function \( y : X \rightarrow T \) that maps input instances \( x \in X \) to output targets \( t \in T \):
  - Classification:
    - The output \( t \in T \) is one of a finite set of discrete categories.
  - Regression:
    - The output \( t \in T \) is continuous, or has a continuous component.

- Supervision = set of training examples:
  \((x_1, t_1), (x_2, t_2), \ldots, (x_n, t_n)\)
Regression: Curve Fitting

- Training: examples \((x_1, t_1), (x_2, t_2), \ldots (x_n, t_n)\)
Regression: Curve Fitting

- Testing: for arbitrary (unseen) instance $x \in X$, compute target output $y(x) = t \in T$. 
Polynomial Curve Fitting

\[ y(x) = y(x, w) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M = \sum_{j=0}^{M} w_j x^j \]
Polynomial Curve Fitting

• Learning = finding the “right” parameters $\mathbf{w}^T = [w_0, w_1, \ldots, w_M]$
  
  - Find $\mathbf{w}$ that minimizes an error function $E(\mathbf{w})$ which measures the misfit between $y(x_n, \mathbf{w})$ and $t_n$.
  
  - Expect that $y(x, \mathbf{w})$ performing well on training examples $x_n \Rightarrow y(x, \mathbf{w})$ will perform well on arbitrary test examples $x \in X$.

• Sum-of-Squares error function:

$$E(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} \{y(x_n, \mathbf{w}) - t_n\}^2$$

Inductive Learning Hypothesis

why squared?
Sum-of-Squares Error Function

• How do we find \( w^* \) that minimizes \( E(w) \)?
  \[
  w^* = \arg \min_w E(w)
  \]
Polynomial Curve Fitting

• \textit{Least Square} solution is found by solving a set of $M + 1$ linear equations:

\[
\sum_{j=0}^{M} A_{ij} w_j = T_i , \text{ where } A_{ij} = \sum_{n=1}^{N} x_n^{i+j}, \text{ and } T_i = \sum_{n=1}^{N} t_n x_n^i
\]

• \textit{Generalization} = how well the parameterized $y(x, w^*)$ performs on arbitrary (unseen) test instances $x \in X$.
  – Generalization performance depends on the value of $M$.  

Lecture 01
0\textsuperscript{th} Order Polynomial
1\textsuperscript{st} Order Polynomial

[M = 1]
3\textsuperscript{rd} Order Polynomial

\begin{align*}
M &= 3
\end{align*}
9\textsuperscript{th} Order Polynomial

\[ M = 9 \]
Polynomial Curve Fitting

- **Model Selection**: choosing the order $M$ of the polynomial.
  - Best generalization obtained with $M = 3$.
  - $M = 9$ obtains poor generalization, even though it fits training examples perfectly:
    - But $M = 9$ polynomials subsume $M = 3$ polynomials!

- **Overfitting** = good performance on training examples, poor performance on test examples.
Overfitting

- Measure fit to training/testing examples using the Root-Mean-Square (RMS) error: 
  \[ E_{RMS} = \sqrt{2E(w^*) / N} \]
- Use 100 random test examples, generated in the same way as the training examples.
### Over-fitting and Parameter Values

<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></td>
<td>-25.43</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>
Overfitting vs. Data Set Size

• More training data ⇒ less overfitting.
• What if we do not have more training data?
  – Use regularization.
  – Use a probabilistic model in a Bayesian setting.