Outline

1. Review of Ridge Regression
2. Review of Non-linear Basis Functions
3. Overfitting and Regularization
4. Hyperparameter Tuning and Cross-Validation
5. Bias-Variance Trade-off: Intuition
6. Bias-variance Trade-off: Analysis
Review of Ridge Regression
What if \( X^\top X \) is not invertible?

\[
W^{LMS} = (X^\top X)^{-1} X^\top y
\]

Why might this happen?

- **Answer 1:** \( N < D \). Not enough data to estimate all parameters.
- **Answer 2:** Columns of \( X \) are not linearly independent, e.g., some features are linear functions of other features. In this case, solution is not unique. Examples:
  - A feature is a re-scaled version of another, for example, having two features correspond to length in meters and feet respectively.
  - Same feature is repeated twice – could happen when there are many features.
  - A feature has the same value for all data points.
  - Sum of two features is equal to a third feature.
What if $X^\top X$ is not invertible?

$$w^{LMS} = (X^\top X)^{-1} X^\top y$$

Why might this happen?

- **Answer 1:** $N < D$. Not enough data to estimate all parameters.
What if $X^\top X$ is not invertible?

$$w^{LMS} = (X^\top X)^{-1} X^\top y$$

Why might this happen?

- **Answer 1:** $N < D$. Not enough data to estimate all parameters.
- **Answer 2:** Columns of $X$ are not linearly independent, e.g., some features are linear functions of other features. In this case, solution is not unique. Examples:
  - A feature is a re-scaled version of another, for example, having two features correspond to length in meters and feet respectively
  - Same feature is repeated twice – could happen when there are many features
  - A feature has the same value for all data points
  - Sum of two features is equal to a third feature
What if $X^\top X$ is not invertible?

$$w^{LMS} = (X^\top X)^{-1} X^\top y$$

Why might this happen?

- **Answer 1:** $N < D$. Not enough data to estimate all parameters.
- **Answer 2:** Columns of $X$ are not linearly independent, e.g., some features are linear functions of other features. In this case, solution is not unique. Examples:
  - A feature is a re-scaled version of another, for example, having two features correspond to length in meters and feet respectively.
What if $X^TX$ is not invertible?

$$w^{LMS} = (X^TX)^{-1}X^Ty$$

Why might this happen?

• **Answer 1:** $N < D$. Not enough data to estimate all parameters.

• **Answer 2:** Columns of $X$ are not linearly independent, e.g., some features are linear functions of other features. In this case, solution is not unique. Examples:
  - A feature is a re-scaled version of another, for example, having two features correspond to length in meters and feet respectively
  - Same feature is repeated twice – could happen when there are many features
What if $X^\top X$ is not invertible?

$$w^{LMS} = (X^\top X)^{-1} X^\top y$$

Why might this happen?

- **Answer 1:** $N < D$. Not enough data to estimate all parameters.
- **Answer 2:** Columns of $X$ are not linearly independent, e.g., some features are linear functions of other features. In this case, solution is not unique. Examples:
  - A feature is a re-scaled version of another, for example, having two features correspond to length in meters and feet respectively
  - Same feature is repeated twice – could happen when there are many features
  - A feature has the same value for all data points
  - Sum of two features is equal to a third feature
Example: Matrix $X^\top X$ is not invertible

<table>
<thead>
<tr>
<th>sqft (1000's)</th>
<th>bathrooms</th>
<th>sale price (100k)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>3.5</td>
</tr>
<tr>
<td>1.5</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>2.5</td>
<td>2</td>
<td>4.5</td>
</tr>
</tbody>
</table>
Example: Matrix $X^\top X$ is not invertible

<table>
<thead>
<tr>
<th>sqft (1000's)</th>
<th>bathrooms</th>
<th>sale price (100k)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>3.5</td>
</tr>
<tr>
<td>1.5</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>2.5</td>
<td>2</td>
<td>4.5</td>
</tr>
</tbody>
</table>

Design matrix and target vector:

$$X = \begin{bmatrix}
1 & 1 & 2 \\
1 & 2 & 2 \\
1 & 1.5 & 2 \\
1 & 2.5 & 2
\end{bmatrix}, \quad w = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \end{bmatrix}, \quad y = \begin{bmatrix} 2 \\ 3.5 \\ 3 \\ 4.5 \end{bmatrix}$$
Example: Matrix $X^\top X$ is not invertible

<table>
<thead>
<tr>
<th>sqft (1000's)</th>
<th>bathrooms</th>
<th>sale price (100k)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>3.5</td>
</tr>
<tr>
<td>1.5</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>2.5</td>
<td>2</td>
<td>4.5</td>
</tr>
</tbody>
</table>

Design matrix and target vector:

$$X = \begin{bmatrix} 1 & 1 & 2 \\ 1 & 2 & 2 \\ 1 & 1.5 & 2 \\ 1 & 2.5 & 2 \end{bmatrix}, \quad w = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \end{bmatrix}, \quad y = \begin{bmatrix} 2 \\ 3.5 \\ 3 \\ 4.5 \end{bmatrix}$$

The 'bathrooms' feature is redundant, so we don't need $w_2$

$$y = w_0 + w_1 x_1 + w_2 x_2$$

$$= w_0 + w_1 x_1 + w_2 \times 2, \text{ since } x_2 \text{ is always 2!}$$

$$= w_{0,\text{eff}} + w_1 x_1, \text{ where } w_{0,\text{eff}} = (w_0 + 2w_2)$$
Ridge regression

**Intuition:** what does a non-invertible $X^\top X$ mean?
Ridge regression

**Intuition:** what does a non-invertible $X^\top X$ mean?

Consider the SVD of this matrix:

$$X^\top X = V \begin{bmatrix} \lambda_1 & 0 & 0 & \cdots & 0 \\ 0 & \lambda_2 & 0 & \cdots & 0 \\ 0 & \cdots & \cdots & \cdots & 0 \\ 0 & \cdots & \cdots & \lambda_r & 0 \\ 0 & \cdots & \cdots & 0 & 0 \end{bmatrix} V^\top$$

where $\lambda_1 \geq \lambda_2 \geq \cdots \lambda_r > 0$ and $r < D$. We will have a divide by zero issue when computing $(X^\top X)^{-1}$.
**Intuition:** what does a non-invertible $X^T X$ mean? Consider the SVD of this matrix:

$$
X^T X = V \begin{bmatrix}
\lambda_1 & 0 & 0 & \cdots & 0 \\
0 & \lambda_2 & 0 & \cdots & 0 \\
0 & \cdots & \cdots & \cdots & 0 \\
0 & \cdots & \cdots & \lambda_r & 0 \\
0 & \cdots & \cdots & 0 & 0 \\
\end{bmatrix} V^T
$$

where $\lambda_1 \geq \lambda_2 \geq \cdots \lambda_r > 0$ and $r < D$. We will have a divide by zero issue when computing $(X^T X)^{-1}$

**Fix the problem:** ensure all singular values are non-zero:

$$
X^T X + \lambda I = V \text{diag}(\lambda_1 + \lambda, \lambda_2 + \lambda, \cdots, \lambda) V^T
$$

where $\lambda > 0$ and $I$ is the identity matrix.
Solution

\[ w = \left( X^T X + \lambda I \right)^{-1} X^T y \]
Regularized least square (ridge regression)

Solution

\[ w = (X^\top X + \lambda I)^{-1} X^\top y \]

This is equivalent to adding an extra term to \( RSS(w) \)

\[
\frac{1}{2} \left\{ w^\top X^\top X w - 2 (X^\top y)^\top w \right\} + \frac{1}{2} \lambda \| w \|^2 \\
\text{regularization}
\]
Regularized least square (ridge regression)

Solution

\[ w = \left( X^\top X + \lambda I \right)^{-1} X^\top y \]

This is equivalent to adding an extra term to \( \text{RSS}(w) \)

\[
\frac{1}{2} \left\{ w^\top X^\top X w - 2 (X^\top y)^\top w \right\} + \frac{1}{2} \lambda \| w \|_2^2
\]

regularization

Benefits

- Numerically more stable, invertible matrix
- Force \( w \) to be small
- Prevent overfitting — more on this later
Applying this to our example

<table>
<thead>
<tr>
<th>sqft (1000’s)</th>
<th>bathrooms</th>
<th>sale price (100k)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>3.5</td>
</tr>
<tr>
<td>1.5</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>2.5</td>
<td>2</td>
<td>4.5</td>
</tr>
</tbody>
</table>
Applying this to our example

<table>
<thead>
<tr>
<th>sqft (1000’s)</th>
<th>bathrooms</th>
<th>sale price (100k)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>3.5</td>
</tr>
<tr>
<td>1.5</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>2.5</td>
<td>2</td>
<td>4.5</td>
</tr>
</tbody>
</table>

The 'bathrooms' feature is redundant, so we don’t need \( w_2 \)

\[
y = w_0 + w_1 x_1 + w_2 x_2 \\
= w_0 + w_1 x_1 + w_2 \times 2, \quad \text{since } x_2 \text{ is always } 2! \\
= w_{0,\text{eff}} + w_1 x_1, \quad \text{where } w_{0,\text{eff}} = (w_0 + 2w_2) \\
= 0.45 + 1.6x_1 \quad \text{Should get this}
\]
Applying this to our example

The 'bathrooms' feature is redundant, so we don’t need $w_2$

\[ y = w_0 + w_1x_1 + w_2x_2 \]
\[ = w_0 + w_1x_1 + w_2 \times 2, \quad \text{since } x_2 \text{ is always } 2! \]
\[ = w_{0,\text{eff}} + w_1x_1, \quad \text{where } w_{0,\text{eff}} = (w_0 + 2w_2) \]
\[ = 0.45 + 1.6x_1 \quad \text{Should get this} \]

Compute the solution for $\lambda = 0.5$

\[
\begin{bmatrix}
  w_0 \\
  w_1 \\
  w_2 \\
\end{bmatrix}
= \left( X^\top X + \lambda I \right)^{-1} X^\top y
\]

\[
\begin{bmatrix}
  w_0 \\
  w_1 \\
  w_2 \\
\end{bmatrix}
= \begin{bmatrix}
  0.208 \\
  1.247 \\
  0.4166 \\
\end{bmatrix}
\]
How does $\lambda$ affect the solution?

$$\begin{bmatrix} w_0 \\ w_1 \\ w_2 \end{bmatrix} = \left( X^\top X + \lambda I \right)^{-1} X^\top y$$

Let us plot $w'_0 = w_0 + 2w_2$ and $w_1$ for different $\lambda \in [0.01, 20]$

Setting small $\lambda$ gives almost the least-squares solution, but it can cause numerical instability in the inversion
How to choose $\lambda$?

$\lambda$ is referred as *hyperparameter*

- Associated with the estimation method, not the dataset
- In contrast $w$ is the parameter vector
- Use validation set or cross-validation to find good choice of $\lambda$

![Graph showing the relationship between Hyperparameter $\lambda$ and Parameter Values $w_{0,eff}$ and $w_1$.]
1. Review of Ridge Regression

2. Review of Non-linear Basis Functions

3. Overfitting and Regularization

4. Hyperparameter Tuning and Cross-Validation

5. Bias-Variance Trade-off: Intuition

6. Bias-variance Trade-off: Analysis
Review of Non-linear Basis Functions
Outline

Review of Ridge Regression

Review of Non-linear Basis Functions

Overfitting and Regularization

Hyperparameter Tuning and Cross-Validation

Bias-Variance Trade-off: Intuition

Bias-variance Trade-off: Analysis
Is a linear modeling assumption always a good idea?

**Figure 1:** Sale price can saturate as sq. footage increases.

**Figure 2:** Temperature has cyclic variations over each year.
General nonlinear basis functions

We can use a nonlinear mapping:

$$\phi(x) : x \in \mathbb{R}^D \rightarrow z \in \mathbb{R}^M$$

- $M$ is dimensionality of new features $z$ (or $\phi(x)$)
- $M$ could be greater than, less than, or equal to $D$
General nonlinear basis functions

We can use a nonlinear mapping:

$$\phi(x) : x \in \mathbb{R}^D \rightarrow z \in \mathbb{R}^M$$

- $M$ is dimensionality of new features $z$ (or $\phi(x)$)
- $M$ could be greater than, less than, or equal to $D$

We can apply existing learning methods on the transformed data:

- linear methods: prediction is based on $w^T \phi(x)$
- other methods: nearest neighbors, decision trees, etc
Regression with nonlinear basis

Residual sum of squares

\[ \sum_n [\mathbf{w}^T \phi(x_n) - y_n]^2 \]

where \( \mathbf{w} \in \mathbb{R}^M \), the same dimensionality as the transformed features \( \phi(x) \).
Regression with nonlinear basis

Residual sum of squares

\[ \sum_n \left[ w^T \phi(x_n) - y_n \right]^2 \]

where \( w \in \mathbb{R}^M \), the same dimensionality as the transformed features \( \phi(x) \).

The LMS solution can be formulated with the new design matrix

\[
\Phi = \begin{pmatrix}
\phi(x_1)^T \\
\phi(x_2)^T \\
\vdots \\
\phi(x_N)^T
\end{pmatrix} \in \mathbb{R}^{N \times M}, \quad w^{\text{LMS}} = \left( \Phi^T \Phi \right)^{-1} \Phi^T y
\]
Example: Lot of Flexibility in Designing New Features!

<table>
<thead>
<tr>
<th>$x_1$, Area (1k sqft)</th>
<th>$x_1^2$, Area$^2$</th>
<th>Price (100k)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>3.5</td>
</tr>
<tr>
<td>1.5</td>
<td>2.25</td>
<td>3</td>
</tr>
<tr>
<td>2.5</td>
<td>6.25</td>
<td>4.5</td>
</tr>
</tbody>
</table>

**Figure 3:** Add $x_1^2$ as a feature to allow us to fit quadratic, instead of linear functions of the house area $x_1$
Example: Lot of Flexibility in Designing New Features!

<table>
<thead>
<tr>
<th>$x_1$, front (100ft)</th>
<th>$x_2$ depth (100ft)</th>
<th>10$x_1$x_2, Lot (1k sqft)</th>
<th>Price (100k)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5</td>
<td>0.5</td>
<td>2.5</td>
<td>2</td>
</tr>
<tr>
<td>0.5</td>
<td>1</td>
<td>5</td>
<td>3.5</td>
</tr>
<tr>
<td>0.8</td>
<td>1.5</td>
<td>12</td>
<td>3</td>
</tr>
<tr>
<td>1.0</td>
<td>1.5</td>
<td>15</td>
<td>4.5</td>
</tr>
</tbody>
</table>

**Figure 4:** Instead of having frontage and depth as two separate features, it may be better to consider the lot-area, which is equal to frontage×depth
Overfitting and Regularization
Outline

Review of Ridge Regression

Review of Non-linear Basis Functions

Overfitting and Regularization

Hyperparameter Tuning and Cross-Validation

Bias-Variance Trade-off: Intuition

Bias-variance Trade-off: Analysis
Non-linear Basis Functions: Polynomial Regression

Polynomial basis functions

\[ \phi(x) = \begin{bmatrix} 1 \\ x \\ x^2 \\ \vdots \\ x^M \end{bmatrix} \Rightarrow f(x) = w_0 + \sum_{m=1}^{M} w_m x^m \]
Polynomial basis functions

\[ \phi(x) = \begin{bmatrix} 1 \\ x \\ x^2 \\ \vdots \\ x^M \end{bmatrix} \Rightarrow f(x) = w_0 + \sum_{m=1}^{M} w_m x^m \]

Fitting samples from a sine function:
Non-linear Basis Functions: Polynomial Regression

Polynomial basis functions

\[ \phi(x) = \begin{bmatrix} 1 \\ x \\ x^2 \\ \vdots \\ x^M \end{bmatrix} \Rightarrow f(x) = w_0 + \sum_{m=1}^{M} w_m x^m \]

Fitting samples from a sine function:

![Graphs showing underfitting with different M values]
Non-linear Basis Functions: Polynomial Regression

**Polynomial basis functions**

\[ \phi(x) = \begin{bmatrix} 1 \\ x \\ x^2 \\ \vdots \\ x^M \end{bmatrix} \Rightarrow f(x) = w_0 + \sum_{m=1}^{M} w_m x^m \]

Fitting samples from a sine function:

Underfitting since \( f(x) \) is too simple.
Adding high-order terms

M=3

More complex features lead to better results on the training data, but potentially worse results on new data, e.g., test data!
Adding high-order terms

- **M=3**

- **M=9**: overfitting

More complex features lead to better results on the training data, but potentially worse results on new data, e.g., test data!
Adding high-order terms

M=3

M=9: overfitting

More complex features lead to better results on the training data, but potentially worse results on new data, e.g., test data!
Overfitting

Parameters for higher-order polynomials are very large

<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>-1.27</td>
<td>7.99</td>
<td>232.37</td>
<td></td>
</tr>
<tr>
<td>$w_2$</td>
<td></td>
<td>-25.43</td>
<td>-5321.83</td>
<td></td>
</tr>
<tr>
<td>$w_3$</td>
<td></td>
<td>17.37</td>
<td>48568.31</td>
<td></td>
</tr>
<tr>
<td>$w_4$</td>
<td></td>
<td></td>
<td>-231639.30</td>
<td></td>
</tr>
<tr>
<td>$w_5$</td>
<td></td>
<td></td>
<td>640042.26</td>
<td></td>
</tr>
<tr>
<td>$w_6$</td>
<td></td>
<td></td>
<td>-1061800.52</td>
<td></td>
</tr>
<tr>
<td>$w_7$</td>
<td></td>
<td></td>
<td>1042400.18</td>
<td></td>
</tr>
<tr>
<td>$w_8$</td>
<td></td>
<td></td>
<td>-557682.99</td>
<td></td>
</tr>
<tr>
<td>$w_9$</td>
<td></td>
<td></td>
<td>125201.43</td>
<td></td>
</tr>
</tbody>
</table>
Overfitting can be quite disastrous

Fitting the housing price data with large $M$:

Predicted price goes to zero (and is ultimately negative) if you buy a big enough house!

This is called poor generalization/overfitting.
Detecting overfitting

Plot model complexity versus objective function:

- X axis: model complexity, e.g., $M$
- Y axis: error, e.g., RSS, RMS (square root of RSS), 0-1 loss

Compute the objective on a training and test dataset.

As a model increases in complexity:

- Training error keeps improving
- Test error may first improve but eventually will deteriorate
Dealing with overfitting: Option 1

Try to use more training data

![Graph showing the effect of using more training data](image)
Dealing with overfitting: Option 1

Try to use more training data

But getting a lot of data can be expensive and time-consuming
Dealing with overfitting: Option 1

Try to use more training data

But getting a lot of data can be expensive and time-consuming
Dealing with overfitting: Option 2

Reduce the Number of Features

![Graphs showing the effect of reducing the number of features on model fit.](image)

- $M = 1$
- $M = 3$
- $M = 9$

May not know which and how many features to remove.
Dealing with overfitting: Option 2

Reduce the Number of Features

May not know which and how many features to remove
Dealing with Overfitting: Option 3

Regularization Methods: Give preference to ‘simpler’ models

- How do we define a simple linear regression model — $\mathbf{w}^\top \mathbf{x}$?
- Intuitively, the weights corresponding to higher order terms should not be “too large”

<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>-1.27</td>
<td>7.99</td>
<td>232.37</td>
<td></td>
</tr>
<tr>
<td>$w_2$</td>
<td>-25.43</td>
<td>-5321.83</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$w_3$</td>
<td>17.37</td>
<td></td>
<td>48568.31</td>
<td></td>
</tr>
<tr>
<td>$w_4$</td>
<td></td>
<td></td>
<td>-231639.30</td>
<td></td>
</tr>
<tr>
<td>$w_5$</td>
<td></td>
<td></td>
<td>640042.26</td>
<td></td>
</tr>
<tr>
<td>$w_6$</td>
<td></td>
<td></td>
<td>-1061800.52</td>
<td></td>
</tr>
<tr>
<td>$w_7$</td>
<td></td>
<td></td>
<td>1042400.18</td>
<td></td>
</tr>
<tr>
<td>$w_8$</td>
<td></td>
<td></td>
<td>-557682.99</td>
<td></td>
</tr>
<tr>
<td>$w_9$</td>
<td></td>
<td></td>
<td>125201.43</td>
<td></td>
</tr>
</tbody>
</table>
Add a term to the objective function.

- Choose the parameters to not just minimize risk, but avoid being too large.

\[
\frac{1}{2} \left\{ w^\top X^\top X w - 2 \left( X^\top y \right)^\top w \right\} + \frac{1}{2} \lambda \| w \|_2^2
\]
Regularization methods

Add a term to the objective function.

- Choose the parameters to not just minimize risk, but avoid being too large.

\[
\frac{1}{2} \left\{ w^\top X^\top X w - 2 (X^\top y)^\top w \right\} + \frac{1}{2} \lambda \| w \|_2^2
\]
Regularization methods

Add a term to the objective function.

- Choose the parameters to not just minimize risk, but avoid being too large.

\[
\frac{1}{2} \left\{ w^T X^T X w - 2 (X^T y)^T w \right\} + \frac{1}{2} \lambda \| w \|^2_2
\]

Advantages

- Forces the magnitude of \( w \) to be small
- Tries to find a simple model with few parameters
- Generalizes well to new data points
Example: Effect of Regularization

- Regularization makes the higher order $w_i$'s smaller
- Regularized polynomial fit will generalize much better
- As $\lambda$ increases, the model becomes simpler

![Graphs showing effect of regularization with $M=9, \lambda = 0.001$ and $M=9, \lambda = 1$.]
Probabilistic interpretation of Regularization

Linear Regression model: \( Y = \mathbf{w}^\top \mathbf{X} + \eta \)

\( \eta \sim N(0, \sigma_0^2) \) is a Gaussian random variable and \( Y \sim N(\mathbf{w}^\top \mathbf{X}, \sigma_0^2) \)
Linear Regression model: $Y = w^\top X + \eta$

$\eta \sim N(0, \sigma_0^2)$ is a Gaussian random variable and $Y \sim N(w^\top X, \sigma_0^2)$

Frequentist interpretation: We assume that $w$ is fixed.

- The likelihood function maps parameters to probabilities
  
  $$L : w, \sigma_0^2 \mapsto p(D|w, \sigma_0^2) = p(y|X, w, \sigma_0^2) = \prod_n p(y_n|x_n, w, \sigma_0^2)$$

- Maximizing the likelihood with respect to $w$ minimizes the RSS and yields the LMS solution:
  
  $$w_{\text{LMS}} = w_{\text{ML}} = \arg \max_w L(w, \sigma_0^2)$$
Proabilistic interpretation of Regularization

Regularized Regression model: \( Y = w^\top X + \eta \)
Probabilistic interpretation of Regularization

Regularized Regression model: \( Y = w^\top X + \eta \)

- \( Y \sim N(w^\top X, \sigma_0^2) \) is a Gaussian random variable (as before)
- \( w_d \sim N(0, \sigma^2) \) are i.i.d. Gaussian random variables (unlike before)
- Note that all \( w_d \) share the same variance \( \sigma^2 \)
Regularized Regression model: \( Y = \mathbf{w}^\top \mathbf{X} + \eta \)

- \( Y \sim N(\mathbf{w}^\top \mathbf{X}, \sigma_0^2) \) is a Gaussian random variable (as before)
- \( w_d \sim N(0, \sigma^2) \) are i.i.d. Gaussian random variables (\textit{unlike before})
- Note that all \( w_d \) share the same variance \( \sigma^2 \)

- To find \( \mathbf{w} \) given data \( \mathcal{D} \), compute the posterior distribution of \( \mathbf{w} \):
  \[
p(\mathbf{w}|\mathcal{D}) = \frac{p(\mathcal{D}|\mathbf{w})p(\mathbf{w})}{p(\mathcal{D})}
\]

- Maximum a posterior (MAP) estimate:
  \[
  \mathbf{w}^{\text{MAP}} = \arg \max_{\mathbf{w}} p(\mathbf{w}|\mathcal{D}) = \arg \max_{\mathbf{w}} p(\mathcal{D}|\mathbf{w})p(\mathbf{w})
  \]
Estimating $w$

Let $x_1, \ldots, x_N$ be i.i.d. with $y|x, w \sim N(w^\top x, \sigma_0^2)$; $w_d \sim N(0, \sigma^2)$. 

Joint likelihood of data and parameters (given $\sigma_0$, $\sigma$):

$$p(D, w) = p(D|w)p(w) = \prod_n p(y_n|x_n, w) \prod_d p(w_d)$$

Plugging in the Gaussian PDF, we get:

$$\log p(D, w) = \sum_n \log p(y_n|x_n, w) + \sum_d \log p(w_d)$$

$$= -\frac{1}{2\sigma_0^2} \sum_n (w^\top x_n - y_n)^2 - \frac{1}{2\sigma^2} \|w\|^2 + \text{const}$$

MAP estimate:

$$w_{\text{map}} = \arg \max_w \log p(D, w)$$

$$w_{\text{map}} = \arg \min_w \sum_n (w^\top x_n - y_n)^2 + \frac{1}{2\sigma^2} \|w\|^2$$
Estimating $w$

Let $x_1, \ldots, x_N$ be i.i.d. with $y \mid w, x \sim N(w^\top x, \sigma_0^2)$; $w_d \sim N(0, \sigma^2)$.

Joint likelihood of data and parameters (given $\sigma_0, \sigma$):

$$p(D, w) = p(D \mid w)p(w) = \prod_n p(y_n \mid x_n, w) \prod_d p(w_d)$$
Estimating $w$

Let $x_1, \ldots, x_N$ be i.i.d. with $y | w, x \sim N(w^\top x, \sigma_0^2)$; $w_d \sim N(0, \sigma^2)$.

**Joint likelihood of data and parameters (given $\sigma_0, \sigma$):**

$$p(D, w) = p(D | w)p(w) = \prod_n p(y_n | x_n, w) \prod_d p(w_d)$$

Plugging in the Gaussian PDF, we get:

$$\log p(D, w) = \sum_n \log p(y_n | x_n, w) + \sum_d \log p(w_d)$$

$$= -\frac{1}{2\sigma_0^2} \sum_n (w^\top x_n - y_n)^2 - \sum_d \frac{1}{2\sigma^2} w_d^2 + \text{const}$$
Estimating $w$

Let $x_1, \ldots, x_N$ be i.i.d. with $y \mid w, x \sim N(w^\top x, \sigma_0^2)$; $w_d \sim N(0, \sigma^2)$.

Joint likelihood of data and parameters (given $\sigma_0, \sigma$):

$$p(D, w) = p(D \mid w)p(w) = \prod_n p(y_n \mid x_n, w) \prod_d p(w_d)$$

Plugging in the Gaussian PDF, we get:

$$\log p(D, w) = \sum_n \log p(y_n \mid x_n, w) + \sum_d \log p(w_d)$$

$$= - \sum_n \frac{(w^\top x_n - y_n)^2}{2\sigma_0^2} - \sum_d \frac{1}{2\sigma^2} w_d^2 + \text{const}$$

MAP estimate: $w^{\text{MAP}} = \arg \max_w \log p(D, w)$

$$w^{\text{MAP}} = \arg \min_w \frac{\sum_n (w^\top x_n - y_n)^2}{2\sigma_0^2} + \frac{1}{2\sigma^2} \| w \|_2^2$$
Maximum a posterior (MAP) estimate

\[ E(w) = \sum_n (wx_n - y_n)^2 + \lambda \|w\|_2^2 \]

where \( \lambda > 0 \) is used to denote \( \sigma_0^2 / \sigma^2 \). This extra term \( \|w\|_2^2 \) is called regularization/regularizer and controls the magnitude of \( w \).

Intuitions

- If \( \lambda \to +\infty \), then \( \sigma_0^2 \gg \sigma^2 \): the variance of noise is far greater than what our prior model can allow for \( w \). In this case, our prior model on \( w \) will give a simpler model. Numerically,

\[ w_{\text{MAP}} \to 0 \]

- If \( \lambda \to 0 \), then we trust our data more. Numerically,

\[ w_{\text{MAP}} \to w_{\text{LMS}} = \arg\min \sum_n (w^\top x_n - y_n)^2 \]
Hyperparameter Tuning and Cross-Validation
Can we tune \( \lambda \) on the training dataset?
Can we tune $\lambda$ on the training dataset?

No: as this will always set $\lambda$ to zero, i.e., no regularization, defeating our intention of controlling model complexity.

$\lambda$ is thus a hyperparameter. To tune it,

- We can use a validation set or do cross validation.
- Pick the value of $\lambda$ that yields lowest error on the testing dataset.

Similar idea applies to tuning learning rate $\eta$ as well.
Training data are used to learn \( f(\cdot) \).

\( N \) samples/instances: \( D^{\text{TRAIN}} = \{(x_1, y_1), (x_2, y_2), \cdots, (x_N, y_N)\} \)

Test data are used to assess the prediction error.

- \( M \) samples/instances: \( D^{\text{TEST}} = \{(x_1, y_1), (x_2, y_2), \cdots, (x_M, y_M)\} \)
- They are used for assessing how well \( f(\cdot) \) will do in predicting an unseen \( x \notin D^{\text{TRAIN}} \)

Validation data are used to optimize hyperparameter(s).

\( L \) samples/instances: \( D^{\text{VAL}} = \{(x_1, y_1), (x_2, y_2), \cdots, (x_L, y_L)\} \)

Training data, validation and test data should **not** overlap!
Recipe

- For each possible value of the hyperparameter (say \( \lambda = 1, 3, \cdots, 100 \))
  - Train a model using \( D^{\text{TRAIN}} \)
  - Evaluate the performance of the model on \( D^{\text{VAL}} \)
- Choose the model with the best performance on \( D^{\text{VAL}} \)
- Evaluate the model on \( D^{\text{TEST}} \)
What if we do not have validation data?

- We split the training data into \( S \) equal parts.
- We use each part in turn as a validation dataset and use the others as a training dataset.
- We choose the hyperparameter such that the model performs the best (based on average, variance, etc.).

**Special case:** when \( S = N \), this will be leave-one-out.
Example: Hyper-parameter Tuning $\lambda$

- $\lambda = 10^{-4}$ gives the smallest validation loss
- Strikes a balance between bias and variance
Example: Hyper-parameter Tuning $M$

- Considering polynomial regression without regularization
- $M = 3$ or $M = 4$ gives the smallest validation loss
- Strikes a balance between bias and variance
Bias-Variance Trade-off: Intuition
Bias-Variance Trade-off: Intuition

- **High Bias:** Model is not rich enough to fit the training dataset and achieve low training loss
- **High Variance:** If the training dataset changes slightly, the model changes a lot
- Regularization helps find a middle ground

**Figure 6:** High Bias  
**Figure 7:** Just Right  
**Figure 8:** High Variance
Supervised learning

We aim to build a function \( h(x) \) to predict the true value \( y \) associated with \( x \). If we make a mistake, we incur a loss

\[
\ell(h(x), y)
\]

Example:

Quadratic loss function for regression when \( y \) is continuous:

\[
\ell(h(x), y) = [h(x) - y]^2
\]

Ex: when \( y = 0 \)
Empirical Risk Minimization

Given the true distribution of data $p(x, y)$, the risk is

$$R[h(x)] = \int_{x, y} \ell(h(x), y) p(x, y) dxdy$$

Key Elements of Risk

- **Limited Data (leads to High Variance):** We don’t know $p(x, y)$, but we have access to the training data $D$ which only allows us to optimize the empirical risk

$$R^{EMP}[h(x)] = \frac{1}{N} \sum_{n} \ell(h(x_n), y_n)$$

As $N \to +\infty$, $R^{EMP}[h(x)] \to R[h(x)]$

- **Limited Function Class (leads to High Bias):** The function $h(x)$ is restricted to a limited class (e.g. linear functions), which does not allow us to perfectly fit $y$, even if we had infinitely many training data points
Goal: to understand the sources of prediction errors

- $\mathcal{D}$: our training data
- $h_\mathcal{D}(x)$: our prediction function
  
  We are using the subscript $\mathcal{D}$ to indicate that the prediction function is learned on the specific set of training data $\mathcal{D}$
Goal: to understand the sources of prediction errors

- \( \mathcal{D} \): our training data
- \( h_\mathcal{D}(x) \): our prediction function
  We are using the subscript \( \mathcal{D} \) to indicate that the prediction function is learned on the specific set of training data \( \mathcal{D} \)
- \( \ell(h(x), y) \): our square loss function for regression

\[
\ell(h_\mathcal{D}(x), y) = [h_\mathcal{D}(x) - y]^2
\]
Bias/variance tradeoff for regression

Goal: to understand the sources of prediction errors

- $\mathcal{D}$: our training data
- $h_{\mathcal{D}}(x)$: our prediction function
  We are using the subscript $\mathcal{D}$ to indicate that the prediction function is learned on the specific set of training data $\mathcal{D}$
- $\ell(h(x), y)$: our square loss function for regression

$$\ell(h_{\mathcal{D}}(x), y) = [h_{\mathcal{D}}(x) - y]^2$$

- Unknown joint distribution $p(x, y)$
The effect of finite training samples

Every training sample $D$ is a sample from the following joint distribution

$$D \sim P(D) = \prod_{n=1}^{N} p(x_n, y_n)$$

Thus, the prediction function $h_D(x)$ is a random function with respect to this distribution. So is also its risk

$$R[h_D(x)] = \int_x \int_y (h_D(x) - y)^2 p(x, y) dx dy$$
The effect of finite training samples

Every training sample $\mathcal{D}$ is a sample from the following joint distribution

$$
\mathcal{D} \sim P(\mathcal{D}) = \prod_{n=1}^{N} p(x_n, y_n)
$$

Thus, the prediction function $h_{\mathcal{D}}(x)$ is a random function with respect to this distribution. So is also its risk

$$
R[h_{\mathcal{D}}(x)] = \int_x \int_y [h_{\mathcal{D}}(x) - y]^2 p(x, y) dx dy
$$

Next we will disentangle the impact of the finite sample $\mathcal{D}$ when assessing the quality of $h(\cdot)$
Bias-Variance Trade-off: Intuition

Error decomposes into 3 terms

\[ \mathbb{E}_D R[h_D(x)] = \text{VARIANCE} + \text{BIAS}^2 + \text{NOISE} \]

We will prove this result, and interpret what it means...

Figure 9: High Bias

Figure 10: Just Right

Figure 11: High Variance
Bias-variance Trade-off: Analysis
Average over the distribution of the training data

**Averaged risk**

\[
\mathbb{E}_D R[h_D(x)] = \int_D \int_x \int_y [h_D(x) - y]^2 p(x, y) dx dy \quad P(D)dD
\]

Namely, the randomness with respect to \( D \) is marginalized out.
Average over the distribution of the training data

Averaged risk

\[ \mathbb{E}_D R[h_D(x)] = \int_D \int_x \int_y [h_D(x) - y]^2 p(x, y) dx dy \, P(D) dD \]

Namely, the randomness with respect to \( D \) is marginalized out.

Averaged prediction

\[ \mathbb{E}_D h_D(x) = \int_D h_D(x) P(D) dD \]

Namely, if we have seen many training datasets, we predict with the average of the predict functions learned on each training dataset.
We will subtract the averaged prediction from the averaged risk

\[ \mathbb{E}_D R[h_D(x)] = \int_D \int_x \int_y [h_D(x) - y]^2 p(x, y) \, dx \, dy \, P(D) \, d'D \]

\[ = \int_D \int_x \int_y [h_D(x) - \mathbb{E}_D h_D(x)] \]
We will subtract the averaged prediction from the averaged risk

$$ \mathbb{E}_D R[h_D(x)] = \int_D \int_x \int_y [h_D(x) - y]^2 p(x, y) dx dy \ P(D) dD $$

$$ = \int_D \int_x \int_y [h_D(x) - \mathbb{E}_D h_D(x)] $$

$$ + \mathbb{E}_D h_D(x) - y]^2 p(x, y) dx dy \ P(D) dD $$
We will subtract the averaged prediction from the averaged risk

$$\mathbb{E}_D R[h_D(x)] = \int_D \int_x \int_y \left[ h_D(x) - y \right]^2 p(x, y) \, dx \, dy \, P(D) \, dD$$

$$= \int_D \int_x \int_y \left[ h_D(x) - \mathbb{E}_D h_D(x) \right] + \mathbb{E}_D h_D(x) - y \right]^2 p(x, y) \, dx \, dy \, P(D) \, dD$$

$$= \int_D \int_x \int_y \left[ h_D(x) - \mathbb{E}_D h_D(x) \right]^2 p(x, y) \, dx \, dy \, P(D) \, dD$$

\[ \text{VARIANCE} \]

$$+ \int_D \int_x \int_y \left[ \mathbb{E}_D h_D(x) - y \right]^2 p(x, y) \, dx \, dy \, P(D) \, dD$$
Where does the cross-term go?

It is zero

\[
\int_D \int_x \int_y \left[ h_D(x) - \mathbb{E}_D h_D(x) \right] \left[ \mathbb{E}_D h_D(x) - y \right] p(x, y) dx dy \ P(D) dD = 0
\]

← (the integral within the braces vanishes, by definition)
Where does the cross-term go?

It is zero

\[
\int_{\mathcal{D}} \int_{x} \int_{y} \left[ h_{\mathcal{D}}(x) - \mathbb{E}_{\mathcal{D}} h_{\mathcal{D}}(x) \right] \left[ \mathbb{E}_{\mathcal{D}} h_{\mathcal{D}}(x) - y \right] p(x, y) \, dx \, dy \, P(\mathcal{D}) \, d\mathcal{D}
\]

\[= \int_{x} \int_{y} \left\{ \int_{\mathcal{D}} \left[ h_{\mathcal{D}}(x) - \mathbb{E}_{\mathcal{D}} h_{\mathcal{D}}(x) \right] P(\mathcal{D}) \, d\mathcal{D} \right\} \left[ \mathbb{E}_{\mathcal{D}} h_{\mathcal{D}}(x) - y \right] p(x, y) \, dx \, dy
\]
Where does the cross-term go?

It is zero

\[
\int_D \int_x \int_y [h_D(x) - \mathbb{E}_D h_D(x)] [\mathbb{E}_D h_D(x) - y] p(x, y) dx dy \ P(D) dD
\]

\[
= \int_x \int_y \left\{ \int_D [h_D(x) - \mathbb{E}_D h_D(x)] P(D) dD \right\} [\mathbb{E}_D h_D(x) - y] p(x, y) dx dy
\]

\[
= 0 \leftarrow \text{(the integral within the braces vanishes, by definition)}
\]
Analyzing the variance

How can we reduce the variance?

\[
\int_D \int_x \int_y [h_D(x) - \mathbb{E}_D h_D(x)]^2 p(x, y) dx dy \ P(D)d\mathcal{D}
\]
Analyzing the variance

How can we reduce the variance?

\[ \int_{\mathcal{D}} \int_x \int_y [h_\mathcal{D}(x) - \mathbb{E}_\mathcal{D} h_\mathcal{D}(x)]^2 p(x, y) dx dy \ P(\mathcal{D}) d\mathcal{D} \]

- Use a lot of data (ie, increase the size of \( \mathcal{D} \))
- Use a simple \( h(\cdot) \) so that \( h_\mathcal{D}(x) \) does not vary much across different training datasets.
  Ex: \( h(x) = \text{const} \)
\[ E_D R[h_D(x)] = \int_D \int_x \int_y [h_D(x) - E_D h_D(x)]^2 p(x, y) dx dy \ P(D) dD + \int_D \int_x \int_y [E_D h_D(x) - y]^2 p(x, y) dx dy \ P(D) dD \]
The remaining item

\[ \mathbb{E}_D R[h_D(x)] = \int_D \int_x \int_y \left[ h_D(x) - \mathbb{E}_D h_D(x) \right]^2 p(x, y) \, dx \, dy \, P(D) \, dD \\
+ \int_D \int_x \int_y \left[ \mathbb{E}_D h_D(x) - y \right]^2 p(x, y) \, dx \, dy \, P(D) \, dD \]

The integrand has no dependency on \(D\) anymore and simplifies to

\[ \int_x \int_y \left[ \mathbb{E}_D h_D(x) - y \right]^2 p(x, y) \, dx \, dy \]
The remaining item

\[
\mathbb{E}_D R[h_D(x)] = \int_D \int_x \int_y \left[ h_D(x) - \mathbb{E}_D h_D(x) \right]^2 p(x, y) dx dy \ P(D) dD \\
+ \int_D \int_x \int_y \left[ \mathbb{E}_D h_D(x) - y \right]^2 p(x, y) dx dy \ P(D) dD
\]

The integrand has no dependency on \( D \) anymore and simplifies to

\[
\int_x \int_y \left[ \mathbb{E}_D h_D(x) - y \right]^2 p(x, y) dx dy
\]

We will apply a similar trick, by using an averaged target \( y \)

\[
\mathbb{E}_y [y] = \int_y y p(y|x) dy
\]
Bias and noise

Decompose again

\[ \int_x \int_y \left[ E_D h_D(x) - y \right]^2 p(x, y) dx dy \]

\[= \int_x \int_y \left[ E_D h_D(x) - E_y[y] + E_y[y] - y \right]^2 p(x, y) dx dy \]

\[= \int_x \int_y \left[ E_D h_D(x) - E_y[y] \right]^2 p(x, y) dx dy \]

\[+ \int_x \int_y \left[ E_y[y] - y \right]^2 p(x, y) dx dy \]

\[\text{BIAS}^2 \]

\[\text{NOISE} \]

Where is the cross-term?

Take-home exercise: Show that it is zero
Analyzing the noise

How can we reduce noise?

\[
\int_x \int_y [\mathbb{E}_y[y] - y]^2 p(x, y) \, dx \, dy = \int_x \left( \int_y [\mathbb{E}_y[y] - y]^2 p(y|x) \, dy \right) p(x) \, dx
\]

There is nothing we can do. This quantity depends on \( p(x, y) \) only; choosing \( h(\cdot) \) or the training dataset \( \mathcal{D} \) will not affect it. Note that the integral inside the parentheses is the variance (noise) of the posterior distribution \( p(y|x) \) at the given \( x \).

**Figure 12:** Somewhat difficult posterior

**Figure 13:** Somewhat easy posterior
Analyzing the bias term

How can we reduce bias?

\[ \int_x \int_y \left[ \mathbb{E}_D h_D(x) - \mathbb{E}_y [y] \right]^2 p(x, y) \, dx \, dy = \int_x \left[ \mathbb{E}_D h_D(x) - \mathbb{E}_y [y] \right]^2 p(x) \, dx \]
Analyzing the bias term

How can we reduce bias?

$$\int_x \int_y \left[ \mathbb{E}_D h_D(x) - \mathbb{E}_y[y] \right]^2 p(x, y) dx dy = \int_x \left[ \mathbb{E}_D h_D(x) - \mathbb{E}_y[y] \right]^2 p(x) dx$$

It can be reduced by using more complex models. We shall choose $h(\cdot)$ to be as flexible as possible: the better $h(\cdot)$ approximates $\mathbb{E}_y[y]$, the smaller the bias. However, this will increase the VARIANCE term.
Bias/variance tradeoff

Error decomposes into 3 terms

$$E_D R[h_D(x)] = \text{VARIANCE} + \text{BIAS}^2 + \text{NOISE}$$

where the first and the second term are inherently in conflict in terms of choosing what kind of $h(x)$ we should use (unless we have an infinite amount of data).
Bias/variance tradeoff

Error decomposes into 3 terms

\[ \mathbb{E}_D R[h_D(x)] = \text{VARIANCE} + \text{BIAS}^2 + \text{NOISE} \]

where the first and the second term are inherently in conflict in terms of choosing what kind of \( h(x) \) we should use (unless we have an infinite amount of data).

If we can compute all terms analytically, they will look like this
Summary of risk components

The average risk (with quadratic loss) can be decomposed as:

$$\mathbb{E}_D R[h_D(x)] = \int_D \int_x \int_y [h_D(x) - \mathbb{E}_D h_D(x)]^2 p(x, y) dx dy \ P(D) dD$$

VARIANCE: error due to training dataset

$$+ \int_x \int_y [\mathbb{E}_D h_D(x) - \mathbb{E}_y [y]]^2 p(x, y) dx dy$$

BIAS$^2$: error due to the model approximation

$$+ \int_x \int_y [\mathbb{E}_y [y] - y]^2 p(x, y) dx dy$$

NOISE: error due to randomness of $y$
Summary of risk components

The average risk (with quadratic loss) can be decomposed as:

\[
\mathbb{E}_D R[h_D(x)] = \int_D \int_x \int_y [h_D(x) - \mathbb{E}_D h_D(x)]^2 p(x, y) \, dx \, dy \, P(D) \, dD
\]

**VARIANCE:** error due to training dataset

\[+
\int_x \int_y [\mathbb{E}_D h_D(x) - \mathbb{E}_y [y]]^2 p(x, y) \, dx \, dy
\]

**BIAS^2:** error due to the model approximation

\[+
\int_x \int_y [\mathbb{E}_y [y] - y]^2 p(x, y) \, dx \, dy
\]

**NOISE:** error due to randomness of \(y\)

Here we define: \(h_D(x)\) as the output of the model trained on \(D\), \(\mathbb{E}_D h_D(x)\) as the expectation of the model over all datasets \(D\), and \(\mathbb{E}_y [y]\) as the expected value of \(y\).
Example: why regularized linear regression could be helpful?

Model

\[ h(x) = w^\top x \]

Consider the best possible (linear) \( h^*(x) \)

\[ w^* = \arg\min_w \int_x \left[ \mathbb{E}_y[y] - w^\top x \right]^2 p(x) dx \]

Note that this linear model assumes the knowledge of joint distribution, thus, not achievable. Intuitively, it is the best linear model that can predict the data most accurately.
More refined decomposition of the bias

\[ \int_{x} \left[ \mathbb{E}_{D} h_{D}(x) - \mathbb{E}_{y}[y] \right]^2 p(x) \, dx = \int_{x} \left[ h^{*}(x) - \mathbb{E}_{y}[y] \right]^2 p(x) \, dx \]

\[ + \int_{x} \left[ \mathbb{E}_{D} h_{D}(x) - h^{*}(x) \right]^2 p(x) \, dx \]

- **Model bias:** the price we pay for choosing linear functions to model data
- **Estimation bias:** the difference between the optimal model and the estimated model

*Normally, the estimation bias is zero if we do not regularize.*
We can only adjust estimation bias

\[
\int_x [E_D h_D(x; \lambda) - h^*(x)]^2 p(x) dx
\]

where \( h(x; \lambda) \) is the estimated model with regularized linear regression (parameterized with \( \lambda \)).

This term will not be zero anymore!

Thus, bias goes up.

But, as long as this is balanced with a decrease in variance, we are willing to do so.
Visualizing the tradeoff

- Reality
- Closest fit in population
- Closest fit
- Model bias
- Estimation Bias
- Model variance
- Shrunken fit
- Restricted model space