Linear Regression and Linear Basis Function Model

Kaushik Sinha

September 22, 2014
A Supervised Learning Problem

Suppose we have a dataset of living area size and monthly rents of apartments in Wichita Area for a number of different apartments.

<table>
<thead>
<tr>
<th>Living area (feet(^2))</th>
<th>Price (1000$s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2104</td>
<td>400</td>
</tr>
<tr>
<td>1600</td>
<td>330</td>
</tr>
<tr>
<td>2400</td>
<td>369</td>
</tr>
<tr>
<td>1416</td>
<td>232</td>
</tr>
<tr>
<td>3000</td>
<td>540</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Suppose we have a dataset of living area size and monthly rents of apartments in Wichita Area for a number of different apartments.

<table>
<thead>
<tr>
<th>Living area (feet$^2$)</th>
<th>Price (1000$$s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2104</td>
<td>400</td>
</tr>
<tr>
<td>1600</td>
<td>330</td>
</tr>
<tr>
<td>2400</td>
<td>369</td>
</tr>
<tr>
<td>1416</td>
<td>232</td>
</tr>
<tr>
<td>3000</td>
<td>540</td>
</tr>
<tr>
<td>:</td>
<td>:</td>
</tr>
</tbody>
</table>

![Graph showing housing prices vs. square feet]
A Supervised Learning Problem

- Suppose we have a dataset of living area size and monthly rents of apartments in Wichita Area for a number of different apartments

<table>
<thead>
<tr>
<th>Living area (feet²)</th>
<th>Price (1000$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2104</td>
<td>400</td>
</tr>
<tr>
<td>1600</td>
<td>330</td>
</tr>
<tr>
<td>2400</td>
<td>369</td>
</tr>
<tr>
<td>1416</td>
<td>232</td>
</tr>
<tr>
<td>3000</td>
<td>540</td>
</tr>
<tr>
<td>:</td>
<td>:</td>
</tr>
</tbody>
</table>

- Given a dataset like this, how can we learn to predict the monthly rent of an apartment in Wichita area as a function of its living area size?
First observe that the outputs are scalar values; hence, this is not a classification but is a regression problem.

Suppose we are given \( N \) input-output pairs \((x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\) where \( x_i \in \mathbb{R}^n \), \( y_i \in \mathbb{R} \) for \( i = 1, \ldots, N \).

In a linear regression model, dependence between \( x_i \) and \( y_i \) is assumed to be linear. This relationship is modeled through a noise term or error variable \( \epsilon_i \) each \( y_i \) can be modeled as:

\[
y_i = x_i^T w + \epsilon_i
\]

where \( w = (w_1, \ldots, w_n) \in \mathbb{R}^n \) and each \( \epsilon_i \) is unobserved and is a zero mean random variable, e.g., a \( N(0, \sigma) \) for some \( \sigma \in \mathbb{R} \).
Linear Regression Model

- First observe that the outputs are scalar values
Linear Regression Model

- First observe that the outputs are scalar values
  - hence, this is not a classification but is a **regression** problem
Linear Regression Model

- First observe that the outputs are scalar values
  - hence, this is not a classification but is a **regression** problem

- Suppose we are given $N$ input-output pairs
  \[(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\] where $x_i \in \mathbb{R}^n$, $y_i \in \mathbb{R}$ for $i = 1, \ldots, N$
First observe that the outputs are scalar values
  
  hence, this is not a classification but is a **regression** problem

Suppose we are given \( N \) input-output pairs

\[
(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)
\]

where \( x_i \in \mathbb{R}^n \), \( y_i \in \mathbb{R} \) for \( i = 1, \ldots, N \)

In a **linear regression model**, dependence between \( x_i \) and \( y_i \) is assumed to be linear
Linear Regression Model

- First observe that the outputs are scalar values
  - hence, this is not a classification but is a regression problem
- Suppose we are given $N$ input-output pairs $(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)$ where $x_i \in \mathbb{R}^n$, $y_i \in \mathbb{R}$ for $i = 1, \ldots, N$
- In a linear regression model, dependence between $x_i$ and $y_i$ is assumed to be linear
  - this relationship is modeled through a noise term or error variable $\epsilon_i$
Linear Regression Model

- First observe that the outputs are scalar values
  - hence, this is not a classification but is a regression problem
- Suppose we are given $N$ input-output pairs
  $$(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)$$
  where $x_i \in \mathbb{R}^n$, $y_i \in \mathbb{R}$ for $i = 1, \ldots, N$
- In a linear regression model, dependence between $x_i$ and $y_i$ is assumed to be linear
  - this relationship is modeled through a noise term or error variable $\epsilon_i$
  - each $y_i$ can be modeled as
    $$y_i = \sum_{j=1}^{n} x_{ij} w_j + \epsilon_i = x_i^T w + \epsilon_i$$
    where $w = (w_1, \ldots, w_n) \in \mathbb{R}^n$ and each $\epsilon_i$ is unobserved and is zero mean random variable, e.g., $N(0, \frac{1}{\sigma})$ for some $\sigma \in \mathbb{R}$
What can we hope to learn in such a model?

Since $\epsilon_i$ are unobserved, the best we can hope for is to learn a parameter vector $\hat{w} = (\hat{w}_1, \ldots, \hat{w}_n) \in \mathbb{R}^n$ that minimizes the sum of errors between $y_i$ and $\sum_{j=1}^{n} x_{ij} w_j$ over all possible $w$ when we measure the error in an appropriate way.

What can be a good way of measuring error?

In linear regression model we consider square error as the choice of error function.

$$\hat{w} = \arg \min_{w \in \mathbb{R}^n} \sum_{i=1}^{N} \left( y_i - \sum_{j=1}^{n} x_{ij} w_j \right)^2$$
What can we hope to learn in such a model?
What can we hope to learn in such a model?

- since $\epsilon_i$ are unobserved, the best we can hope for is to learn a parameter vector $\hat{\mathbf{w}} = (\hat{w}_1, \ldots, \hat{w}_n) \in \mathbb{R}^n$ that minimizes the sum of errors between $y_i$ and $\sum_{j=1}^n x_{ij} w_j$ over all possible $\mathbf{w}$ when we measure the error in an appropriate way.
What can we hope to learn in such a model?

- Since $\epsilon_i$ are unobserved, the best we can hope for is to learn a parameter vector $\hat{w} = (\hat{w}_1, \ldots, \hat{w}_n) \in \mathbb{R}^n$ that minimizes the sum of errors between $y_i$ and $\sum_{j=1}^{n} x_{ij} w_j$ over all possible $w$ when we measure the error in an appropriate way.

- What can be a good way of measuring error?
What can we hope to learn in such a model?

- Since $\epsilon_i$ are unobserved, the best we can hope for is to learn a parameter vector $\hat{w} = (\hat{w}_1, \ldots, \hat{w}_n) \in \mathbb{R}^n$ that minimizes the sum of errors between $y_i$ and $\sum_{j=1}^{n} x_{ij}w_j$ over all possible $w$ when we measure the error in an appropriate way.
- What can be a good way of measuring error?

In linear regression model we consider square error as the choice of error function.

$$\hat{w} = \arg\min_{w \in \mathbb{R}^n} \sum_{i=1}^{N} \left( y_i - \sum_{j=1}^{n} x_{ij}w_j \right)^2 = \arg\min_{w \in \mathbb{R}^n} \sum_{i=1}^{N} \left( y_i - x_i^T w \right)^2$$
Suppose we use the following compact notation

\[ \Phi = \begin{bmatrix} 
    x_{11} & x_{12} & \cdots & x_{1n} \\
    x_{21} & x_{22} & \cdots & x_{2n} \\
    \vdots & \vdots & \ddots & \vdots \\
    x_{N1} & x_{N2} & \cdots & x_{Nn} 
\end{bmatrix} = \begin{bmatrix} 
    x_1^T \\
    x_2^T \\
    \vdots \\
    x_N^T 
\end{bmatrix} \quad y = \begin{bmatrix} 
    y_1 \\
    y_2 \\
    \vdots \\
    y_N 
\end{bmatrix} \]
Linear Regression Model

- Suppose we use the following compact notation

$$\Phi = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{N1} & x_{N2} & \cdots & x_{Nn} \end{bmatrix} = \begin{bmatrix} x_1^T \\ x_2^T \\ \vdots \\ x_N^T \end{bmatrix}, \quad y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}$$

- $\Phi$ called the design matrix and we seek to find $w \in \mathbb{R}^n$ such that

$$\Phi w = y$$
Linear Regression Model

- Suppose we use the following compact notation

\[
\Phi = \begin{bmatrix}
  x_{11} & x_{12} & \cdots & x_{1n} \\
  x_{21} & x_{22} & \cdots & x_{2n} \\
  \vdots & \vdots & \ddots & \vdots \\
  x_{N1} & x_{N2} & \cdots & x_{Nn}
\end{bmatrix} = \begin{bmatrix}
  x_1^T \\
  x_2^T \\
  \vdots \\
  x_N^T
\end{bmatrix}
\]

\[
y = \begin{bmatrix}
  y_1 \\
  y_2 \\
  \vdots \\
  y_N
\end{bmatrix}
\]

- \(\Phi\) called the design matrix and we seek to find \(w \in \mathbb{R}^n\) such that

\[
\Phi w = y
\]

- Then, linear regression model translates to finding a \(\hat{w} \in \mathbb{R}^n\) such that

\[
\hat{w} = \arg \min_{w \in \mathbb{R}^n} \sum_{i=1}^{N} \left( y_i - x_i^T w \right)^2 = \arg \min_{w \in \mathbb{R}^n} \| y - \Phi w \|^2_2
\]

Kaushik Sinha | Linear Regression and Linear Basis Function Model
We can get close form solution for $\hat{w}$
We can get close form solution for $\hat{w}$

$$S(w) = \|y - \Phi w\|_2^2$$
We can get close form solution for $\hat{w}$

$$S(w) = \|y - \Phi w\|_2^2$$

$$= (y - \Phi w)^T (y - \Phi w)$$
We can get close form solution for $\hat{w}$

$$S(w) = \|y - \Phi w\|_2^2$$

$$= (y - \Phi w)^T (y - \Phi w)$$

$$= y^T y - 2w^T \Phi^T y + w^T \Phi^T \Phi w$$

To find $\hat{w}$ that minimizes the above sum squared error $S(w)$ we can simply set

$$\frac{dS(w)}{dw} \bigg|_{w = \hat{w}} = 0$$

$$\hat{w} = (\Phi^T \Phi)^{-1} \Phi^T y$$

Where $\Phi^\dagger$ is the Moore-Penrose pseudo inverse of $\Phi$.
We can get close form solution for \( \hat{w} \)

\[
S(w) = \|y - \Phi w\|^2_2 \\
= (y - \Phi w)^T(y - \Phi w) \\
= y^Ty - 2w^T\Phi^Ty + w^T\Phi^T\Phi w
\]

To find \( w \) that minimizes the above sum squared error \( S(w) \), we can simply set

\[
\frac{dS(w)}{dw}|_{w=\hat{w}} = -2\Phi^Ty + 2\Phi^T\Phi\hat{w} = 0
\]
We can get close form solution for $\hat{w}$

$$S(w) = \|y - \Phi w\|_2^2$$

$$= (y - \Phi w)^T(y - \Phi w)$$

$$= y^Ty - 2w^T\Phi^Ty + w^T\Phi^T\Phi w$$

To find $w$ that minimizes the above sum squared error $S(w)$, we can simply set

$$\frac{dS(w)}{dw}|_{w=\hat{w}} = -2\Phi^Ty + 2\Phi^T\Phi\hat{w} = 0$$

$$\hat{w} = (\Phi^T\Phi)^{-1}\Phi^Ty = \Phi^\dagger y$$

Where $\Phi^\dagger$ is the Moore-Penrose pseudo inverse of $\Phi$
Some Unresolved Issues of Linear Regression Model

What is the probabilistic interpretation of least square linear regression?

In what ways can we generalize $\Phi$?

What happens if $\Phi^T\Phi$ is singular and thus has no inverse?

What is the geometric interpretation of least square linear regression?

What if the assumed linear model doesn’t pass through origin?
Some Unresolved Issues of Linear Regression Model

- What is the probabilistic interpretation of least square linear regression?

- What are the assumptions of linear regression?

- What happens if \( \Phi^T \Phi \) is singular and thus has no inverse?

- What is the geometric interpretation of least square linear regression?

- What if the assumed linear model doesn’t pass through origin?
Some Unresolved Issues of Linear Regression Model

- What is the probabilistic interpretation of least square linear regression?
- In what ways can we generalize $\Phi$?

Kaushik Sinha
Linear Regression and Linear Basis Function Model
What is the probabilistic interpretation of least square linear regression?

In what ways can we generalize $\Phi$?

What happens if $\Phi^T \Phi$ is singular and thus has no inverse?
Some Unresolved Issues of Linear Regression Model

- What is the probabilistic interpretation of least square linear regression?
- In what ways can we generalize $\Phi$?
- What happens if $\Phi^T \Phi$ is singular and thus has no inverse?
- What is the geometric interpretation of least square linear regression?
Some Unresolved Issues of Linear Regression Model

- What is the probabilistic interpretation of least square linear regression?
- In what ways can we generalize $\Phi$?
- What happens if $\Phi^T \Phi$ is singular and thus has no inverse?
- What is the geometric interpretation of least square linear regression?
- What if the assumed linear model doesn’t pass through origin?
Probabilistic Interpretation of Linear Regression

- In linear regression model an input-output pair $(x, y) \in \mathbb{R}^n \times \mathbb{R}$ is modeled as
  \[ y = x^T w + \epsilon \]

- where $\epsilon$ follows a Gaussian distribution with 0 mean and precision (inverse variance) $\sigma \in \mathbb{R}$, i.e. $\epsilon \sim \mathcal{N}(0, \frac{1}{\sigma})$

- this is equivalent to saying that $y \sim \mathcal{N}(x^T w, \frac{1}{\sigma})$ i.e.,
  \[ p(y|x, w, \sigma) = \sqrt{\frac{\sigma}{2\pi}} e^{-\frac{\sigma(y-x^T w)^2}{2}} \]

- First observation
  \[ E[y|x] = x^T w \]
Probabilistic Interpretation of Linear Regression

Given $N$ pairs of iid $(x_1, y_1), \ldots, (x_N, y_N)$, the likelihood of seeing $y = (y_1, \ldots, y_N)$ given $x_1, \ldots, x_N, w, \sigma$ is
Given $N$ pairs of iid $(x_1, y_1), \ldots, (x_N, y_N)$, the likelihood of seeing $y = (y_1, \ldots, y_N)$ given $x_1, \ldots, x_N, w, \sigma$ is

$$p(y|x_1, \ldots, x_N, w, \sigma) = \prod_{i=1}^{N} p(y_i|x_i, w, \sigma)$$
Given $N$ pairs of iid $(x_1, y_1), \ldots, (x_N, y_N)$, the likelihood of seeing $y = (y_1, \ldots, y_N)$ given $x_1, \ldots, x_N, w, \sigma$ is

$$p(y|x_1, \ldots, x_N, w, \sigma) = \prod_{i=1}^{N} p(y_i|x_i, w, \sigma)$$

Taking logarithm, the log likelihood is given by

$$\ln p(y|x_1, \ldots, x_N, w, \sigma) = \sum_{i=1}^{N} \ln p(y_i|x_i, w, \sigma)$$
Given $N$ pairs of iid $(x_1, y_1), \ldots, (x_N, y_N)$, the likelihood of seeing $y = (y_1, \ldots, y_N)$ given $x_1, \ldots, x_N, w, \sigma$ is

$$p(y|x_1, \ldots, x_N, w, \sigma) = \prod_{i=1}^{N} p(y_i|x_i, w, \sigma)$$

Taking logarithm, the log likelihood is given by

$$\ln p(y|x_1, \ldots, x_N, w, \sigma) = \sum_{i=1}^{N} \ln p(y_i|x_i, w, \sigma)$$

$$= \frac{N}{2} \ln \sigma - \frac{N}{2} \ln(2\pi) - \frac{\sigma}{2} \sum_{i=1}^{N} \left(y_i - x_i^T w \right)^2$$
Given $N$ pairs of iid $(x_1, y_1), \ldots, (x_N, y_N)$, the likelihood of seeing $y = (y_1, \ldots, y_N)$ given $x_1, \ldots, x_N, w, \sigma$ is

$$p(y|x_1, \ldots, x_N, w, \sigma) = \prod_{i=1}^{N} p(y_i|x_i, w, \sigma)$$

Taking logarithm, the log likelihood is given by

$$\ln p(y|x_1, \ldots, x_N, w, \sigma) = \sum_{i=1}^{N} \ln p(y_i|x_i, w, \sigma)$$

$$= \frac{N}{2} \ln \sigma - \frac{N}{2} \ln(2\pi) - \frac{\sigma}{2} \sum_{i=1}^{N} (y_i - x_i^T w)^2$$

$$= \frac{N}{2} \ln \sigma - \frac{N}{2} \ln(2\pi) - \frac{\sigma}{2} \|y - \Phi w\|^2$$
Given \( N \) pairs of iid \((x_1, y_1), \ldots, (x_N, y_N)\), the likelihood of seeing \( y = (y_1, \ldots, y_N) \) given \( x_1, \ldots, x_N, w, \sigma \) is

\[
p(y|x_1, \ldots, x_N, w, \sigma) = \prod_{i=1}^{N} p(y_i|x_i, w, \sigma)
\]

Taking logarithm, the log likelihood is given by

\[
\ln p(y|x_1, \ldots, x_N, w, \sigma) = \sum_{i=1}^{N} \ln p(y_i|x_i, w, \sigma)
\]

\[
= \frac{N}{2} \ln \sigma - \frac{N}{2} \ln(2\pi) - \frac{\sigma}{2} \sum_{i=1}^{N} (y_i - x_i^T w)^2
\]

\[
= \frac{N}{2} \ln \sigma - \frac{N}{2} \ln(2\pi) - \frac{\sigma}{2} \|y - \Phi w\|^2 \quad (1)
\]
Given $N$ pairs of iid $(x_1, y_1), \ldots, (x_N, y_N)$, the likelihood of seeing $y = (y_1, \ldots, y_N)$ given $x_1, \ldots, x_N, w, \sigma$ is

$$p(y | x_1, \ldots, x_N, w, \sigma) = \prod_{i=1}^{N} p(y_i | x_i, w, \sigma)$$

Taking logarithm, the log likelihood is given by

$$\ln p(y | x_1, \ldots, x_N, w, \sigma) = \sum_{i=1}^{N} \ln p(y_i | x_i, w, \sigma)$$

$$= \frac{N}{2} \ln \sigma - \frac{N}{2} \ln(2\pi) - \frac{\sigma}{2} \sum_{i=1}^{N} \left(y_i - x_i^T w \right)^2$$

$$= \frac{N}{2} \ln \sigma - \frac{N}{2} \ln(2\pi) - \frac{\sigma}{2} \| y - \Phi w \|^2 \quad (1)$$

We need to maximize log likelihood with respect to $w$ and $\sigma$.
Maximizing Log Likelihood

For any $\sigma$, to maximize Equation 1, we need to minimize

$$\sum_{i=1}^{N} (y_i - x_i^T w)^2$$

Our choice of error function before was no coincidence!

Suppose $w_{ML}$ be the choice of $w$ that maximizes log likelihood of Equation 1.

The gradient of the log likelihood with respect to $w$ gives $\sigma (-\Phi^T y + \Phi^T \Phi w)$.

Setting this to zero gives the value for optimal $w_{ML}$ that maximizes log likelihood in Equation 1, this is the same expression for optimal $\hat{w}$ that we have seen before.

Let $\sigma_{ML}$ be the value of $\sigma$ that maximizes the log likelihood in Equation 1 (take derivative with respect to $\sigma$ and set it to zero).

$$\frac{1}{\sigma_{ML}} = \frac{1}{N} \| y - \Phi w_{ML} \|^2 = \frac{1}{N} \sum_{i=1}^{N} (y_i - x_i^T w_{ML})^2$$

The inverse of noise precision is variance of target value around regression function.
Maximizing Log Likelihood

For any $\sigma$, to maximize Equation 1, we need to minimize

$$\sum_{i=1}^{N} (y_i - x_i^T w)^2$$

Our choice of error function before was no coincidence!

Suppose $w_{ML}$ be the choice of $w$ that maximizes log likelihood of Equation 1.

The gradient of the log likelihood with respect to $w$ gives

$$\sigma \left( -\Phi^T y + \Phi^T \Phi w \right)$$

Setting this to zero gives the value for optimal $w_{ML}$ that maximizes log likelihood in Equation 1. This is the same expression for optimal $\hat{w}$ that we have seen before.

Let $\sigma_{ML}$ be the value of $\sigma$ that maximizes the log likelihood in Equation 1.

(1) $\frac{1}{\sigma_{ML}} = \frac{1}{N} \| y - \Phi w_{ML} \|_2^2 = \frac{1}{N} \sum_{i=1}^{N} (y_i - x_i^T w_{ML})^2$

The inverse of noise precision is variance of target value around regression function.
For any $\sigma$, to maximize Equation 1, we need to minimize
\[ \sum_{i=1}^{N} (y_i - x_i^T w)^2 \]
- Our choice of error function before was no coincidence!

Let $\sigma_{ML}$ be the value of $\sigma$ that maximizes the log likelihood in Equation 1 (take derivative with respect to $\sigma$ and set it to zero)
\[ 1/\sigma_{ML}^2 = 1/N \sum_{i=1}^{N} (y_i - \Phi w_{ML})^2 \]
Maximizing Log Likelihood

For any $\sigma$, to maximize Equation 1, we need to minimize

$$\sum_{i=1}^{N} (y_i - x_i^T w)^2$$

- Our choice of error function before was no coincidence!

Suppose $w_{ML}$ be the choice of $w$ that maximizes log likelihood of Equation 1.
Maximizing Log Likelihood

- For any $\sigma$, to maximize Equation 1, we need to minimize
  $$\sum_{i=1}^{N} (y_i - x_i^T w)^2$$
  - Our choice of error function before was no coincidence!
- Suppose $w_{ML}$ be the choice of $w$ that maximizes log likelihood of Equation 1
  - gradient of the log likelihood with respect to $w$ gives
    $$\sigma \left( -\Phi^T y + \Phi^T \Phi w \right)$$
For any $\sigma$, to maximize Equation 1, we need to minimize
$$\sum_{i=1}^{N} (y_i - x_i^T w)^2$$
- Our choice of error function before was no coincidence!

Suppose $w_{ML}$ be the choice of $w$ that maximizes log likelihood of Equation 1
- gradient of the log likelihood with respect to $w$ gives $\sigma (-\Phi^T y + \Phi^T \Phi w)$
- setting this to zero gives the value for optimal $w_{ML}$ that maximizes log likelihood in Equation 1
For any $\sigma$, to maximize Equation 1, we need to minimize
\[
\sum_{i=1}^{N} \left(y_i - x_i^T w\right)^2
\]
- Our choice of error function before was no coincidence!

Suppose $w_{ML}$ be the choice of $w$ that maximizes log likelihood of Equation 1
- gradient of the log likelihood with respect to $w$ gives
  \[
  \sigma \left(-\Phi^T y + \Phi^T \Phi w\right)
  \]
- setting this to zero gives the value for optimal $w_{ML}$ that maximizes log likelihood in Equation 1
- this is the same expression for optimal $\hat{w}$ that we have seen before

Let $\sigma_{ML}$ be the value of $\sigma$ that maximizes the log likelihood in Equation 1 (take derivative with respect to $\sigma$ and set it to zero)
\[
\frac{1}{\sigma_{ML}} = \frac{1}{N} \sum_{i=1}^{N} \left(y_i - \Phi w_{ML}\right)^2
\]
inverse of noise precision is variance of target value around regression function
Maximizing Log Likelihood

For any $\sigma$, to maximize Equation 1, we need to minimize
\[
\sum_{i=1}^{N} (y_i - x_i^T w)^2
\]
- Our choice of error function before was no coincidence!

Suppose $w_{ML}$ be the choice of $w$ that maximizes log likelihood of Equation 1
- gradient of the log likelihood with respect to $w$ gives
  \[
  \sigma \left( -\Phi^T y + \Phi^T \Phi w \right)
  \]
  setting this to zero gives the value for optimal $w_{ML}$ that maximizes log likelihood in Equation 1
  - this is the same expression for optimal $\hat{w}$ that we have seen before

Let $\sigma_{ML}$ be the value of $\sigma$ that maximizes the log likelihood in Equation 1 (take derivative with respect to $\sigma$ and set it to zero
Maximizing Log Likelihood

- For any $\sigma$, to maximize Equation 1, we need to minimize
  \[ \sum_{i=1}^{N} (y_i - x_i^T \mathbf{w})^2 \]
  - Our choice of error function before was no coincidence!
- Suppose $\mathbf{w}_{ML}$ be the choice of $\mathbf{w}$ that maximizes log likelihood of Equation 1
  - gradient of the log likelihood with respect to $\mathbf{w}$ gives
    \[ \sigma \left( -\Phi^T \mathbf{y} + \Phi^T \Phi \mathbf{w} \right) \]
  - setting this to zero gives the value for optimal $\mathbf{w}_{ML}$ that maximizes log likelihood in Equation 1
  - this is the same expression for optimal $\hat{\mathbf{w}}$ that we have seen before
- Let $\sigma_{ML}$ be the value of $\sigma$ that maximizes the log likelihood in Equation 1 (take derivative with respect to $\sigma$ and set it to zero
  \[ \frac{1}{\sigma_{ML}} = \frac{1}{N} \| \mathbf{y} - \Phi \mathbf{w}_{ML} \|_2^2 = \frac{1}{N} \sum_{i=1}^{N} (y_i - x_i^T \mathbf{w}_{ML})^2 \]
Maximizing Log Likelihood

- For any $\sigma$, to maximize Equation 1, we need to minimize
  \[ \sum_{i=1}^{N} (y_i - x_i^T w)^2 \]
  - Our choice of error function before was no coincidence!
- Suppose $w_{ML}$ be the choice of $w$ that maximizes log likelihood of Equation 1
  - gradient of the log likelihood with respect to $w$ gives
    \[ \sigma (-\Phi^T y + \Phi^T \Phi w) \]
    - setting this to zero gives the value for optimal $w_{ML}$ that maximizes log likelihood in Equation 1
  - this is the same expression for optimal $\hat{w}$ that we have seen before
- Let $\sigma_{ML}$ be the value of $\sigma$ that maximizes the log likelihood in Equation 1 (take derivative with respect to $\sigma$ and set it to zero
  \[ \frac{1}{\sigma_{ML}} = \frac{1}{N} ||y - \Phi w_{ML}||_2^2 = \frac{1}{N} \sum_{i=1}^{N} (y_i - x_i^T w_{ML})^2 \]
  - inverse of noise precision is variance of target value around regression function

Kaushik Sinha
Linear Regression and Linear Basis Function Model
The design matrix that we have seen so far can also be expressed in a more general form

\[
\Phi = \begin{bmatrix}
\phi_1(x_1) & \phi_2(x_1) & \cdots & \phi_M(x_1) \\
\phi_1(x_2) & \phi_2(x_2) & \cdots & \phi_M(x_2) \\
\vdots & \vdots & \ddots & \vdots \\
\phi_1(x_N) & \phi_2(x_N) & \cdots & \phi_M(x_N)
\end{bmatrix}
= \begin{bmatrix}
\phi(x_1)^T \\
\phi(x_2)^T \\
\vdots \\
\phi(x_N)^T
\end{bmatrix}
\]
The design matrix that we have seen so far can also be expressed in a more general form

\[
\Phi = \begin{bmatrix}
\phi_1(x_1) & \phi_2(x_1) & \cdots & \phi_M(x_1) \\
\phi_1(x_2) & \phi_2(x_2) & \cdots & \phi_M(x_2) \\
\vdots & \vdots & \ddots & \vdots \\
\phi_1(x_N) & \phi_2(x_N) & \cdots & \phi_M(x_N)
\end{bmatrix}
= \begin{bmatrix}
\phi(x_1)^T \\
\phi(x_2)^T \\
\vdots \\
\phi(x_N)^T
\end{bmatrix}
\]

where each \( \phi_i \) for \( i = 1, \ldots, M \) is a function \( \phi_i : \mathbb{R}^n \rightarrow \mathbb{R} \) and is in general called basis function.
The design matrix that we have seen so far can also be expressed in a more general form

\[
\Phi = \begin{bmatrix}
\phi_1(x_1) & \phi_2(x_1) & \cdots & \phi_M(x_1) \\
\phi_1(x_2) & \phi_2(x_2) & \cdots & \phi_M(x_2) \\
\vdots & \vdots & \ddots & \vdots \\
\phi_1(x_N) & \phi_2(x_N) & \cdots & \phi_M(x_N)
\end{bmatrix}
= \begin{bmatrix}
\phi(x_1)^T \\
\phi(x_2)^T \\
\vdots \\
\phi(x_N)^T
\end{bmatrix}
\]

where each \( \phi_i \) for \( i = 1, \ldots, M \) is a function \( \phi_i : \mathbb{R}^n \rightarrow \mathbb{R} \) and is in general called basis function.

- in particular, \( \phi(\cdot)^T = (\phi_1(\cdot), \ldots, \phi_M(\cdot)) \)
The design matrix that we have seen so far can also be expressed in a more general form:

\[ \Phi = \begin{bmatrix} \phi_1(x_1) & \phi_2(x_1) & \cdots & \phi_M(x_1) \\ \phi_1(x_2) & \phi_2(x_2) & \cdots & \phi_M(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_1(x_N) & \phi_2(x_N) & \cdots & \phi_M(x_N) \end{bmatrix} = \begin{bmatrix} \phi(x_1)^T \\ \phi(x_2)^T \\ \vdots \\ \phi(x_N)^T \end{bmatrix} \]

where each \( \phi_i \) for \( i = 1, \ldots, M \) is a function \( \phi_i : \mathbb{R}^n \to \mathbb{R} \) and is in general called basis function.

- in particular, \( \phi(\cdot)^T = (\phi_1(\cdot), \ldots, \phi_M(\cdot)) \)
- in case of linear regression (that we have seen before)
The design matrix that we have seen so far can also be expressed in a more general form

\[
\Phi = \begin{bmatrix}
\phi_1(x_1) & \phi_2(x_1) & \cdots & \phi_M(x_1) \\
\phi_1(x_2) & \phi_2(x_2) & \cdots & \phi_M(x_2) \\
\vdots & \vdots & \ddots & \vdots \\
\phi_1(x_N) & \phi_2(x_N) & \cdots & \phi_M(x_N)
\end{bmatrix} = \begin{bmatrix}
\phi(x_1)^T \\
\phi(x_2)^T \\
\vdots \\
\phi(x_N)^T
\end{bmatrix}
\]

where each \( \phi_i \) for \( i = 1, \ldots, M \) is a function \( \phi_i : \mathbb{R}^n \rightarrow \mathbb{R} \) and is in general called basis function

- in particular, \( \phi(\cdot)^T = (\phi_1(\cdot), \ldots, \phi_M(\cdot)) \)
- in case of linear regression (that we have seen before)
  - \( \phi_i(x) \) gives \( i^{th} \) coordinate of \( x \) and thus for any \( j \), \( \phi_i(x_j) = x_{ji} \)
The design matrix that we have seen so far can also be expressed in a more general form

\[
\Phi = \begin{bmatrix}
\phi_1(x_1) & \phi_2(x_1) & \cdots & \phi_M(x_1) \\
\phi_1(x_2) & \phi_2(x_2) & \cdots & \phi_M(x_2) \\
\vdots & \vdots & \ddots & \vdots \\
\phi_1(x_N) & \phi_2(x_N) & \cdots & \phi_M(x_N)
\end{bmatrix} = \begin{bmatrix}
\phi(x_1)^T \\
\phi(x_2)^T \\
\vdots \\
\phi(x_N)^T
\end{bmatrix}
\]

where each \( \phi_i \) for \( i = 1, \ldots, M \) is a function \( \phi_i : \mathbb{R}^n \to \mathbb{R} \) and is in general called basis function.

- in particular, \( \phi(\cdot)^T = (\phi_1(\cdot), \ldots, \phi_M(\cdot)) \)

- in case of linear regression (that we have seen before)
  - \( \phi_i(x) \) gives \( i^{th} \) coordinate of \( x \) and thus for any \( j \), \( \phi_i(x_j) = x_{ji} \)
  - \( M = n \)
The design matrix that we have seen so far can also be expressed in a more general form

$$\Phi = \begin{bmatrix} \phi_1(x_1) & \phi_2(x_1) & \cdots & \phi_M(x_1) \\ \phi_1(x_2) & \phi_2(x_2) & \cdots & \phi_M(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_1(x_N) & \phi_2(x_N) & \cdots & \phi_M(x_N) \end{bmatrix} = \begin{bmatrix} \phi(x_1)^T \\ \phi(x_2)^T \\ \vdots \\ \phi(x_N)^T \end{bmatrix}$$

where each $\phi_i$ for $i = 1, \ldots, M$ is a function $\phi_i : \mathbb{R}^n \rightarrow \mathbb{R}$ and is in general called basis function.

- in particular, $\phi(\cdot)^T = (\phi_1(\cdot), \ldots, \phi_M(\cdot))$
- in case of linear regression (that we have seen before)
  - $\phi_i(x)$ gives $i^{th}$ coordinate of $x$ and thus for any $j$, $\phi_i(x_j) = x_{ji}$
  - $M = n$
- the $i^{th}$ column of $\Phi$ is the function $\phi_i$ evaluated at $x_1, \ldots, x_N$
In general the $i$th column of the design matrix $\Phi$ can be described by a function $\phi_i$. The function $\phi_i$ can be non-linear and several choices exist, e.g.,

- **Polynomial function**: $\phi_j(x) = \text{some polynomial of } j\text{-th coordinate of } x$
- **Gaussian function**: $\phi_j(x) = \exp(-\|x - \mu_j\|^2/s^2)$, where $\mu_j$ and $s$ are location and scale parameters respectively
- **Sigmoidal function**: $\phi_j(x) = \sigma(\|x - \mu_j\|^2/s^2)$, where $\sigma(a) = 1/(1+e^{-a})$ is the logistic sigmoid function

Even though $\phi_i$ can be non-linear, we are seeking a linear combination of non-linear functions that best describe output variables for $(x, y) \in \mathbb{R}^n \times \mathbb{R}$, we are seeking $w \in \mathbb{R}^M$, such that,

$$\sum_{i=1}^M \phi_i(x)w_i$$

is close to $y$. 

Kaushik Sinha

Linear Regression and Linear Basis Function Model
In general the $i^{th}$ column of the design matrix $\Phi$ can be described by a function $\phi_i$. 

- Polynomial function: $\phi_j(x) = \text{some polynomial of } j\text{-th coordinate of } x$
- Gaussian function: $\phi_j(x) = \exp(-\|x - \mu_j\|^2/s^2)$, where $\mu_j$ and $s$ are location and scale parameters respectively.
- Sigmoidal function: $\phi_j(x) = \sigma(\|x - \mu_j\|^2/s)$, where $\sigma(a) = \frac{1}{1+e^{-a}}$ is the logistic sigmoid function.
In general the $i^{th}$ column of the design matrix $\Phi$ can be described by a function $\phi_i$.

The function $\phi_i$ can be non-linear and several choices exists, e.g.,

- **Polynomial function**: $\phi_j(x) = \text{some polynomial of } j\text{-th coordinate of } x$
- **Gaussian function**: $\phi_j(x) = \exp\left(-\|x - \mu_j\|^2 / 2s^2\right)$, where $\mu_j$ and $s$ are location and scale parameters respectively.
- **Sigmoidal function**: $\phi_j(x) = \sigma\left(\|x - \mu_j\|^2 / s\right)$, where $\sigma(a) = 1 / (1 + e^{-a})$ is the logistic sigmoid function.
Linear Basis Function Model

- In general the $i^{th}$ column of the design matrix $\Phi$ can be described by a function $\phi_i$.
- The function $\phi_i$ can be non-linear and several choices exist, e.g.,
  - **Polynomial function**:
    \[ \phi_j(x) = \text{some polynomial of j-th coordinate of } x \]
  - **Gaussian function**:
    \[ \phi_j(x) = \exp\left(-\frac{\|x-\mu_j\|^2}{s^2}\right) \]
    where $\mu_j$ and $s$ are location and scale parameters respectively.
  - **Sigmoidal function**:
    \[ \phi_j(x) = \sigma\left(\frac{\|x-\mu_j\|^2}{s}\right) \]
    where $\sigma(a) = \frac{1}{1+e^{-a}}$ is the logistic sigmoid function.

Even though $\phi_i$ can be non-linear, we are seeking a linear combination of non-linear functions that best describes output variables for $(x, y) \in \mathbb{R}^n \times \mathbb{R}$, we are seeking $w \in \mathbb{R}^M$, such that,
\[ \sum_{i=1}^{M} \phi_i(x) w_i \text{ is close to } y \]
In general the $i^{th}$ column of the design matrix $\Phi$ can be described by a function $\phi_i$

The function $\phi_i$ can be non-linear and several choices exists, e.g.,

- **Polynomial function**:
  $\phi_j(x) = \text{some polynomial of j-th coordinate of } x$

- **Gaussian function**:
  $\phi_j(x) = \exp\left(-\frac{\|x - \mu_j\|_2^2}{2s^2}\right)$, where $\mu_j$ and $s$ are location and scale parameters respectively
In general the $i^{th}$ column of the design matrix $\Phi$ can be described by a function $\phi_i$.

The function $\phi_i$ can be non-linear and several choices exists, e.g.,

- **Polynomial function**: $\phi_j(x) = \text{some polynomial of } j\text{-th coordinate of } x$

- **Gaussian function**: $\phi_j(x) = \exp \left( -\frac{\|x - \mu_j\|^2_2}{2s^2} \right)$, where $\mu_j$ and $s$ are location and scale parameters respectively

- **Sigmoidal function**: $\phi_j(x) = \sigma \left( \frac{\|x - \mu_j\|^2_2}{s} \right)$, where $\sigma(a) = \frac{1}{1 + e^{-a}}$ is the logistic sigmoid function
In general the $i^{th}$ column of the design matrix $\Phi$ can be described by a function $\phi_i$

The function $\phi_i$ can be non-linear and several choices exists, e.g.,

- **Polynomial function** :
  \[ \phi_j(x) = \text{some polynomial of } j\text{-th coordinate of } x \]

- **Gaussian function** : \[ \phi_j(x) = \exp \left( -\frac{\|x-\mu_j\|^2}{2s^2} \right) \], where $\mu_j$ and $s$ are location and scale parameters respectively

- **Sigmoidal function** : \[ \phi_j(x) = \sigma \left( \frac{\|x-\mu_j\|}{s} \right) \], where $\sigma(a) = \frac{1}{1+e^{-a}}$ is the logistic sigmoid function

Even though $\phi_i$ can be non-linear, we are seeking a linear combination of non-linear functions that best describe output variables
In general the $i^{th}$ column of the design matrix $\Phi$ can be described by a function $\phi_i$.

The function $\phi_i$ can be non-linear and several choices exists, e.g.,

- **Polynomial function**: $\phi_j(x) = \text{some polynomial of j-th coordinate of } x$
- **Gaussian function**: $\phi_j(x) = \exp \left( -\frac{\|x - \mu_j\|_2^2}{2s^2} \right)$, where $\mu_j$ and $s$ are location and scale parameters respectively
- **Sigmoidal function**: $\phi_j(x) = \sigma \left( \frac{\|x - \mu_j\|_2}{s} \right)$, where $\sigma(a) = \frac{1}{1+e^{-a}}$ is the logistic sigmoid function

Even though $\phi_i$ can be non-linear, we are seeking a linear combination of non-linear functions that best describe output variables.

- for $(x, y) \in \mathbb{R}^n \times \mathbb{R}$, we are seeking $w \in \mathbb{R}^M$, such that, $\sum_{i=1}^M \phi_i(x)w_i$ is close to $y$
Regularized Least Square

Suppose the design matrix \( \Phi \in \mathbb{R}^{N \times M} \) is constructed using \( M \) basis functions and \( N \) input-output pairs (training set).

Let the Singular Value Decomposition (SVD) of \( \Phi \) be

\[
\Phi = U \Sigma V^T,
\]

where

\( U \in \mathbb{R}^{N \times r} \) is an orthonormal matrix

\( \Sigma \in \mathbb{R}^{r \times r} \) is diagonal matrix of singular values

\( \sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0 \)

\( V \in \mathbb{R}^{M \times r} \) is an orthonormal matrix

How does the least square linear regression solution look like in terms of \( U \), \( \Sigma \), and \( V \)?
Suppose the design matrix $\Phi \in \mathbb{R}^{N \times M}$ is constructed using $M$ basis functions and $N$ input-output pairs (training set).
Suppose the design matrix \( \Phi \in \mathbb{R}^{N \times M} \) is constructed using \( M \) basis functions and \( N \) input-output pairs (training set).

Let the Singular Value Decomposition (SVD) of \( \Phi \) be \( \Phi = U \Sigma V^T \), where
Suppose the design matrix $\Phi \in \mathbb{R}^{N \times M}$ is constructed using $M$ basis functions and $N$ input-output pairs (training set).

Let the Singular Value Decomposition (SVD) of $\Phi$ be $\Phi = U \Sigma V^T$, where

- $U \in \mathbb{R}^{N \times r}$ is an orthonormal matrix.
Regularized Least Square

Suppose the design matrix $\Phi \in \mathbb{R}^{N \times M}$ is constructed using $M$ basis functions and $N$ input-output pairs (training set).

Let the Singular Value Decomposition (SVD) of $\Phi$ be $\Phi = U \Sigma V^T$, where:

- $U \in \mathbb{R}^{N \times r}$ is an orthonormal matrix
- $\Sigma \in \mathbb{R}^{r \times r}$ is diagonal matrix of singular values
  $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0$
Suppose the design matrix $\Phi \in \mathbb{R}^{N \times M}$ is constructed using $M$ basis functions and $N$ input-output pairs (training set).

Let the Singular Value Decomposition (SVD) of $\Phi$ be $\Phi = U \Sigma V^T$, where

- $U \in \mathbb{R}^{N \times r}$ is an orthonormal matrix
- $\Sigma \in \mathbb{R}^{r \times r}$ is diagonal matrix of singular values $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0$
- $V \in \mathbb{R}^{M \times r}$ is an orthonormal matrix
Suppose the design matrix $\Phi \in \mathbb{R}^{N \times M}$ is constructed using $M$ basis functions and $N$ input-output pairs (training set).

Let the Singular Value Decomposition (SVD) of $\Phi$ be $\Phi = U \Sigma V^T$, where

- $U \in \mathbb{R}^{N \times r}$ is an orthonormal matrix
- $\Sigma \in \mathbb{R}^{r \times r}$ is diagonal matrix of singular values with $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0$
- $V \in \mathbb{R}^{M \times r}$ is an orthonormal matrix

How does the least square linear regression solution look like in terms of $U$, $\Sigma$ and $V$?
Regularized Least Square

\[ \hat{w} = (\Phi^T \Phi)^{-1} \Phi^T y \]
Regularized Least Square

\[ \hat{w} = (\Phi^T \Phi)^{-1} \Phi^T y \]

If even a single singular value \( \sigma_i \) is small, then slight perturbation in \( y \) can lead to a large error in the least square linear regression solution!
\[ \hat{w} = (\Phi^T \Phi)^{-1} \Phi^T y \]
\[ = (V \Sigma U^T \Sigma U \Sigma V^T)^{-1} (U \Sigma V^T)^T y \]
\[ = (V \Sigma^2 V^T)^{-1} (U \Sigma V^T)^T y \]
\[
\hat{\mathbf{w}} = (\Phi^T\Phi)^{-1}\Phi^T\mathbf{y} \\
= (V\Sigma U^T U\Sigma V^T)^{-1}(U\Sigma V^T)^T\mathbf{y} \\
= (V\Sigma^2 V^T)^{-1}(U\Sigma V^T)^T\mathbf{y} \\
= V\Sigma^{-2} V^T (U\Sigma V^T)^T\mathbf{y}
\]

If even a single singular value \(\sigma_i\) is small, then slight perturbation in \(\mathbf{y}\) can lead to a large error in the least square linear regression solution!
\[ \hat{w} = (\Phi^T \Phi)^{-1} \Phi^T y \]

= \((V \Sigma U^T U \Sigma V^T)^{-1}(U \Sigma V^T)^T y\)

= \((V \Sigma^2 V^T)^{-1}(U \Sigma V^T)^T y\)

= \(V \Sigma^{-2} V^T (U \Sigma V^T)^T y\)

= \(V \Sigma^{-2} V^T V \Sigma U^T y\)

If even a single singular value \(\sigma_i\) is small, then slight perturbation in \(y\) can lead to a large error in the least square linear regression solution!
\[ \hat{w} = (\Phi^T \Phi)^{-1} \Phi^T y \]
\[ = (V \Sigma U^T U \Sigma V^T)^{-1} (U \Sigma V^T)^T y \]
\[ = (V \Sigma^2 V^T)^{-1} (U \Sigma V^T)^T y \]
\[ = V \Sigma^{-2} V^T (U \Sigma V^T)^T y \]
\[ = V \Sigma^{-2} V^T V \Sigma U^T y \]
\[ = V \Sigma^{-2} \Sigma U^T y \]
\[
\hat{w} = (\Phi^T \Phi)^{-1} \Phi^T y \\
= (V \Sigma U^T U \Sigma V^T)^{-1} (U \Sigma V^T)^T y \\
= (V \Sigma^2 V^T)^{-1} (U \Sigma V^T)^T y \\
= V \Sigma^{-2} V^T (U \Sigma V^T)^T y \\
= V \Sigma^{-2} V^T V \Sigma U^T y \\
= V \Sigma^{-2} \Sigma U^T y \\
= V \Sigma^{-1} U^T y
\]
Regularized Least Square

\[ \hat{w} = (\Phi^T \Phi)^{-1} \Phi^T y \]

\[ = (V \Sigma U^T U \Sigma V^T)^{-1} (U \Sigma V^T)^T y \]

\[ = (V \Sigma^2 V^T)^{-1} (U \Sigma V^T)^T y \]

\[ = V \Sigma^{-2} V^T (U \Sigma V^T)^T y \]

\[ = V \Sigma^{-2} V^T V \Sigma U^T y \]

\[ = V \Sigma^{-2} \Sigma U^T y \]

\[ = V \Sigma^{-1} U^T y \]

\[ = \sum_{i=1}^{r} \frac{u_i^T y}{\sigma_i} v_i \]

If even a single singular value \( \sigma_i \) is small, then slight perturbation in \( y \) can lead to a large error in the least square linear regression solution!
If even a single singular value $\sigma_i$ is small, then slight perturbation in $y$ can lead to a large error in the least square linear regression solution!
There are two potential problems with least squared linear regression.

If the design matrix $\Phi$ doesn't have full rank, then least squared linear regression doesn't have a solution since the inverse of $\Phi^T \Phi$ doesn't exist.

If at least one of the singular values of the design matrix $\Phi$ is very small, then a small perturbation in the output values will give a large error in the solution.

The trick to solving these problems is to use regularization and essentially solve the following problem:

$$w_{\text{reg}} = \arg\min_{w \in \mathbb{R}^M} \| y - \Phi w \|^2_2 + \lambda \| w \|^2_2$$

where $\lambda > 0$ is called the regularization parameter.
There are two potential problems with least squared linear regression.

1. If the design matrix $\Phi$ doesn't have full rank, then least squared linear regression doesn't have a solution since the inverse of $\Phi^T \Phi$ doesn't exist.
2. If at least one of the singular values of the design matrix $\Phi$ is very small, then a small perturbation in the output values will give a large error in the solution.

The trick to solve these problems is to use regularization and essentially solve the following problem:

$$w_{\text{reg}} = \text{arg min}_{w \in \mathbb{R}^M} \| y - \Phi w \|^2_2 + \lambda \| w \|^2_2$$

where $\lambda > 0$ is called the regularization parameter.
There are two potential problems with least squared linear regression:

- If design matrix $\Phi$ doesn’t have full rank then least squared linear regression doesn’t have a solution since inverse of $\Phi^T \Phi$ doesn’t exist.

The trick to solve these problems is to use regularization and essentially solve the following problem:

$$
\mathbf{w}_{\text{r}} = \text{arg min}_{\mathbf{w} \in \mathbb{R}^M} \| \mathbf{y} - \Phi \mathbf{w} \|_2^2 + \lambda \| \mathbf{w} \|_2^2
$$

where $\lambda > 0$ is called regularization parameter.
There are two potential problems with least squared linear regression

- If design matrix $\Phi$ doesn’t have full rank then least squared linear regression doesn’t have a solution since inverse of $\Phi^T \Phi$ doesn’t exist.
- If at least one of the singular values of design matrix $\Phi$ is very small then a small perturbation in output values will give a large error in solution.

The trick to solve these problems is to use regularization and essentially solve the following problem:

$$w_{\text{r}} = \arg \min_{w \in \mathbb{R}^M} \| y - \Phi w \|_2^2 + \lambda \| w \|_2^2$$

where $\lambda > 0$ is called the regularization parameter.
There are two potential problems with least squared linear regression:

1. If design matrix $\Phi$ doesn’t have full rank then least squared linear regression doesn’t have a solution since inverse of $\Phi^T\Phi$ doesn’t exist.
2. If at least one of the singular values of design matrix $\Phi$ is very small then a small perturbation in output values will give a large error in solution.

The trick to solve these problems is to use regularization and essentially solve the following problem:

$$w^* = \arg \min_{w \in \mathbb{R}^M} \|y - \Phi w\|^2 + \lambda \|w\|^2$$
There are two potential problems with least squared linear regression:

- If design matrix $\Phi$ doesn’t have full rank then least squared linear regression doesn’t have a solution since inverse of $\Phi^T\Phi$ doesn’t exist.
- If at least one of the singular values of design matrix $\Phi$ is very small then a small perturbation in output values will give a large error in solution.

The trick to solve these problems is to use regularization and essentially solve the following problem:

$$w^r = \arg \min_{w \in \mathbb{R}^M} \| y - \Phi w \|^2 + \lambda \| w \|^2$$

where $\lambda > 0$ is called regularization parameter.
We will later see that this problem is equivalent to the following minimization problem under constraint

$$\min_{w \in \mathbb{R}^M} \|y - \Phi w\|^2$$

such that $\|w\|^2 \leq \eta$

for an appropriate choice of $\eta > 0$
We will later see that this problem is equivalent to the following minimization problem under constraint

$$\min_{w \in \mathbb{R}^M} \|y - \Phi w\|^2$$

such that $\|w\|^2 \leq \eta$

for an appropriate choice of $\eta > 0$

This is called *constraint optimization* because
We will later see that this problem is equivalent to the following minimization problem under constraint

\[
\min_{\mathbf{w} \in \mathbb{R}^M} \| \mathbf{y} - \Phi \mathbf{w} \|^2 \\
\text{such that } \| \mathbf{w} \|^2 \leq \eta
\]

for an appropriate choice of \( \eta > 0 \)

This is called \textit{constraint optimization} because

- we want to minimize \( \| \mathbf{y} - \Phi \mathbf{w} \|^2 \) (the unconstrained version)
We will later see that this problem is equivalent to the following minimization problem under constraint

\[
\min_{w \in \mathbb{R}^M} \| y - \Phi w \|^2
\]

such that \( \| w \|^2 \leq \eta \)

- for an appropriate choice of \( \eta > 0 \)
- This is called \textit{constraint optimization} because
  - we want to minimize \( \| y - \Phi w \|^2 \) (the unconstrained version)
  - under the constraint \( \| w \|^2 \leq \eta \)
We will later see that this problem is equivalent to the following minimization problem under constraint

$$\min_{w \in \mathbb{R}^M} \|y - \Phi w\|_2^2$$

such that $\|w\|_2^2 \leq \eta$

for an appropriate choice of $\eta > 0$

This is called *constraint optimization* because
- we want to minimize $\|y - \Phi w\|_2^2$ (the unconstrained version)
- under the constraint $\|w\|_2^2 \leq \eta$

These two techniques can be related using *Lagrange multipliers*
It is easy to show (in the same way by first taking the derivative with respect to $w$ and setting it to zero) that the regularized least square linear regression solution $w^r$ takes the form

$$w^r = (\Phi^T \Phi + \lambda I)^{-1} \Phi^T y$$
It is easy to show (in the same way by first taking the derivative with respect to $\mathbf{w}$ and setting it to zero) that the regularized least square linear regression solution $\mathbf{w}^r$ takes the form

$$\mathbf{w}^r = (\Phi^T \Phi + \lambda \mathbf{I})^{-1} \Phi^T \mathbf{y}$$

For well chosen $\lambda$ the matrix $(\Phi^T \Phi + \lambda \mathbf{I})$ is diagonally dominant and has an inverse.
It is easy to show (in the same way by first taking the derivative with respect to $w$ and setting it to zero) that the regularized least square linear regression solution $w^r$ takes the form

$$w^r = (\Phi^T \Phi + \lambda I)^{-1} \Phi^T y$$

- For well chosen $\lambda$ the matrix $(\Phi^T \Phi + \lambda I)$ is diagonally dominant and has a inverse
- Each eigenvalue of $(\Phi^T \Phi + \lambda I)$ is at least $\lambda$
Sparse Regularized Least Square

The regularized term can also be
\[ \lambda \sum_{i=1}^{M} |w_i|^q \]
for any \( q > 0 \) so far we have seen \( q = 2 \), i.e.,
\[ \lambda \|w\|_2^2 \] It turns out that \( q = 1 \) situation is of much interest, in particular, the solution \( w_{rs} \) becomes sparse
\[ w_{rs} = \arg \min_{w \in \mathbb{R}^M} \|y - \Phi w\|_2 + \lambda \sum_{i=1}^{M} |w_i| \] i.e., most of the \( w_{rs} \)’s are zeros only few co-ordinates of \( w_{rs} \) are non zero this is called \( \ell_1 \) regularization however, there is no closed form solution for \( w_{rs} \)
Sparse Regularized Least Square

The regularized term can also be \( \lambda \sum_{i=1}^{M} |w_i|^q \) for any \( q > 0 \)
The regularized term can also be $\lambda \sum_{i=1}^{M} |w_i|^q$ for any $q > 0$.

- so far we have seen $q = 2$, i.e., $\lambda \|w\|_2^2$

This is called $\ell_1$ regularization but there is no closed form solution for $w_{rs}$. Only few coordinates of $w_{rs}$ are non-zero.
The regularized term can also be \( \lambda \sum_{i=1}^{M} |w_i|^q \) for any \( q > 0 \)

- so far we have seen \( q = 2 \), i.e., \( \lambda \|w\|_2^2 \)
- It turns out that \( q = 1 \) situation is of much interest, in particular, the solution \( w^{rs} \) becomes sparse

\[
w^{rs} = \arg \min_{w \in \mathbb{R}^M} \|y - \Phi w\|^2 + \lambda \sum_{i=1}^{M} |w_i|
\]
Sparse Regularized Least Square

- The regularized term can also be \( \lambda \sum_{i=1}^{M} |w_i|^q \) for any \( q > 0 \)
  - so far we have seen \( q = 2 \), i.e., \( \lambda \| \mathbf{w} \|_2^2 \)

- It turns out that \( q = 1 \) situation is of much interest, in particular, the solution \( \mathbf{w}^{rs} \) becomes \textit{sparse}

\[
\mathbf{w}^{rs} = \arg \min_{\mathbf{w} \in \mathbb{R}^M} \| \mathbf{y} - \Phi \mathbf{w} \|_2^2 + \lambda \sum_{i=1}^{M} |w_i|
\]

- i.e., most of the \( w_i^{rs} \)'s are zeros
Sparse Regularized Least Square

- The regularized term can also be \( \lambda \sum_{i=1}^{M} |w_i|^q \) for any \( q > 0 \)
- so far we have seen \( q = 2 \), i.e., \( \lambda \|w\|_2^2 \)
- It turns out that \( q = 1 \) situation is of much interest, in particular, the solution \( w^{rs} \) becomes sparse

\[
\begin{align*}
\mathbf{w}^{rs} &= \arg \min_{\mathbf{w} \in \mathbb{R}^M} \|\mathbf{y} - \Phi \mathbf{w}\|^2 + \lambda \sum_{i=1}^{M} |w_i| \\
\end{align*}
\]

- i.e., most of the \( w_i^{rs} \)'s are zeros
- only few co-ordinates of \( \mathbf{w}^{rs} \) are non zero
The regularized term can also be \( \lambda \sum_{i=1}^{M} |w_i|^q \) for any \( q > 0 \)

so far we have seen \( q = 2 \), i.e., \( \lambda \|w\|_2^2 \)

It turns out that \( q = 1 \) situation is of much interest, in particular, the solution \( w^{rs} \) becomes \textit{sparse}

\[
\begin{align*}
\mathbf{w}^{rs} &= \arg \min_{\mathbf{w} \in \mathbb{R}^M} \| \mathbf{y} - \Phi \mathbf{w} \|^2 + \lambda \sum_{i=1}^{M} |w_i| \\
&= \mathbf{w}_{rs}
\end{align*}
\]

i.e., most of the \( w_i^{rs} \)’s are zeros

only few co-ordinates of \( \mathbf{w}^{rs} \) are non zero

this is called \textit{\( \ell_1 \) regularization}
The regularized term can also be $\lambda \sum_{i=1}^{M} |w_i|^q$ for any $q > 0$

so far we have seen $q = 2$, i.e., $\lambda \|w\|_2^2$

It turns out that $q = 1$ situation is of much interest, in particular, the solution $w^{rs}$ becomes sparse

$$w^{rs} = \arg \min_{w \in \mathbb{R}^M} \|y - \Phi w\|^2 + \lambda \sum_{i=1}^{M} |w_i|$$

i.e., most of the $w_i^{rs}$'s are zeros

only few co-ordinates of $w^{rs}$ are non zero

this is called $\ell_1$ regularization

however, there is no closed form solution for $w^{rs}$
Sparse Regularized Least Square

Why does $\ell_1$ regularization provide sparse solution?

Regularization term contour works as constraints.

Square error along with $\ell_1$ regularization constraints limits the different values that $w_{rs}$ is allowed to take.
Why does $\ell_1$ regularization provide a sparse solution?
Sparse Regularized Least Square

Why does $\ell_1$ regularization provides sparse solution?
Regularization term contour works as constraints
Why does $\ell_1$ regularization provides sparse solution?
Regularization term contour works as constraints

$q = 1$
$q = 2$
Why does $\ell_1$ regularization provides sparse solution?

- Regularization term contour works as constraints

Square error along with $\ell_1$ regularization constraints limits the different values that $w^{rs}$ is allowed to take
Given a $N \times M$ design matrix $\Phi$ and $y = (y_1, \ldots, y_N) \in \mathbb{R}^N$ the least square linear regression solution $\hat{w}$ is such that linear combination of the columns of $\Phi$ (with coefficients $\hat{w}_1, \ldots, \hat{w}_M$) is the orthogonal projection of $y$ onto the space spanned by the columns of $\Phi$. 
Given a $N \times M$ design matrix $\Phi$ and $y = (y_1, \ldots, y_N) \in \mathbb{R}^N$, the least square linear regression solution $\hat{w}$ is such that linear combination of the columns of $\Phi$ (with coefficients $\hat{w}_1, \ldots, \hat{w}_M$) is the orthogonal projection of $y$ onto the space spanned by the columns of $\Phi$.

Picture?
Suppose we are given \(N\) input-output pairs \((x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\) where \(x_i \in \mathbb{R}^n\), \(y_i \in \mathbb{R}\) for \(i = 1, \ldots, N\). Consider the linear regression model, where dependence between \(x_i\) and \(y_i\) is assumed to be linear and each \(y_i\) can be modeled as

\[
y_i = \sum_{j=1}^n x_{ij} w_j + w_0 + \epsilon_i = x_i^T w + w_0 + \epsilon_i
\]

where \(w = (w_1, \ldots, w_n) \in \mathbb{R}^n\), \(w_0 \in \mathbb{R}\) and each \(\epsilon_i \sim N(0, \sigma^2)\) for some \(\sigma \in \mathbb{R}\). This situation can be dealt exactly as before by setting \(\tilde{x} = (1, x_1, \ldots, x_n) \in \mathbb{R}^{n+1}\), \(\tilde{w} = (w_0, w_1, \ldots, w_n) \in \mathbb{R}^{n+1}\) such that

\[
y_i = \tilde{x}_i^T \tilde{w} + \epsilon_i
\]
Suppose we are given $N$ input-output pairs
$(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)$ where $x_i \in \mathbb{R}^n, y_i \in \mathbb{R}$ for $i = 1, \ldots, N$
Suppose we are given \( N \) input-output pairs 
\((x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\) where \( x_i \in \mathbb{R}^n, y_i \in \mathbb{R} \) for \( i = 1, \ldots, N \).

Consider the linear regression model, where dependence between \( x_i \) and \( y_i \) is assumed to be linear and each \( y_i \) can be modeled as

\[
y_i = \sum_{j=1}^{n} x_{ij} w_j + w_0 + \epsilon_i = x_i^T w + w_0 + \epsilon_i
\]

where \( w = (w_1, \ldots, w_n) \in \mathbb{R}^n, w_0 \in \mathbb{R} \) and each \( \epsilon_i \sim \mathcal{N}(0, \frac{1}{\sigma}) \) for some \( \sigma \in \mathbb{R} \).
Suppose we are given $N$ input-output pairs $(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)$ where $x_i \in \mathbb{R}^n, y_i \in \mathbb{R}$ for $i = 1, \ldots, N$.

Consider the linear regression model, where dependence between $x_i$ and $y_i$ is assumed to be linear and each $y_i$ can be modeled as

$$y_i = \sum_{j=1}^{n} x_{ij}w_j + w_0 + \epsilon_i = x_i^T \mathbf{w} + w_0 + \epsilon_i$$

where $\mathbf{w} = (w_1, \ldots, w_n) \in \mathbb{R}^n, w_0 \in \mathbb{R}$ and each $\epsilon_i \sim \mathcal{N}\left(0, \frac{1}{\sigma}\right)$ for some $\sigma \in \mathbb{R}$.

This situation can be dealt exactly as before by setting $\tilde{x} = (1, x_1, \ldots, x_n) \in \mathbb{R}^{n+1}, \tilde{\mathbf{w}} = (w_0, w_1, \ldots, w_n) \in \mathbb{R}^{n+1}$ such that

$$y_i = \tilde{x}_i^T \tilde{\mathbf{w}} + \epsilon_i$$
Bias Variance Trade-off

In linear regression model we are given \((x_1, y_1), \ldots, (x_N, y_N)\).

Linear regression model outputs \(y(x)\) as an estimate of \(y_i\) and \(y(x_i)\) is a linear function of \(x_i\).

So what happens when a pair \((x, y)\) is given and Linear regression model is applied on it to predict its output as \(y(x)\)?

What is the error or loss \(L\), as it is often called, we suffer on average?

\[
E[L] = \int \int (y(x) - y)^2 p(x, y) \, dx \, dy
\]

Our goal is to choose \(y(x)\) that minimizes \(E[L]\).

Using calculus of variation we get

\[
dE[L] = 2 \int (y(x) - y)p(x, y) \, dy = 0
\]

Solving for \(y(x)\) we get

\[
y(x) = \int y p(x, y) \, dy = E[y|x]
\]
In linear regression model we are given \((x_1, y_1), \ldots, (x_N, y_N)\)
In linear regression model we are given $\left( x_1, y_1 \right), \ldots, \left( x_N, y_N \right)$.

Linear regression model outputs $y(x_i)$ as an estimate of $y_i$ and $y(x_i)$ is a linear function of $x_i$.
In linear regression model we are given \((x_1, y_1), \ldots, (x_N, y_N)\).

Linear regression model outputs \(y(x_i)\) as an estimate of \(y_i\) and
\(y(x_i)\) is a linear function of \(x_i\).

So what happens when a pair \((x, y)\) is given and linear regression model is applied on it to predict its output as \(y(x)\)?
Bias Variance Trade-off

- In linear regression model we are given \((x_1, y_1), \ldots, (x_N, y_N)\)
- Linear regression model outputs \(y(x_i)\) as an estimate of \(y_i\) and \(y(x_i)\) is a linear function of \(x_i\)
- So what happens when a pair \((x, y)\) is given and Linear regression model is applied on it to predict its output as \(y(x)\)?

  - what is the error or loss \(L\), as it is often called, we suffer on average?
Bias Variance Trade-off

- In linear regression model we are given \((x_1, y_1), \ldots, (x_N, y_N)\)
- Linear regression model outputs \(y(x_i)\) as an estimate of \(y_i\) and \(y(x_i)\) is a linear function of \(x_i\)
- So what happens when a pair \((x, y)\) is given and Linear regression model is applied on it to predict its output as \(y(x)\)?
  - what is the error or loss \(L\), as it is often called, we suffer on average?

\[
E[L] = \int \int (y(x) - y)^2 p(x, y) dx dy
\]
In linear regression model we are given \((x_1, y_1), \ldots, (x_N, y_N)\).

Linear regression model outputs \(y(x_i)\) as an estimate of \(y_i\) and \(y(x_i)\) is a linear function of \(x_i\).

So what happens when a pair \((x, y)\) is given and Linear regression model is applied on it to predict its output as \(y(x)\)?

- what is the error or loss \(L\), as it is often called, we suffer on average?

\[
E[L] = \int \int (y(x) - y)^2 p(x, y) dx dy
\]

- our goal is to choose \(y(x)\) that minimizes \(E[L]\)
In linear regression model we are given \((x_1, y_1), \ldots, (x_N, y_N)\).

Linear regression model outputs \(y(x_i)\) as an estimate of \(y_i\) and \(y(x_i)\) is a linear function of \(x_i\).

So what happens when a pair \((x, y)\) is given and linear regression model is applied on it to predict its output as \(y(x)\)?

- what is the error or loss \(L\), as it is often called, we suffer on average?

\[
E[L] = \int \int (y(x) - y)^2 p(x, y) dx dy
\]

- our goal is to choose \(y(x)\) that minimizes \(E[L]\)
- Using calculus of variation we get

\[
\frac{dE[L]}{dy(x)} = 2 \int (y(x) - y) p(x, y) dy = 0
\]
In linear regression model we are given \((x_1, y_1), \ldots, (x_N, y_N)\).

Linear regression model outputs \(y(x_i)\) as an estimate of \(y_i\) and \(y(x_i)\) is a linear function of \(x_i\).

So what happens when a pair \((x, y)\) is given and Linear regression model is applied on it to predict its output as \(y(x)\)?

- what is the error or loss \(L\), as it is often called, we suffer on average?

\[
E[L] = \int \int (y(x) - y)^2 p(x, y) dx dy
\]

- our goal is to choose \(y(x)\) that minimizes \(E[L]\)
- Using calculus of variation we get

\[
\frac{dE[L]}{dy(x)} = 2 \int (y(x) - y) p(x, y) dy = 0
\]

- solving for \(y(x)\) we get

\[
y(x) = \frac{\int yp(x, y) dy}{p(x)} = \int yp(y|x) dy = E[y|x]
\]
Note that

\[(y(x) - y)^2 = (y(x) - \mathbb{E}(y|x) + \mathbb{E}[y|x] - y)^2\]
Bias Variance Trade-off

Note that

\[(y(x) - y)^2 = (y(x) - E(y|x) + E[y|x] - y)^2\]
\[= (y(x) - E[y|x])^2 + (E[y|x] - y)^2 + 2(y(x) - E[y|x])(E[y|x] - y)\]
Bias Variance Trade-off

Note that

\[(y(x) - y)^2 = (y(x) - E(y|x) + E[y|x] - y)^2 = (y(x) - E[y|x])^2 + (E[y|x] - y)^2 + 2(y(x) - E[y|x])(E[y|x] - y)\]

To find \(E[L]\), we need to integrate the above equation
Bias Variance Trade-off

Note that

\[(y(x) - y)^2 = (y(x) - E(y|x) + E[y|x] - y)^2\]
\[= (y(x) - E[y|x])^2 + (E[y|x] - y)^2 + 2(y(x) - E[y|x])(E[y|x] - y)\]

To find \(E[L]\), we need to integrate the above equation

Let’s focus on the cross product term
Note that
\[
(y(x) - y)^2 = (y(x) - E[y|x] + E[y|x] - y)^2
= (y(x) - E[y|x])^2 + (E[y|x] - y)^2 + 2(y(x) - E[y|x])(E[y|x] - y)
\]

To find $E[L]$, we need to integrate the above equation.

Let's focus on the cross product term

\[
\int \int (y(x) - E[y|x])(E[y|x] - y)p(x, y)dx dy = \int \int (y(x) - E[y|x])(E[y|x] - y)p(x)p(y|x)dx dy
\]
Note that

\[(y(x) - y)^2 = (y(x) - E(y|x) + E[y|x] - y)^2 = (y(x) - E[y|x])^2 + (E[y|x] - y)^2 + 2(y(x) - E[y|x])(E[y|x] - y)\]

To find \(E[L]\), we need to integrate the above equation.

Let’s focus on the cross product term

\[
\int \int (y(x) - E[y|x])(E[y|x] - y)p(x, y)dx\,dy = \int \int (y(x) - E[y|x])(E[y|x] - y)p(x)p(y|x)dx\,dy
\]

\[
= \int (y(x) - E[y|x]) \left( \int (E[y|x] - y)p(y|x)dy \right) p(x)dx
\]
Note that

\[(y(x) - y)^2 = (y(x) - E(y|x) + E[y|x] - y)^2\]

\[= (y(x) - E[y|x])^2 + (E[y|x] - y)^2 + 2(y(x) - E[y|x])(E[y|x] - y)\]

To find \(E[L]\), we need to integrate the above equation.

Let's focus on the cross product term

\[
\int \int (y(x) - E[y|x])(E[y|x] - y)p(x, y)dxdy = \int \int (y(x) - E[y|x])(E[y|x] - y)p(x)p(y|x)dxdy
\]

\[= \int (y(x) - E[y|x]) \left( \int (E[y|x] - y)p(y|x)dy \right) p(x)dx\]

\[= \int (y(x) - E[y|x]) \left( E[y|x] - \int yp(y|x)dy \right) p(x)dx\]

\[= \int \left( y(x) - E[y|x] \right) \left( E[y|x] - \int yp(y|x)dy \right) p(x)dx\]

\[= 0\]
Bias Variance Trade-off

- Note that

\[
(y(x) - y)^2 = (y(x) - E(y|x) + E[y|x] - y)^2
= (y(x) - E[y|x])^2 + (E[y|x] - y)^2 + 2(y(x) - E[y|x])(E[y|x] - y)
\]

- To find \( E[L] \), we need to integrate the above equation

Let's focus on the cross product term

\[
\int \int (y(x) - E[y|x])(E[y|x] - y)p(x, y)dxdy = \int \int (y(x) - E[y|x])(E[y|x] - y)p(x)p(y|x)dxdy
= \int (y(x) - E[y|x]) \left( \int (E[y|x] - y)p(y|x)dy \right) p(x)dx
= \int (y(x) - E[y|x]) \left( \begin{array}{c}
E[y|x] - \int yp(y|x)dy
\end{array} \right) p(x)dx
= \left( \begin{array}{c}
E[y|x] - \int yp(y|x)dy
\end{array} \right) p(x)dx
= 0
\]
Bias Variance Trade-off

Let \( h(x) = \mathbb{E}[y|x] \)

Therefore

\[
\mathbb{E}[L] = \int \int \left( (y(x) - \mathbb{E}[y|x])^2 + (\mathbb{E}[y|x] - y)^2 \right) p(x,y) \, dx \, dy = \int \left( y(x) - h(x) \right)^2 p(x) \, dx + \int \int \left( h(x) - y \right)^2 p(x,y) \, dx \, dy
\]

\[= \text{noise}(2) \]

Let's focus on the first term. The prediction \( y(x) \) depends on what training size we use. We would better represent \( y(x) \) as \( (x; D) \), where \( D \) is a random training set. So the first term in the above equation should read

\[
\int \left( y(x; D) - h(x) \right)^2 p(x) \, dx
\]
Let $h(x) = E[y|x]$
• Let $h(x) = E[y|x]$
• Therefore

$$E[L] = \int \int ((y(x) - E[y|x])^2 + (E[y|x] - y)^2) \, p(x, y) \, dx \, dy$$
Let $h(x) = E[y|x]$

Therefore

$$E[L] = \int \int ((y(x) - E[y|x])^2 + (E[y|x] - y)^2) \, p(x, y) \, dx \, dy$$

$$= \int \int ((y(x) - h(x))^2 + (h(x) - y)^2) \, p(x, y) \, dx \, dy$$
Bias Variance Trade-off

- Let $h(x) = E[y|x]$
- Therefore

\[
E[L] = \int \int ((y(x) - E[y|x])^2 + (E[y|x] - y)^2) p(x, y) dx dy
\]

\[
= \int \int ((y(x) - h(x))^2 + (h(x) - y)^2) p(x, y) dx dy
\]

\[
= \int (y(x) - h(x))^2 p(x) dx + \int \int (h(x) - y)^2 p(x, y) dx dy
\]

(2)
Bias Variance Trade-off

- Let \( h(x) = E[y|x] \)
- Therefore

\[
E[L] = \int \int \left( (y(x) - E[y|x])^2 + (E[y|x] - y)^2 \right) p(x, y) dxdy
\]

\[
= \int \int \left( (y(x) - h(x))^2 + (h(x) - y)^2 \right) p(x, y) dxdy
\]

\[
= \int (y(x) - h(x))^2 p(x) dx + \int \int (h(x) - y)^2 p(x, y) dxdy
\]

\[
\text{noise}
\]

- Lets focus on the first term
Bias Variance Trade-off

- Let \( h(x) = E[y|x] \)
- Therefore

\[
E[L] = \int \int ((y(x) - E[y|x])^2 + (E[y|x] - y)^2) p(x, y) dx dy
\]

\[
= \int \int ((y(x) - h(x))^2 + (h(x) - y)^2) p(x, y) dx dy
\]

\[
= \int (y(x) - h(x))^2 p(x) dx + \int \int (h(x) - y)^2 p(x, y) dx dy
\]

\[\text{noise}\] (2)

- Lets focus on the first term
  - the prediction \( y(x) \) depends on what training size we use
Bias Variance Trade-off

- Let \( h(x) = E[y|x] \)
- Therefore

\[
E[L] = \int \int ((y(x) - E[y|x])^2 + (E[y|x] - y)^2) p(x, y) dx dy
\]

\[
= \int \int ((y(x) - h(x))^2 + (h(x) - y)^2) p(x, y) dx dy
\]

\[
= \int (y(x) - h(x))^2 p(x) dx + \int \int (h(x) - y)^2 p(x, y) dx dy
\]

(2)

- Lets focus on the first term
  - the prediction \( y(x) \) depends on what training size we use
  - we would better represent \( y(x) \) as \( (x; D) \), where \( D \) is a random training set
Let $h(x) = E[y|x]$

Therefore

$$E[L] = \int \int ((y(x) - E[y|x])^2 + (E[y|x] - y)^2) \, p(x,y) \, dx \, dy$$

$$= \int \int ((y(x) - h(x))^2 + (h(x) - y)^2) \, p(x,y) \, dx \, dy$$

$$= \int (y(x) - h(x))^2 \, p(x) \, dx + \int \int (h(x) - y)^2 \, p(x,y) \, dx \, dy \tag{2}$$

\[\text{noise}\]

Let's focus on the first term
- the prediction $y(x)$ depends on what training size we use
- we would better represent $y(x)$ as $(x; D)$, where $D$ is a random training set
- so the first term in the above equation should read

$$\int (y(x; D) - h(x))^2 \, p(x) \, dx$$
Now we can write:

\[(y(x; D - h(x)))^2 = (y(x; D - E_D [y(x; D)]) + E_D [y(x; D)] - h(x))^2 = (y(x; D - E_D [y(x; D)]))^2 + (E_D [y(x; D)] - h(x))^2 + 2(y(x; D - E_D [y(x; D)])(E_D [y(x; D)] - h(x)))\]

Taking expectation with respect to \(D\), the cross term vanishes.

Thus we are left with:

\[E_D [(y(x; D) - h(x))^2] = (E_D [y(x; D) - h(x))^2 \equiv \text{bias}_2^2 + E_D [y(x; D) - E_D [y(x; D)]]^2 \equiv \text{variance}\]

Expected loss = (bias)^2 + variance + noise

Bias tells us how close the output estimation is to the true underlying output.

Variance tells us how reliable our estimation is, i.e., how much it will vary if we change a different training set.
Now we can write

\[(y(x; D - h(x))^2\]
\[= (y(x; D - E_D[y(x; D)]) + E_D[y(x; D)] - h(x))^2\]
Now we can write

\[(y(x; D - h(x))^2 = (y(x; D - E_D[y(x; D)] + E_D[y(x; D)] - h(x))^2 = (y(x; D - E_D[y(x; D)]))^2 + (E_D[y(x; D)] - h(x))^2\]
Now we can write

\[
(y(x; D - h(x)))^2 \\
= (y(x; D - E_D[y(x; D)]) + E_D[y(x; D)] - h(x))^2 \\
= (y(x; D - E_D[y(x; D)]))^2 + (E_D[y(x; D)] - h(x))^2 \\
+ 2(y(x; D - E_D[y(x; D)])(E_D[y(x; D)] - h(x))
\]
Now we can write

\[(y(x; D - h(x))^2 = (y(x; D - E_D[y(x; D)]) + E_D[y(x; D)] - h(x))^2 = (y(x; D - E_D[y(x; D)]))^2 + (E_D[y(x; D)] - h(x))^2 + 2(y(x; D - E_D[y(x; D)])(E_D[y(x; D)] - h(x))\]

Taking expectation with respect to \(D\) the cross term vanishes.
Now we can write

\[
(y(x; D - h(x)))^2 \\
= (y(x; D - E_D[y(x; D)]) + E_D[y(x; D)]) - h(x))^2 \\
= (y(x; D - E_D[y(x; D)]))^2 + (E_D[y(x; D)] - h(x))^2 \\
+ 2(y(x; D - E_D[y(x; D)])(E_D[y(x; D)] - h(x))
\]

Taking expectation with respect to \( D \) the cross term vanishes

Thus we are left with

\[
E_D[(y(x; D) - h(x))^2] = (E_D[y(x; D) - h(x)])^2 + E_D[y(x; D) - E_D[(y(x; D))]^2
\]

Bias tells us how close the output estimation is to the true underlying output

Variance tells us how reliable our estimation is, i.e., if we change a different training set how much it going to vary
Now we can write

\[(y(x; \mathcal{D} - h(x))^2 = (y(x; \mathcal{D} - E_{\mathcal{D}}[y(x; \mathcal{D})] + E_{\mathcal{D}}[y(x; \mathcal{D})] - h(x))^2
= (y(x; \mathcal{D} - E_{\mathcal{D}}[y(x; \mathcal{D})])^2 + (E_{\mathcal{D}}[y(x; \mathcal{D})] - h(x))^2
+ 2(y(x; \mathcal{D} - E_{\mathcal{D}}[y(x; \mathcal{D})])(E_{\mathcal{D}}[y(x; \mathcal{D})] - h(x))\]

Taking expectation with respect to \(\mathcal{D}\) the cross term vanishes

Thus we are left with

\[E_{\mathcal{D}}[(y(x; \mathcal{D}) - h(x))^2] = (E_{\mathcal{D}}[y(x; \mathcal{D})] - h(x))^2 + E_{\mathcal{D}}[y(x; \mathcal{D}) - E_{\mathcal{D}}[(y(x; \mathcal{D}))]^2\]

expected loss = (bias)^2 + variance + noise
Now we can write

\[(y(x; D - h(x)))^2 = (y(x; D - E_D[y(x; D)]) + E_D[y(x; D)] - h(x))^2\]

= \[(y(x; D - E_D[y(x; D)]))^2 + (E_D[y(x; D)] - h(x))^2 + 2(y(x; D - E_D[y(x; D)])(E_D[y(x; D)] - h(x))\]

Taking expectation with respect to \(D\) the cross term vanishes

Thus we are left with

\[E_D[(y(x; D) - h(x))^2] = (E_D[y(x; D)] - h(x))^2 + E_D[y(x; D) - E_D[y(x; D)]]^2\]

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

Bias tells us how close the output estimation is to the true underlying output
Bias Variance Trade-off

Now we can write

\[
(y(x; D - h(x))^2 = (y(x; D - E_D[y(x; D)]) + E_D[y(x; D)] - h(x))^2 \\
= (y(x; D - E_D[y(x; D)]))^2 + (E_D[y(x; D)] - h(x))^2 \\
+ 2(y(x; D - E_D[y(x; D)])(E_D[y(x; D)] - h(x))
\]

Taking expectation with respect to \( D \) the cross term vanishes

Thus we are left with

\[
E_D[(y(x; D) - h(x))^2] = (E_D[y(x; D)] - h(x))^2 + E_D[y(x; D) - E_D[(y(x; D))]^2
\]

expected loss = \((bias)^2 + variance + noise\)

- Bias tells us how close the output estimation is to the true underlying output
- Variance tells us how reliable our estimation is, i.e., if we change a different training set how much it going to vary.
Over-fitting and Under-fitting

When do we say the model is over fitted?

When do we say the model is under fitted?

Kaushik Sinha

Linear Regression and Linear Basis Function Model
When do we say the model is *over fitted*?
Over-fitting and Under-fitting

- When do we say the model is over fitted?
- When do we say the model is under fitted?
Over-fitting and Under-fitting

- When do we say the model is **over fitted**?
- When do we say the model is **under fitted**?
Over-fitting and Under-fitting

The model is over-fitted when training error is very small but testing error is large. It is too specific for the training sample, with no generalization ability.

The model is under-fitted when both training error and testing errors are large. The model is not complex enough to capture/represent a functional relationship $f: x \rightarrow y$.

Kaushik Sinha

Linear Regression and Linear Basis Function Model
Over-fitting and Under-fitting

The model is **over-fitted** when training error is very small but testing error is large.

- **High Bias, Low Variance**
- **Low Bias, High Variance**

- The model is under-fitted when both training error and testing errors are large. The model is not complex enough to capture/represent a functional relationship $f: x \rightarrow y$.
Over-fitting and Under-fitting

- The model is **over-fitted** when training error is very small but testing error is large
  - too specific for the training sample no generalization ability

![Graph showing the relationship between model complexity and prediction error](image-url)

- High Bias, Low Variance
- Low Bias, High Variance

Kaushik Sinha
Linear Regression and Linear Basis Function Model
Over-fitting and Under-fitting

The model is over-fitted when training error is very small but testing error is large
- too specific for the training sample no generalization ability

The model is under-fitted when both training error and testing errors are large
The model is **over-fitted** when training error is very small but testing error is large
- too specific for the training sample no generalization ability

The model is **under-fitted** when both training error and testing errors are large
- the model is not complex enough to capture/represent a functional relationship $f : x \rightarrow y$

**Over-fitting and Under-fitting**

![Graph showing the relationship between model complexity and prediction error.](image-url)