CS2109S Tutorial 5

Regularisation and Kernel Trick

(AY 25/26 Semester 2)

March 12, 2026

(Prepared by Benson)

Contents

Recap: Regularisation

Why do we need regularisation?

A. To decrease model complexity.

B. To improve training time efficiency.

C. To avoid underfitting.

D. To avoid overfitting.

Recap: Regularisation

How does regularisation address overfitting?

A. By penalizing weights for transformed features.
B. By reducing noise in the training data.
C. By penalizing large weights.
D. By reducing the number of transformed features.

Q1. L1-norm vs. L2-norm.

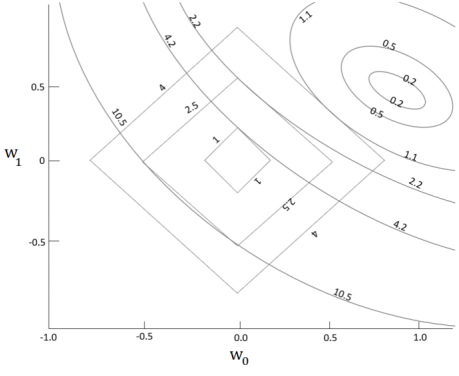
L2-norm (Ridge Regression):

$$J(\boldsymbol{w}) = \frac{1}{2m}\left[\sum_{i=1}^{n}(h_{\boldsymbol{w}}(\boldsymbol{x}^{(i)}) - y^{(i)})^2\right] + \lambda\sum_{i=1}^{n}\boldsymbol{w}_i^2$$

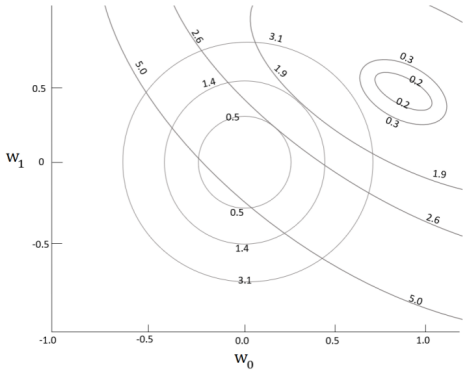
L1-norm (Lasso Regression):

$$J(\boldsymbol{w}) = \frac{1}{2m}\left[\sum_{i=1}^{n}(h_{\boldsymbol{w}}(\boldsymbol{x}^{(i)}) - y^{(i)})^2\right] + \lambda\sum_{i=1}^{n}|\boldsymbol{w}_i|$$

Q1. L1-norm vs. L2-norm.



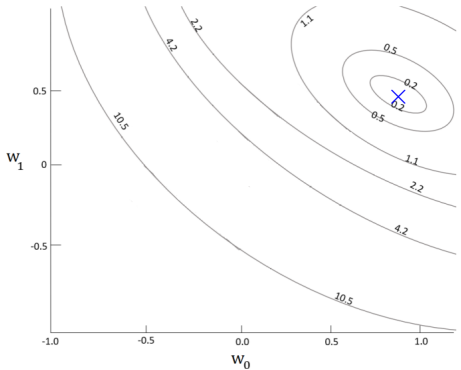
L1 regularisation



L2 regularisation

Q1. L1-norm vs. L2-norm.

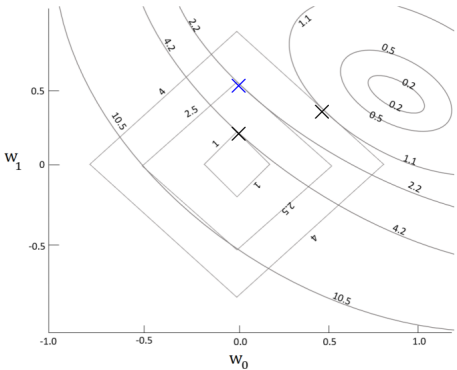
(a) No regularisation.



$w_0 = 0.9, w_1 = 0.5$
Cost ≈ 0

Q1. L1-norm vs. L2-norm.

(b) L1 regularisation with $\lambda = 5$.



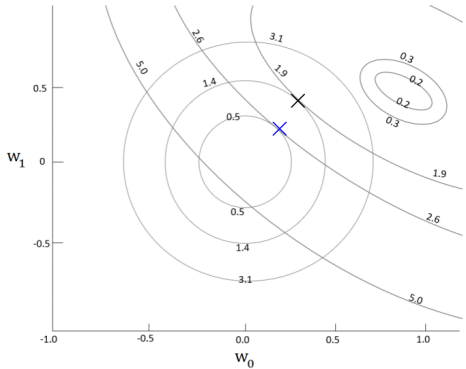
Total cost of the three points (from left to right):

- $4.2 + 1 = 5.2$
- $2.2 + 2.5 = 4.7$
- $1.1 + 4 = 5.1$

$w_0 = 0.0, w_1 = 0.5$
Cost $= 4.7$

Q1. L1-norm vs. L2-norm.

(c) L2 regularisation with $\lambda = 5$.

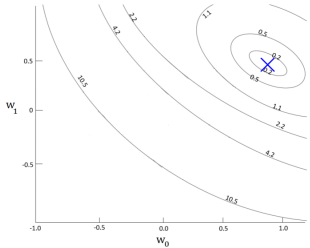


Total cost of the two points (from left to right):

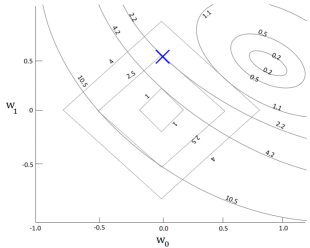
- $0.5 + 2.6 = 3.1$
- $1.9 + 1.4 = 3.3$

$w_0 = 0.2, w_1 = 0.25$
Cost $= 3.1$

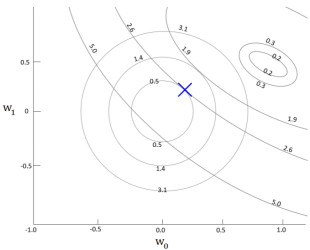
Q1. L1-norm vs. L2-norm.



No regularisation
$w_0 = 0.9, w_1 = 0.5$
Cost ≈ 0



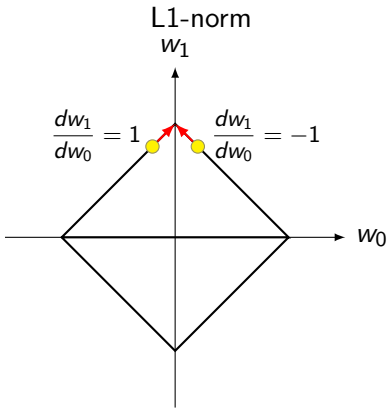
L1 regularisation
$w_0 = 0.0, w_1 = 0.5$
Cost $= 4.7$



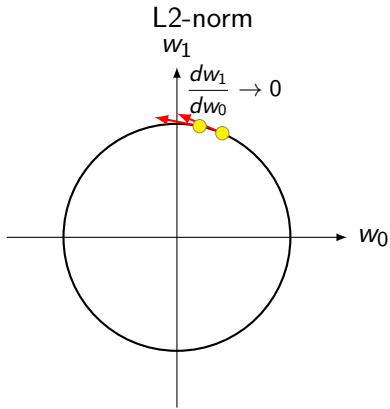
L2 regularisation
$w_0 = 0.2, w_1 = 0.25$
Cost $= 3.1$

- L2 heavily penalizes larger parameters, preferring **all** smaller values.
- L1 may set values of certain parameters to 0 (why?).

Q1. L1-norm vs. L2-norm.



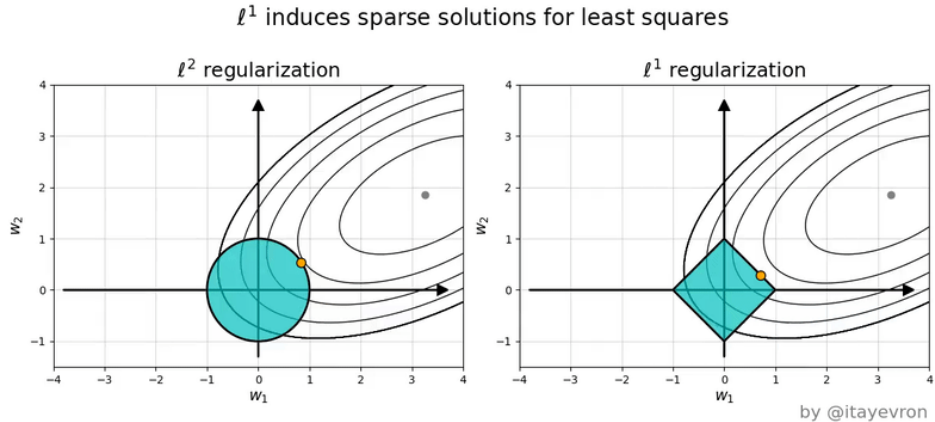
If w_1 is more important than w_0, pushing towards w_1 is free lunch ("one for one").



As $w_0 \to 0$, pushing towards w_1 has little gain ($\because$ push will be "less aggressive").

Q1. L1-norm vs. L2-norm.

Animation (See HTML slides):



Q1. L1-norm vs. L2-norm.

L2-norm (Ridge Regression):

$$J(\boldsymbol{w}) = \frac{1}{2m}\left[\sum_{i=1}^{n}(h_{\boldsymbol{w}}(\boldsymbol{x}^{(i)}) - y^{(i)})^2\right] + \lambda\sum_{i=1}^{n}\boldsymbol{w}_i^2$$

- L2 heavily penalizes larger parameters, preferring **all** smaller values.

L1-norm (Lasso Regression):

$$J(\boldsymbol{w}) = \frac{1}{2m}\left[\sum_{i=1}^{n}(h_{\boldsymbol{w}}(\boldsymbol{x}^{(i)}) - y^{(i)})^2\right] + \lambda\sum_{i=1}^{n}|\boldsymbol{w}_i|$$

- L1 may set values of certain parameters to 0 $\rightarrow$ effectively "feature selection" (select the more important features, and zero out the rest).

Q2. Normal Equation with L2 Regularization

(a) Explain why the unnormalized normal equation cannot be used:

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

- The two features x_1 and x_2 are linearly dependent. So $\boldsymbol{X}^\top \boldsymbol{X}$ is not invertible.

x_1	x_2	y
1	2	3
2	4	5

Q2. Normal Equation with L2 Regularization

(b) Suppose we add regularisation with parameter $\lambda = 1$.
Compute the weight vector $\boldsymbol{w}$.

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

x_1	x_2	y
1	2	3
2	4	5

Solution.

$$\boldsymbol{X} = \begin{bmatrix} 1 & 1 & 2 \\ 1 & 2 & 4 \end{bmatrix}, \ \boldsymbol{y} = \begin{bmatrix} 3 \\ 5 \end{bmatrix}$$

$$\boldsymbol{w} = (\boldsymbol{X}^\top \boldsymbol{X})^{-1} \boldsymbol{X}^\top \boldsymbol{y} = \begin{bmatrix} \frac{13}{33} & \frac{5}{11} & \frac{10}{11} \end{bmatrix}^\top$$

$\therefore \hat{y} = \frac{13}{33} + \frac{5}{11} x_1 + \frac{10}{11} x_2.$

Note. $\boldsymbol{X}^\top \boldsymbol{X} + \lambda I$ is always invertible for all $\lambda > 0$.

Recap: Dual Representation and Kernel Trick

Primal Form of Linear Regression:

$$h_{\boldsymbol{w}}(\boldsymbol{x}) = \boldsymbol{w} \cdot \boldsymbol{x}$$

Dual Form of Linear Regression:

$$h_{\boldsymbol{\alpha}}(\boldsymbol{x}) = \left(\sum_{j=1}^{n} \alpha_j \boldsymbol{x}^{(j)} \right) \cdot \boldsymbol{x} = \sum_{j=1}^{n} \alpha_j (\boldsymbol{x}^{(j)} \cdot \boldsymbol{x})$$

What happens if we apply the **feature transformation** $\phi(x) = \begin{bmatrix} x & x & x & x \end{bmatrix}^\top$?

Recap: Dual Representation and Kernel Trick

(Demo: See HTML slides)

Recap: Dual Representation and Kernel Trick

Important Property of a Kernel Function

$$K(\boldsymbol{u}, \boldsymbol{v}) = \phi(\boldsymbol{u}) \cdot \phi(\boldsymbol{v}) \text{ for some function } \phi$$

If we have a model that **ONLY** relies on the dot products of $\boldsymbol{x}^{(i)}$:

$$h_{\boldsymbol{\alpha}}(\boldsymbol{x}) = \sum_{j=1}^{n} \alpha_j (\boldsymbol{x}^{(j)} \cdot \boldsymbol{x})$$

Applying the kernel function $\equiv$ applying the transformed features!

$$h_{\boldsymbol{\alpha}}(\boldsymbol{x}) = \sum_{j=1}^{n} \alpha_j (\phi(\boldsymbol{x}^{(j)}) \cdot \phi(\boldsymbol{x})) = \sum_{j=1}^{n} \alpha_j (k(\boldsymbol{x}^{(j)}, \boldsymbol{x}))$$

Q3. Dual Representation of Logistic Regression

Primal Form of Logistic Regression:

$$h_{\boldsymbol{w}}(\boldsymbol{x}) = \sigma(\boldsymbol{w}^\top \boldsymbol{x}) = \frac{1}{1 + e^{-\boldsymbol{w}^\top \boldsymbol{x}}}$$

(a) Given $\boldsymbol{w} = \sum_{i=1}^{n} \alpha_i \boldsymbol{x}^{(i)}$, derive the dual representation of $h_{\boldsymbol{w}}(x)$.

$$\begin{aligned} h_{\boldsymbol{\alpha}}(\boldsymbol{x}) &= \sigma\left(\left(\sum_{i=1}^{n} \alpha_i \boldsymbol{x}^{(i)}\right)^\top \boldsymbol{x}\right) \\ &= \sigma\left(\sum_{i=1}^{n} \alpha_i (\boldsymbol{x}^{(i)} \cdot \boldsymbol{x})\right) \end{aligned}$$

(b) Write down the expression for the BCE loss $J^{BCE}(\boldsymbol{\alpha})$ in dual form.

$$J^{BCE}(\alpha) = \frac{1}{n} \sum_{i=1}^{n} BCE\left(y^{(i)}, \sigma\left(\sum_{i=1}^{n} \alpha_i (\boldsymbol{x}^{(i)} \cdot \boldsymbol{x})\right)\right)$$

We do not need to explicitly compute the weight vector $\boldsymbol{w}$!

Q3. Dual Representation of Logistic Regression

Primal Form of Logistic Regression:

$$h_{\boldsymbol{w}}(\boldsymbol{x}) = \sigma(\boldsymbol{w}^\top \boldsymbol{x})$$

Dual Form of Logistic Regression:

$$h_{\boldsymbol{\alpha}}(\boldsymbol{x}) = \sigma\left(\sum_{i=1}^{n} \alpha_i (\boldsymbol{x}^{(i)} \cdot \boldsymbol{x})\right)$$

(c) How many parameters does the primal/dual logistic regression model have?

- Primal: d parameters.
- Dual: n parameters.

(d) When is primal/dual representation more efficient?

- Primal: When d is much smaller than n.
- Dual: When d is large. Especially when the kernel trick could be used.

Q4. Kernel Trick

(a) The linear kernel has a mapping $\phi(x) = x$. Compute $k(\boldsymbol{u}, \boldsymbol{v})$ for $\boldsymbol{u} = \begin{bmatrix} 2 \\ 3 \end{bmatrix}$ and $\boldsymbol{v} = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$.

$$k(\boldsymbol{u}, \boldsymbol{v}) = \begin{bmatrix} 2 \\ 3 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ -1 \end{bmatrix} = -1$$

(b) Consider the polynomial kernel of degree 2: $k(\boldsymbol{u}, \boldsymbol{v}) = (\boldsymbol{u}^\top \boldsymbol{v} + 1)^2$. Compute $k(\boldsymbol{u}, \boldsymbol{v})$ for $\boldsymbol{u} = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$ and $\boldsymbol{v} = \begin{bmatrix} 3 \\ 1 \end{bmatrix}$.

$$k(\boldsymbol{u}, \boldsymbol{v}) = \left(\begin{bmatrix} 1 \\ 2 \end{bmatrix} \cdot \begin{bmatrix} 3 \\ 1 \end{bmatrix} + 1 \right)^2 = (5 + 1)^2 = 36$$

Q4. Kernel Trick

(c) Consider the polynomial kernel of degree 2: $k(\boldsymbol{u}, \boldsymbol{v}) = (\boldsymbol{u}^\top \boldsymbol{v} + 1)^2$. Find its explicit feature mapping.

$$
\begin{aligned}
k(\boldsymbol{u}, \boldsymbol{v}) &= (\boldsymbol{u}^\top \boldsymbol{v} + 1)^2 \\
&= (u_1 v_1 + u_2 v_2 + 1)^2 \\
&= u_1^2 v_1^2 + u_2^2 v_2^2 + 2u_1 u_2 v_1 v_2 + 2u_1 v_1 + 2u_2 v_2 + 1 \\
&= \begin{bmatrix} u_1^2 \\ u_2^2 \\ \sqrt{2} u_1 u_2 \\ \sqrt{2} u_1 \\ \sqrt{2} u_2 \\ 1 \end{bmatrix} \cdot \begin{bmatrix} v_1^2 \\ v_2^2 \\ \sqrt{2} v_1 v_2 \\ \sqrt{2} v_1 \\ \sqrt{2} v_2 \\ 1 \end{bmatrix}
\end{aligned}
$$

Hence $\phi(\boldsymbol{x}) = \begin{bmatrix} x_1^2 & x_2^2 & \sqrt{2} x_1 x_2 & \sqrt{2} x_1 & \sqrt{2} x_2 & 1 \end{bmatrix}^\top$.

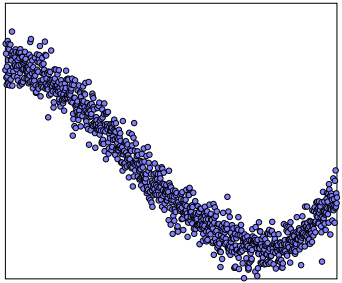
Q4. Kernel Trick

(d) What are the advantages of kernel trick?

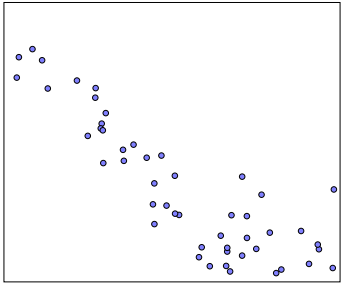
- We never explicitly compute $\phi(x)$, saving time and memory!
- We can change the feature transformation just by swapping $\phi(x)$.
- For infinite dimensional kernels (RBF / Gaussian kernel), explicit computation is impossible.

Bias and Variance

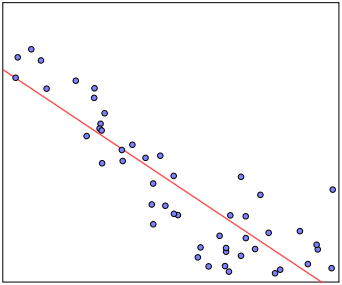
Population



Sample



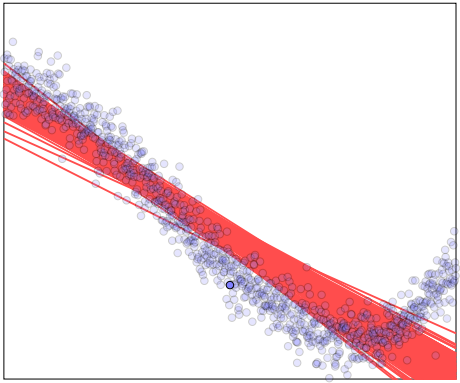
Trained Model

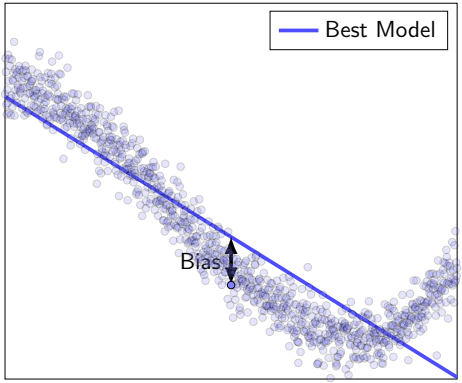


Hypothesis class:
$h_w(x) = w_0 x + b$

Bias and Variance

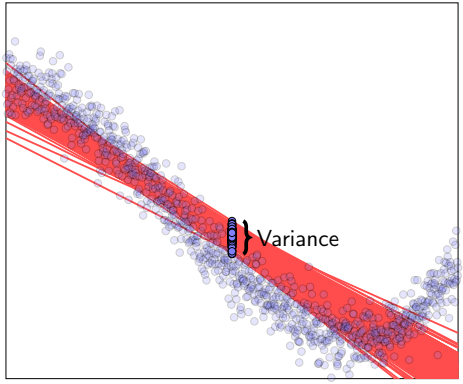
Bias measures how much the **expected** *predicted value* differs from the *actual value*. Intuition: The error of the **best model** when you are given **infinite amount of training data**.





Bias and Variance

Variance measures the consistency of predictions due to sampling.



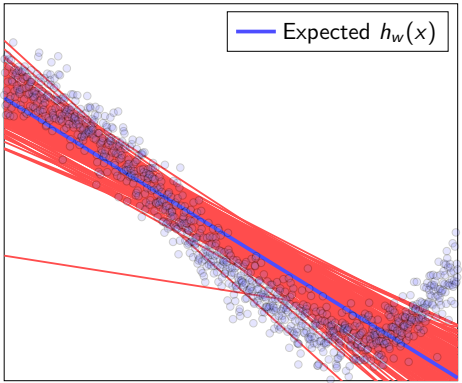
Bias and Variance

Poll:

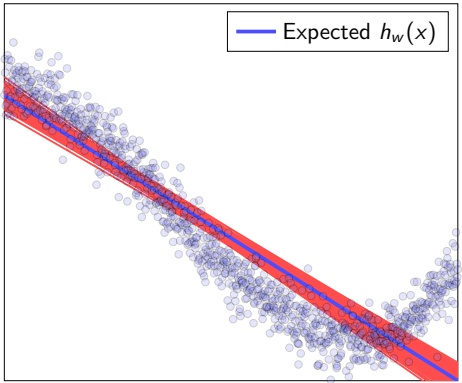
1. If we increase the amount of training samples:
   The bias remains unchanged and the variance decreases.
2. If we increase the degree of the polynomial in our hypothesis class:
   The bias decreases and the variance increases.

Bias and Variance

Less samples (Higher variance)

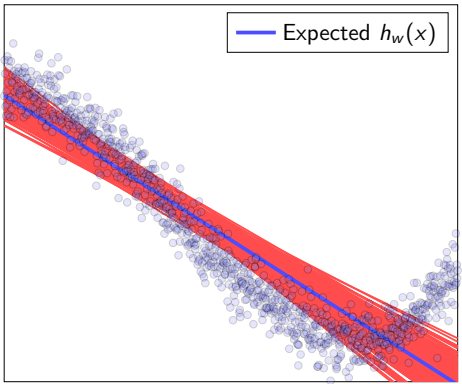


More samples (Lower variance)

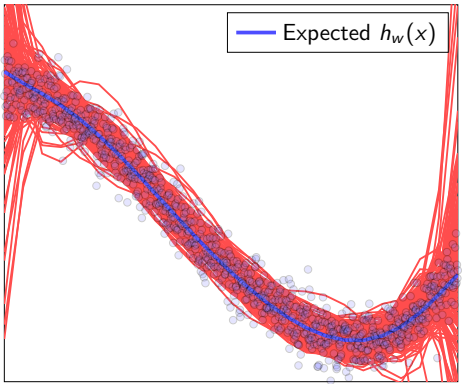


Bias and Variance

Lower degree (High bias; Low variance)

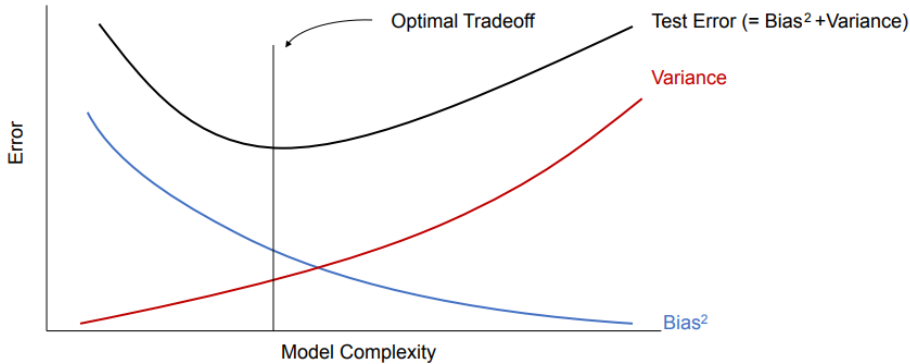


Higher degree (Low bias; High variance)



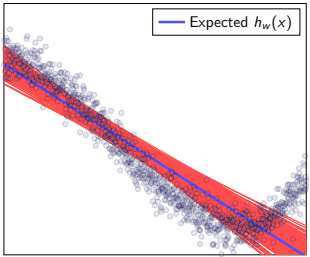
Bias-Variance Tradeoff

$$\mathrm{Err}(x) = \left(\underbrace{y - \mathbb{E}[h_w(x)]}_{\text{bias}} \right)^2 + \underbrace{\mathbb{E}\left[(\mathbb{E}[h_w(x)] - h_w(x))^2 \right]}_{\text{variance}}$$

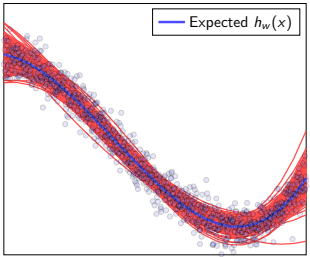


Bias-Variance Tradeoff

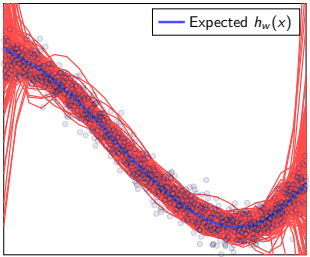
Underfitting



Just Right



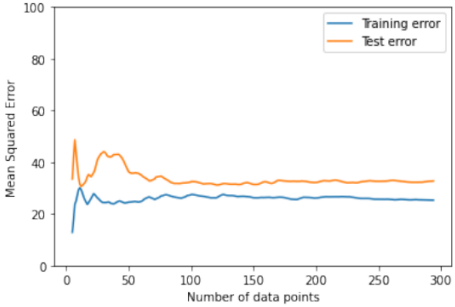
Overfitting

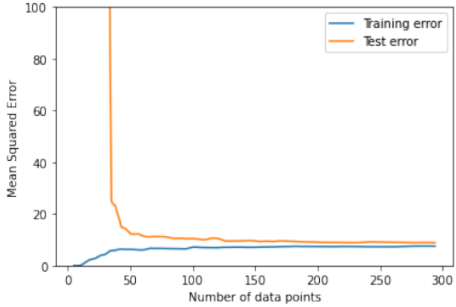


Extra. Bias and Variance

The model hypotheses are as below:

1. $H_w(x) = w_0 + w_1 x$
2. $H_w(x) = w_0 + w_1 x + w_2 x^2 + \cdots + w_{10} x^{10}$





Extra. Bias and Variance

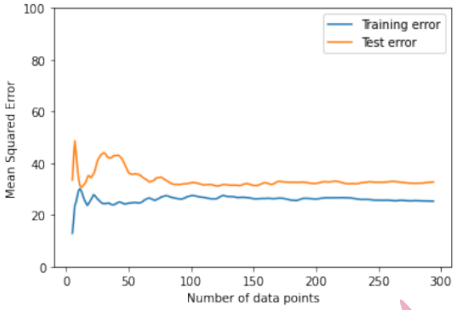
Main Idea:

- Both bias and variance contribute to the mean squared error.
- Bias does not decrease with the number of training samples. Bias causes error in **both** the training set and the test set.
- Variance decreases with the number of training samples. Variance causes a difference between the error in the training set and that in the test set.

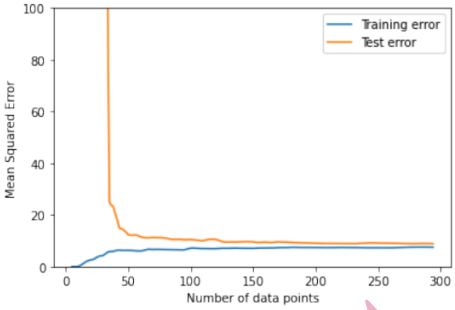
Q5. Bias and Variance

The model hypotheses are as below:

1. $H_w(x) = w_0 + w_1 x$ High bias
2. $H_w(x) = w_0 + w_1 x + w_2 x^2 + \cdots + w_{10} x^{10}$ High variance



High bias



High variance

Bonus. Gaussian Kernel

Extra Slide

Prove that the Gaussian Kernel, $K(\boldsymbol{u}, \boldsymbol{v}) = e^{-\frac{\|\boldsymbol{u}-\boldsymbol{v}\|^2}{2\sigma^2}}$, has infinite dimensional features.
(We assume $\sigma^2 = 1$ for simplicity.)

$$
\begin{aligned}
K(\boldsymbol{u}, \boldsymbol{v}) &= e^{-\frac{\|\boldsymbol{u}-\boldsymbol{v}\|^2}{2}} \\
&= e^{-\frac{\boldsymbol{u}\cdot\boldsymbol{u}-2\boldsymbol{u}\cdot\boldsymbol{v}+\boldsymbol{v}\cdot\boldsymbol{v}}{2}} \\
&= e^{-\frac{\boldsymbol{u}\cdot\boldsymbol{u}}{2}} e^{\boldsymbol{u}\cdot\boldsymbol{v}} e^{-\frac{\boldsymbol{v}\cdot\boldsymbol{v}}{2}} \\
&= e^{-\frac{\boldsymbol{u}\cdot\boldsymbol{u}}{2}} \left(1 + \boldsymbol{u}\cdot\boldsymbol{v} + \frac{1}{2!}(\boldsymbol{u}\cdot\boldsymbol{v})^2 + \frac{1}{3!}(\boldsymbol{u}\cdot\boldsymbol{v})^3 + \frac{1}{4!}(\boldsymbol{u}\cdot\boldsymbol{v})^4 + \cdots\right) e^{-\frac{\boldsymbol{v}\cdot\boldsymbol{v}}{2}} \\
&= e^{-\frac{\boldsymbol{u}\cdot\boldsymbol{u}}{2}} \left(\phi_0(\boldsymbol{u}) \cdot \phi_0(\boldsymbol{v})\right) e^{-\frac{\boldsymbol{v}\cdot\boldsymbol{v}}{2}} \\
&= \left(e^{-\frac{\boldsymbol{u}\cdot\boldsymbol{u}}{2}} \phi_0(\boldsymbol{u})\right) \cdot \left(e^{-\frac{\boldsymbol{v}\cdot\boldsymbol{v}}{2}} \phi_0(\boldsymbol{v})\right)
\end{aligned}
$$

$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \cdots$

Sum of polynomial kernels (in lecture)
⇒ Also a kernel (by bonus Q1)

scalars (multiplied to the transformed features)