Northeastern University
Khoury College of
Computer Sciences

Practical Issues and Feature Normalization

DS 4400 | Machine Learning and Data Mining I

Zohair Shafi

Spring 2026

Wednesday | January 14, 2026

Today's Outline

1. Recap
2. Practical Issues in Linear Regression
3. Feature Pre-processing and Normalization

Today's Outline

1. **Recap**
2. Practical Issues in Linear Regression
3. Feature Pre-processing and Normalization

Recap

Derivative of the Sigmoid Function

- Sigmoid: $\sigma(x) = \dfrac{1}{1 + e^{-x}}$

Let $f(x) = 1 + e^{-x}$ and $g(x) = \dfrac{1}{x} = x^{-1}$

$$\sigma(x) = g(f(x))$$

$$\sigma(x) = (1 + e^{-x})^{-1}$$

$$\sigma'(x) = -1 \cdot (1 + e^{-x})^{-2} \cdot -e^{-x}$$

$$\sigma'(x) = \frac{e^{-x}}{(1 + e^{-x})^2}$$

$$\sigma'(x) = \sigma(x) \cdot \frac{e^{-x}}{1 + e^{-x}}$$

$$\sigma'(x) = \sigma(x) \cdot (1 - \sigma(x))$$

Recap

Linear Regression Derivation

$$L(\theta) = \frac{1}{m}\sum_{i=1}^{m}[f_\theta(x_i) - y_i]^2$$

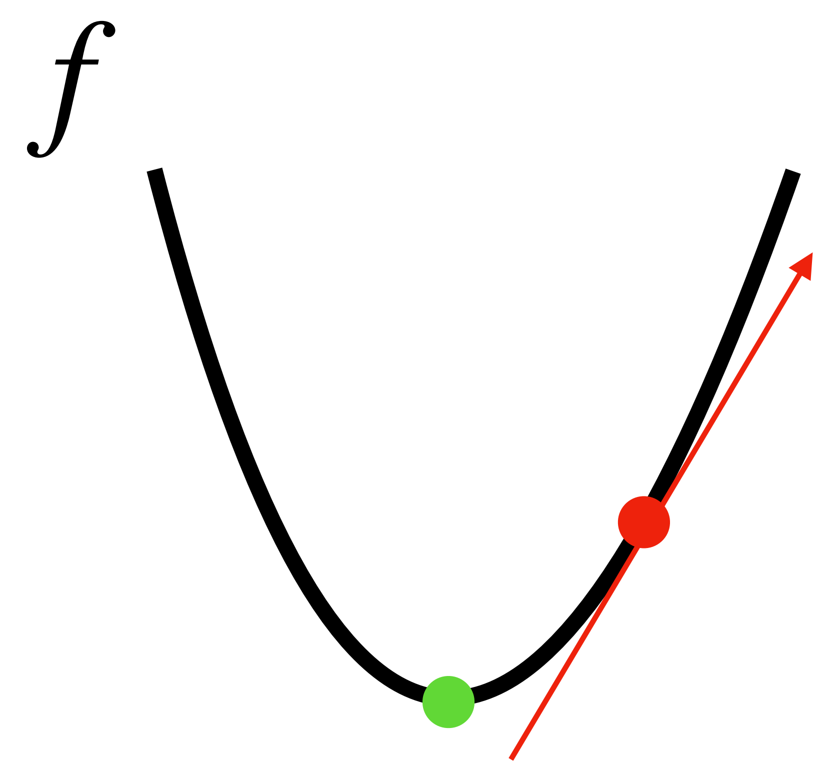
$$L(\theta) = \frac{1}{m}\sum_{i=1}^{m}[\theta_0 + \theta_1 \cdot x_i - y_i]^2$$

Find the point where $\nabla L(\theta) = 0$

$$\frac{\partial L(\theta)}{\partial \theta_0} = \frac{2}{m}\sum_{i=1}^{m}(\theta_0 + \theta_1 x_i - y_i) = 0$$

$$\frac{\partial L(\theta)}{\partial \theta_1} = \frac{2}{m}\sum_{i=1}^{m}x_i \cdot (\theta_0 + \theta_1 x_i - y_i) = 0$$

$\nabla f(\bullet)$ points in direction of steepest ascent



$\nabla f(\bullet) = 0$

Recap

Linear Regression Derivation

Find the point where $\nabla L(\theta) = 0$

$$\frac{\partial L(\theta)}{\partial \theta_0} = \frac{2}{m} \sum_{i=1}^{m} (\theta_0 + \theta_1 x_i - y_i) = 0$$

$$\frac{\partial L(\theta)}{\partial \theta_1} = \frac{2}{m} \sum_{i=1}^{m} x_i \cdot (\theta_0 + \theta_1 x_i - y_i) = 0$$

$$\theta_0 = \bar{y} - \theta_1 \bar{x}$$

$$\theta_1 = \frac{Cov(x, y)}{Var(x)}$$

Recap

Linear Regression Derivation - Matrix Form

$$L(\theta) = \frac{1}{m} \sum (Y - X\theta)^2$$

$$Y \in \mathbb{R}^{m \times 1}$$
$$X \in \mathbb{R}^{m \times n}$$
$$\theta \in \mathbb{R}^{n \times 1}$$

Recap

Linear Regression Derivation - Matrix Form

$$L(\theta) = \frac{1}{m}\sum(Y - X\theta)^2$$

$$Y \in \mathbb{R}^{m\times 1}$$
$$X \in \mathbb{R}^{m\times n}$$
$$\theta \in \mathbb{R}^{n\times 1}$$
$$X\theta \in \mathbb{R}^{m\times n} \cdot \mathbb{R}^{n\times 1} = \mathbb{R}^{m\times 1}$$

Recap

Linear Regression Derivation - Matrix Form

$$L(\theta) = \frac{1}{m} \sum (Y - X\theta)^2$$

$$Y \in \mathbb{R}^{m \times 1}$$
$$X \in \mathbb{R}^{m \times n}$$
$$\theta \in \mathbb{R}^{n \times 1}$$
$$X\theta \in \mathbb{R}^{m \times n} \cdot \mathbb{R}^{n \times 1} = \mathbb{R}^{m \times 1}$$

$$\vec{u} = \begin{bmatrix} 5 \\ 7 \\ 9 \end{bmatrix} \quad \vec{u}^2 = \begin{bmatrix} 25 \\ 49 \\ 81 \end{bmatrix}$$

Recap

Linear Regression Derivation - Matrix Form

$$L(\theta) = \frac{1}{m} \sum (Y - X\theta)^2$$

$$\vec{u} = \begin{bmatrix} 5 \\ 7 \\ 9 \end{bmatrix} \quad \vec{u}^2 = \begin{bmatrix} 25 \\ 49 \\ 81 \end{bmatrix}$$

$$u^T u = [5 \;\; 7 \;\; 9] \cdot \begin{bmatrix} 5 \\ 7 \\ 9 \end{bmatrix} = [5 \cdot 5 + 7 \cdot 7 + 9 \cdot 9] = [155]$$

Recap

Linear Regression Derivation - Matrix Form

$$L(\theta) = \frac{1}{m}\sum(Y - X\theta)^2$$

$$\vec{u} = \begin{bmatrix}5\\7\\9\end{bmatrix} \quad \vec{u}^2 = \begin{bmatrix}25\\49\\81\end{bmatrix}$$

$$u^T u = [5 \;\; 7 \;\; 9] \cdot \begin{bmatrix}5\\7\\9\end{bmatrix} = [5 \cdot 5 + 7 \cdot 7 + 9 \cdot 9] = [155]$$

$$u^T u = [u_1^2 + u_2^2 + u_3^2 + \cdots u_n^2] = \sum_i^n u_i^2$$

Recap

Linear Regression Derivation - Matrix Form

Each of these is a vector

$$L(\theta) = \frac{1}{m}\sum(Y - X\theta)^2$$

$$\vec{u} = \begin{bmatrix}5\\7\\9\end{bmatrix} \quad \vec{u}^2 = \begin{bmatrix}25\\49\\81\end{bmatrix}$$

$$u^T u = [5\ 7\ 9]\cdot\begin{bmatrix}5\\7\\9\end{bmatrix} = [5\cdot 5 + 7\cdot 7 + 9\cdot 9] = [155]$$

$$u^T u = [u_1^2 + u_2^2 + u_3^2 + \cdots u_n^2] = \sum_i^n u_i^2$$

Recap

Linear Regression Derivation - Matrix Form

This whole thing is then also a vector

$$L(\theta) = \frac{1}{m} \sum (Y - X\theta)^2$$

$$\vec{u} = \begin{bmatrix} 5 \\ 7 \\ 9 \end{bmatrix} \quad \vec{u}^2 = \begin{bmatrix} 25 \\ 49 \\ 81 \end{bmatrix}$$

$$u^T u = [5 \ 7 \ 9] \cdot \begin{bmatrix} 5 \\ 7 \\ 9 \end{bmatrix} = [5 \cdot 5 + 7 \cdot 7 + 9 \cdot 9] = [155]$$

$$u^T u = [u_1^2 + u_2^2 + u_3^2 + \cdots u_n^2] = \sum_i^n u_i^2$$

Recap

Linear Regression Derivation - Matrix Form

These two representations are now similar

$$L(\theta) = \frac{1}{m} \boxed{\sum (Y - X\theta)^2}$$

$$\vec{u} = \begin{bmatrix} 5 \\ 7 \\ 9 \end{bmatrix} \quad \vec{u}^2 = \begin{bmatrix} 25 \\ 49 \\ 81 \end{bmatrix}$$

$$u^T u = [5 \ 7 \ 9] \cdot \begin{bmatrix} 5 \\ 7 \\ 9 \end{bmatrix} = [5 \cdot 5 + 7 \cdot 7 + 9 \cdot 9] = [155]$$

$$u^T u = [u_1^2 + u_2^2 + u_3^2 + \cdots u_n^2] = \boxed{\sum_i^n u_i^2}$$

Recap

Linear Regression Derivation - Matrix Form

So we can replace this with $(Y - X\theta)^T(Y - X\theta)$

$$L(\theta) = \frac{1}{m}\boxed{\sum (Y - X\theta)^2}$$

$$\vec{u} = \begin{bmatrix} 5 \\ 7 \\ 9 \end{bmatrix} \quad \vec{u}^2 = \begin{bmatrix} 25 \\ 49 \\ 81 \end{bmatrix}$$

$$u^T u = [5 \;\; 7 \;\; 9] \cdot \begin{bmatrix} 5 \\ 7 \\ 9 \end{bmatrix} = [5 \cdot 5 + 7 \cdot 7 + 9 \cdot 9] = [155]$$

$$\boxed{u^T u} = [u_1^2 + u_2^2 + u_3^2 + \cdots u_n^2] = \sum_i^n u_i^2$$

Recap

Linear Regression Derivation - Matrix Form

$$L(\theta) = \frac{1}{m} \sum (Y - X\theta)^2$$

$$L(\theta) = (Y - X\theta)^T (Y - X\theta)$$

(why is this true?)

$$L(\theta) = (Y^T - \theta^T X^T)(Y - X\theta)$$

(Take the transpose inside. And then, because $(AB)^T = B^T A^T$)

$$L(\theta) = Y^T Y - Y^T X\theta - \theta^T X^T Y + \theta^T X^T X\theta$$

(the two terms in the centre are equivalent, why?)

$$L(\theta) = Y^T Y - 2\theta^T X^T Y + \theta^T X^T X\theta$$

Recap

Linear Regression Derivation - Matrix Form

$$L(\theta) = \frac{1}{m}\sum (Y - X\theta)^2$$

$$L(\theta) = (Y - X\theta)^T(Y - X\theta)$$

(why is this true?)

$$L(\theta) = (Y^T - \theta^T X^T)(Y - X\theta)$$

(Take the transpose inside. And then, because $(AB)^T = B^T A^T$)

$$L(\theta) = Y^T Y - Y^T X\theta - \theta^T X^T Y + \theta^T X^T X\theta$$

(the two terms in the centre are equivalent, why?)

$$L(\theta) = Y^T Y - 2\theta^T X^T Y + \theta^T X^T X\theta$$

Recap

Linear Regression Derivation - Matrix Form

$$Y^T X\theta \in \mathbb{R}^{1\times m} \cdot \mathbb{R}^{m\times n} \cdot \mathbb{R}^{n\times 1}$$

Which means

$$Y^T X\theta \in \mathbb{R}^{1\times 1}$$

Which means

$Y^T X\theta$ is **symmetric**

Which is why

$$Y^T X\theta = (Y^T X\theta)^T = \theta^T X^T Y$$

Recap

Linear Regression Derivation - Matrix Form

$$L(\theta) = \frac{1}{m} \sum (Y - X\theta)^2$$

$$L(\theta) = (Y - X\theta)^T (Y - X\theta)$$

(why is this true?)

$$L(\theta) = (Y^T - \theta^T X^T)(Y - X\theta)$$

(because $(AB)^T = B^T A^T$)

$$L(\theta) = Y^T Y - Y^T X\theta - \theta^T X^T Y + \theta^T X^T X\theta$$

(the two terms in the centre are equivalent, why?)

$$L(\theta) = Y^T Y - 2\theta^T X^T Y + \theta^T X^T X\theta$$

Recap

Matrix Form - Derivative

$$L(\theta) = Y^TY - 2\theta^TX^TY + \theta^TX^TX\theta$$

For any vector A

$$\nabla_A(B^TA) = B$$

For any symmetric matrix B

$$\nabla_A(A^TBA) = 2BA$$

So we have

$$\nabla L(\theta) = -2X^TY + 2X^TX\theta$$

Recap

Matrix Form - Derivative

$$L(\theta) = Y^T Y - 2\theta^T X^T Y + \theta^T X^T X\theta$$

For any vector A

$$\nabla_A (B^T A) = B$$

Recap

Matrix Form - Derivative

$$L(\theta) = Y^T Y - 2\theta^T X^T Y + \theta^T X^T X\theta$$

Lets look at $\theta^T X^T Y$

Recap

Matrix Form - Derivative

$$L(\theta) = Y^T Y - 2\theta^T X^T Y + \theta^T X^T X\theta$$

Lets look at $\theta^T X^T Y$

$X^T Y \in \mathbb{R}^{n\times 1}$ - This is a vector.

Recap

Matrix Form - Derivative

$$L(\theta) = Y^T Y - 2\theta^T X^T Y + \theta^T X^T X\theta$$

Lets look at $\theta^T X^T Y$

$\theta^T B$ where $B = X^T Y \in \mathbb{R}^{n \times 1}$

Recap

Matrix Form - Derivative

$$L(\theta) = Y^T Y - 2\theta^T X^T Y + \theta^T X^T X\theta$$

Lets look at $\theta^T X^T Y$

$\theta^T B$ where $B = X^T Y \in \mathbb{R}^{n\times 1}$

We know this is symmetric because the result is $\in \mathbb{R}^{1\times 1}$

Recap

Matrix Form - Derivative

$$L(\theta) = Y^T Y - 2\theta^T X^T Y + \theta^T X^T X\theta$$

Lets look at $\theta^T X^T Y$

$\theta^T B$ where $B = X^T Y \in \mathbb{R}^{n\times 1}$

$$\theta^T B = (\theta^T B)^T = B^T \theta$$

The derivative rule we had:
$\nabla_A (B^T A) = B$

Recap

Matrix Form - Derivative

$$L(\theta) = Y^T Y - 2\theta^T X^T Y + \theta^T X^T X\theta$$

Lets look at $\theta^T X^T Y$

$\theta^T B$ where $B = X^T Y \in \mathbb{R}^{n \times 1}$

$$\theta^T B = (\theta^T B)^T = B^T \theta$$

The derivative rule we had:

$$\nabla_\theta (B^T \theta) = B$$

Recap

Matrix Form - Derivative

$$L(\theta) = Y^TY - 2\theta^TX^TY + \theta^TX^TX\theta$$

For any vector A

$$\nabla_A(B^TA) = B$$

For any symmetric matrix B

$$\nabla_A(A^TBA) = 2BA$$

So we have

$$\nabla L(\theta) = -2X^TY + 2X^TX\theta$$

Recap

Matrix Form - Derivative

$$L(\theta) = Y^TY - 2\theta^TX^TY + \theta^TX^TX\theta$$

For any vector A

$$\nabla_A(B^TA) = B$$

For any symmetric matrix B

$$\nabla_A(A^TBA) = 2BA$$

So we have

$$\nabla L(\theta) = -2X^TY + 2X^TX\theta$$

Recap

Matrix Form - Derivative

Is $X^T X$ symmetric?

$(X^T X)^T = X^T \cdot (X^T)^T = X^T X$

$X^T X$ is **always** symmetric

$$L(\theta) = Y^T Y - 2\theta^T X^T Y + \theta^T X^T X\theta$$

For any vector A

$$\nabla_A (B^T A) = B$$

For any symmetric matrix B

$$\nabla_A (A^T B A) = 2BA$$

So we have

$$\nabla L(\theta) = -2X^T Y + 2X^T X\theta$$

Recap

Solution

We want to find the minimum so set gradient to zero

$$\nabla L(\theta) = -2X^TY + 2X^TX\theta = 0$$

$$2X^TX\theta = 2X^TY$$

$$X^TX\theta = X^TY$$

If X^TX is invertible, then

$$\theta = (X^TX)^{-1}X^TY$$

Practical Example

https://zohairshafi.github.io/pages/lectures/Lecture_2_Notebook.ipynb

Today’s Outline

1. Recap
2. **Practical Issues in Linear Regression**
3. Feature Pre-processing and Normalization

Train / Test Splits

- Generally data is split into a training dataset and a testing data
- Rough rule of thumb is that this is an 80-20 split

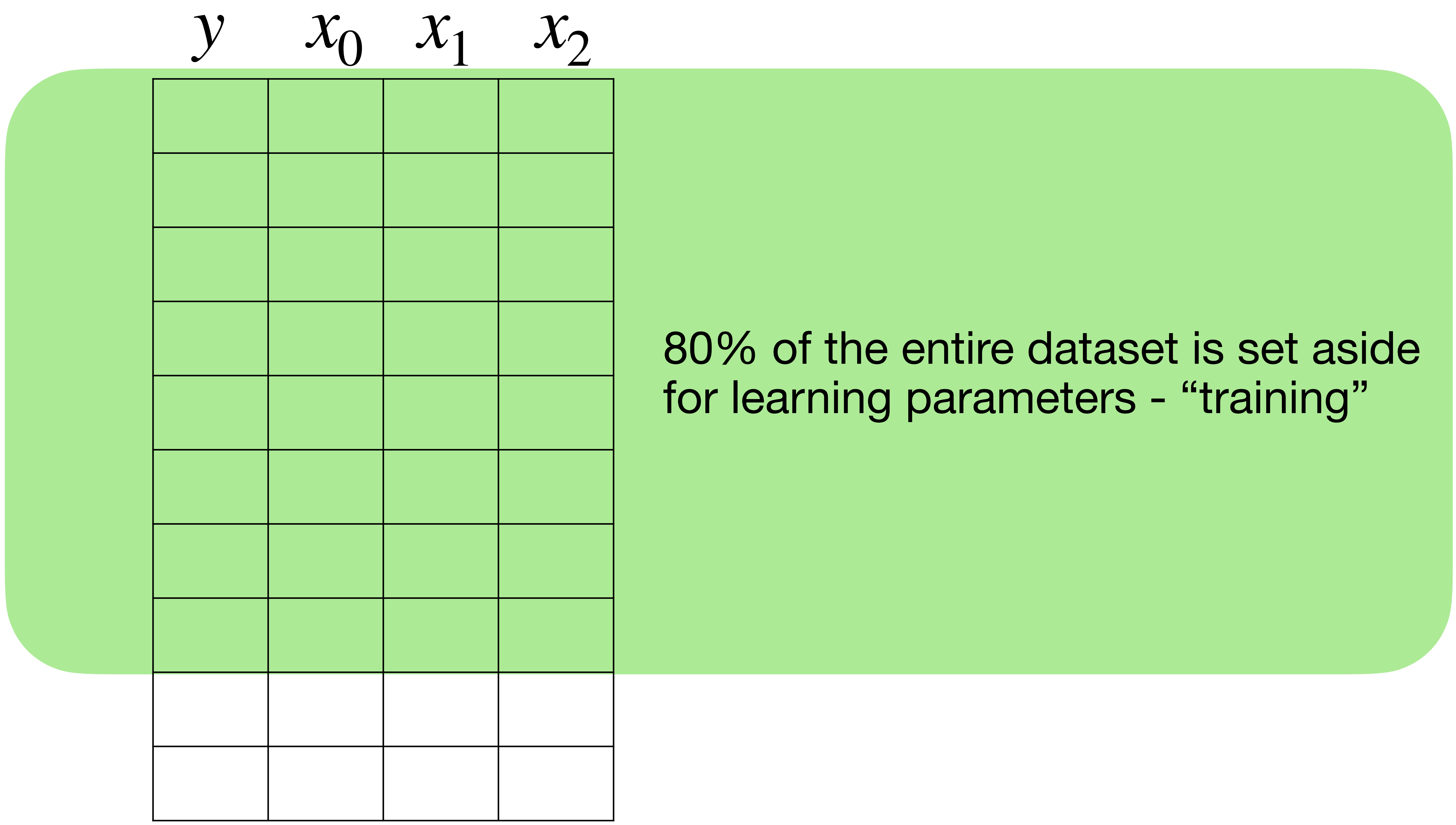
Train / Test Splits

- Generally data is split into a training dataset and a testing data
- Rough rule of thumb is that this is an 80-20 split

y	x_0	x_1	x_2

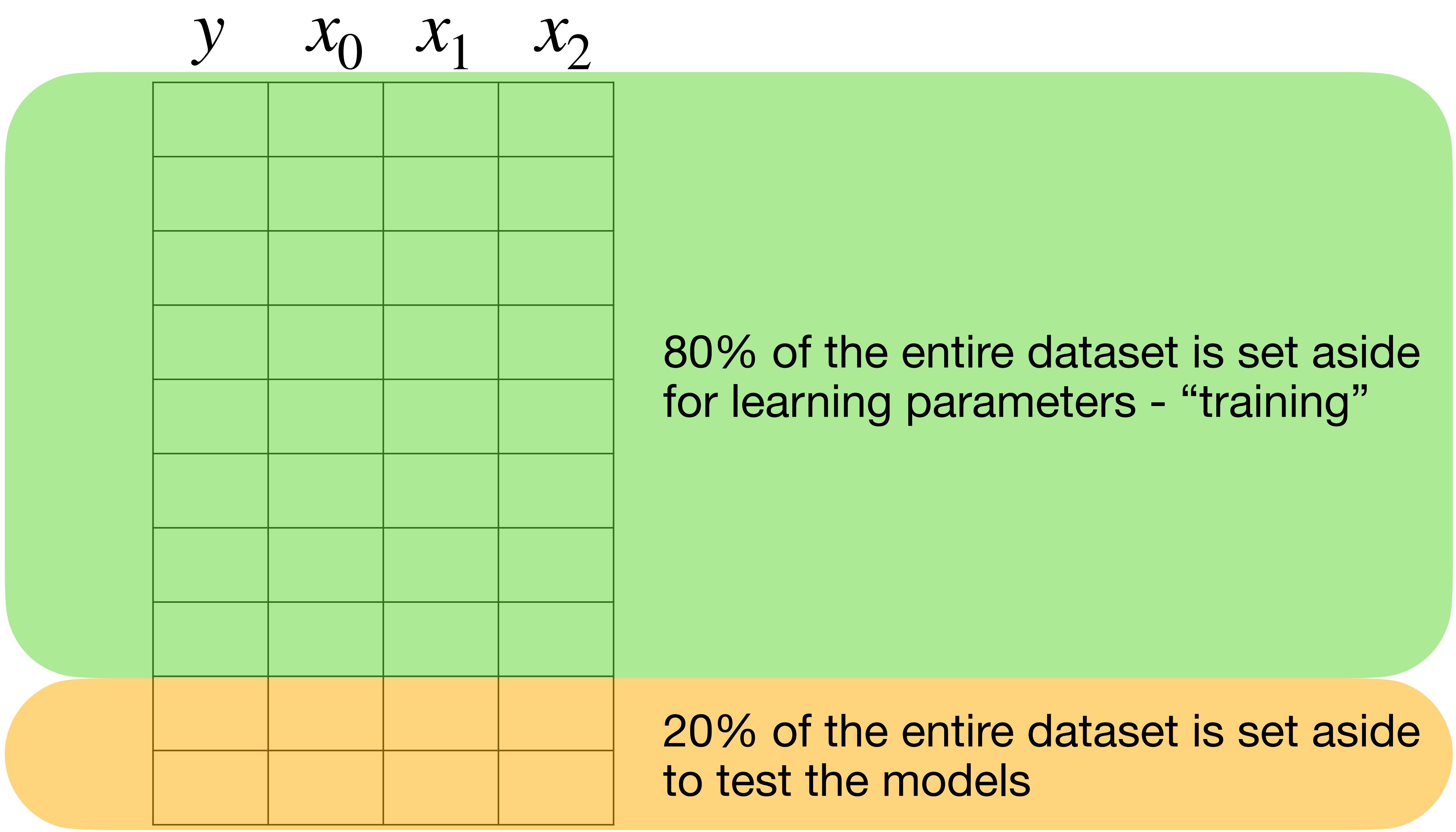
Train / Test Splits

- Generally data is split into a training dataset and a testing data
- Rough rule of thumb is that this is an 80-20 split



Train / Test Splits

- Generally data is split into a training dataset and a testing data
- Rough rule of thumb is that this is an 80-20 split



This is **unseen** data and tells you if the model can generalize well

Train / Test Splits

- However, in practice, if you are given only one train and test set, its easy to accidentally pick model architectures that work well on the test set, even though test set data is unseen
- To counter this, we use two unseen datasets - “validation” set and “test” set
- The split is generally of the form 80-10-10 where 80% is training data, 10% is validation data and 10% is test data

Practical Issues in Linear Regression

Multicollinearity

- When two features are highly correlated or are linearly dependent on each other

Practical Issues in Linear Regression

Multicollinearity

- When two features are highly correlated or are linearly dependent on each other
- Why it's a problem: $\theta = (X^T X)^{-1} X^T Y$
    - $X^T X$ becomes nearly singular (ill-conditioned)
    - Small changes in data cause huge changes in coefficients
    - Coefficients become unreliable and hard to interpret
    - Standard errors blow up

Practical Issues in Linear Regression

Multicollinearity

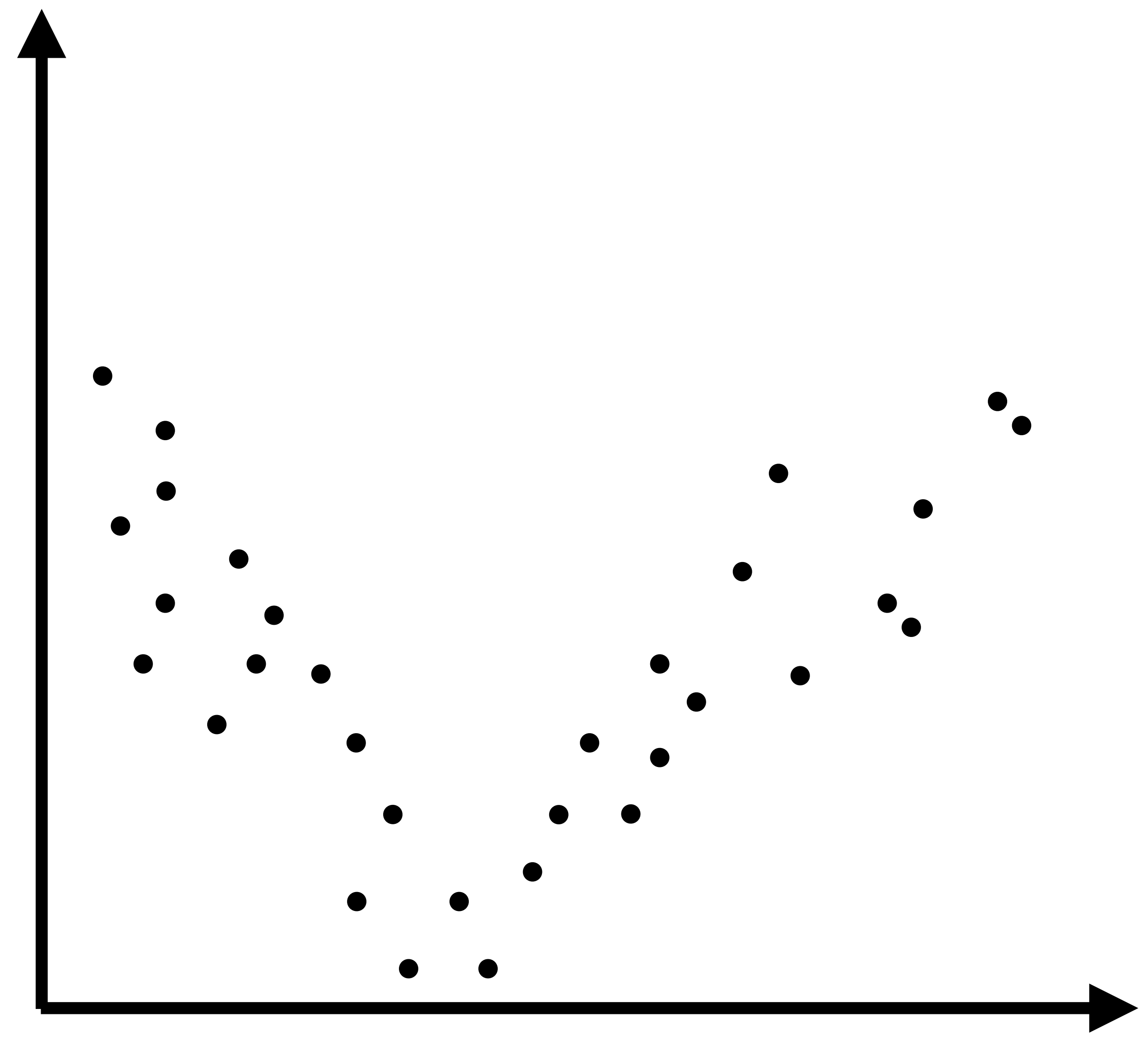
- When two features are highly correlated or are linearly dependent on each other
- Why it's a problem: $\theta = (X^T X)^{-1} X^T Y$

Simple Detection:
If correlation between features ≥ 0.8

  - $X^T X$ becomes nearly singular (ill-conditioned)
  - Small changes in data cause huge changes in coefficients
  - Coefficients become unreliable and hard to interpret
  - Standard errors blow up

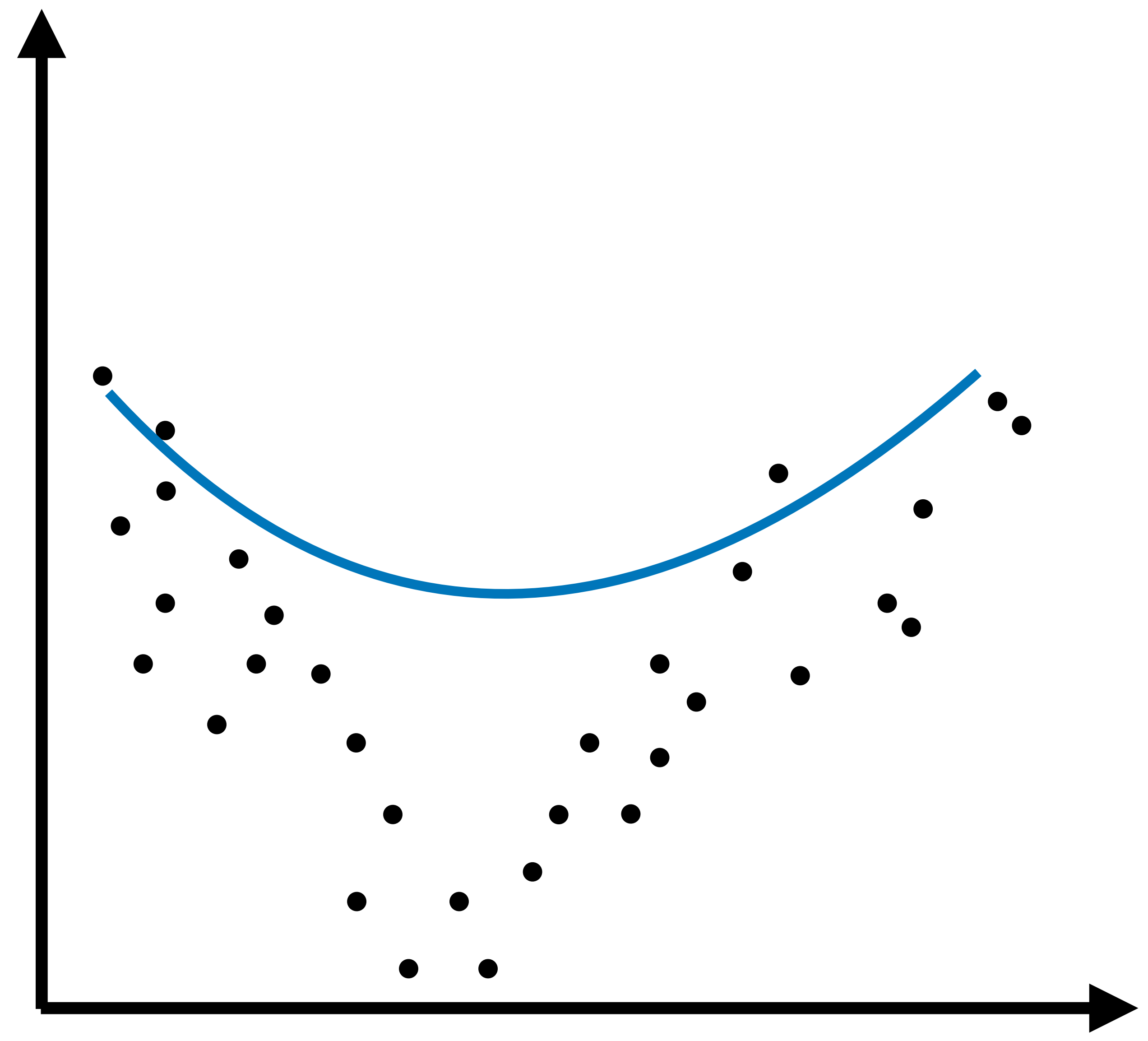
Practical Issues in Linear Regression

Overfitting vs Underfitting



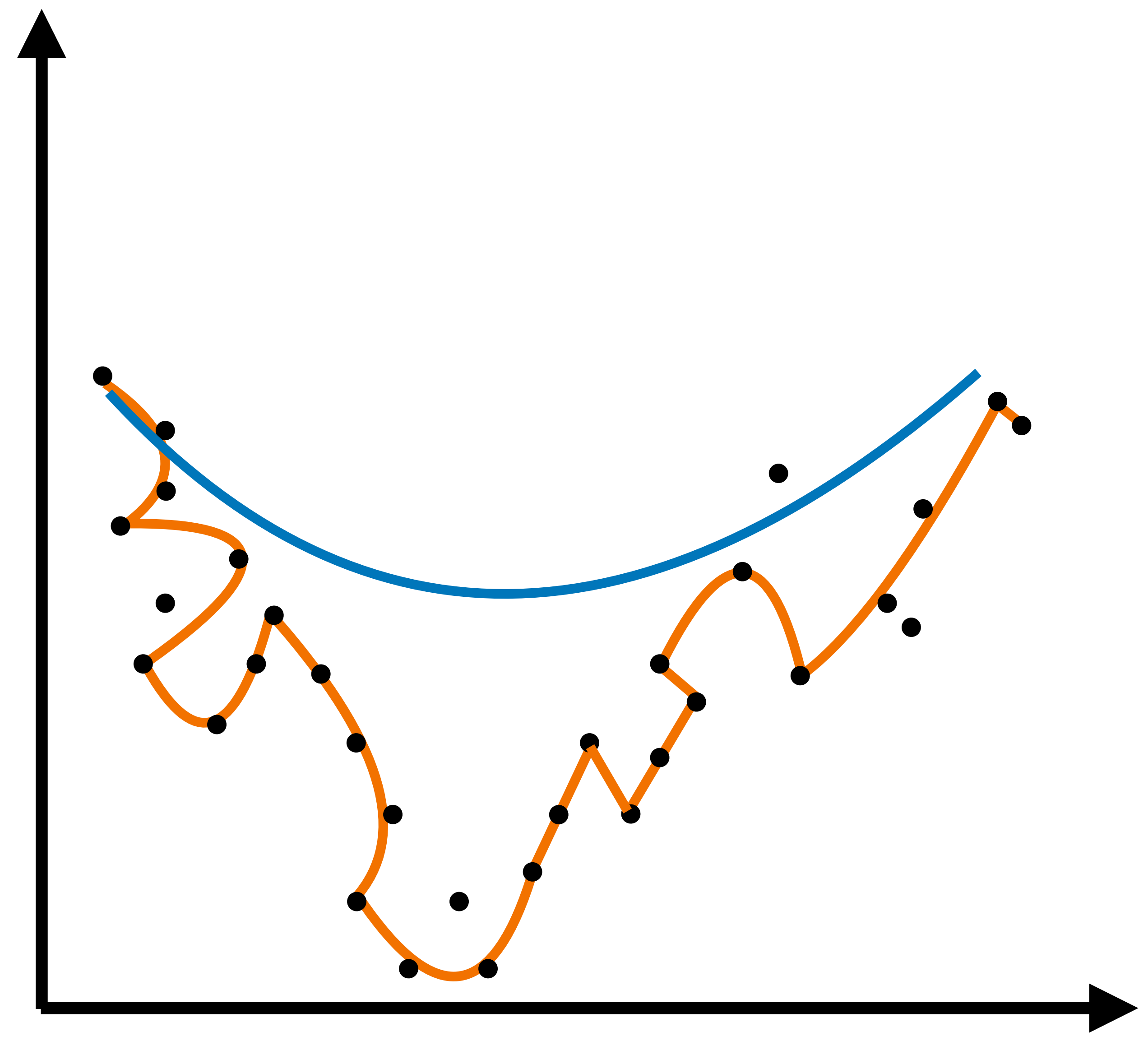
Practical Issues in Linear Regression

Overfitting vs Underfitting



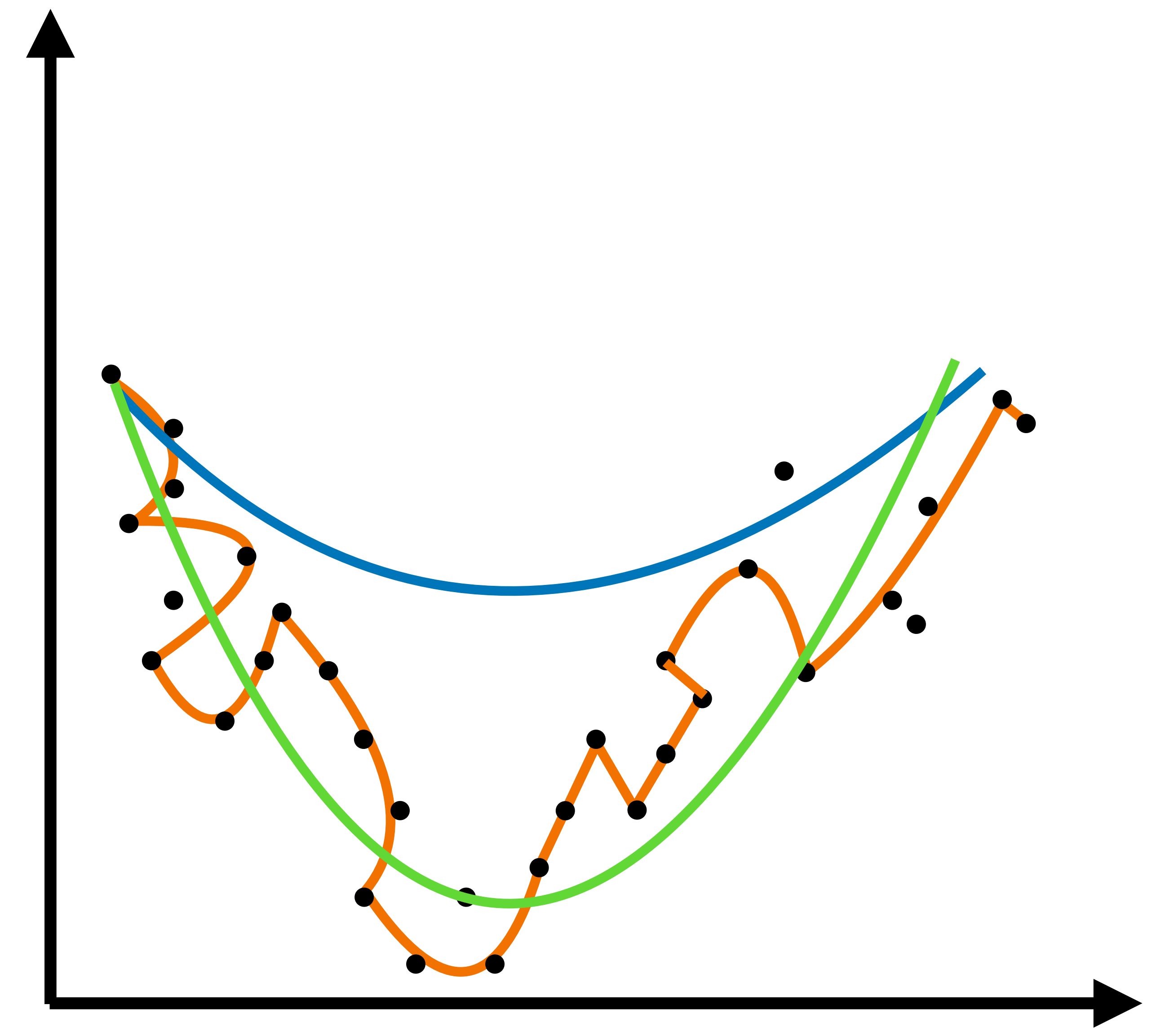
Practical Issues in Linear Regression

Overfitting vs Underfitting



Practical Issues in Linear Regression

Overfitting vs Underfitting



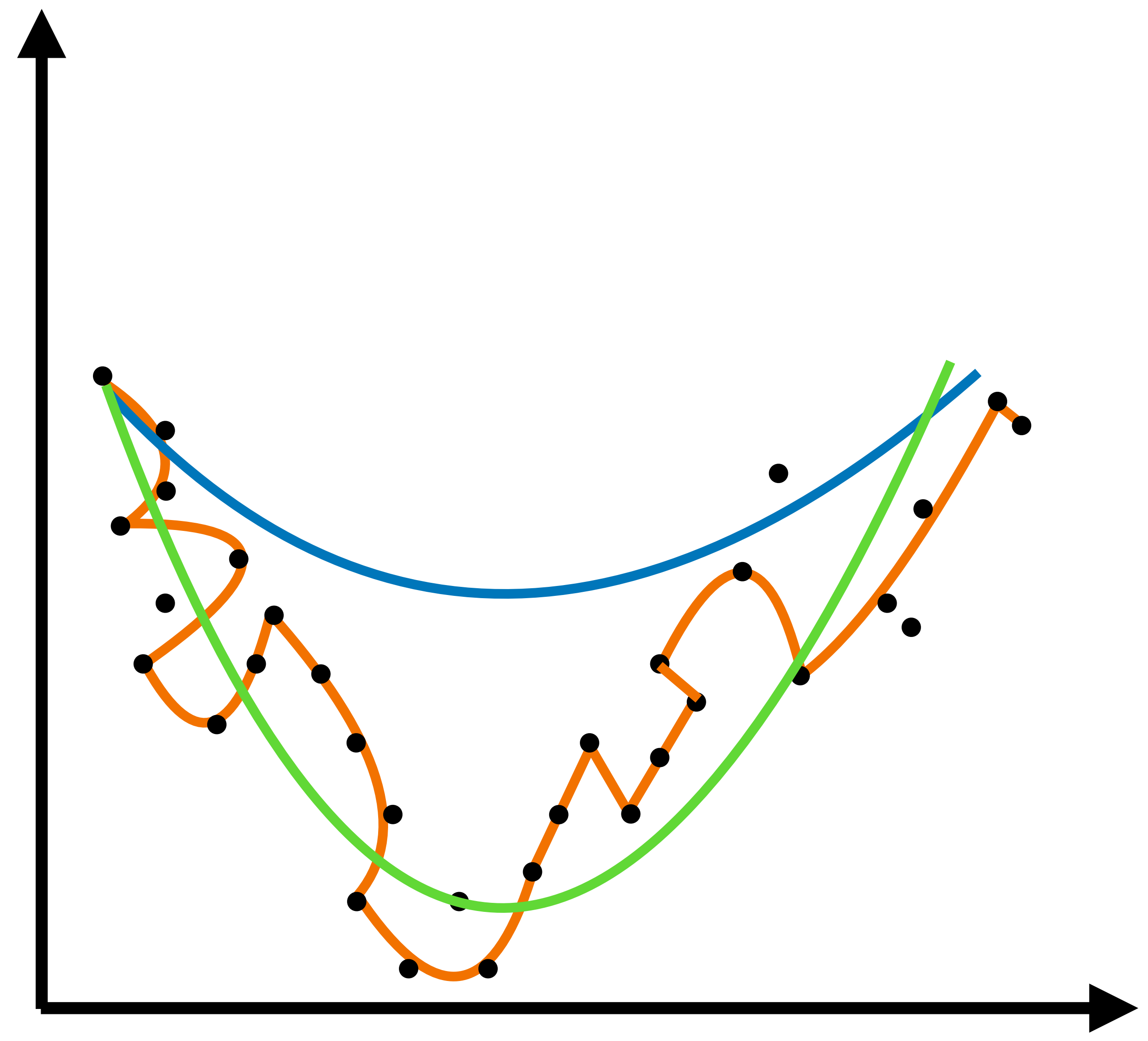
Practical Issues in Linear Regression

Overfitting vs Underfitting

The blue model is **underfitting** the data

The orange model is **overfitting** the data

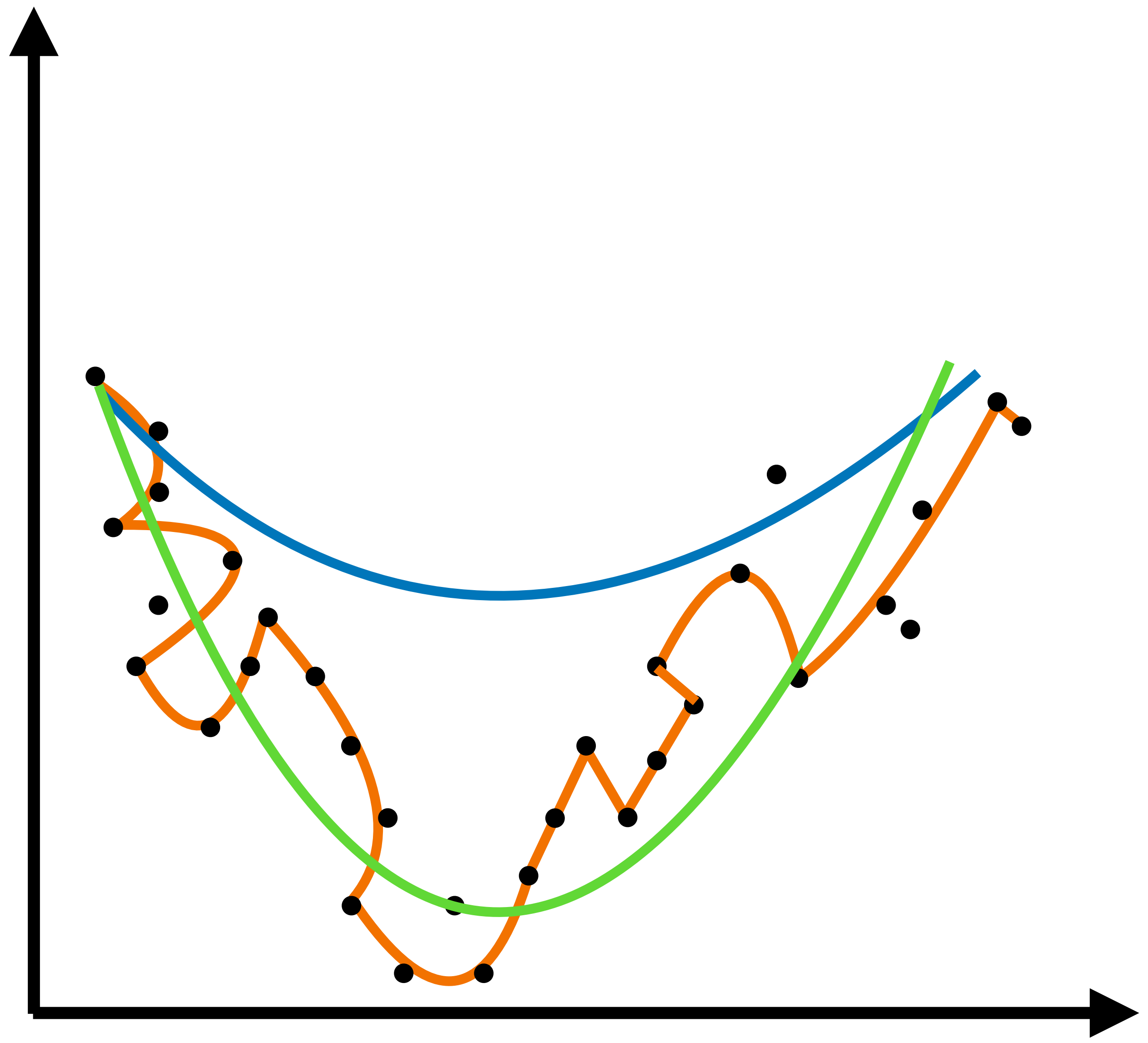
The green model is a good fit of the data



Practical Issues in Linear Regression

Underfitting

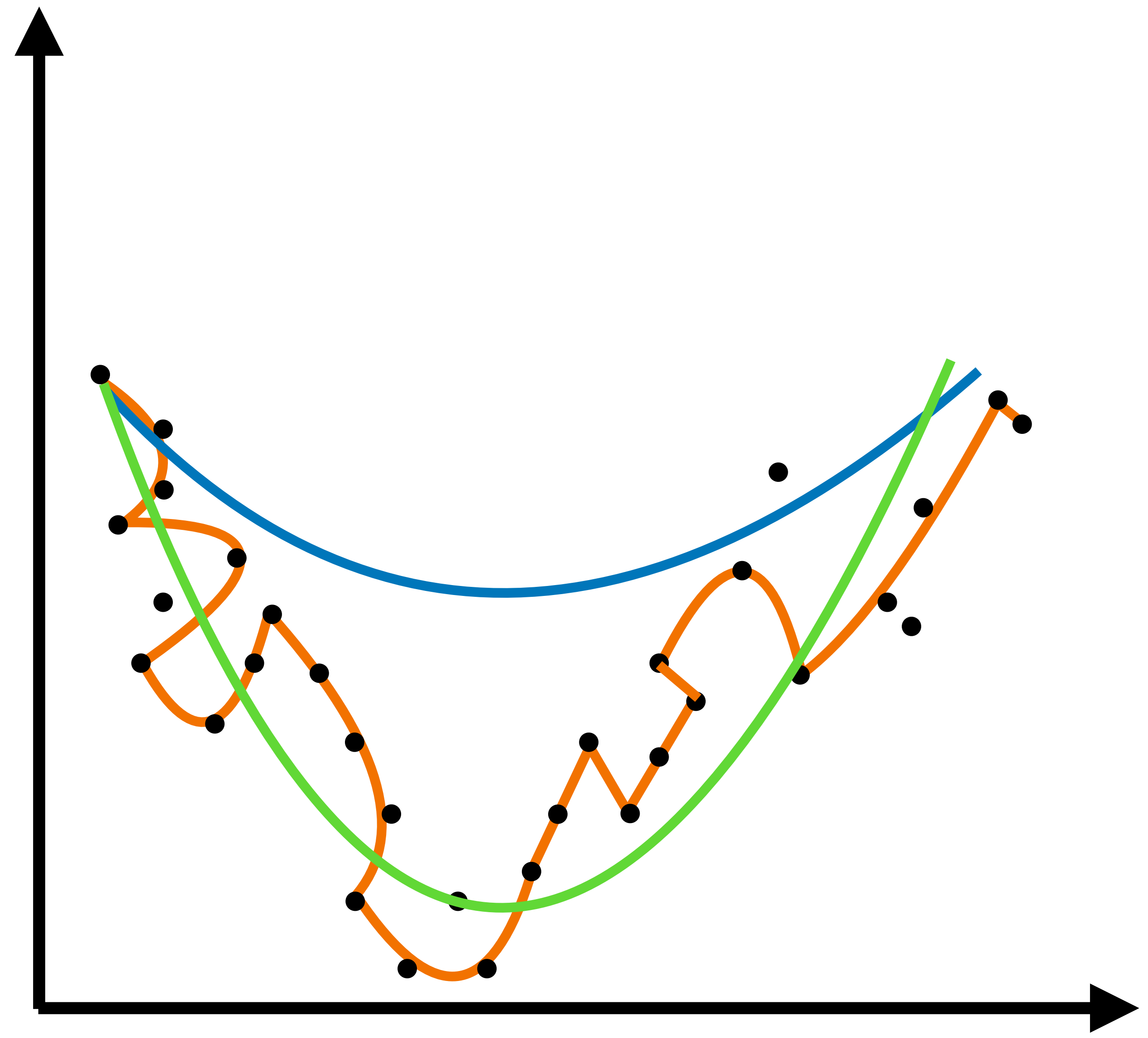
- What is happening?
  - The model is too simple to be able to capture the data
- How do you identify it?
  - Training loss is **high**
  - Test loss is **high**
- Solutions
  - Add more features
  - Add polynomial features $(x_1^2, x_2^2, x_1 x_2, \cdots)$
  - Use a more complex model



Practical Issues in Linear Regression

Overfitting

- What is happening?
  - The model is too complex, so it learns the noise distribution and outliers and hence does not generalize well to new data points
- How do you identify it?
  - Training loss is **low**
  - Test loss is **high**
  - Coefficients have **large** magnitudes
- Solutions
  - Regularization (L_1, L_2)
  - Cross-validation for model selection
  - Reduce number of features
  - Get more training data



Practical Issues in Linear Regression

A more mathematical look - Bias / Variance Tradeoff

Every model's prediction error/loss can be decomposed into three parts:

$$\text{Expected Loss} = \text{Bias}^2 + \text{Variance} + \text{Irreducible Noise}$$

Practical Issues in Linear Regression

A more mathematical look - Bias / Variance Tradeoff

Every model's prediction error/loss can be decomposed into three parts:

$$\text{Expected Loss} = \text{Bias}^2 + \text{Variance} + \text{Irreducible Noise}$$

Error from wrong assumptions due to the model being too simple

Practical Issues in Linear Regression

A more mathematical look - Bias / Variance Tradeoff

Every model's prediction error/loss can be decomposed into three parts:

$$\text{Expected Loss} = \text{Bias}^2 + \text{Variance} + \text{Irreducible Noise}$$

Error from high sensitivity to each data point and noise due to the model being too complex

Practical Issues in Linear Regression

A more mathematical look - Bias / Variance Tradeoff

Every model's prediction error/loss can be decomposed into three parts:

$$\text{Expected Loss} = \text{Bias}^2 + \text{Variance} + \text{Irreducible Noise}$$

Inherent randomness in data. Cannot be removed.

Practical Issues in Linear Regression

A more mathematical look - Bias / Variance Tradeoff

Every model's prediction error/loss can be decomposed into three parts:

Expected Loss = Bias^2 + Variance + Irreducible Noise

$$\mathbb{E}[(Y-\hat{Y})^2] == (\mathbb{E}[\hat{Y}]-Y)^2 + \mathbb{E}[(\hat{Y}-\mathbb{E}[\hat{Y}])^2] + \sigma^2$$

Practical Issues in Linear Regression

A more mathematical look - Bias / Variance Tradeoff

Every model's prediction error/loss can be decomposed into three parts:

Expected Loss = Bias^2 + Variance + Irreducible Noise

$$\mathbb{E}[(Y-\hat{Y})^2] = = (\mathbb{E}[\hat{Y}] - Y)^2 + \mathbb{E}[(\hat{Y} - \mathbb{E}[\hat{Y}])^2] + \sigma^2$$

How far is the average prediction from the true labels?

Practical Issues in Linear Regression

A more mathematical look - Bias / Variance Tradeoff

Every model's prediction error/loss can be decomposed into three parts:

Expected Loss = Bias² + Variance + Irreducible Noise

$$\mathbb{E}[(Y - \hat{Y})^2] = = (\mathbb{E}[\hat{Y}] - Y)^2 + \mathbb{E}[(\hat{Y} - \mathbb{E}[\hat{Y}])^2] + \sigma^2$$

If we use different training datasets, how much does $\hat{Y}$ vary?

Practical Issues in Linear Regression

A more mathematical look - Bias / Variance Tradeoff

Every model's prediction error/loss can be decomposed into three parts:

Expected Loss = Bias^2 + Variance + Irreducible Noise

$$\mathbb{E}[(Y - \hat{Y})^2] == (\mathbb{E}[\hat{Y}] - Y)^2 + \mathbb{E}[(\hat{Y} - \mathbb{E}[\hat{Y}])^2] + \sigma^2$$

This is not the Sigmoid function. This is just irreducible noise in the true data Y

Practical Issues in Linear Regression

Bias / Variance Tradeoff

Why is it called a **tradeoff**?

Practical Issues in Linear Regression

Bias / Variance Tradeoff

Why is it called a **tradeoff**?

Model Complexity	**Bias**	**Variance**	**Train Error**	**Test Error**
Too Simple	High	Low	High	High

Practical Issues in Linear Regression

Bias / Variance Tradeoff

Why is it called a **tradeoff**?

Model Complexity	Bias	Variance	Train Error	Test Error
Too Simple	High	Low	High	High
Too Complex	Low	High	Low	High

Practical Issues in Linear Regression

Bias / Variance Tradeoff

Why is it called a **tradeoff**?

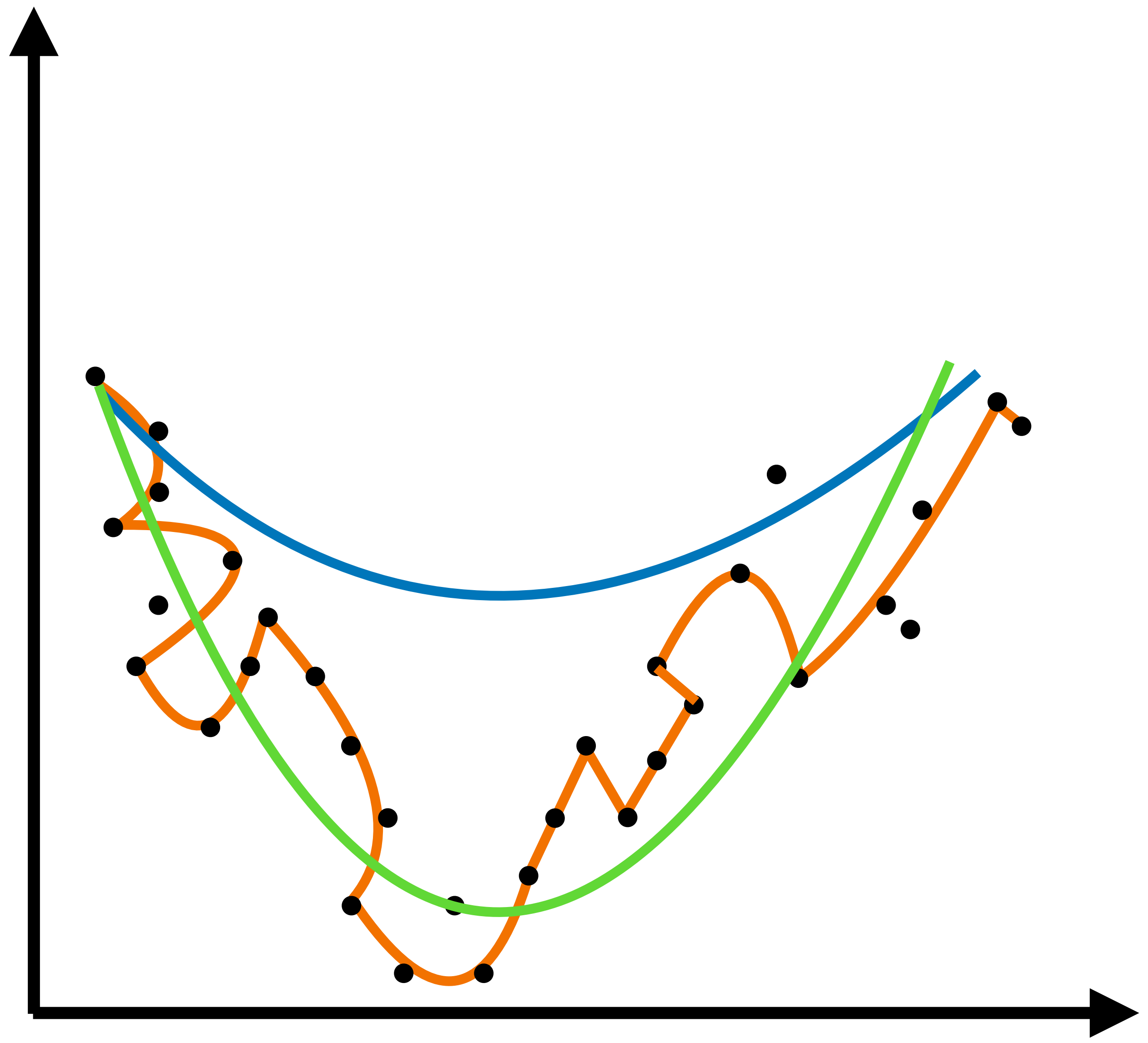
Model Complexity	Bias	Variance	Train Error	Test Error
Too Simple	High	Low	High	High
Sweet Spot	Medium	Medium	Medium	Medium
Too Complex	Low	High	Low	High

Practical Issues in Linear Regression

Regularization

- Regularization explicitly trades bias for variance.

$$L(\theta) = \frac{1}{m} \sum (Y - X\theta)^2$$



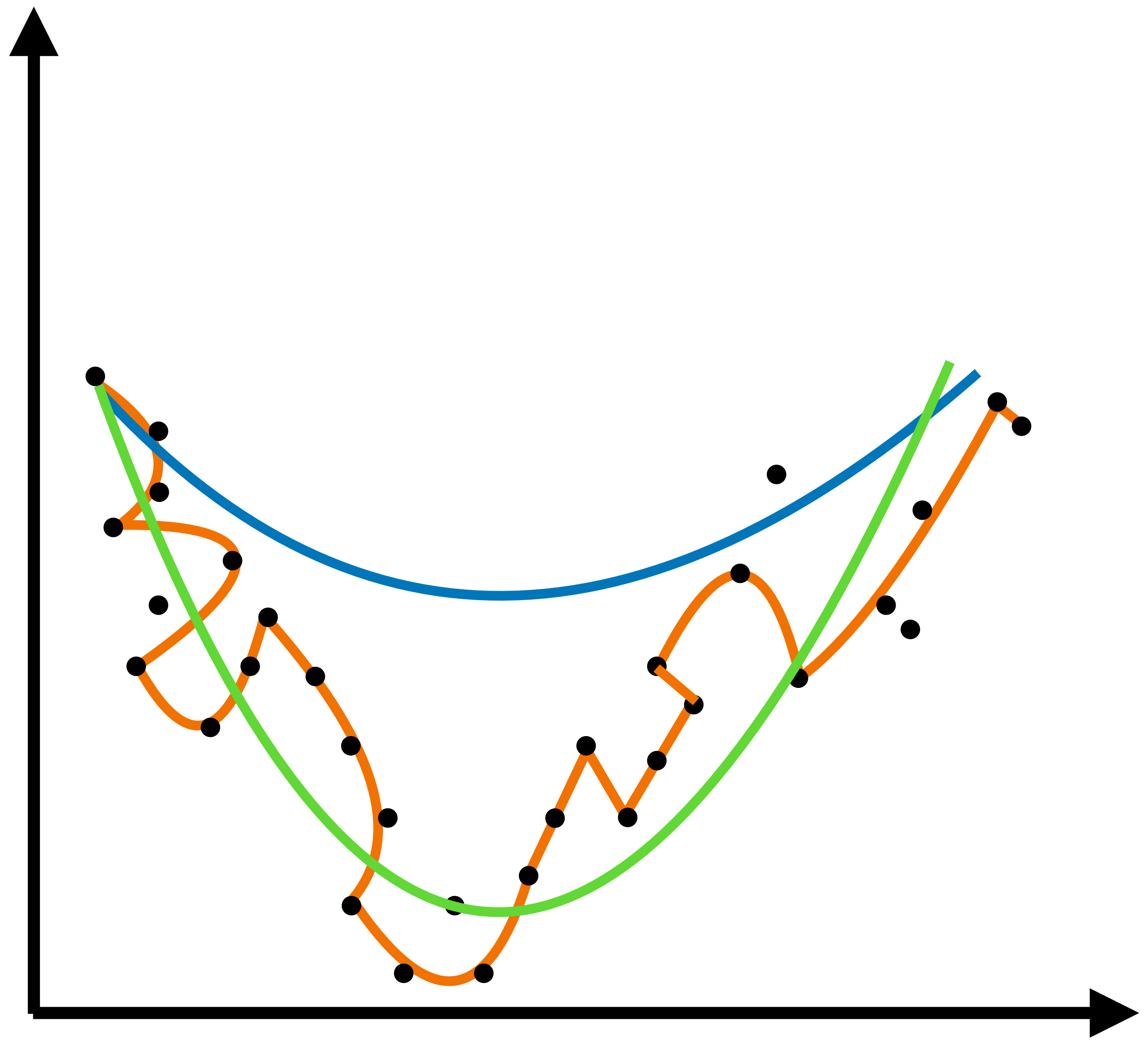
Practical Issues in Linear Regression

Regularization

- Regularization explicitly trades bias for variance.

$$L(\theta) = \frac{1}{m}\sum (Y - X\theta)^2$$

$$L(\theta) = \frac{1}{m}\sum (Y - X\theta)^2 + \lambda ||\theta||^2$$



Practical Issues in Linear Regression

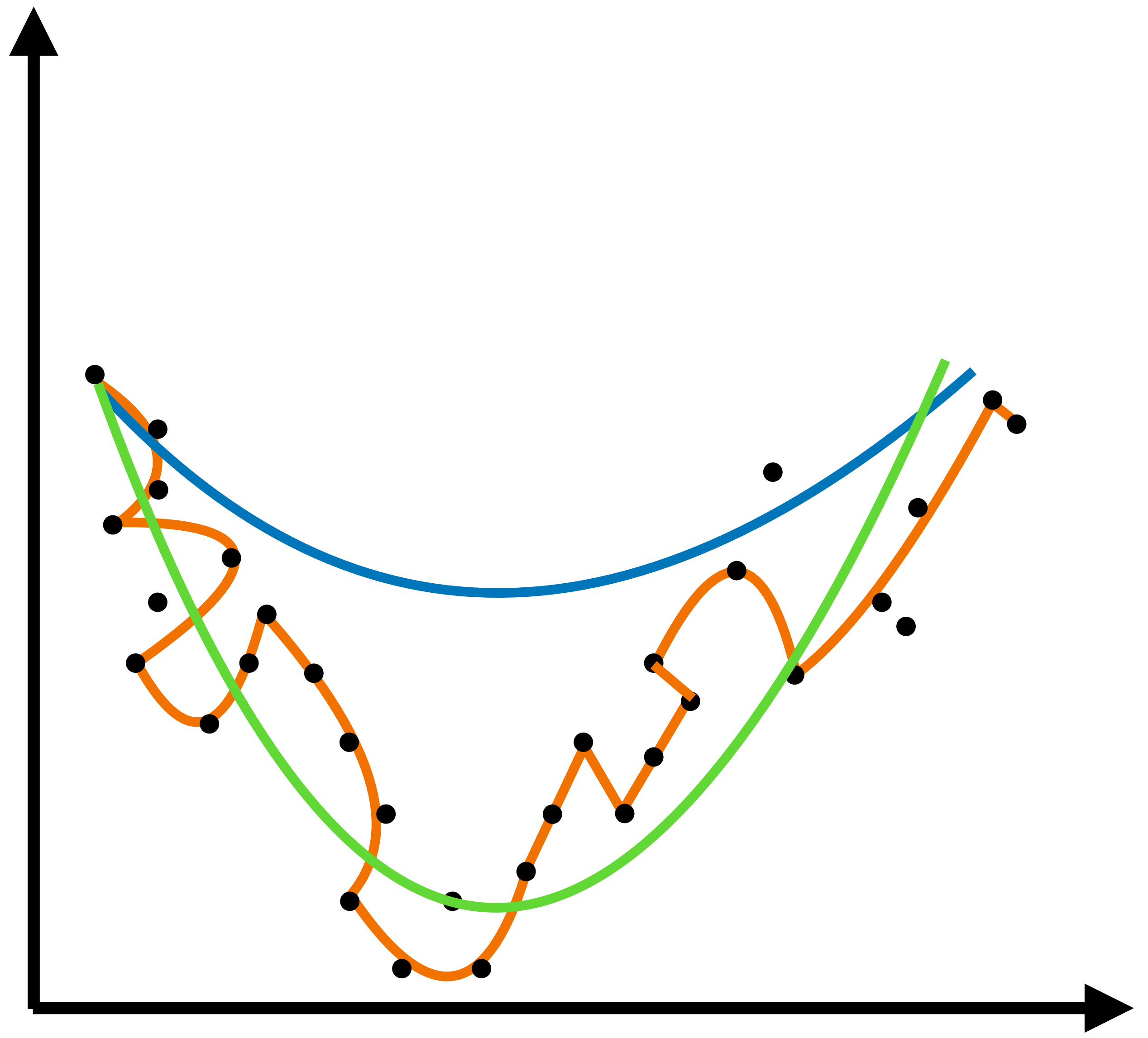
Regularization

- Regularization explicitly trades bias for variance.

$$L(\theta) = \frac{1}{m}\sum (Y - X\theta)^2$$

$$L(\theta) = \frac{1}{m}\sum (Y - X\theta)^2 + \lambda||\theta||^2$$

$$\theta = (X^TX + \lambda I)^{-1}X^TY$$



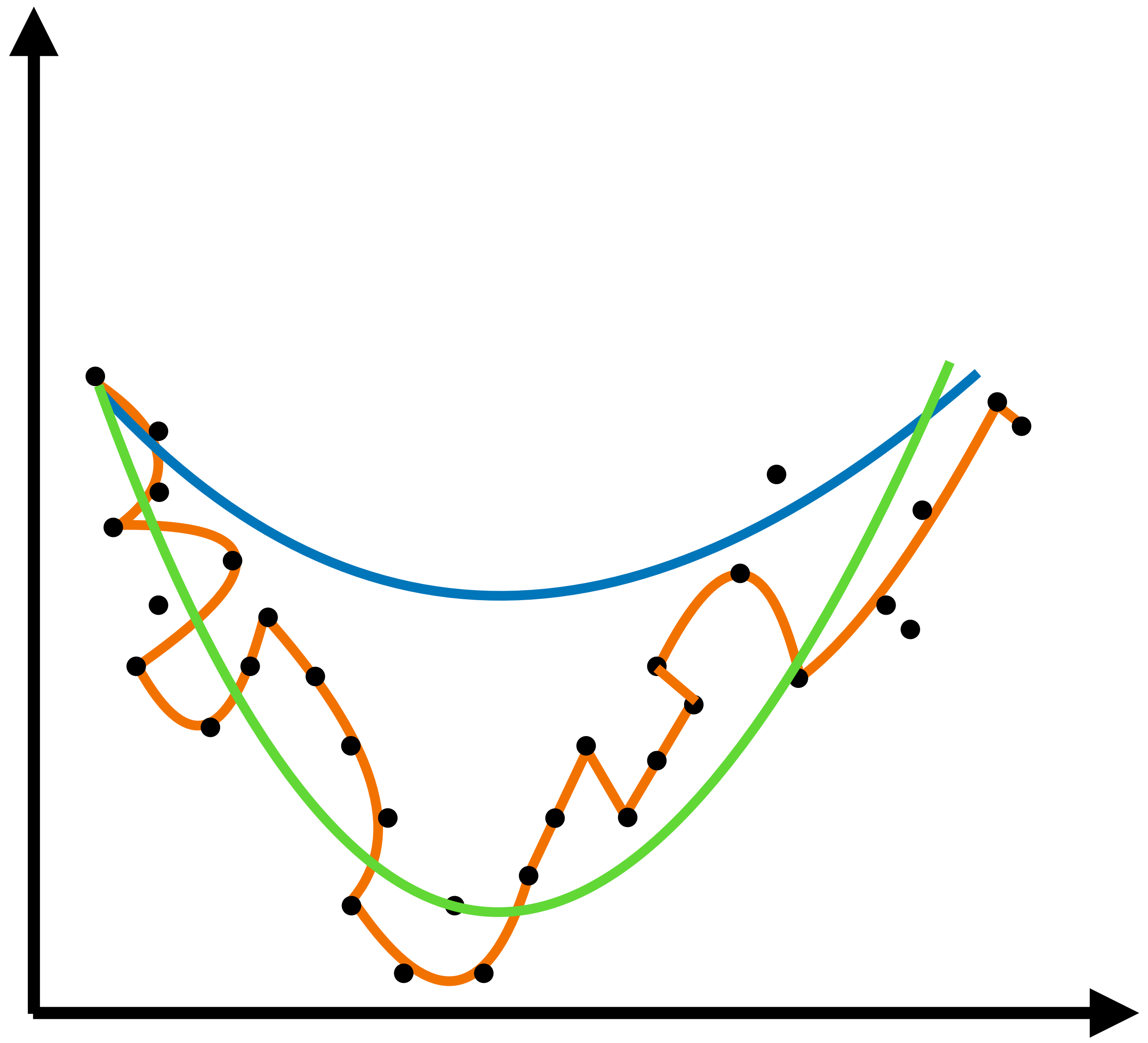
Practical Issues in Linear Regression

Regularization

- Regularization explicitly trades bias for variance.

$$L(\theta) = \frac{1}{m}\sum (Y - X\theta)^2 + \lambda\|\theta\|^2$$

$$\theta = (X^T X + \lambda I)^{-1} X^T Y$$



- As λ increases:
  - Coefficients shrink toward zero
  - Bias increases (we're constraining the model)
  - Variance decreases (less sensitive to data)
  - At some $\lambda*$, test error is minimized

Practical Issues in Linear Regression

Regularization

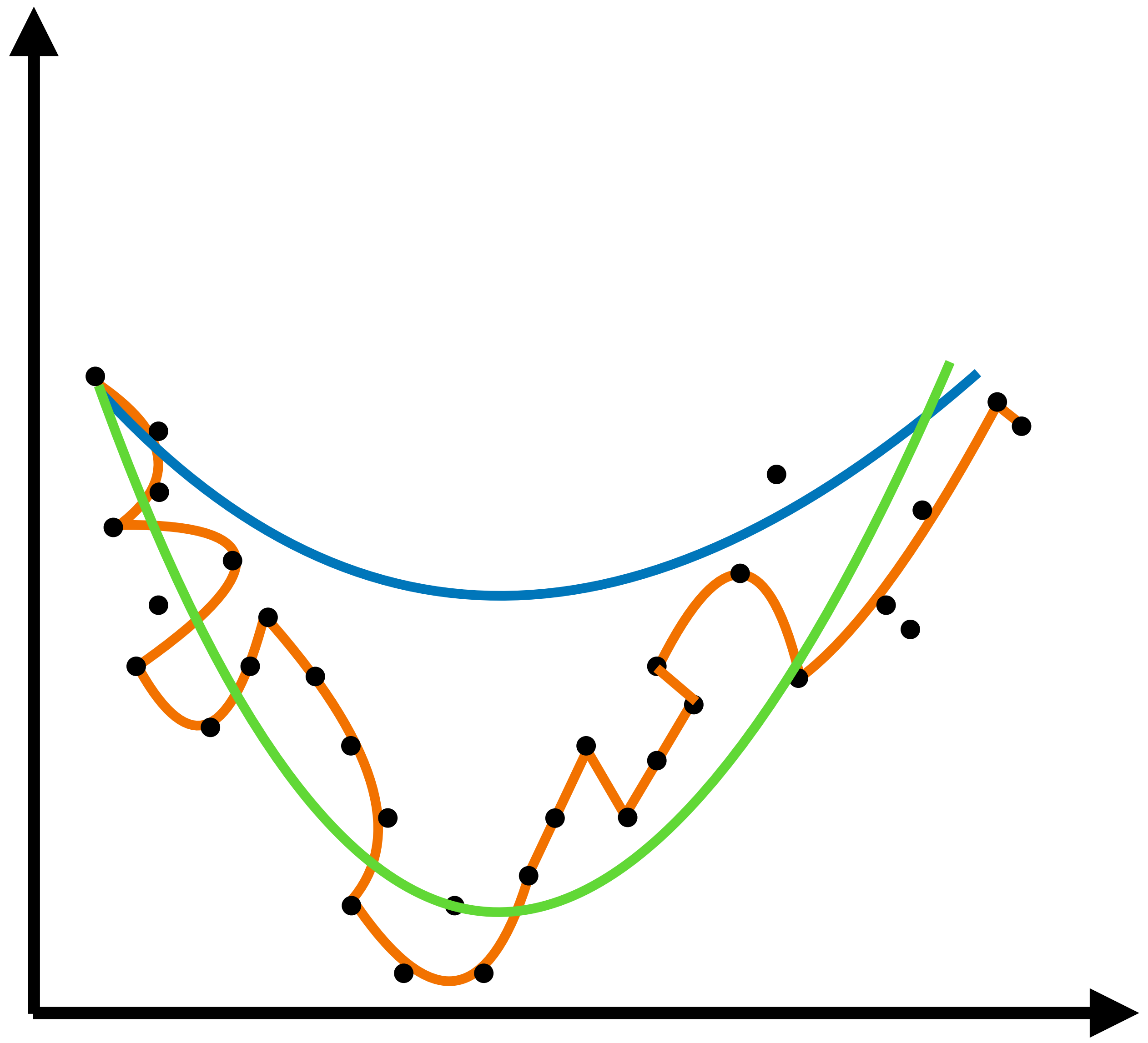
- Regularization explicitly trades bias for variance.

$$L(\theta) = \frac{1}{m}\sum (Y - X\theta)^2 + \lambda||\theta||^2$$

$$\theta = (X^T X + \lambda I)^{-1} X^T Y$$

These sort of parameters are usually called **hyper-parameters**

They are **not learnable** but are human defined

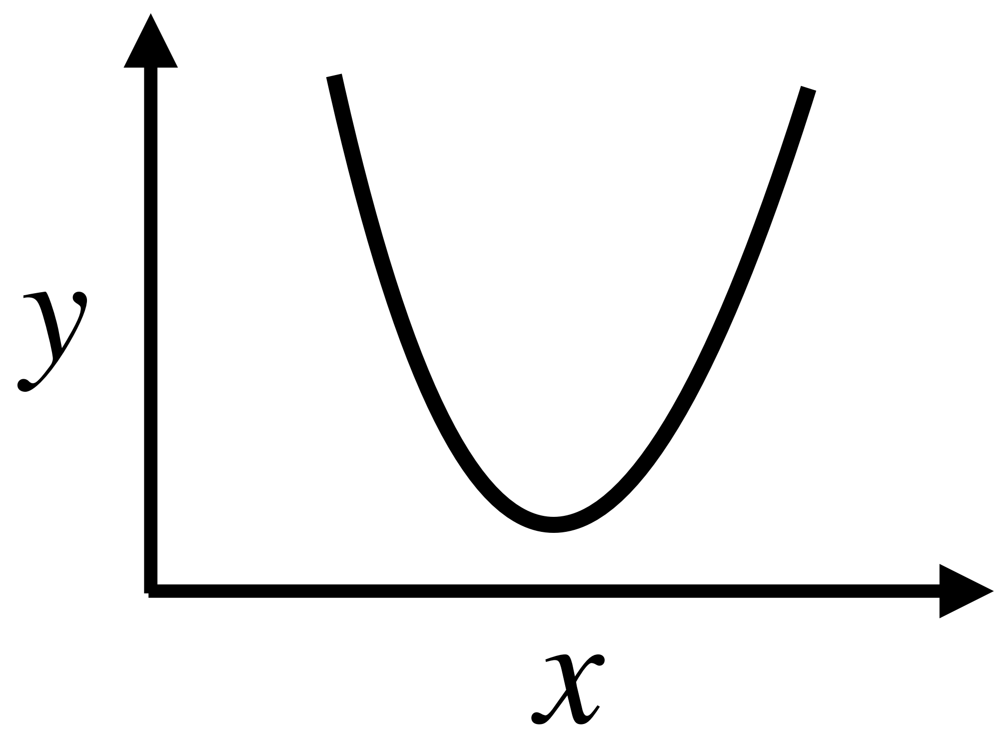


- As λ increases:
    - Coefficients shrink toward zero
    - Bias increases (we're constraining the model)
    - Variance decreases (less sensitive to data)
    - At some $\lambda *$, test error is minimized

Practical Issues in Linear Regression

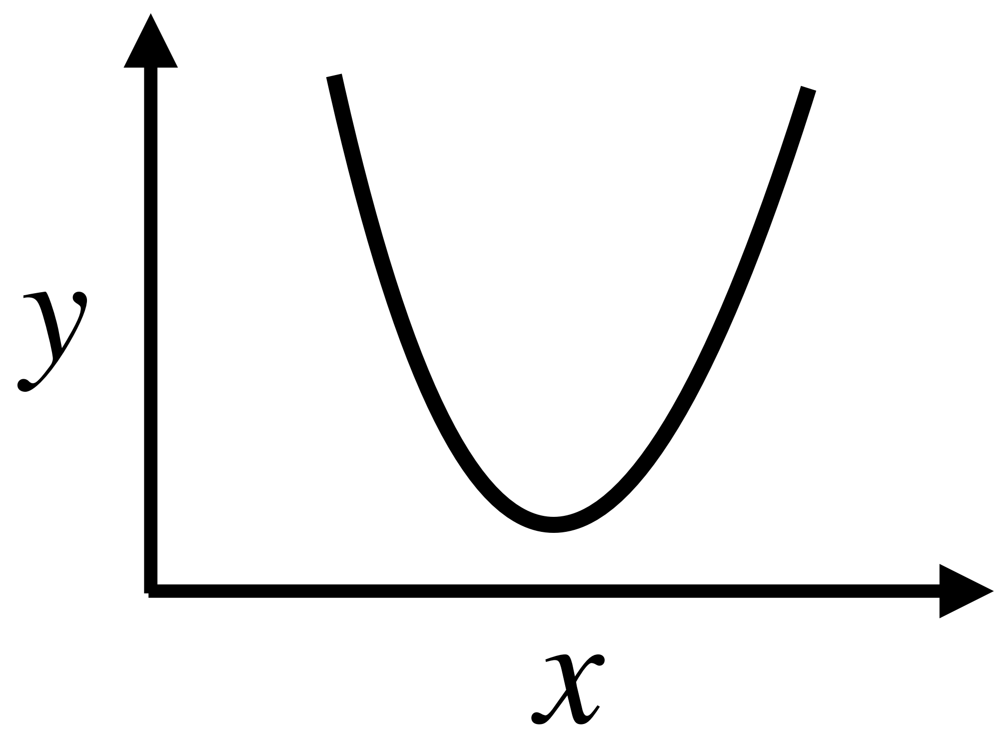
Non-Linearity

- True relationship between x and y is not linear



Practical Issues in Linear Regression

Non-Linearity



- True relationship between x and y is not linear
- Solutions:
    - Add polynomial terms like x^2, x^3, etc..
    - Add interaction terms like $x_1 \cdot x_2$
    - Transform input features like $log(x), \sqrt{x}$
    - Use a non-linear model

Today’s Outline

1. Recap
2. Practical Issues in Linear Regression
3. Feature Pre-processing and Normalization

Today's Outline

1. Recap
2. Practical Issues in Linear Regression
3. **Feature Pre-processing and Normalization**

Feature Normalization

Why Normalize?

- If feature x_1 ranges from 0 to 1 and feature x_2 ranges from 0 to 1,000,000, this could lead to numerical instability in the solving process
  - This is particular relevant to gradient descent

Feature Normalization

Why Normalize?

- If feature x_1 ranges from 0 to 1 and feature x_2 ranges from 0 to 1,000,000, this could lead to numerical instability in the solving process
  - This is particular relevant to gradient descent
- Regularization unfairness
  - If x_2 is much larger, θ_2 must be much smaller to produce similar predictions.
  - The regularization penalty then affects features unequally based on arbitrary scale choices.

Feature Normalization

Why Normalize?

- If feature x_1 ranges from 0 to 1 and feature x_2 ranges from 0 to 1,000,000, this could lead to numerical instability in the solving process
    - This is particular relevant to gradient descent
- Regularization unfairness
    - If x_2 is much larger, θ_2 must be much smaller to produce similar predictions.
    - The regularization penalty then affects features unequally based on arbitrary scale choices.
- Distance-based algorithms

Feature Normalization
Categorical vs Continuous Features

- Predict credit card balance
  - Age
  - Income
  - Number of cards
  - Credit limit
  - Credit rating
- Categorical variables
  - Student (Yes/No)
  - State (50 different states)

Feature Normalization

Indicator Variables and One-Hot Encoding

- For a variable like “Student” that takes True/False values:
  - We can simply replace with 0/1

Feature Normalization

Indicator Variables and One-Hot Encoding

- For a variable like “Student” that takes True/False values:
  - We can simply replace with 0/1
- For a variable like “State” which in the US can take 50 values, we use something called One-Hot encoding
  - $state = [x_{NY}, x_{MA}, x_{NJ}, x_{WA}, x_{CA}, \cdots x_{RI}]$
  - If the particular data point is from MA, that element of the vector is set to 1 and everything else 0
    - $state = [0,1,0,0,0,\cdots 0]$

Feature Normalization

Indicator Variables and One-Hot Encoding

- For a variable like “Student” that takes True/False values:
  - We can simply replace with 0/1
- For a variable like “State” which in the US can take 50 values, we use something called One-Hot encoding
  - $state = [x_{NY}, x_{MA}, x_{NJ}, x_{WA}, x_{CA}, \cdots x_{RI}]$
  - If the particular data point is from MA, that element of the vector is set to 1 and everything else 0
    - $state = [0, 1, 0, 0, 0, \cdots 0]$

Feature Normalization

Indicator Variables and One-Hot Encoding

- For a variable like “Student” that takes True/False values:
  - We can simply replace with 0/1
- For a variable like “State” which in the US can take 50 values, we use something called One-Hot encoding
  - $state = [x_{NY}, x_{MA}, x_{NJ}, x_{WA}, x_{CA}, \cdots x_{RI}]$
  - If the particular data point is from MA, that element of the vector is set to 1 and everything else 0
    - $state = [0,1,0,0,0,\cdots 0]$

A key disadvantage of one-hot encoding is that the feature space grows extremely large

Feature Normalization

Normalization Methods

1. Min-Max Normalization
2. Mean-Variance Normalization
3. Max-Absolute Normalization
4. Robust Normalization

Feature Normalization

Min-Max Normalization

- For every column in the input data, i.e., for each x_0, x_1, x_2, x_4 etc., this normalization method will scale each column to 0 and 1

$$x' = \frac{x - min(x)}{max(x) - min(x)}$$

- This method preserves zero entries in sparse data
- But is very sensitive to **outliers**

Feature Normalization

Mean-Variance Normalization

- For every column in the input data, i.e., for each x_0, x_1, x_2, x_4 etc., this normalization method will scale to have mean 0 and standard deviation 1

$$x' = \frac{x - \mu(x)}{\sigma(x)}$$

- Most common in practice
- Less sensitive to outliers than min-max
- Does not bound the range to 0 and 1

Feature Normalization

Max-Absolute Normalization

- For every column in the input data, i.e., for each x_0, x_1, x_2, x_4 etc., this normalization method will scale each column to -1 and 1

$$x' = \frac{x}{|max(x)|}$$

- Good for sparse data since it preserves sparsity (zeros stay zero)

Feature Normalization

Robust Normalization

- For every column in the input data, i.e., for each x_0, x_1, x_2, x_4 etc., this normalization method will scale each column as

$$x' = \frac{x - median(x)}{IQR(x)}$$

- Robust to outliers

- Use when data has many outliers

Feature Normalization

Robust Normalization

- For every column in the input data, i.e., for each x_0, x_1, x_2, x_4 etc., this normalization method will scale each column as

$$x' = \frac{x - median(x)}{IQR(x)}$$

This is just the second quartile Q_2

Inter-quartile range $Q_3 - Q1$

- Robust to outliers

- Use when data has many outliers

Conclusion

- We saw practical issues and considerations in linear regression like
  - Train/test splits
  - Multicollinearity
  - Overfitting and Underfitting
  - Bias-Variance tradeoffs
  - Regularization
- Feature pre-processing
  - One-hot encoding
- Normalization methods