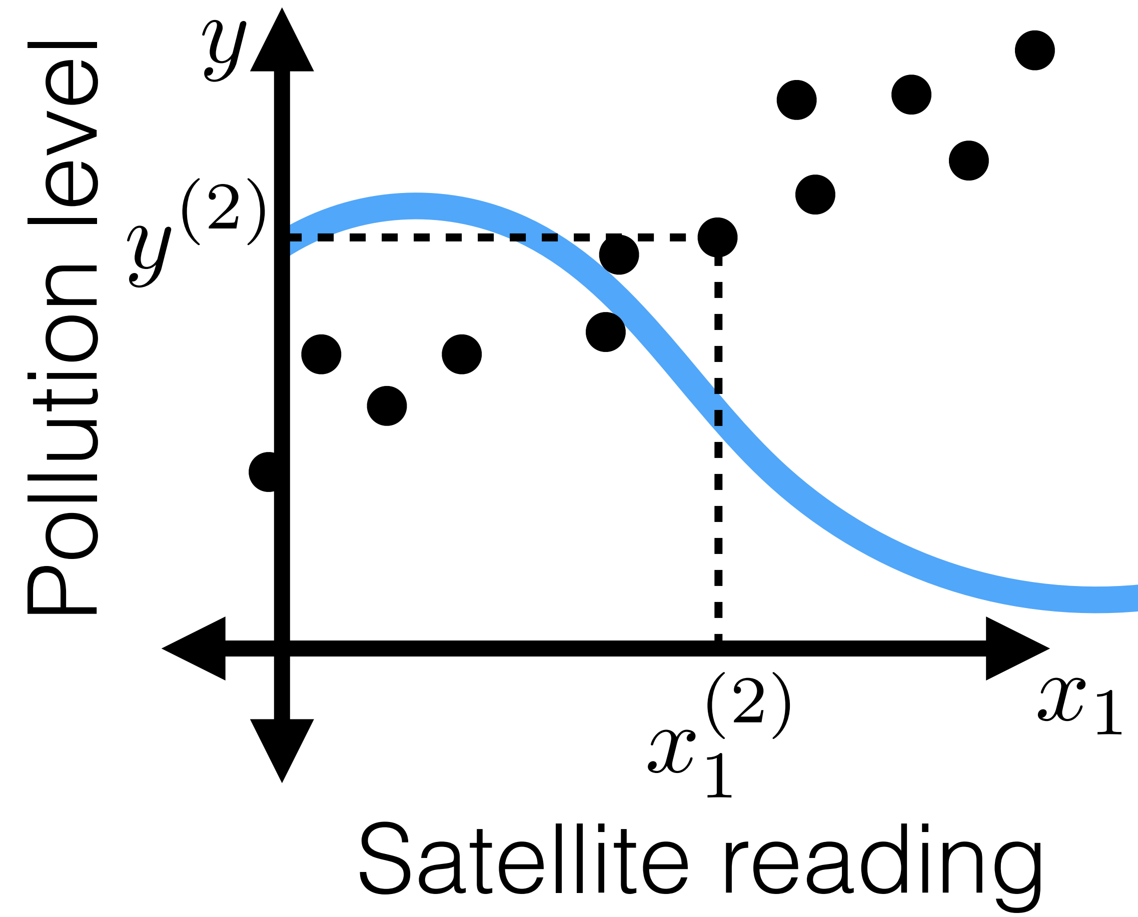
Analytical Regression

Prof. Tamara Broderick

Edited From 6.036 Fall21 Offering

Getting started: regression



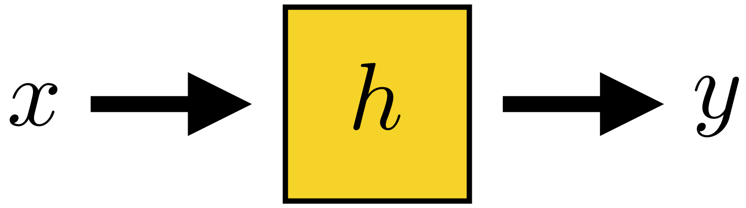
Example: predict pollution level

What do we have? (Training) data

- *n* training data points
- For data point $i \in \{1, \ldots, n\}$
  - Feature vector $x^{(i)} = (x_1^{(i)}, \ldots, x_d^{(i)})^\top \in \mathbb{R}^d$
  - Label $y^{(i)} \in \mathbb{R}$
- Training data $\mathcal{D}_n = \{(x^{(1)}, y^{(1)}), \ldots, (x^{(n)}, y^{(n)})\}$

What do we want? A good way to label new points

- How to label? Hypothesis $h : \mathbb{R}^d \to \mathbb{R}$



Is this a ***good*** hypothesis?

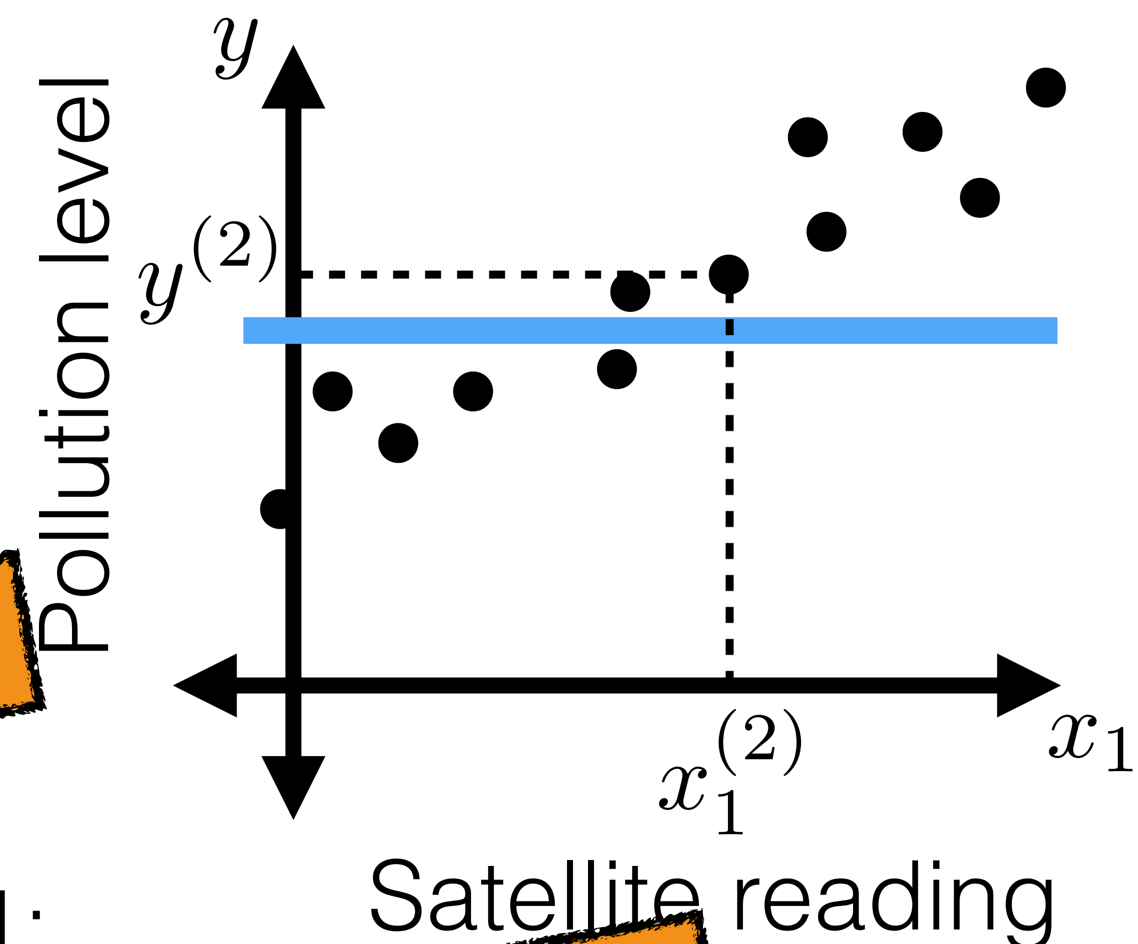
- Example *h*: For any *x*, *h*(*x*) = 1,000,000

Linear regressors

- Hypothesis class $\mathcal{H}$: set of h
  - Example: all constant functions
- A linear regression hypothesis when d=1:

parameters

$$h(x;\theta,\theta_0) = \theta x + \theta_0$$



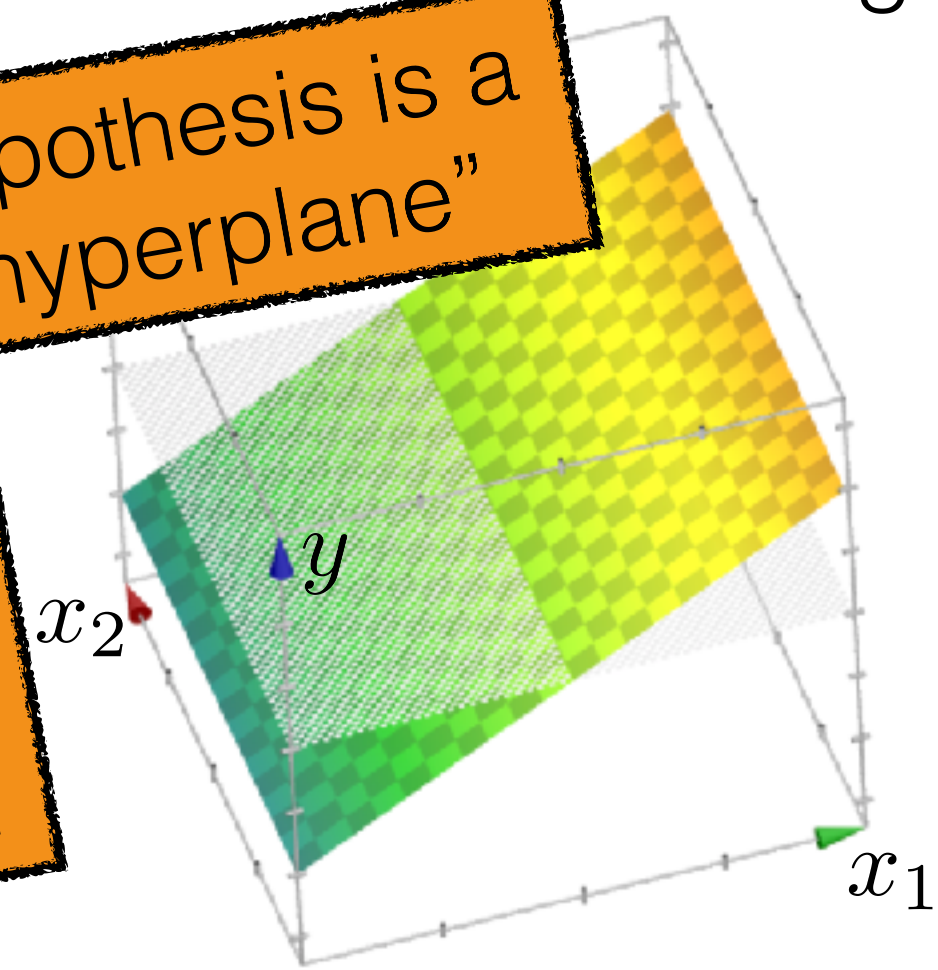
- A linear reg. hypothesis when d≥1:

$$\begin{aligned} h(x;\theta,\theta_0) &= \theta_1 x_1 + \cdots + \theta_d x_d + \theta_0 \\ &= \underbrace{\theta^\top x}_{\text{1x2,2x1}} + \theta_0 \end{aligned}$$

Hypothesis is a "hyperplane"

OR

$$\begin{aligned} h(x;\theta) &= \theta_1 x_1 + \cdots + \theta_d x_d + (\theta_0)(1) \\ &= \underbrace{\theta^\top x}_{\text{1x3,3x1}} \end{aligned}$$



Notational trick: *not* the same θ & x!

- Our hypothesis class in linear regression will be the set of all such h

How good is a regression hypothesis?

- Should predict well on future data
- How good is a regressor at one point? Loss $L(g,a)$

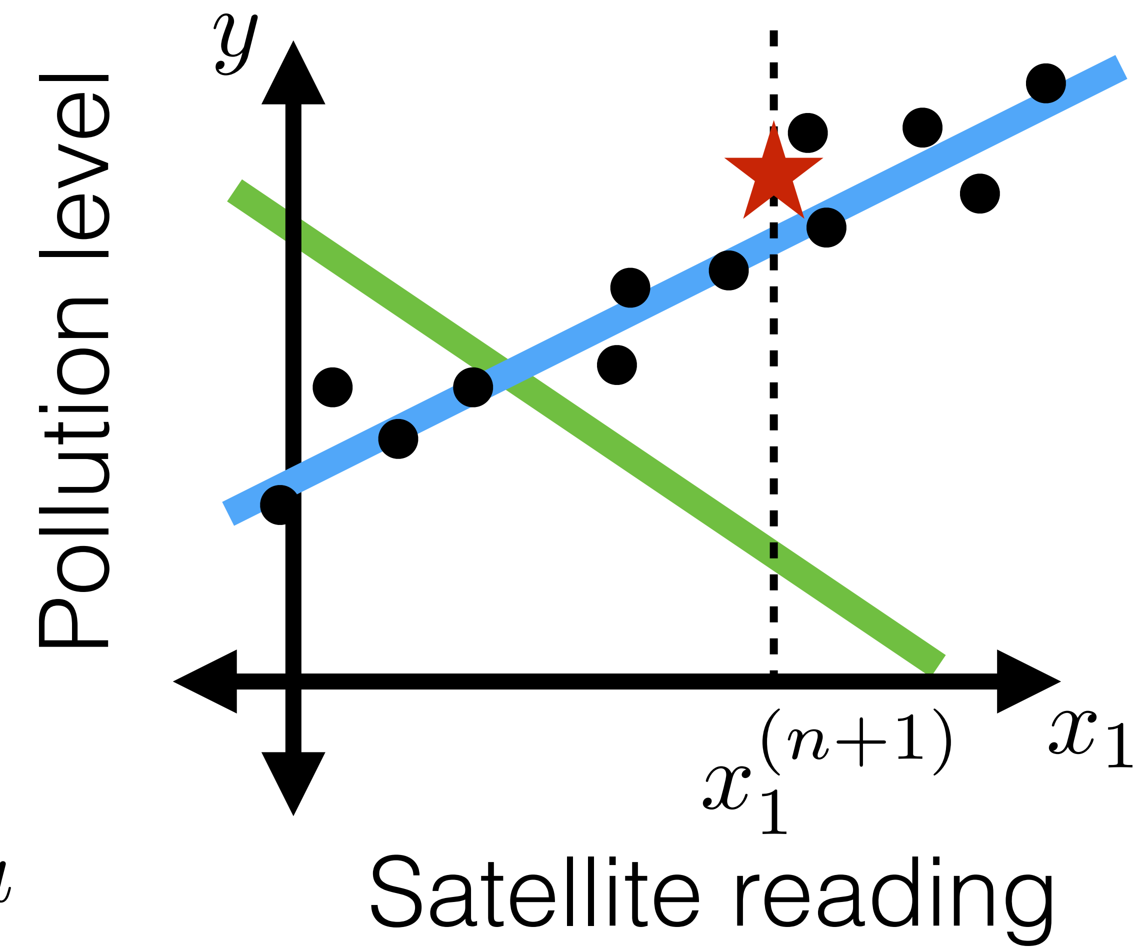
  g: guess,
  a: actual

  - Ex: squared loss

    $$L(g,a) = (g-a)^2$$

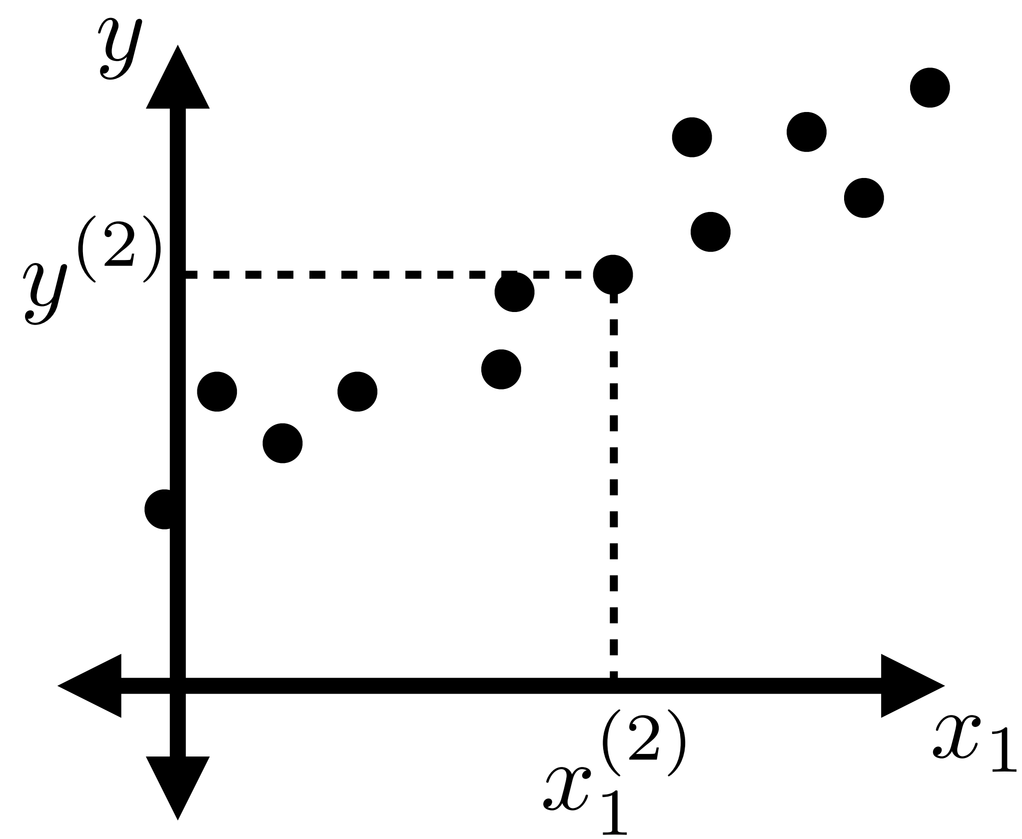
  - Example: asymmetric loss

    $$L(g,a) = \begin{cases} (g-a)^2 \text{ if } g > a \\ 2(g-a)^2 \text{ if } g \le a \end{cases}$$



- Test error (n' new points): $\mathcal{E}(h) = \frac{1}{n'} \sum_{i=n+1}^{n+n'} L(h(x^{(i)}), y^{(i)})$
- Training error: $\mathcal{E}_n(h) = \frac{1}{n} \sum_{i=1}^{n} L(h(x^{(i)}), y^{(i)})$
- One idea: prefer h to $\tilde{h}$ if $\mathcal{E}_n(h) < \mathcal{E}_n(\tilde{h})$

Learning a regressor



- Have data; have hypothesis class
- Want to choose a good regressor
  - Recall: $x \rightarrow \boxed{h} \rightarrow y$
  - New: $\mathcal{D}_n \rightarrow \boxed{\text{learning algorithm}} \rightarrow h$
- Example:
  - Suppose someone already generated 1 trillion hypotheses, e.g. at random, indexed by *j*:

$$h^{(j)}(x) = h(x; \theta^{(j)}, \theta_0^{(j)})$$

hyperparameter

```
Ex_learning_alg( 𝒟ₙ ; k )
   Set j* = the j ∈ {1,...,k} with lowest ℰₙ(h^(j))
   Return h^(j*)
```

- How does training error of `Ex_learning_alg`($\mathcal{D}_n$;1) compare to the training error of `Ex_learning_alg`($\mathcal{D}_n$;2)?

Linear regression: Another way

- How about we just consider all hypotheses in our class and choose the one with lowest training error?
  - We'll see: not typically straightforward
  - But for linear regression with square loss: can do it!
- Recall: training error: $\mathcal{E}_n(h) = \frac{1}{n}\sum_{i=1}^{n} L(h(x^{(i)}), y^{(i)})$
- Training error: square loss, linear regr., extra "1" feature

$$J(\theta) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} - y^{(i)})^2$$

Define $\underset{\text{nxd}}{\tilde{X}} = \begin{bmatrix} x_1^{(1)} & \cdots & x_d^{(1)} \\ \vdots & \ddots & \vdots \\ x_1^{(n)} & \cdots & x_d^{(n)} \end{bmatrix}$

Linear regression: Another way

- How about we just consider all hypotheses in our class and choose the one with lowest training error?
  - We'll see: not typically straightforward
  - But for linear regression with square loss: can do it!
- Recall: training error: $\mathcal{E}_n(h) = \frac{1}{n} \sum_{i=1}^{n} L(h(x^{(i)}), y^{(i)})$
- Training error: square loss, linear regr., extra "1" feature

$$J(\theta) = \frac{1}{n} \sum_{i=1}^{n} (\theta^\top x^{(i)} - y^{(i)})^2$$

Define $\underset{\text{nxd}}{\tilde{X}} = \begin{bmatrix} \boxed{x_1^{(1)} \quad \cdots \quad x_d^{(1)}} \\ \vdots & \ddots & \vdots \\ x_1^{(n)} & \cdots & x_d^{(n)} \end{bmatrix}$

Linear regression: Another way

- How about we just consider all hypotheses in our class and choose the one with lowest training error?
  - We'll see: not typically straightforward
  - But for linear regression with square loss: can do it!
- Recall: training error: $\mathcal{E}_n(h) = \frac{1}{n}\sum_{i=1}^{n} L(h(x^{(i)}), y^{(i)})$
- Training error: square loss, linear regr., extra "1" feature

$$J(\theta) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} - y^{(i)})^2$$

Define $\underset{\text{nxd}}{\tilde{X}} = \begin{bmatrix} x_1^{(1)} & \cdots & x_d^{(1)} \\ \vdots & \ddots & \vdots \\ x_1^{(n)} & \cdots & x_d^{(n)} \end{bmatrix}$ $\quad \underset{\text{nx1}}{\tilde{Y}} = \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(n)} \end{bmatrix}$

Linear regression: Another way

- How about we just consider all hypotheses in our class and choose the one with lowest training error?
  - We'll see: not typically straightforward
  - But for linear regression with square loss: can do it!
- Recall: training error: $\mathcal{E}_n(h) = \frac{1}{n}\sum_{i=1}^{n} L(h(x^{(i)}), y^{(i)})$
- Training error: square loss, linear regr., extra "1" feature

$$J(\theta) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} - y^{(i)})^2 \boxed{= \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y})}$$

Define $\underset{\text{nxd}}{\tilde{X}} = \begin{bmatrix} x_1^{(1)} & \cdots & x_d^{(1)} \\ \vdots & \ddots & \vdots \\ x_1^{(n)} & \cdots & x_d^{(n)} \end{bmatrix}$ $\underset{\text{nx1}}{\tilde{Y}} = \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(n)} \end{bmatrix}$

Linear regression: Another way

- How about we just consider all hypotheses in our class and choose the one with lowest training error?
  - We'll see: not typically straightforward
  - But for linear regression with square loss: can do it!

- Recall: training error: $\mathcal{E}_n(h) = \frac{1}{n}\sum_{i=1}^{n} L(h(x^{(i)}), y^{(i)})$

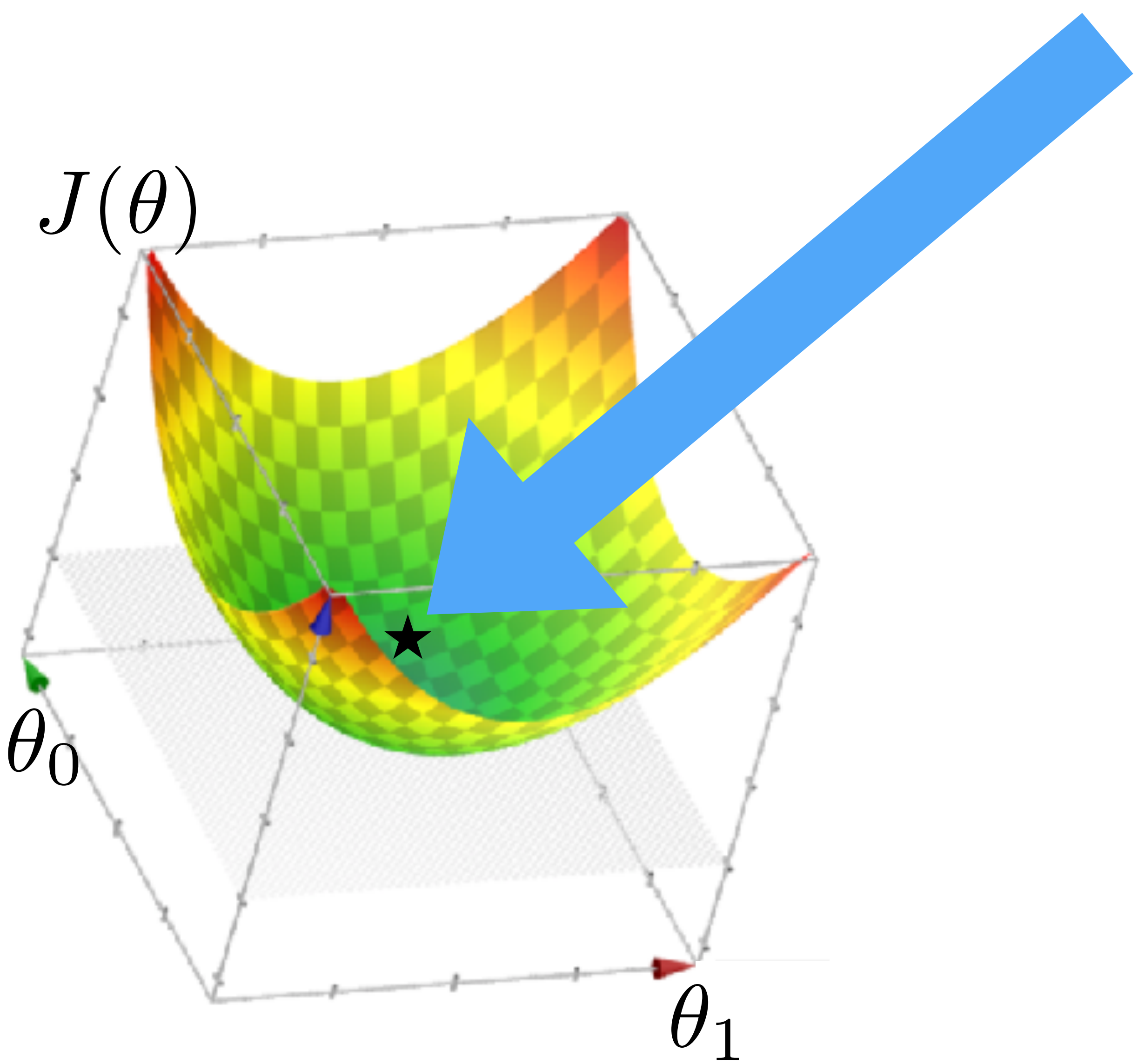
- Training error: square loss, linear regr., extra "1" feature

$$J(\theta) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} - y^{(i)})^2 = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y})$$

Define $\underset{\text{nxd}}{\tilde{X}} = \begin{bmatrix} x_1^{(1)} & \cdots & x_d^{(1)} \\ \vdots & \ddots & \vdots \\ x_1^{(n)} & \cdots & x_d^{(n)} \end{bmatrix}$ $\quad \underset{\text{nx1}}{\tilde{Y}} = \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(n)} \end{bmatrix}$

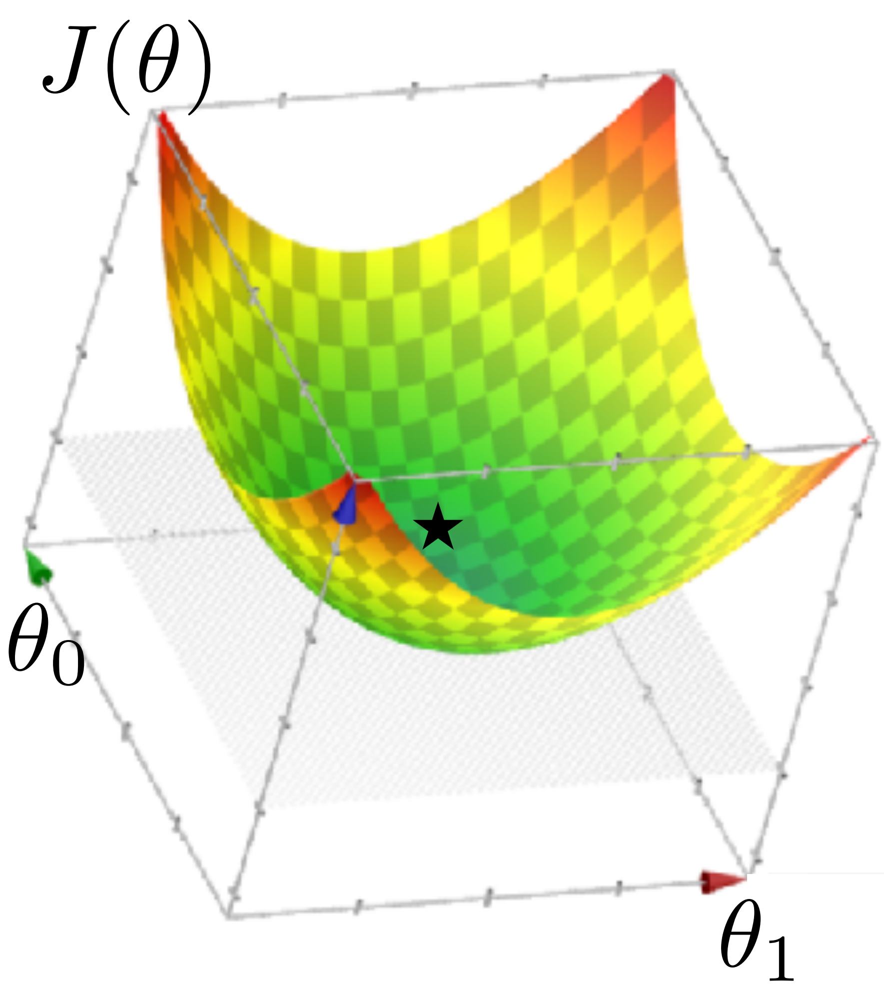
Linear regression: A Direct Solution

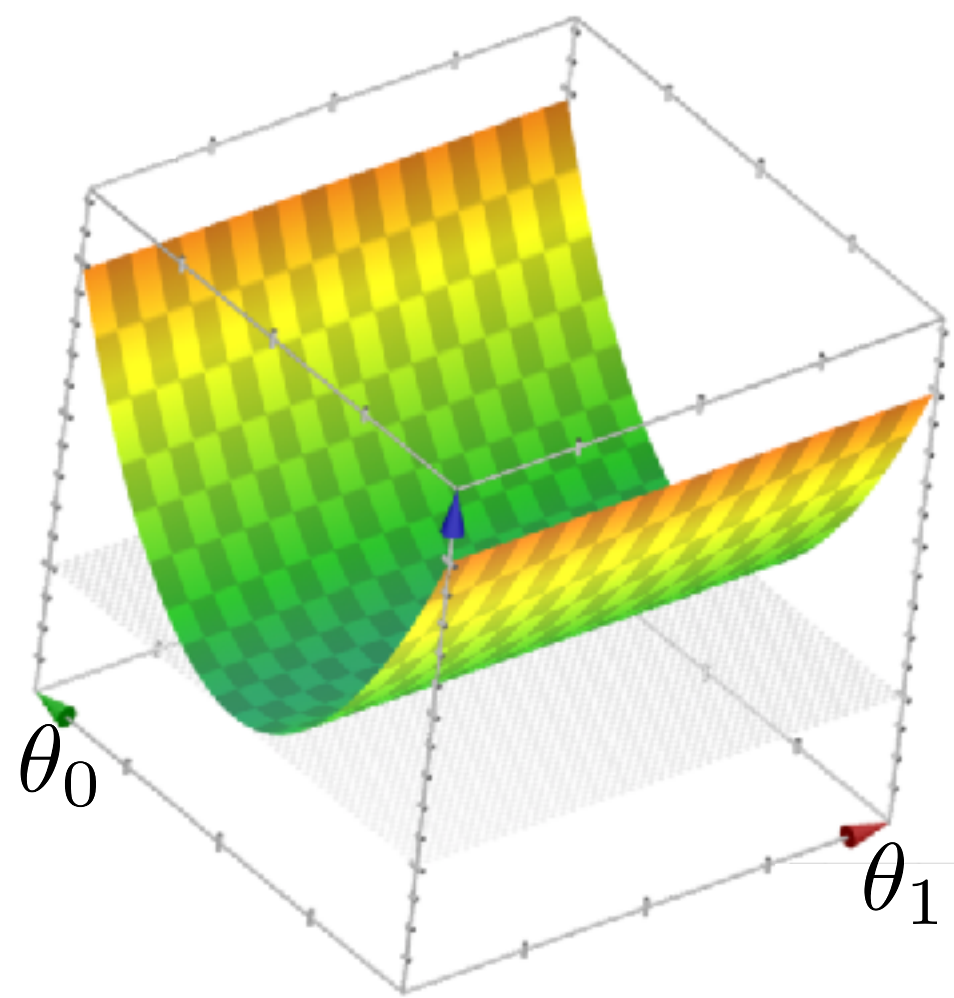
- Goal: minimize $J(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y})$



Linear regression: A Direct Solution

- Goal: minimize $J(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y})$
- Uniquely minimized at a point if gradient at that point is zero and function "curves up" [see linear algebra]





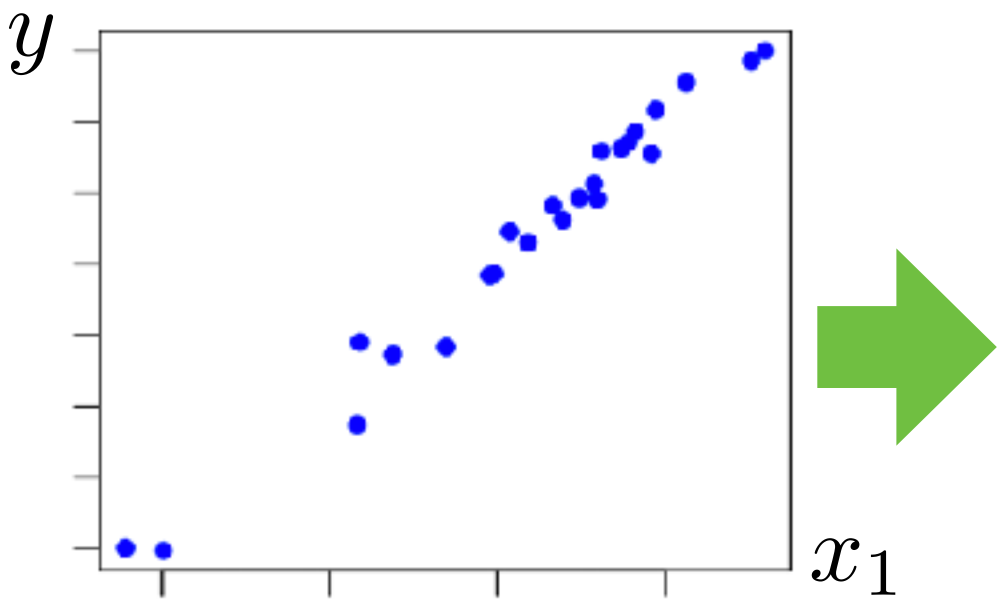
Linear regression: A Direct Solution

- Goal: minimize $J(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y})$
- Uniquely minimized at a point if gradient at that point is zero and function “curves up” [see linear algebra]
- Gradient $\overset{\text{dx1}}{\nabla_\theta J(\theta)} \overset{\text{set}}{=} 0$

Linear regression: A Direct Solution

- Goal: minimize $J(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y})$
- Uniquely minimized at a point if gradient at that point is zero and function "curves up" [see linear algebra]
- Gradient $\underbrace{\nabla_\theta J(\theta)}_{\text{dx1}} \overset{\text{set}}{=} 0 \Rightarrow \theta = (\tilde{X}^\top\tilde{X})^{-1}\tilde{X}^\top\tilde{Y}$

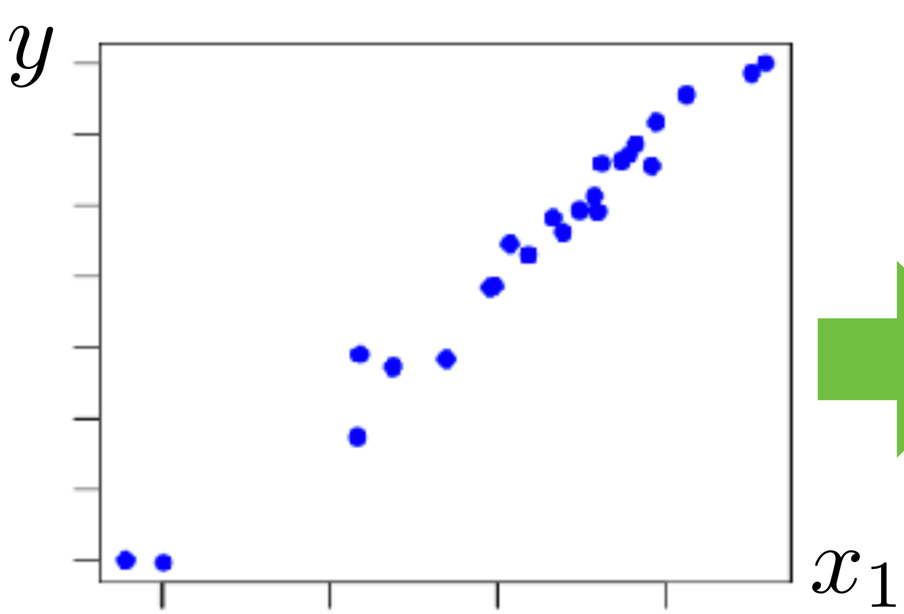
Exercise: check n,d=1

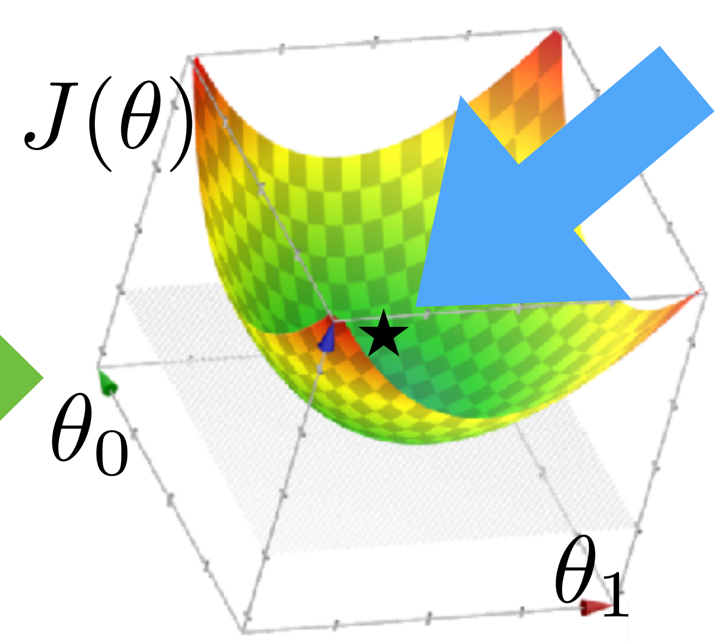


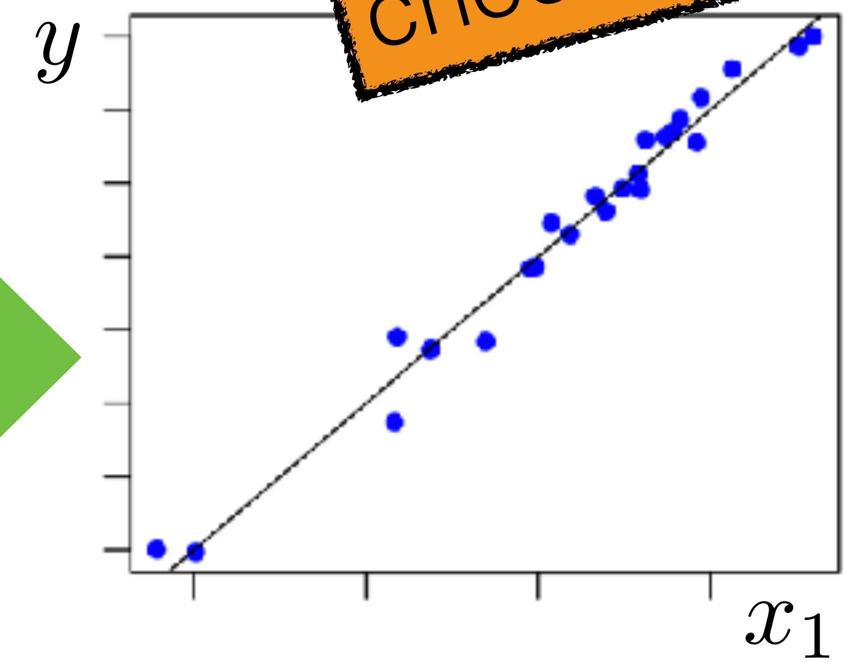
Linear regression: A Direct Solution

- Goal: minimize $J(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y})$
- Uniquely minimized at a point if gradient at that point is zero and function “curves up” [see linear algebra]
- Gradient $\nabla_\theta \overset{\text{dx1}}{J(\theta)} \overset{\text{set}}{=} 0 \;\Rightarrow\; \theta = (\tilde{X}^\top \tilde{X})^{-1}\tilde{X}^\top \tilde{Y}$

Exercise: check n,d=1



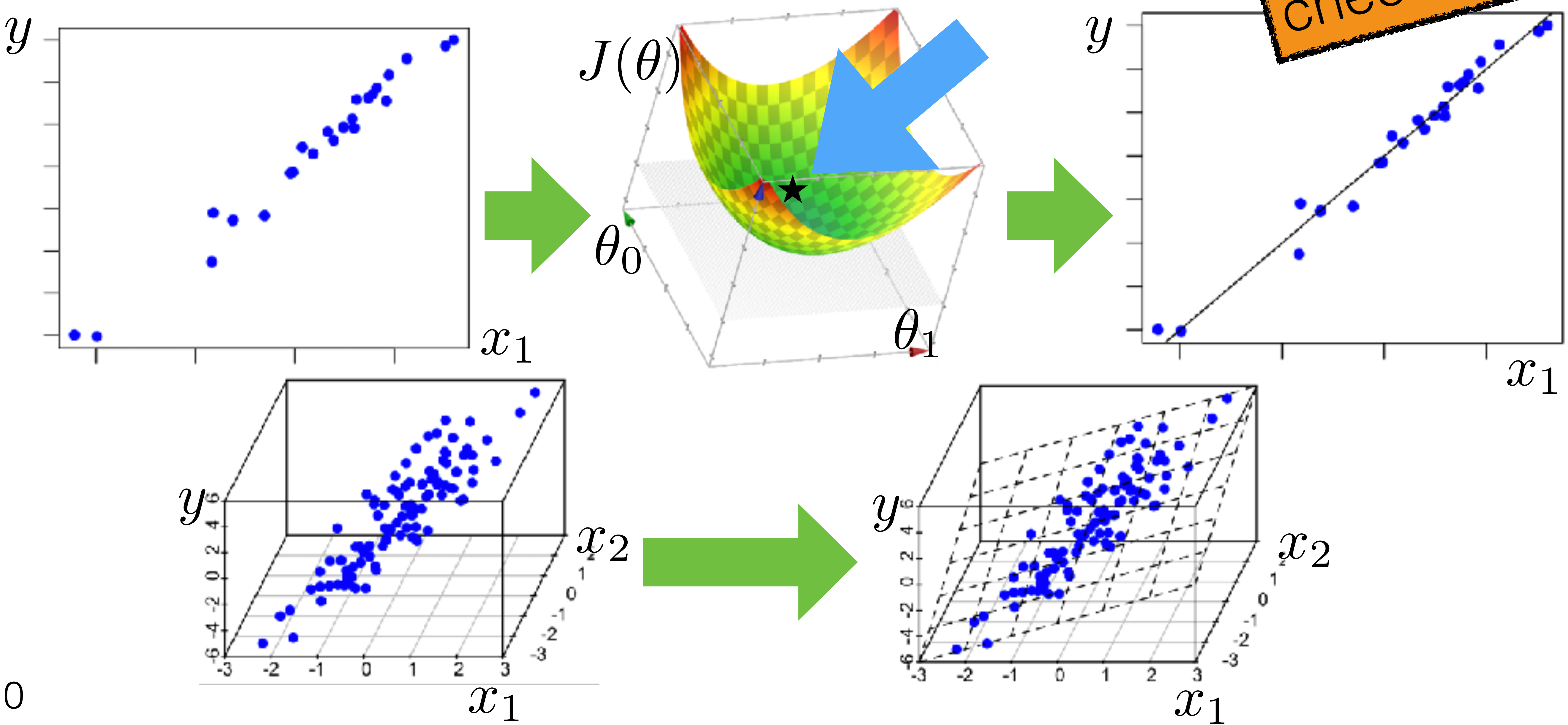




Linear regression: A Direct Solution

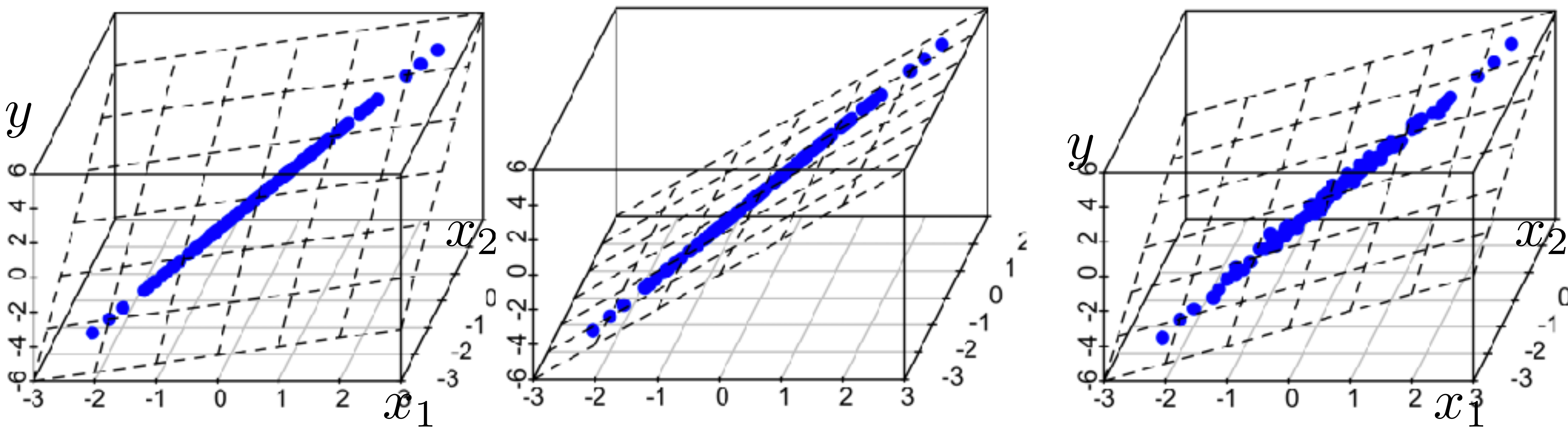
- Goal: minimize $J(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y})$
- Uniquely minimized at a point if gradient at that point is zero and function "curves up" [see linear algebra]
- Gradient $\overset{\text{dx1}}{\nabla_\theta J(\theta)} \overset{\text{set}}{=} 0 \Rightarrow \theta = (\tilde{X}^\top\tilde{X})^{-1}\tilde{X}^\top\tilde{Y}$

Exercise: check n,d=1



What can go wrong in practice?

- Does the linear regr. objective always “curve up”? No!
- Sometimes there isn’t a unique best hyperplane
  - Then $X^\top X$ not invertible



- Sometimes there’s technically a unique best hyperplane, but just because of noise
- Practical: real-life features often have this issue
- How to choose among hyperplanes? Preference for θ components being near zero

Regularizing linear regression

- Linear regression with square penalty: ridge regression

$$J_{\text{ridge}}(\theta, \theta_0) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2 + \lambda\|\theta\|^2$$

Regularizing linear regression

- Linear regression with square penalty: ridge regression

$$J_{\text{ridge}}(\theta, \theta_0) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2 + \lambda\|\theta\|^2$$

Regularizing linear regression

- Linear regression with square penalty: ridge regression

$$J_{\text{ridge}}(\theta, \theta_0) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2 + \lambda\|\theta\|^2 \quad (\lambda > 0)$$

Regularizing linear regression

- Linear regression with square penalty: ridge regression

$$J_{\text{ridge}}(\theta, \theta_0) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2 + \lambda\|\theta\|^2 \quad (\lambda > 0)$$

- Special case: ridge regression with no offset

$$J_{\text{ridge}}(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y}) + \lambda\|\theta\|^2$$

What happens if $\lambda < 0$?

  - Min at: $\nabla_\theta J_{\text{ridge}}(\theta) = 0$

$$\Rightarrow \theta = (\underbrace{\tilde{X}^\top \tilde{X}}_{\text{dxn,nxd}} + n\lambda \underbrace{I}_{\text{dxd}})^{-1}\tilde{X}^\top \tilde{Y}$$

Exercise: write out the learning algorithm

  - When $\lambda > 0$, always “curves up” & can invert
  - Can also solve with an offset

- Can think of λ as hyperparameter of a learning algorithm
- How to choose λ? One option: cross validation (see HW!)