LINEAR MODELS

Based on slides by Andrew Ng

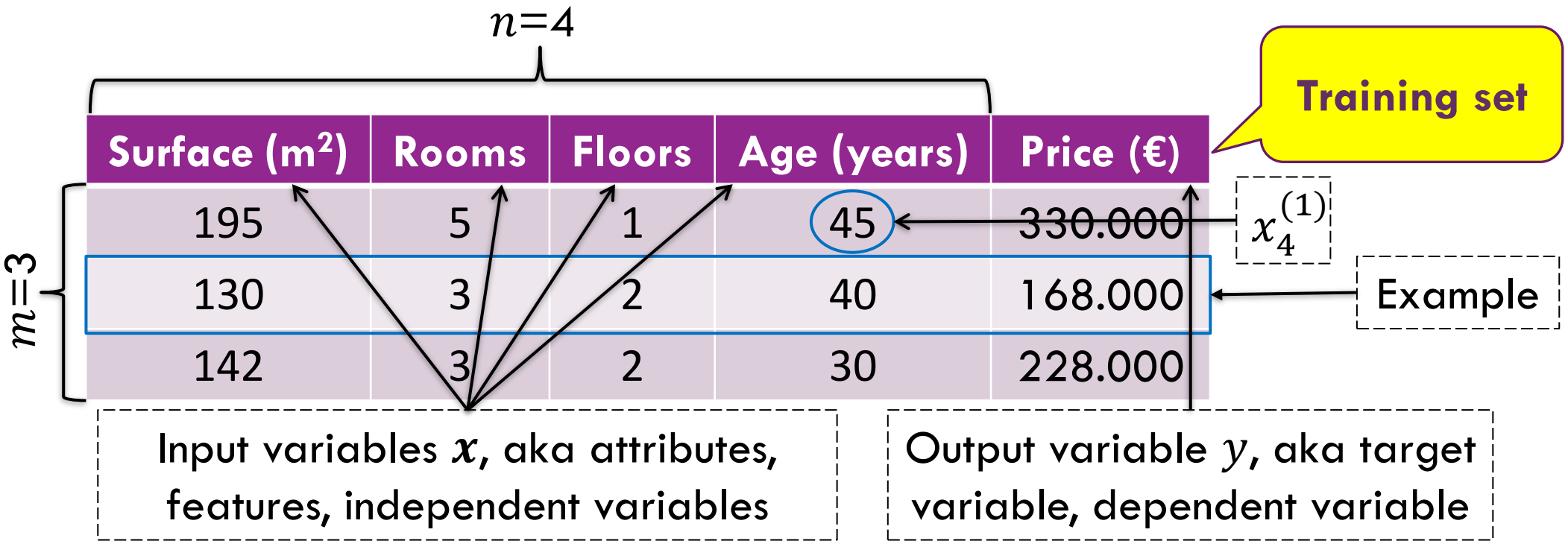
Grigorios Tsoumakas,
School of informatics,
Aristotle university of Thessaloniki

OUTLINE

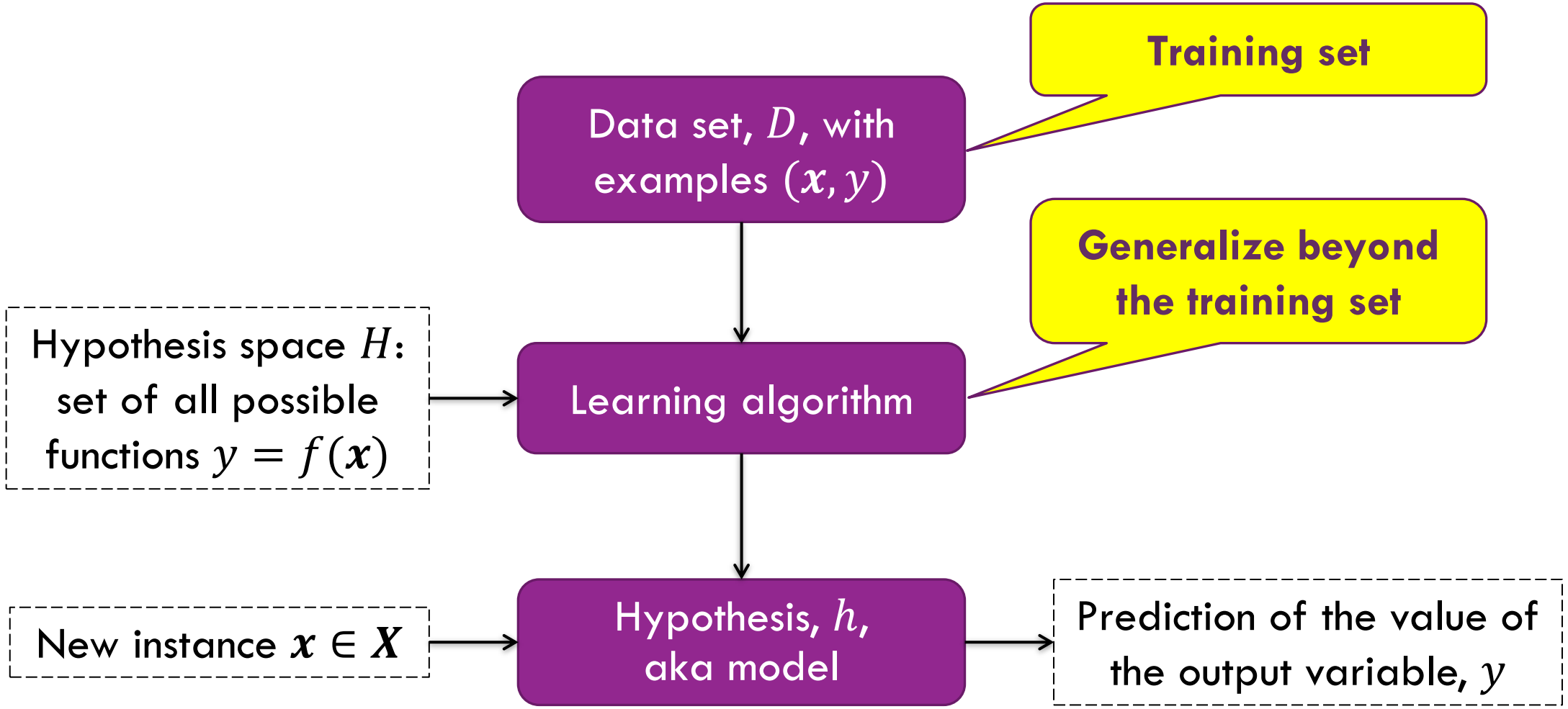
Linear regression

Logistic regression

SUPERVISED LEARNING NOTATION AND FORMALISM



LEARNING PROCESS



THE 3 COMPONENTS OF LEARNING ALGORITHMS

Representation

- Instances
- Hyperplanes
- Decision Trees
- Sets of Rules
- Neural Networks

Evaluation

- Accuracy
- Precision, recall
- Squared error
- Likelihood
- Posterior prob.
- Information gain
- KL divergence
- Margin

Optimization

- Combinatorial
  - Greedy search
  - Beam search
  - Branch-and-bound
- Continuous
  - Gradient descent
  - Linear programming
  - Quadratic program.

REPRESENTATION

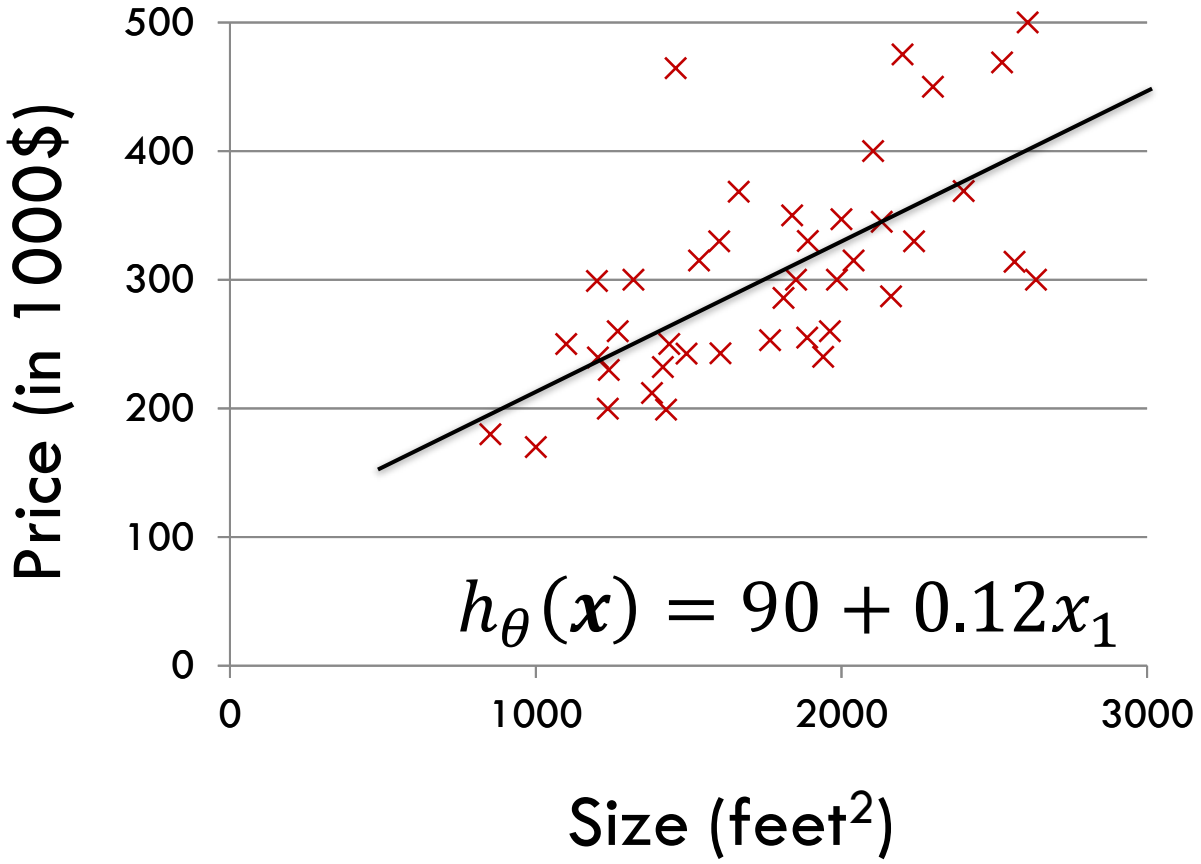
In linear regression, we represent the hypothesis, h, as follows

$$h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$

The real-valued constants $\theta_0, \theta_1, \ldots, \theta_n$ are called parameters

- θ_0 is called bias or intercept, $\theta_1, \ldots, \theta_n$ are called feature weights

EXAMPLE: HOME VALUATION



EVALUATION

How good is a given hypothesis?

A loss function $L(h_{\boldsymbol{\theta}}(\boldsymbol{x}), y)$ measures the difference between the value of the output variable y of a training example $(\boldsymbol{x}, y)$ and the output of the hypothesis given $\boldsymbol{x}$, $h_{\boldsymbol{\theta}}(\boldsymbol{x})$

A cost function iterates over the training corpus $\left(\boldsymbol{x}^{(i)}, y^{(i)}\right)$, $i = 1 \ldots m$ and measures the average loss between the ground truth $y^{(i)}$ and the output of the hypothesis $h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right)$

REPRESENTATION + EVALUATION

In linear regression, we represent the hypothesis, h, as follows

$$h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$

The real-valued constants $\theta_0, \theta_1, \ldots, \theta_n$ are called parameters

- θ_0 is called bias or intercept, $\theta_1, \ldots, \theta_n$ are called feature weights

We evaluate linear regression hypotheses using the (half of) the mean squared error cost function

- $J(\theta_0, \ldots, \theta_n) = \frac{1}{2m} \sum_{i=1}^{m} [h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}) - y^{(i)}]^2$

AN EXAMPLE WITH A SIMPLIFIED HYPOTHESIS

Hypothesis with one variable

- $h_\theta(\boldsymbol{x}) = \theta_0 + \theta_1 x_1$

Parameters θ_0, θ_1

Cost function

- $J(\theta_0, \theta_1) = \frac{1}{2m}\sum_{i=1}^{m}[h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}) - y^{(i)}]^2$

Goal: $\min_{\theta_0,\theta_1} J(\theta_0, \theta_1)$

Simplified hypothesis

- $h_\theta(\boldsymbol{x}) = \theta_1 x_1$ (i.e., $\theta_0 = 0$)

Parameter θ_1

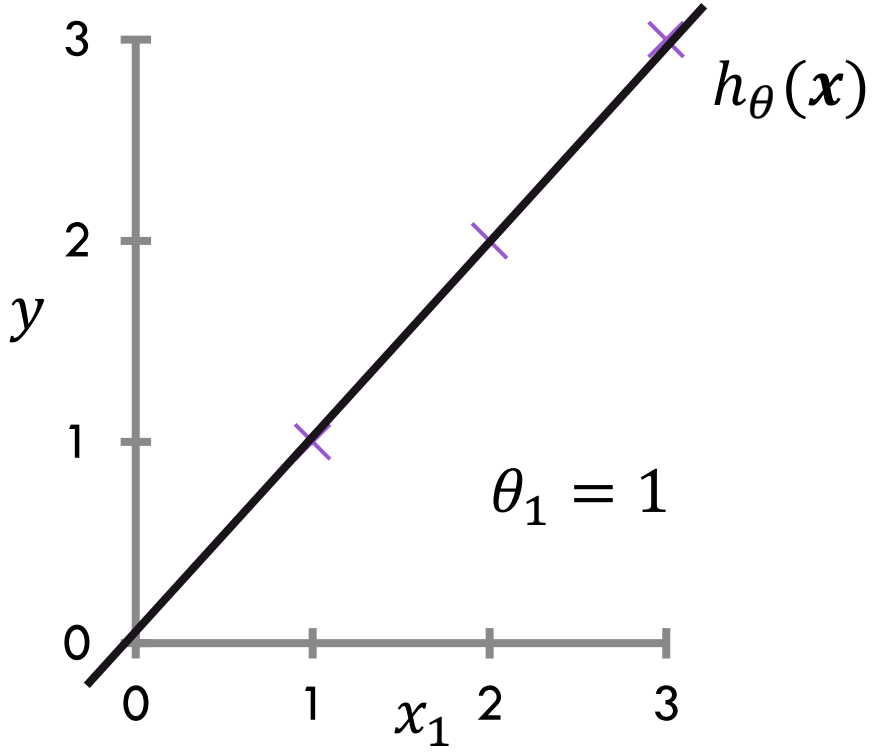
Cost function

- $J(\theta_1) = \frac{1}{2m}\sum_{i=1}^{m}[h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}) - y^{(i)}]^2$

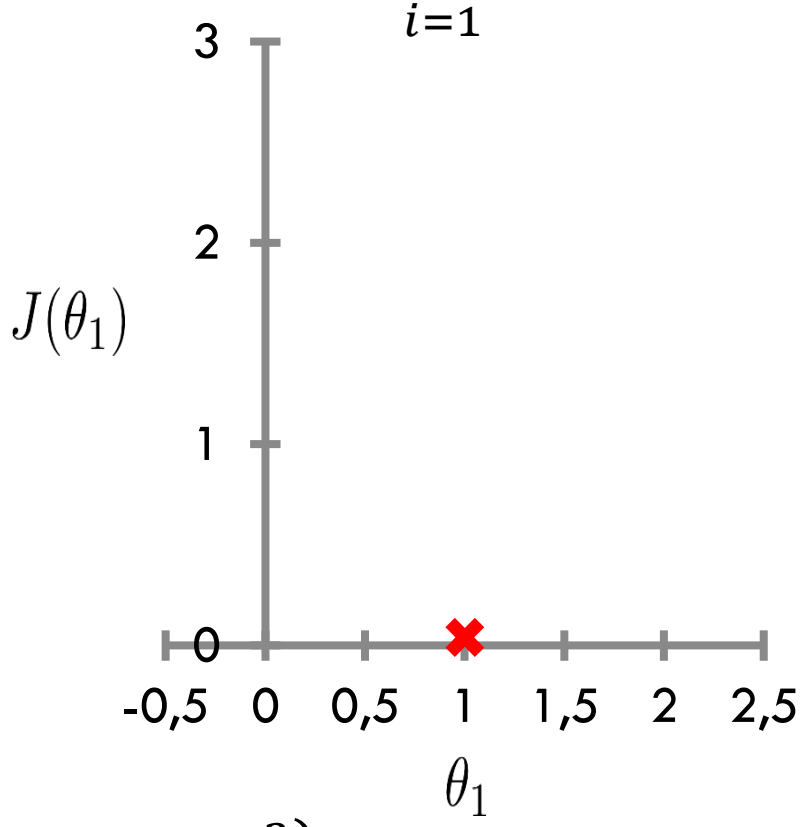
Goal: $\min_{\theta_1} J(\theta_1)$

AN EXAMPLE WITH A SIMPLIFIED HYPOTHESIS

$$h_\theta(\boldsymbol{x}) = \theta_1 x_1$$



$$J(\theta_1) = \frac{1}{2m}\sum_{i=1}^{m}\left[h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right]^2$$

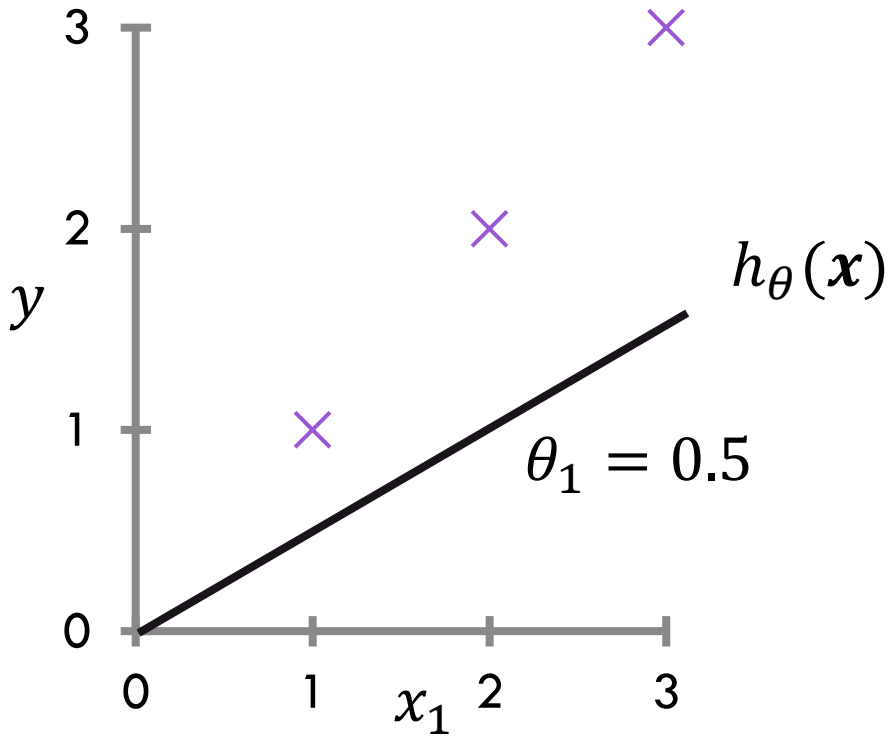


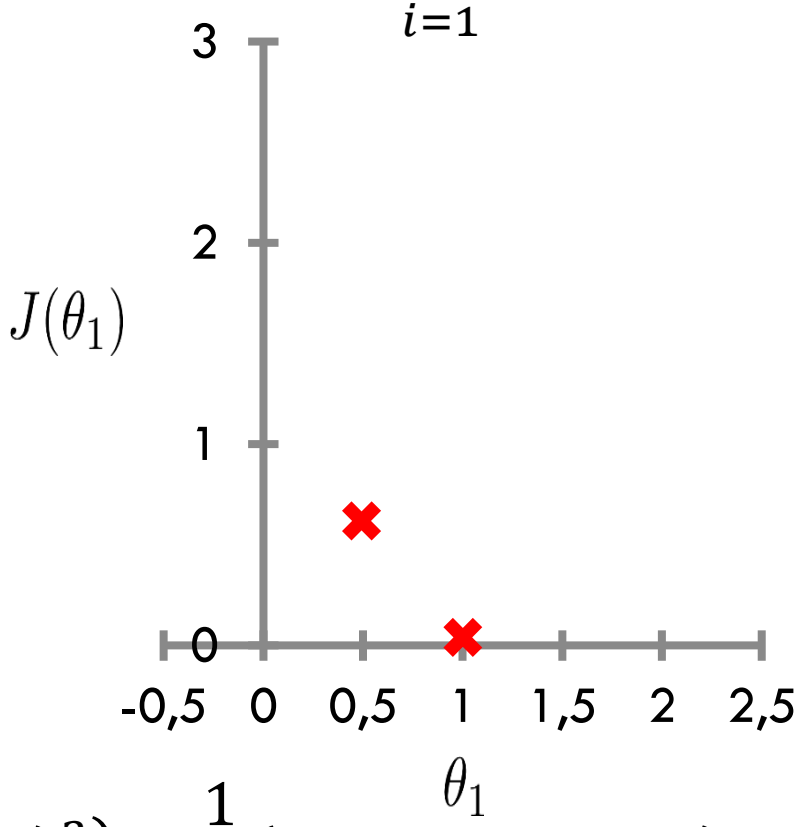
$$J(\theta_1) = \frac{1}{6}\left((1-1)^2 + (2-2)^2 + (3-3)^2\right) = 0$$

AN EXAMPLE WITH A SIMPLIFIED HYPOTHESIS

$$h_\theta(\boldsymbol{x}) = \theta_1 x_1$$

$$J(\theta_1) = \frac{1}{2m}\sum_{i=1}^{m}[h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}) - y^{(i)}]^2$$

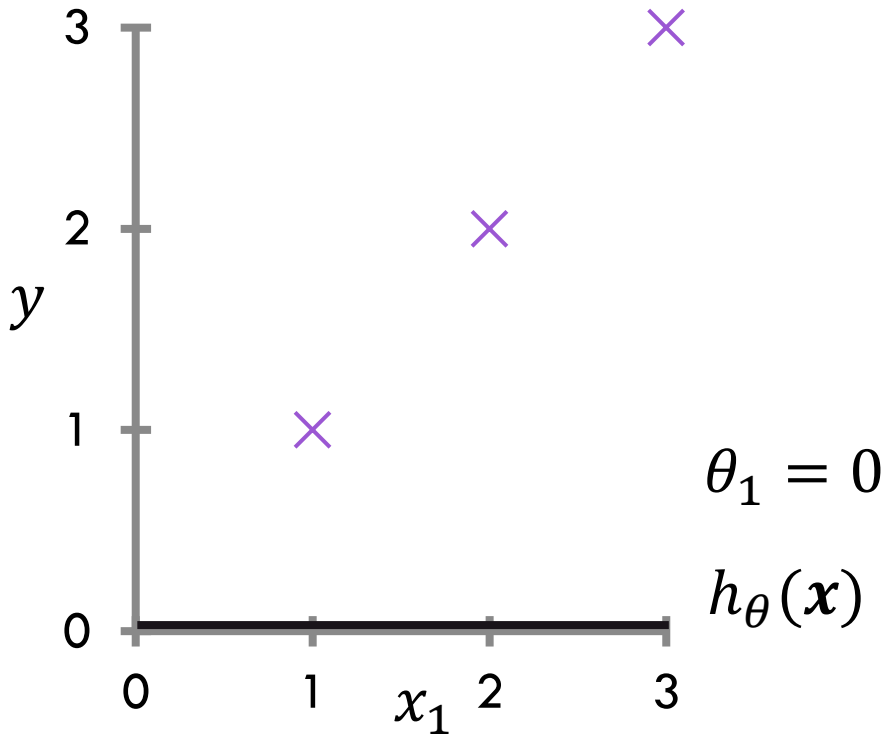




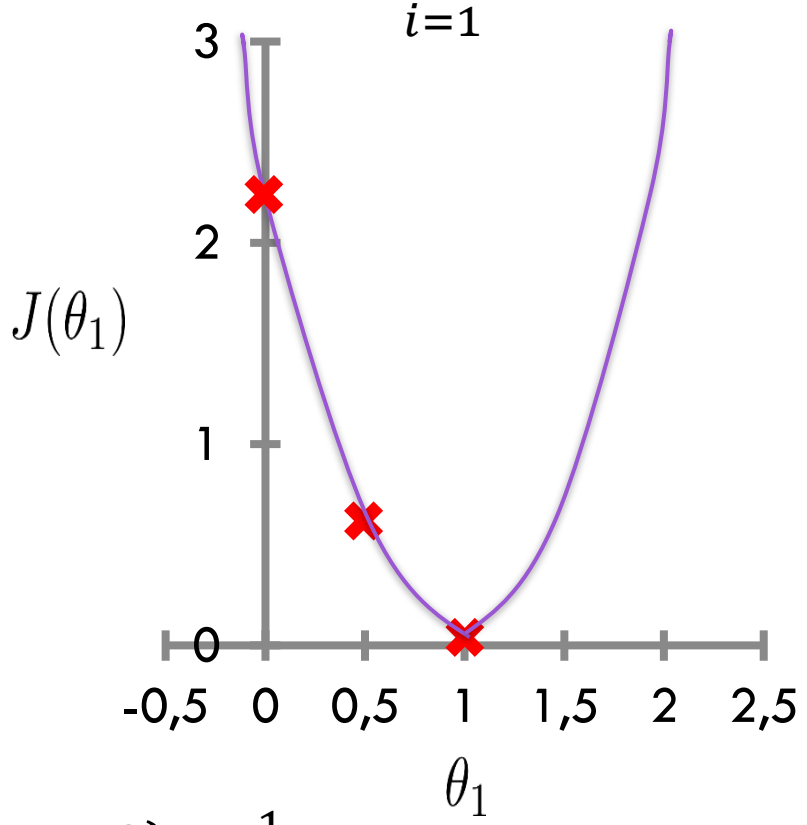
$$J(\theta_1) = \frac{1}{6}\left((0{,}5-1)^2 + (1-2)^2 + (1{,}5-3)^2\right) = \frac{1}{6}(0{,}25 + 1 + 2{,}25) \cong 0{,}58$$

AN EXAMPLE WITH A SIMPLIFIED HYPOTHESIS

$$h_\theta(x) = \theta_1 x_1$$



$$J(\theta_1) = \frac{1}{2m}\sum_{i=1}^{m}[h_\theta(x^{(i)}) - y^{(i)}]^2$$



$$J(\theta_1) = \frac{1}{6}\left((0-1)^2 + (0-2)^2 + (0-3)^2\right) = \frac{1}{6}(1+4+9) \cong 2{,}33$$

VISUALIZING THE COST WITH ONE VARIABLE

Hypothesis with one variable

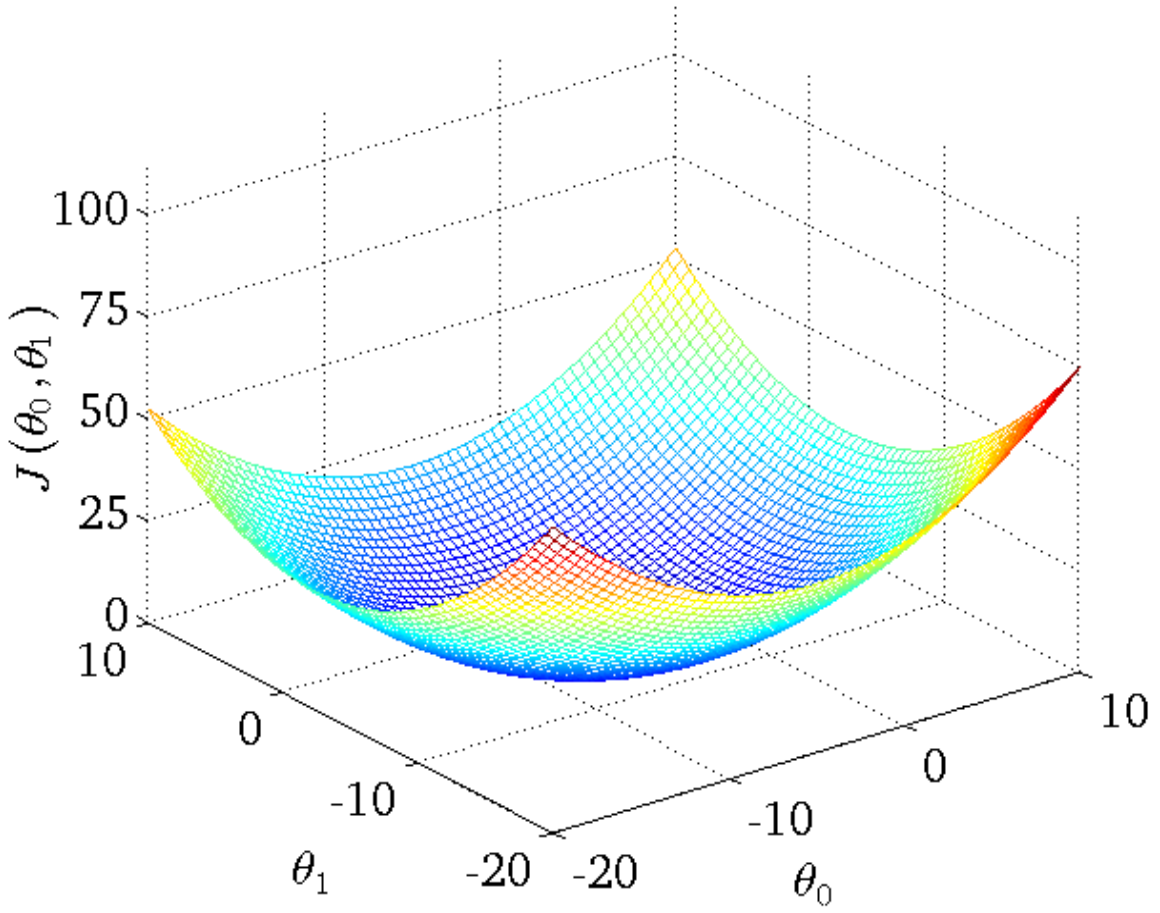
- $h_\theta(\boldsymbol{x}) = \theta_0 + \theta_1 x_1$

Parameters θ_0, θ_1

Cost function

- $J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} [h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}) - y^{(i)}]^2$

Goal: $\min_{\theta_0, \theta_1} J(\theta_0, \theta_1)$



REPRESENTATION + EVALUATION + OPTIMIZATION

We represent the hypothesis, h, as follows

$$h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$

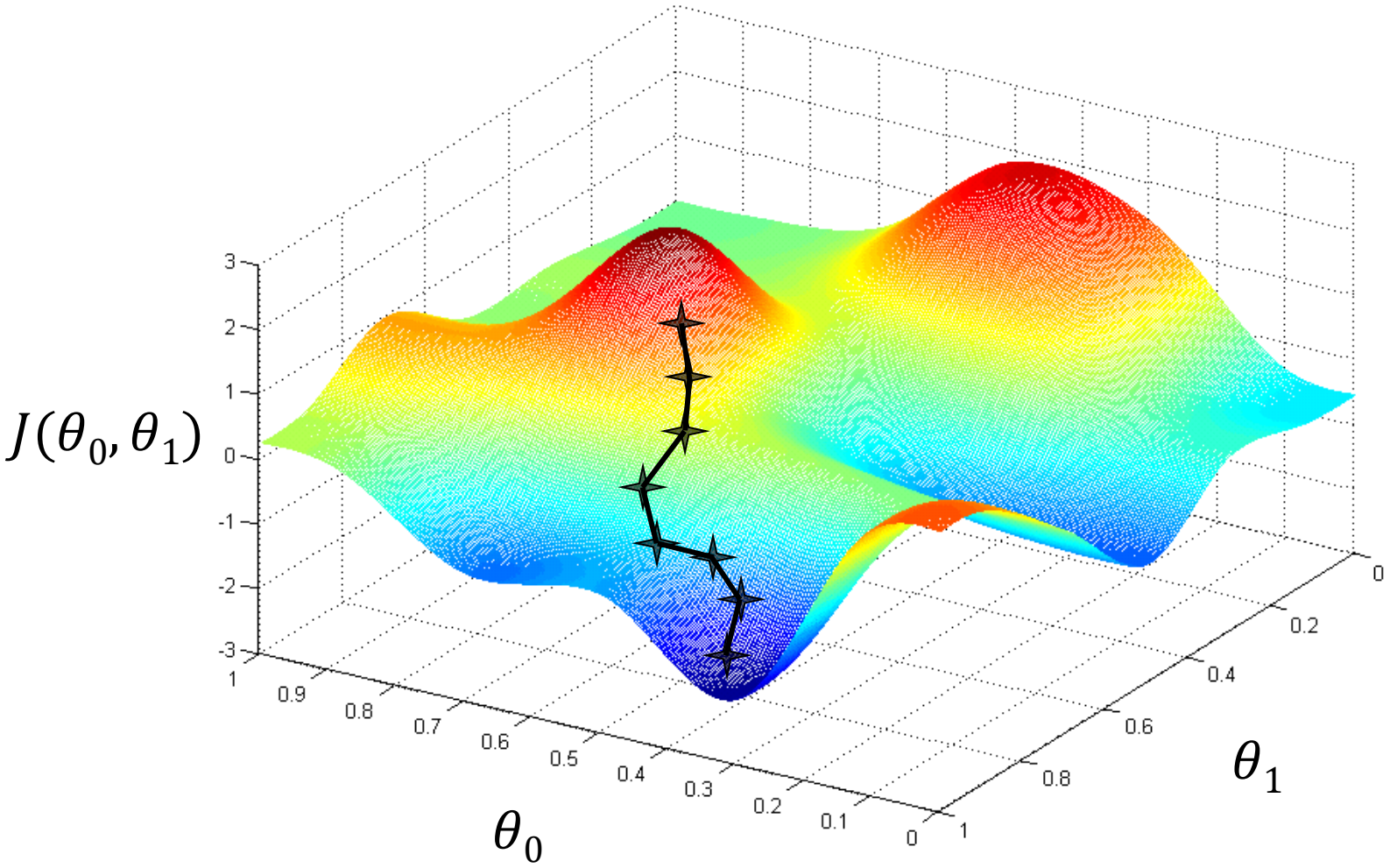
We evaluate hypotheses using (half of the) mean squared error

- $J(\theta_0, \ldots, \theta_n) = \frac{1}{2m} \sum_{i=1}^{m} [h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}) - y^{(i)}]^2$

We search for the optimal hypothesis using gradient descent

- Iterative optimization algorithm for finding the minimum of a function
- Here we want to find the $\theta_0, \ldots, \theta_n$ that minimize $J(\theta_0, \ldots, \theta_n)$

GD IN ARBITRARY FUNCTION OF TWO PARAMETERS



Start with random initial values for the parameters

Take steps proportional to the negative of the gradient of the function at the current point

GRADIENT DESCENT

Iterate the following update of parameters until convergence

- $\forall j \in \{0, \dots, n\}$: $\theta_j := \theta_j - \eta \frac{\partial}{\partial \theta_j} J(\theta_0, \dots, \theta_n)$
- η is a small positive number called the learning rate

In linear regression

- $\forall j \in \{0, \dots, n\}$: $\theta_j := \theta_j - \eta \frac{\partial}{\partial \theta_j} \frac{1}{2m} \sum_{i=1}^{m} [h_{\boldsymbol{\theta}}(x^{(i)}) - y^{(i)}]^2$
- $\theta_0 := \theta_0 - \eta \frac{1}{m} \sum_{i=1}^{m} [h_{\boldsymbol{\theta}}(x^{(i)}) - y^{(i)}]$
- $\forall j > 0$: $\theta_j := \theta_j - \eta \frac{1}{m} \sum_{i=1}^{m} [h_{\boldsymbol{\theta}}(x^{(i)}) - y^{(i)}] \, x_j^{(i)}$

SETTING THE LEARNING RATE

Gradient descent will converge for sufficiently small η

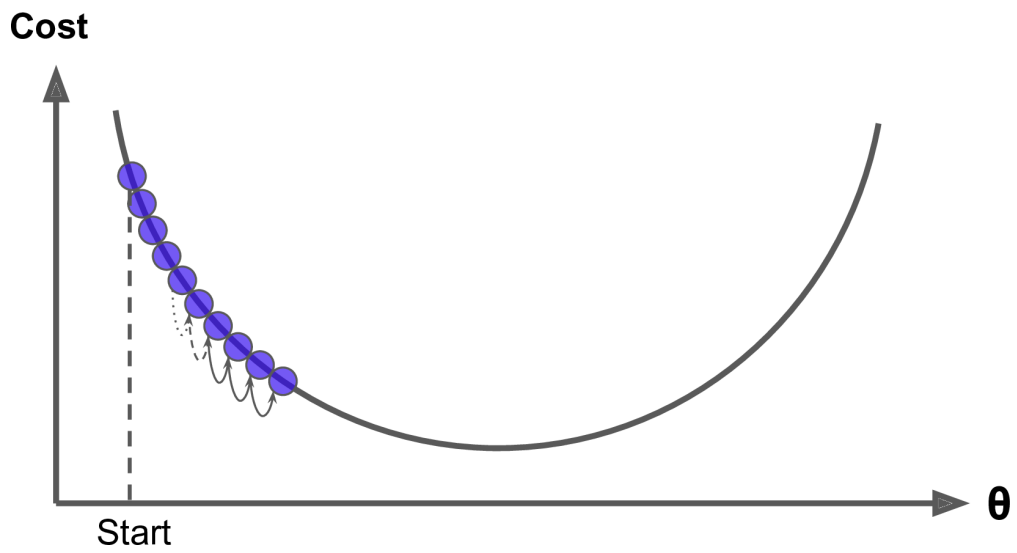
- Plot of cost over time can reveal convergence
- Stop when change of cost is smaller than $e = 10^{-3}$

Should we reduce η over iterations (aka epochs)?

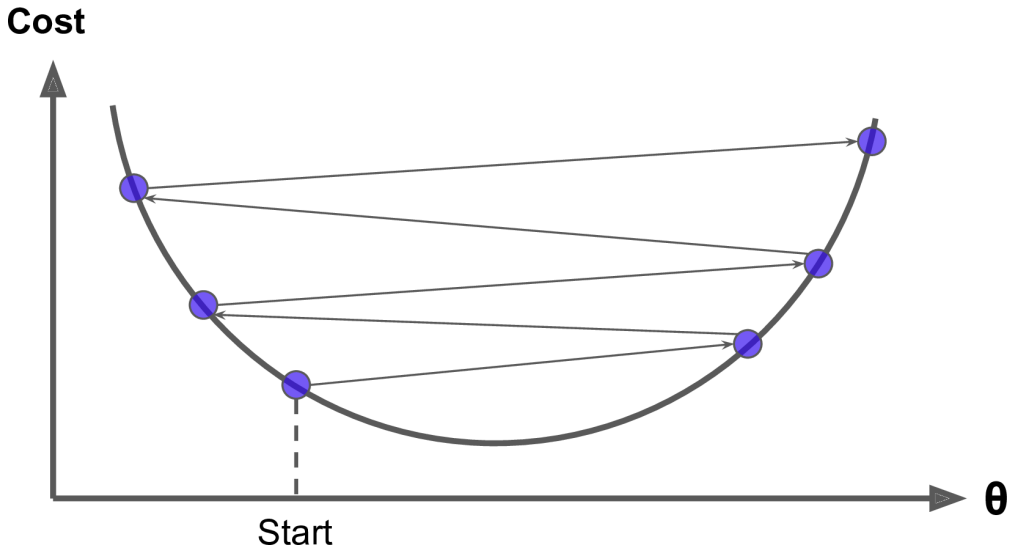
- No, as the magnitude of updates reduces automatically when approaching the minimum, due to the smaller gradient

SETTING THE LEARNING RATE

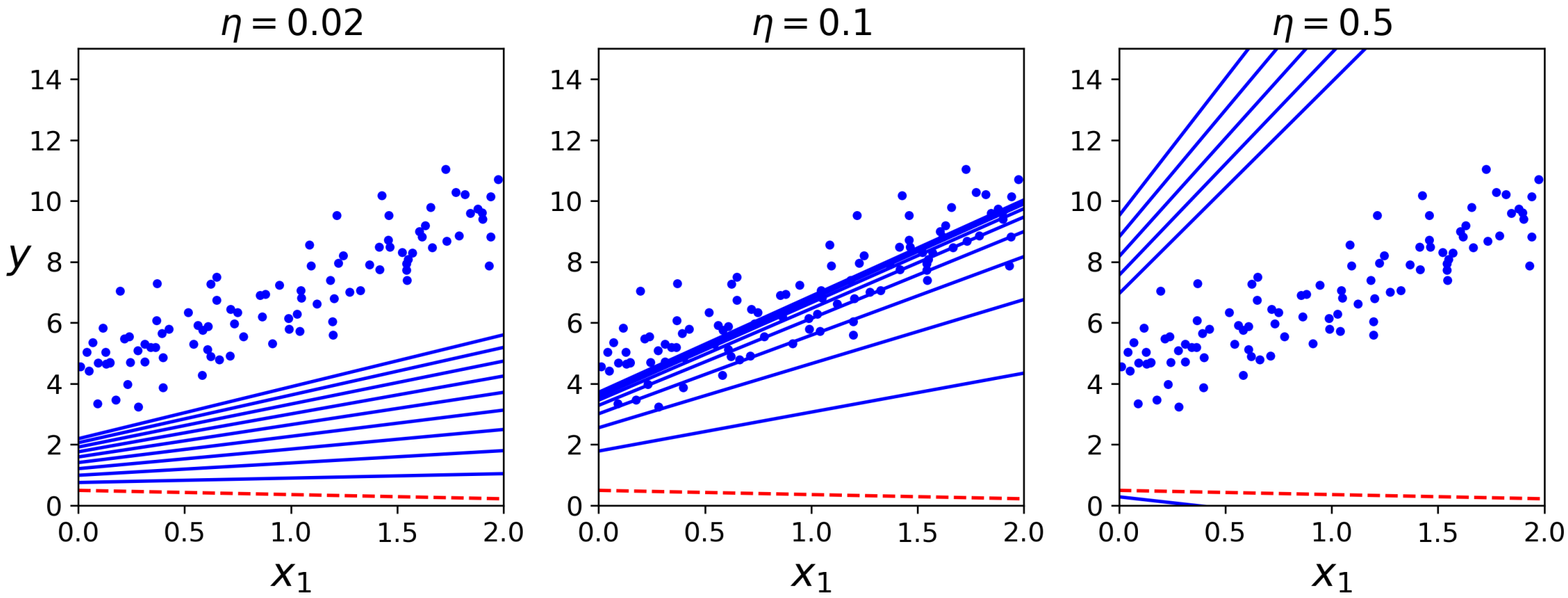
If η is too small, gradient descent will delay



If η is too large, gradient descent will not converge and may even diverge



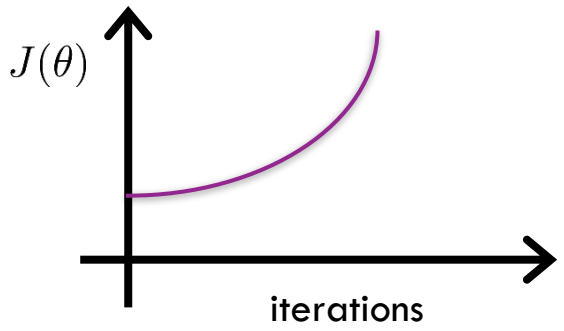
SETTING THE LEARNING RATE

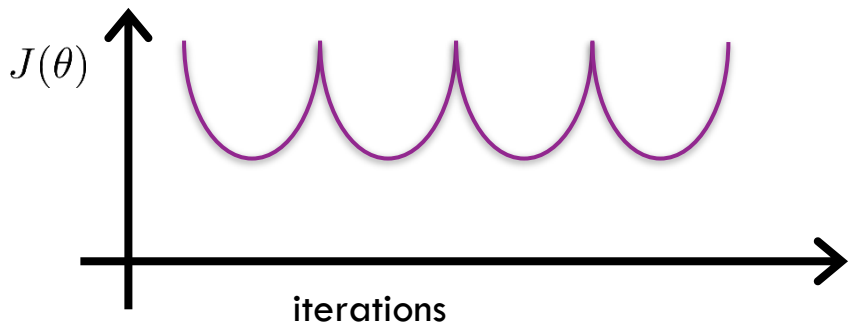


SETTING THE LEARNING RATE

In practice we experiment with different values and plot the cost over iterations to detect small/large η problems

- … 0.001, 0.003, 0.01, 0.03, 0.1, 0.3, 1, …





NORMALIZATION AND STANDARDIZATION

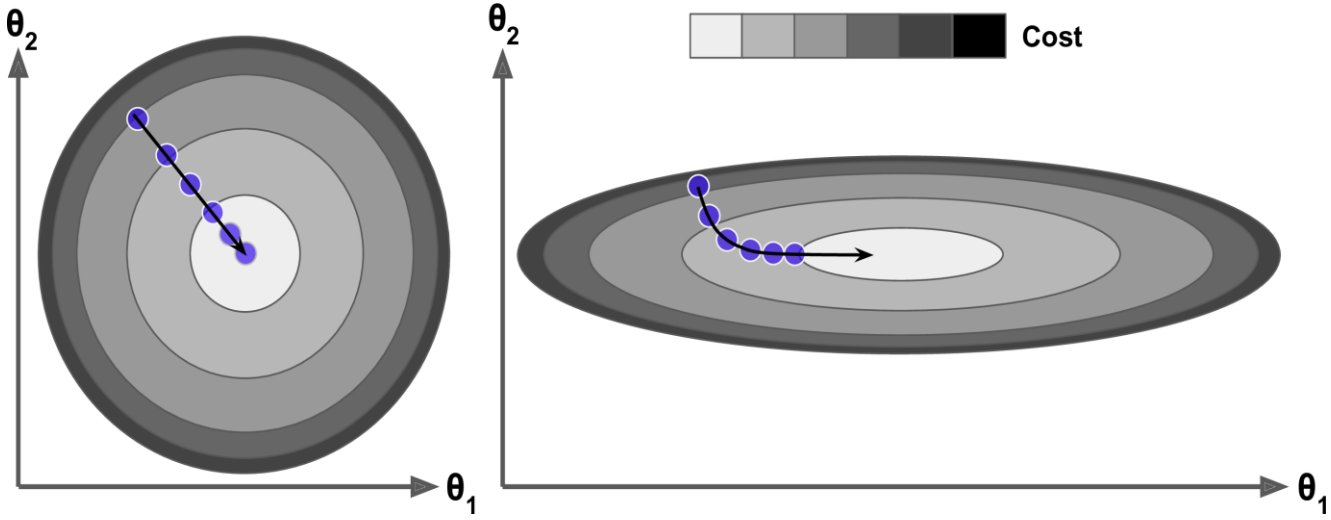
When the values of the different input variables are in very different scales, the convergence of gradient descent delays

Normalization

$$\acute{x}_i^{(j)} = \frac{x_i^{(j)} - \min(x_i)}{\max(x_i) - \min(x_i)}$$

Standardization

$$\acute{x}_i^{(j)} = \frac{x_i^{(j)} - \mu(x_i)}{\sigma(x_i)}$$



UPDATES MUST BE DONE CONCURRENTLY

Non-concurrent (wrong) updates

- $\theta_0 := \theta_0 - \eta \frac{1}{m} \sum_{i=1}^{m} [h_{\boldsymbol{\theta}}(x^{(i)}) - y^{(i)}]$
- $\forall j > 0: \ \theta_j := \theta_j - \eta \frac{1}{m} \sum_{i=1}^{m} [h_{\boldsymbol{\theta}}(x^{(i)}) - y^{(i)}] \, x_j^{(i)}$

Concurrent (correct) updates

- $\boldsymbol{\theta}' = \boldsymbol{\theta}$
- $\theta_0 := \theta_0 - \eta \frac{1}{m} \sum_{i=1}^{m} [h_{\boldsymbol{\theta}'}(x^{(i)}) - y^{(i)}]$
- $\forall j > 0: \ \theta_j := \theta_j - \eta \frac{1}{m} \sum_{i=1}^{m} [h_{\boldsymbol{\theta}'}(x^{(i)}) - y^{(i)}] \, x_j^{(i)}$

SIMPLIFYING NOTATION

In linear regression, we represent the hypothesis, h, as follows

$$h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$

To simplify the exposition, we define $x_0 = 1$, hence

$$h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n = \boldsymbol{\theta} \cdot \boldsymbol{x}$$

USING LINEAR ALGEBRA NOTATION

To simplify the exposition, we define $x_0 = 1$, hence

$$h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n = \boldsymbol{\theta} \cdot \boldsymbol{x}$$

Using linear algebra notation, we can further simplify the description of algorithms and parallelize computations

- $\boldsymbol{\theta} = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \vdots \\ \theta_n \end{bmatrix}$ $\boldsymbol{x} = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$ $h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \boldsymbol{\theta}^{\mathrm{T}} \boldsymbol{x}$

GRADIENT DESCENT IN LINEAR ALGEBRA NOTATION

Without linear algebra notation

- $\boldsymbol{\theta}' = \boldsymbol{\theta}$
- $\forall j \in \{0, \dots, n\}\colon \theta_j := \theta_j - \eta \frac{1}{m} \sum_{i=1}^{m} [\boldsymbol{\theta}' \cdot \boldsymbol{x}^{(i)} - y^{(i)}]\, x_j^{(i)}$

With linear algebra notation

- $\boldsymbol{\theta} = \boldsymbol{\theta} - \eta \frac{1}{m} \boldsymbol{X}^T (\boldsymbol{X}\boldsymbol{\theta} - \boldsymbol{y})$

NORMAL EQUATION

Set the partial derivatives to zero to find minimum

- $\frac{1}{m}\boldsymbol{X}^T(\boldsymbol{X\theta} - \boldsymbol{y}) = \vec{0}$

Closed form solution

- $\boldsymbol{\theta} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y}$

STOCHASTIC GRADIENT DESCENT

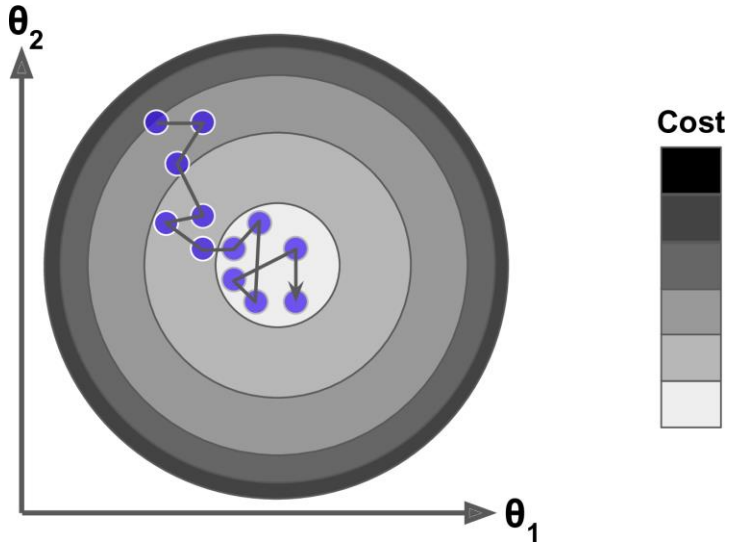
Update the parameters after seeing each example $(\boldsymbol{x}, y)$ in different random (stochastic) order per training set pass

Without linear algebra notation

- $\boldsymbol{\theta}' = \boldsymbol{\theta}$
- $\forall j \in \{0, \ldots, n\}$: $\theta_j := \theta_j - \eta\,[\boldsymbol{\theta}' \cdot \boldsymbol{x} - y]x_j$

With linear algebra notation

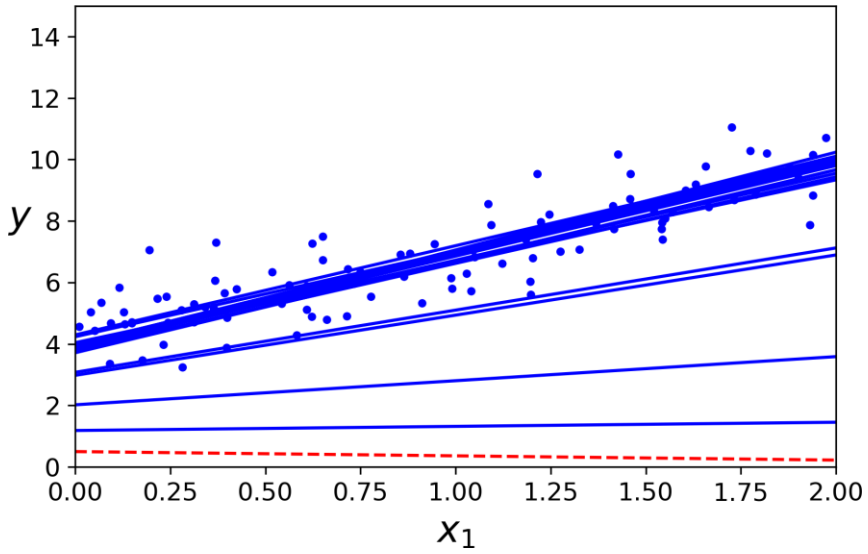
- $\boldsymbol{\theta} = \boldsymbol{\theta} - \eta\,\boldsymbol{x}(\boldsymbol{\theta}^T\boldsymbol{x} - y)$

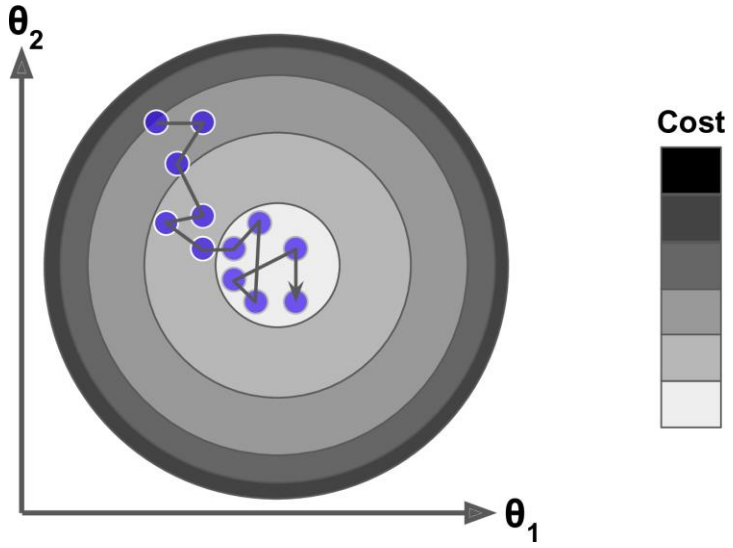


STOCHASTIC GRADIENT DESCENT

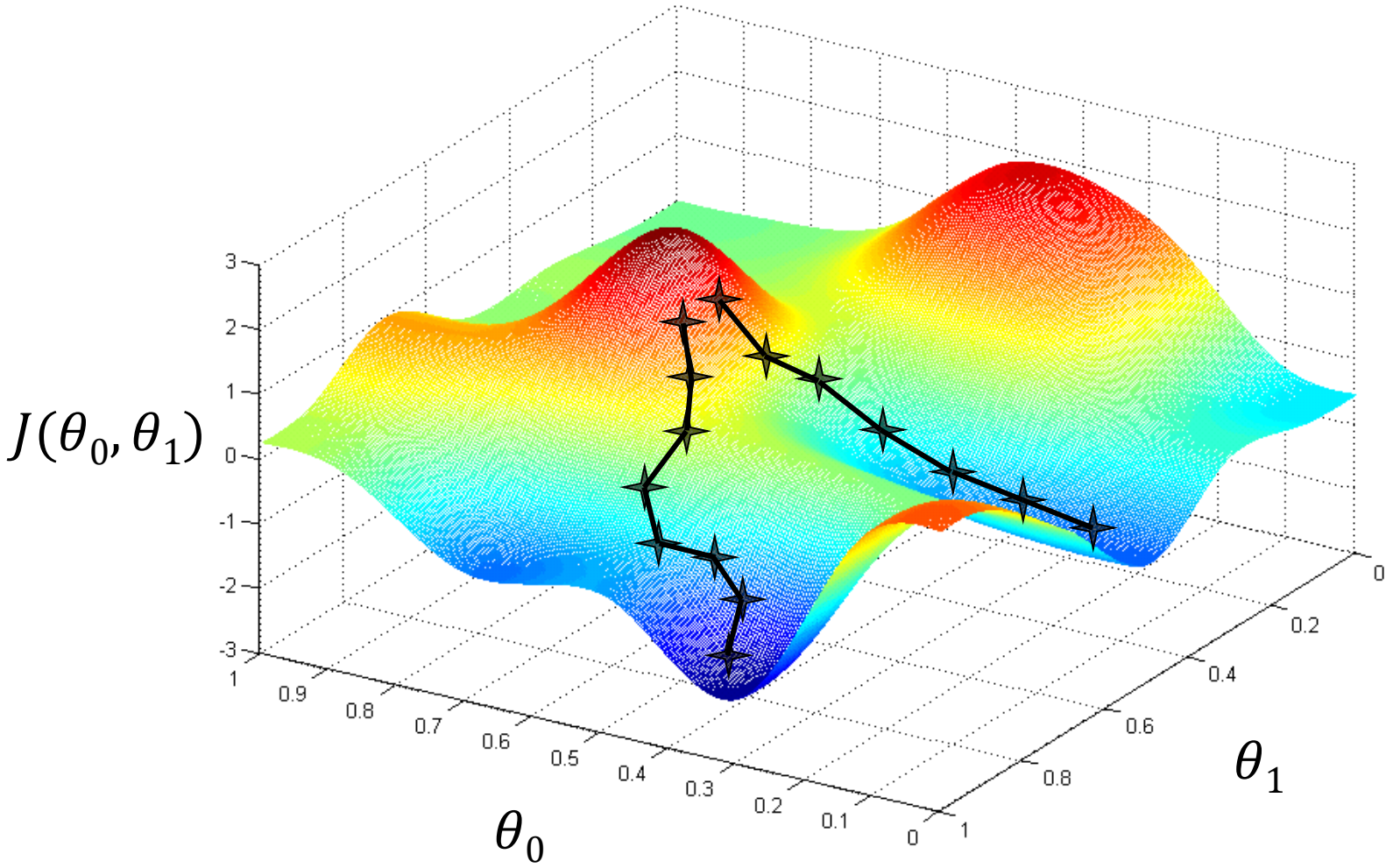
Gradually reduce the learning rate

- The function that determines the learning rate at each iteration is called the learning schedule



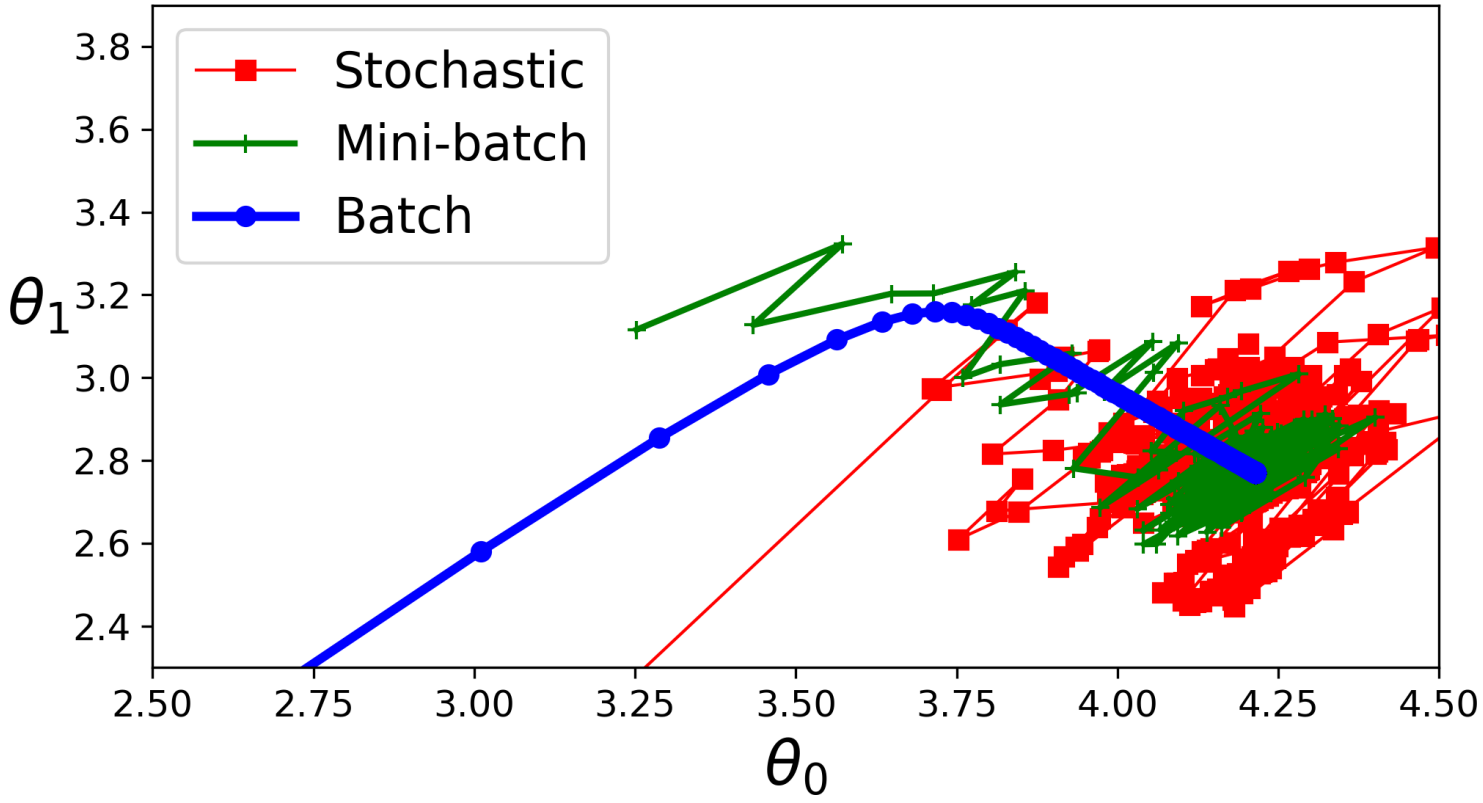


GD IN ARBITRARY FUNCTION OF TWO PARAMETERS



BATCH, STOCHASTIC AND MINI-BATCH GD

In mini-batch gradient descent, weights are updated after a batch of training examples



OUTLINE

Linear regression

Logistic regression

CLASSIFICATION

The output takes discrete values

- Spam vs legitimate e-mails
- Fraudulent vs normal credit card transactions
- Benign vs malignant tumor

Notation

- $y \in \{0,1\}$: binary classification, $0/1 =$ negative/positive class
- $y \in \{0, 1, 2, \dots, c-1\}$: multi-class classification

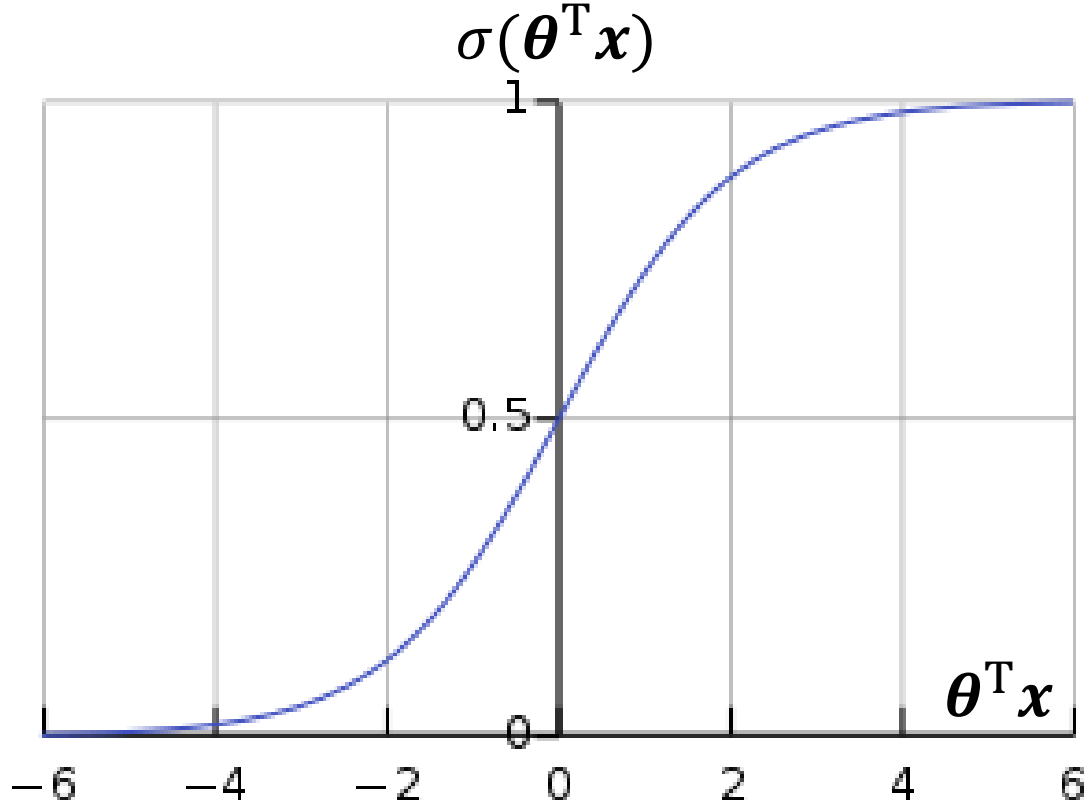
REPRESENTATION

A logistic regression model first computes $z = \sum_{i=0}^{n} \theta_i x_i = \boldsymbol{\theta}^{\mathrm{T}} \boldsymbol{x}$

Then passes z through the sigmoid function $\sigma(z) = \frac{1}{1+e^{-z}}$

$$p(y = 1|\boldsymbol{x}) = \sigma\left(\boldsymbol{\theta}^{\mathrm{T}} \boldsymbol{x}\right)$$

$$p(y = 0|\boldsymbol{x}) = 1 - \sigma\left(\boldsymbol{\theta}^{\mathrm{T}} \boldsymbol{x}\right)$$

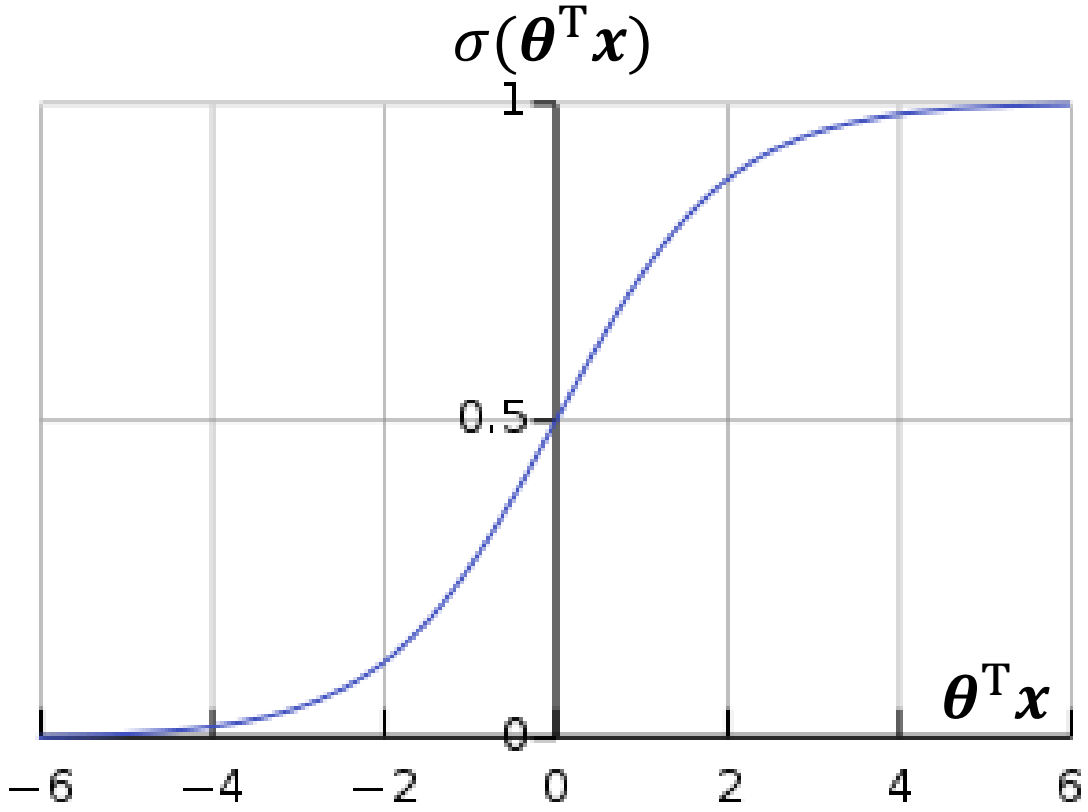


REPRESENTATION

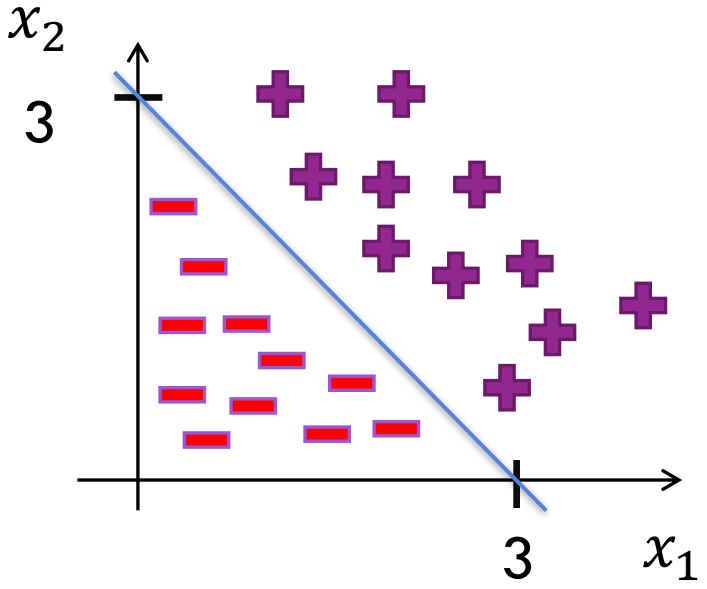
$$p(y = 1|\boldsymbol{x}) = \sigma(\boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{x})$$

$$y = \begin{cases} 1, \text{if } p(y = 1|\boldsymbol{x}) \geq 0.5 \\ 0, \text{otherwise} \end{cases}$$

$$y = \begin{cases} 1, \text{if } \boldsymbol{\theta}^{T}\boldsymbol{x} \geq 0 \\ 0, \text{otherwise} \end{cases}$$



THE DECISION BOUNDARY = A HYPERPLANE



Consider a learning task with two features x_1 and x_2

Let $\theta_0 = -3$ and $\theta_1 = \theta_2 = 1$

Classifier outputs 1 if

$$(-3) + x_1 + x_2 \geq 0 \Rightarrow x_1 + x_2 \geq 3$$

Classifier outputs 0 if

$$(-3) + x_1 + x_2 < 0 \Rightarrow x_1 + x_2 < 3$$

EVALUATION

How good is a given hypothesis?

A loss function $L(h_{\boldsymbol{\theta}}(\boldsymbol{x}), y)$ measures the difference between the value of the output variable y of a training example $(\boldsymbol{x}, y)$ and th output of the hypothesis given $\boldsymbol{x}$, $h_{\boldsymbol{\theta}}(\boldsymbol{x})$

A cost function iterates over the training corpus $\left(\boldsymbol{x}^{(i)}, y^{(i)}\right)$, $i = 1 \ldots m$ and measures the average loss between the ground truth $y^{(i)}$ and the output of the hypothesis $h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right)$

THE CROSS-ENTROPY LOSS

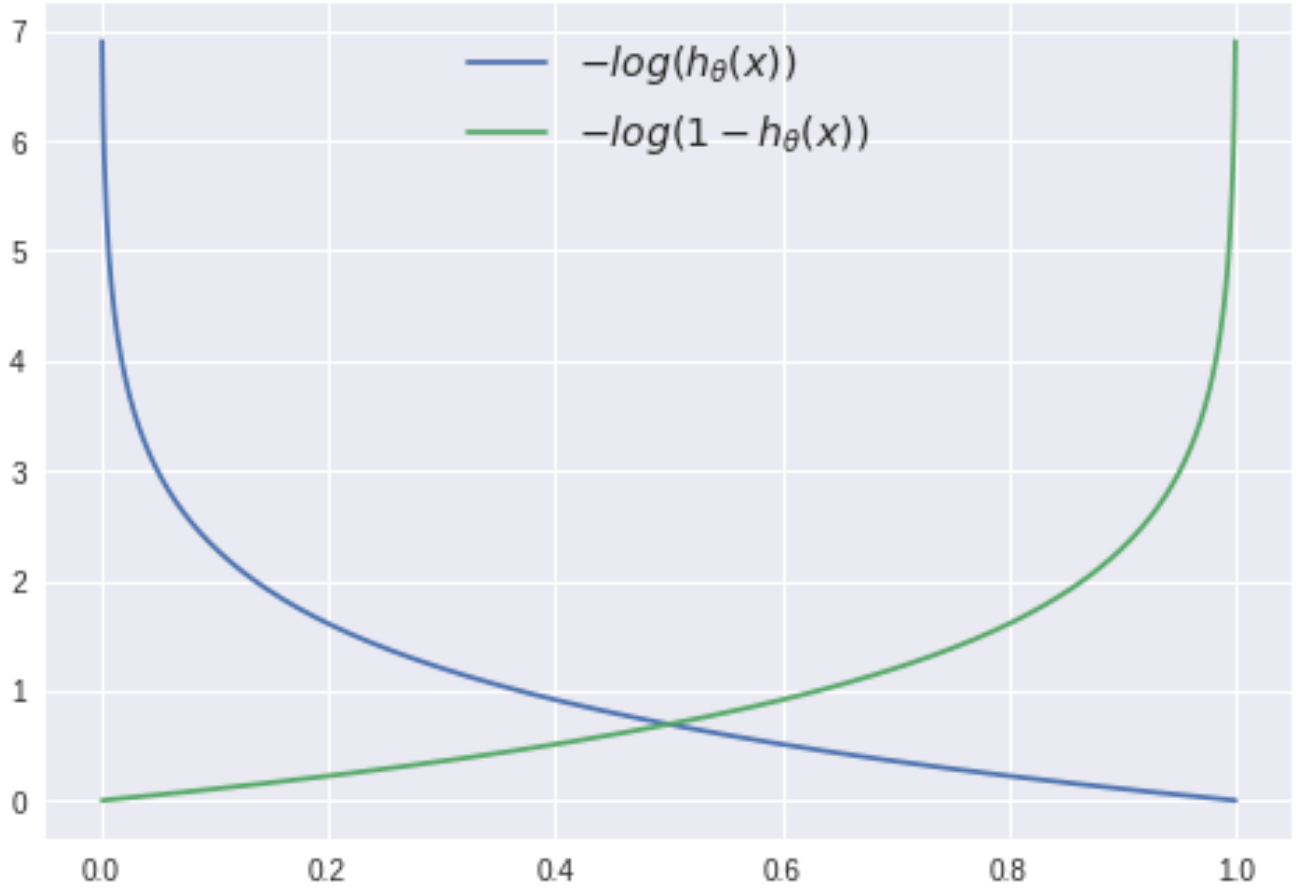
$y \in \{0,1\}$ is the correct output, and $h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sigma\left(\boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{x}\right)$ is the output of the logistic regression classifier for example $(\boldsymbol{x}, y)$

$$L_{CE}(h_{\boldsymbol{\theta}}(\boldsymbol{x}), y) = -\log p(y|\boldsymbol{x})$$

$$L_{CE}(h_{\boldsymbol{\theta}}(\boldsymbol{x}), y) = \begin{cases} -\log h_{\boldsymbol{\theta}}(\boldsymbol{x}), \text{if } y = 1 \\ -\log(1 - h_{\boldsymbol{\theta}}(\boldsymbol{x})), \text{if } y = 0 \end{cases}$$

$$L_{CE}(h_{\boldsymbol{\theta}}(\boldsymbol{x}), y) = -[y \log h_{\boldsymbol{\theta}}(\boldsymbol{x}) + (1 - y) \log(1 - h_{\boldsymbol{\theta}}(\boldsymbol{x}))]$$

THE CROSS-ENTROPY LOSS



$$L_{CE}(h_{\boldsymbol{\theta}}(\boldsymbol{x}), y) =$$

- $-\log h_{\boldsymbol{\theta}}(\boldsymbol{x})$, if $y = 1$
- $-\log(1 - h_{\boldsymbol{\theta}}(\boldsymbol{x}))$, if $y = 0$

THE COST FUNCTION OF LOGISTIC REGRESSION

$$J_{CE}(\boldsymbol{\theta}) = \frac{1}{m}\sum_{i=1}^{m} L_{CE}\left(h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}), y^{(i)}\right)$$

$$J_{CE}(\boldsymbol{\theta}) = -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)} \log h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}) + \left(1 - y^{(i)}\right) \log\left(1 - h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)})\right)\right]$$

$$J_{CE}(\boldsymbol{\theta}) = -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)} \log \sigma(\boldsymbol{\theta} \cdot \boldsymbol{x}^{(i)}) + \left(1 - y^{(i)}\right) \log\left(1 - \sigma(\boldsymbol{\theta} \cdot \boldsymbol{x}^{(i)})\right)\right]$$

GRADIENT DESCENT IN LOGISTIC REGRESSION

Without linear algebra notation

- $\boldsymbol{\theta}' = \boldsymbol{\theta}$
- $\forall j \in \{0, \dots, n\}$: $\theta_j := \theta_j - \eta \frac{1}{m} \sum_{i=1}^{m} [\sigma(\boldsymbol{\theta}' \cdot \boldsymbol{x}^{(i)}) - y^{(i)}] \, x_j^{(i)}$

With linear algebra notation

- $\boldsymbol{\theta} = \boldsymbol{\theta} - \eta \frac{1}{m} \boldsymbol{X}^T (\sigma(\boldsymbol{X}\boldsymbol{\theta}) - \boldsymbol{y})$

MULTICLASS CLASSIFICATION

E-mail classification

- work ($y = 0$), friends ($y = 1$), family ($y = 2$)

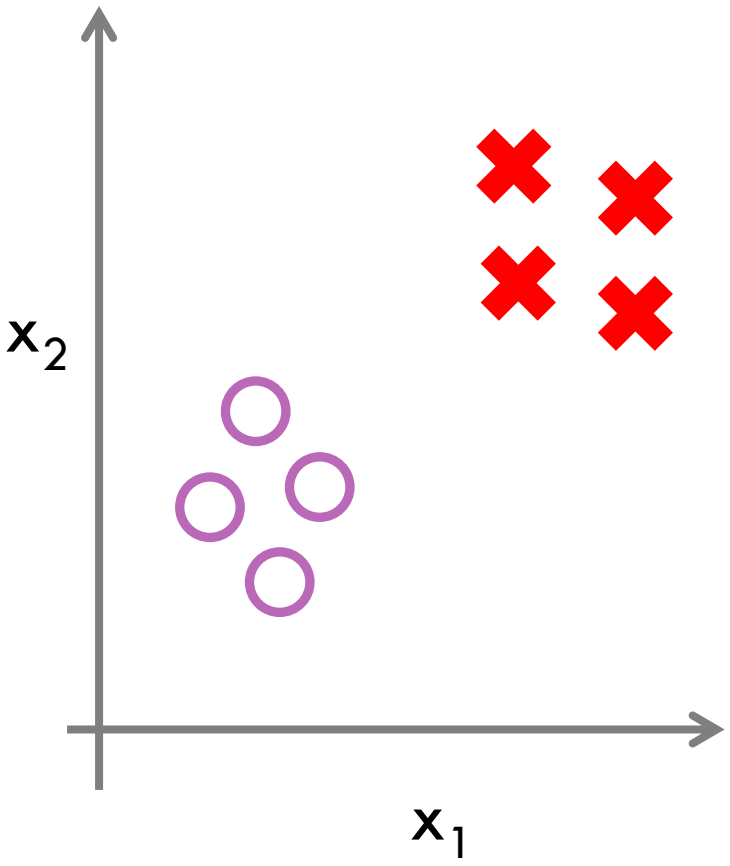
Medical diagnosis

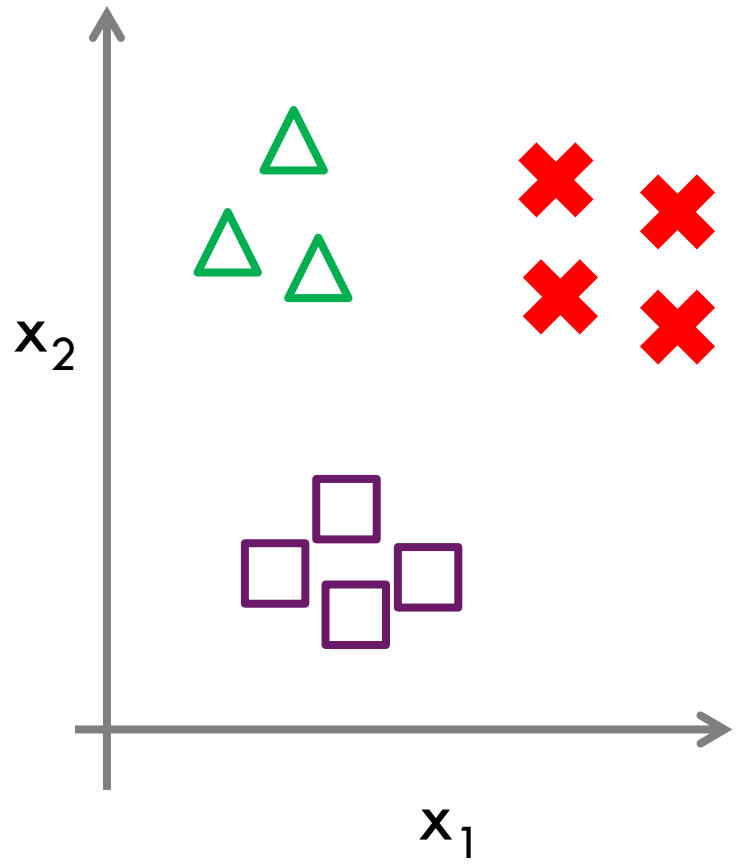
- healthy ($y = 0$), cold ($y = 1$), flu ($y = 2$)

Weather forecast

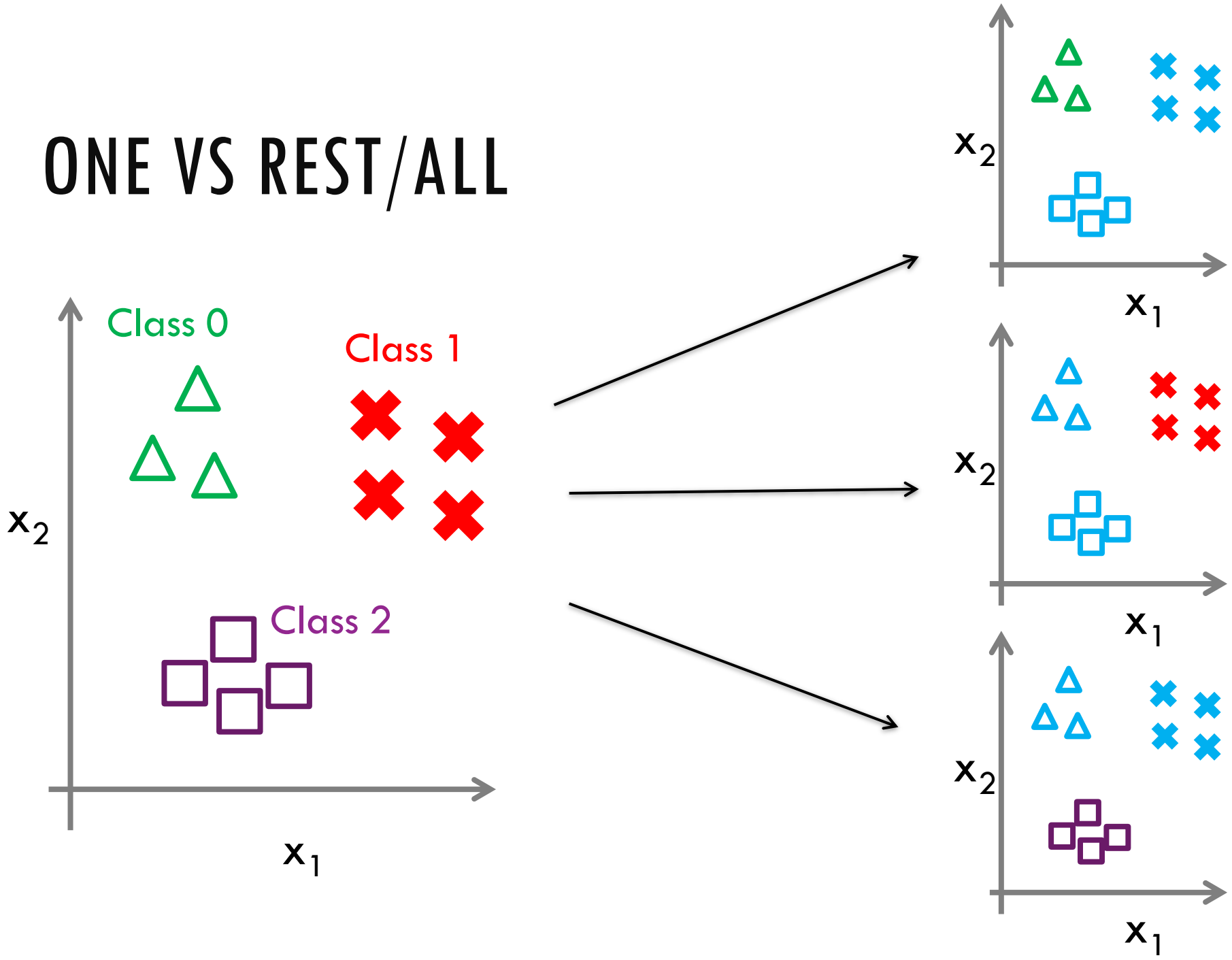
- sun ($y = 0$), clouds ($y = 1$), rain ($y = 2$) , snow ($y = 3$)

BINARY VS MULTICLASS CLASSIFICATION

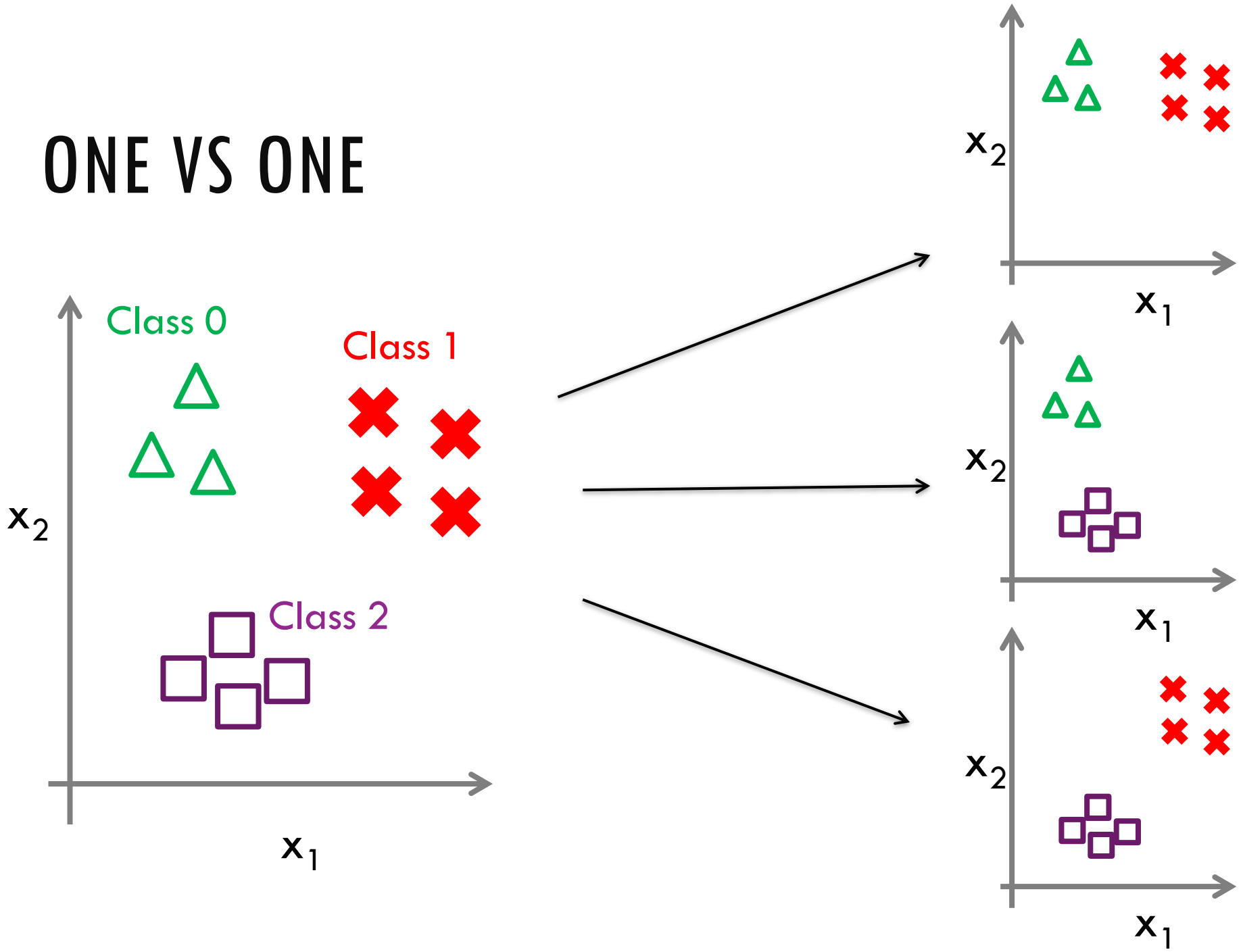




ONE VS REST/ALL



ONE VS ONE

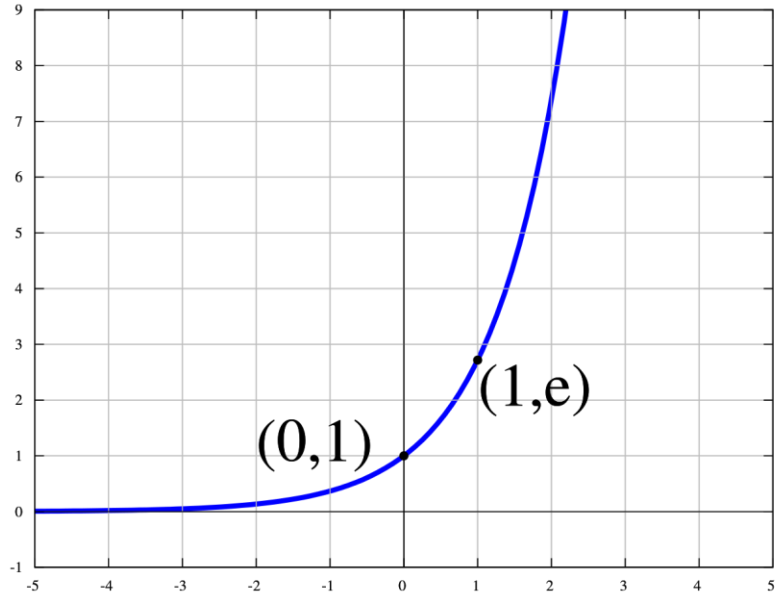


MULTINOMIAL LOGISTIC REGRESSION

For each class $k = 1 \ldots c$, learn a separate set of parameters $\boldsymbol{\theta}_k$

Output a probability distribution using the softmax function

- $p(y = k|\boldsymbol{x}) = \frac{\exp(\boldsymbol{\theta}_k \cdot \boldsymbol{x})}{\sum_{l=1}^{c} \exp(\boldsymbol{\theta}_l \cdot \boldsymbol{x})}$



$$\boldsymbol{\Theta} = [\boldsymbol{\theta}_1 \ldots \boldsymbol{\theta}_c],\ h_{\boldsymbol{\Theta}}(\boldsymbol{x}) = \left(\frac{\exp(\boldsymbol{\theta}_1 \cdot \boldsymbol{x})}{\sum_{l=1}^{c} \exp(\boldsymbol{\theta}_l \cdot \boldsymbol{x})}, \ldots, \frac{\exp(\boldsymbol{\theta}_c \cdot \boldsymbol{x})}{\sum_{l=1}^{c} \exp(\boldsymbol{\theta}_l \cdot \boldsymbol{x})} \right)$$

THE COST OF MULTINOMIAL LOGISTIC REGRESSION

$$J_{CE}(\boldsymbol{\Theta}) = \frac{1}{m}\sum_{i=1}^{m} L_{CE}\left(h_{\boldsymbol{\Theta}}(\boldsymbol{x}^{(i)}), y^{(i)}\right)$$

$$J_{CE}(\boldsymbol{\Theta}) = -\frac{1}{m}\sum_{i=1}^{m}\sum_{k=1}^{c}\left[y^{(i)} = k\right]\log p(y = k|\boldsymbol{x})$$

$$J_{CE}(\boldsymbol{\Theta}) = -\frac{1}{m}\sum_{i=1}^{m}\sum_{k=1}^{c}\left[y^{(i)} = k\right]\log \frac{\exp(\boldsymbol{\theta}_k \cdot \boldsymbol{x})}{\sum_{l=1}^{c}\exp(\boldsymbol{\theta}_l \cdot \boldsymbol{x})}$$

GD IN MULTINOMIAL LOGISTIC REGRESSION

Without linear algebra notation

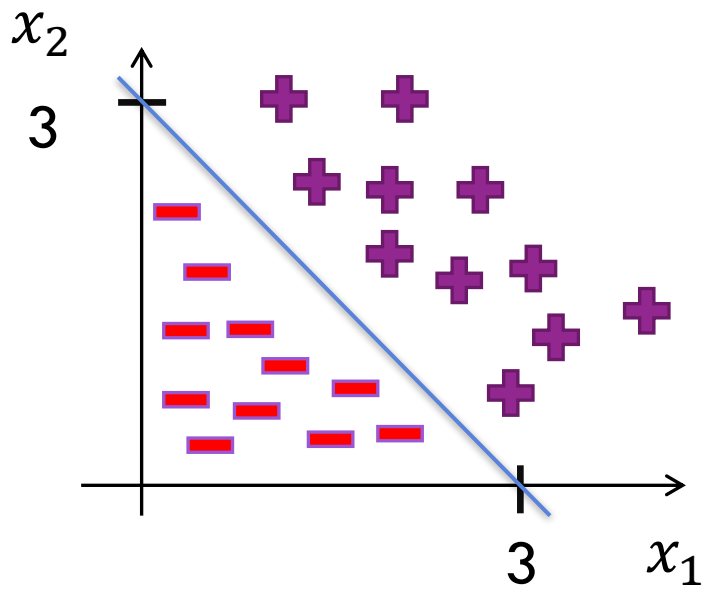
- $\boldsymbol{\Theta}' = \boldsymbol{\Theta}$
- $\forall k \in \{1, \dots, c\}, j \in \{0, \dots, n\},:$

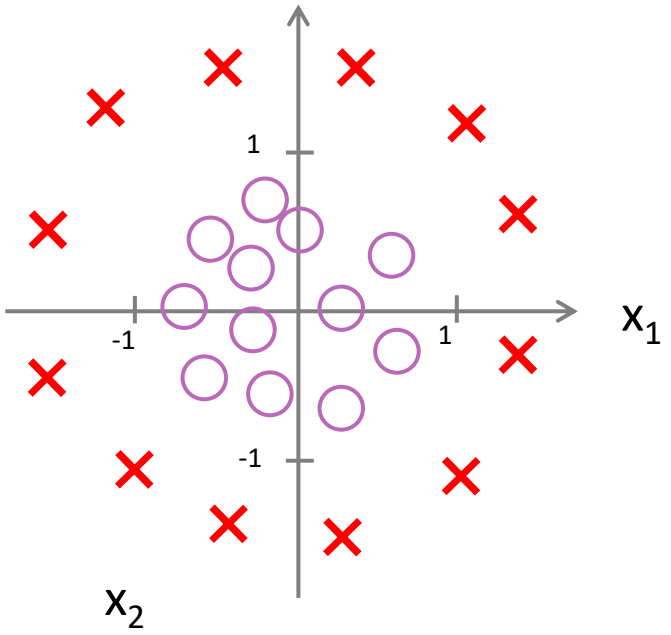
$$\theta_{kj} \coloneqq \theta_{kj} - \eta \frac{1}{m} \sum_{i=1}^{m} [\frac{\exp(\boldsymbol{\theta'}_k \cdot \boldsymbol{x})}{\sum_{l=1}^{c} \exp(\boldsymbol{\theta'}_l \cdot \boldsymbol{x})} - y_k^{(i)}] \, x_j^{(i)}$$

NON-LINEAR DECISION BOUNDARIES

$\theta_1 = 1, \theta_2 = 1, \theta_0 = -3$

Can we model this with LR?





NON-LINEAR DECISION BOUNDARIES

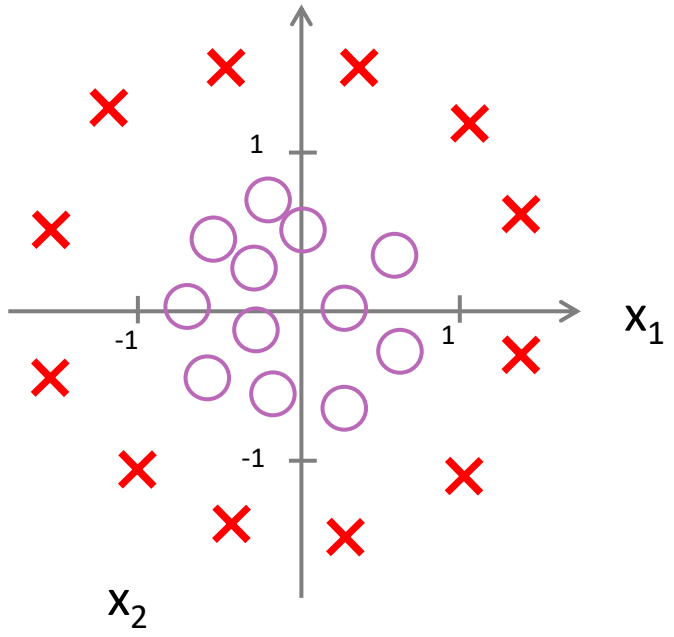
What if we add features

- $x_3 = x_1^2$
- $x_4 = x_2^2$

Consider parameters

- $\theta_1 = \theta_2 = 0$
- $\theta_3 = \theta_4 = 1$
- $\theta_0 = -1$

Can we model this with LR?



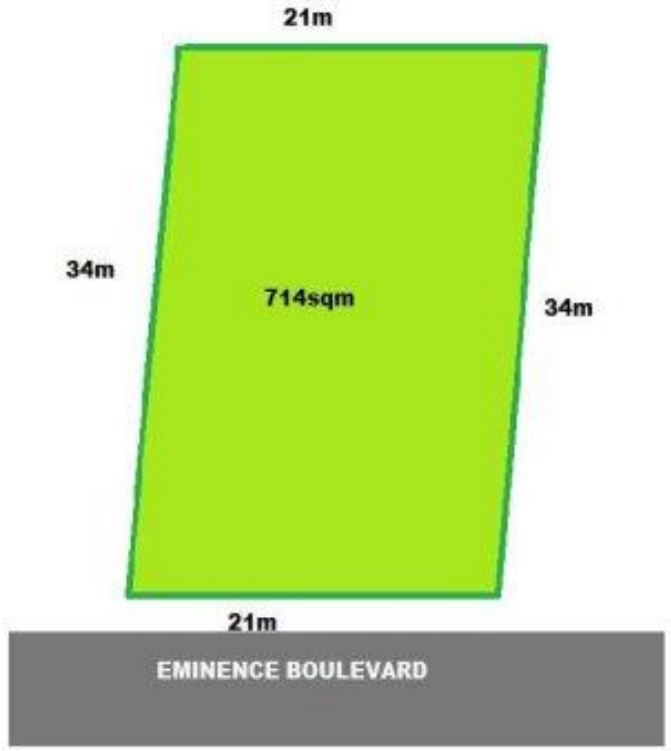
PROPERTY VALUATION

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2$$

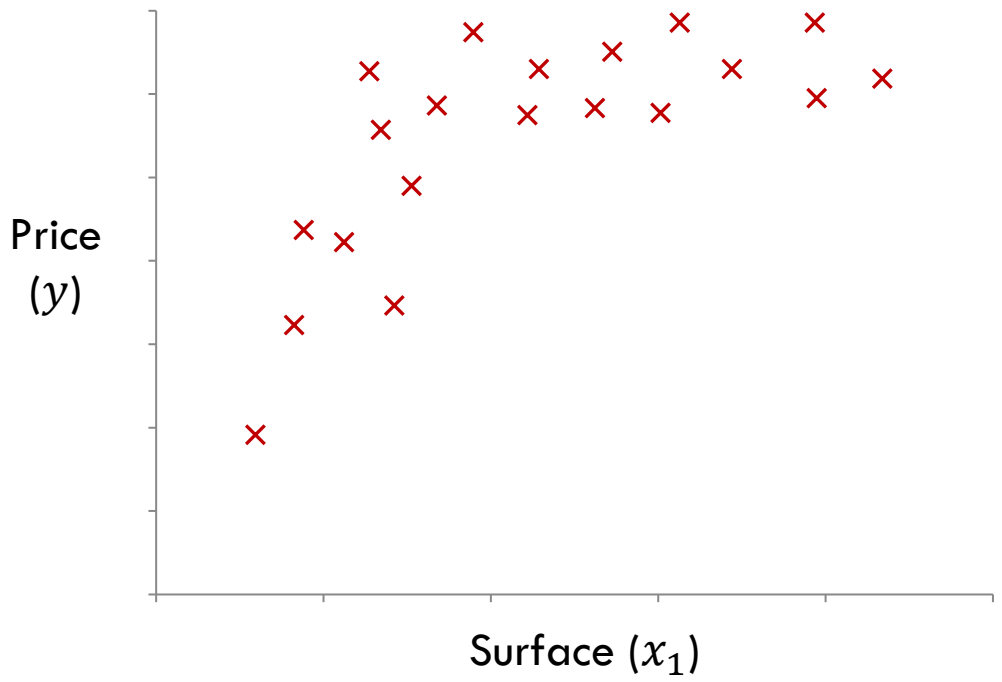
- x_1: face
- x_2: depth

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3$$

- x_3: surface = face x depth



POLYNOMIAL REGRESSION

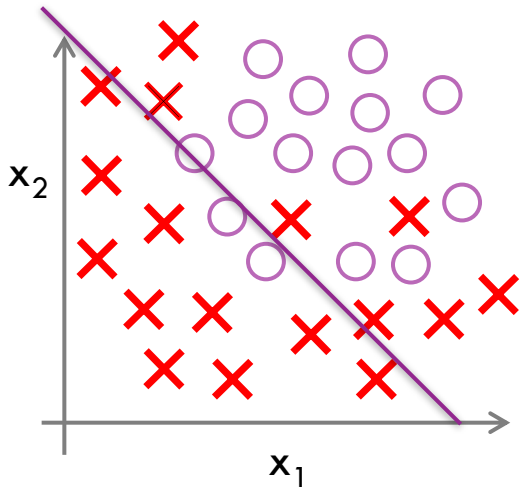


$$h_\theta(x) = \theta_0 + \theta_1 x_1$$

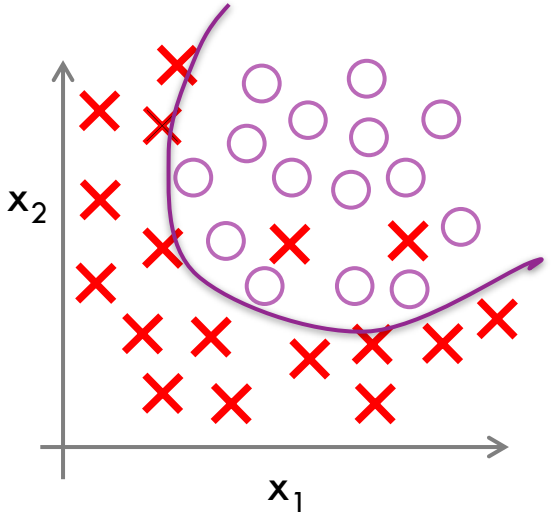
$$x_2 = x_1^2, x_3 = x_1^3$$

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3$$

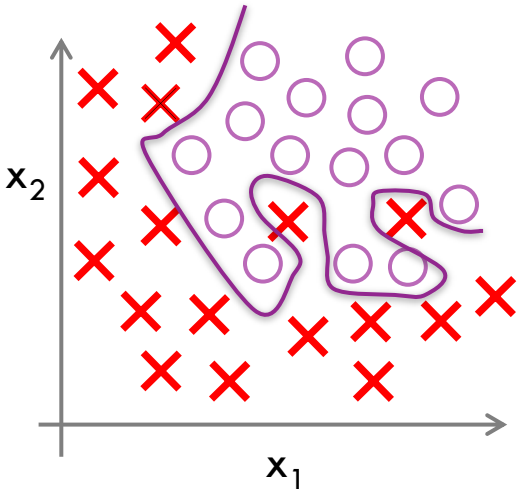
UNDERFITTING AND OVERFITTING



$$\theta_0 + \theta_1 x_1 + \theta_2 x_2$$

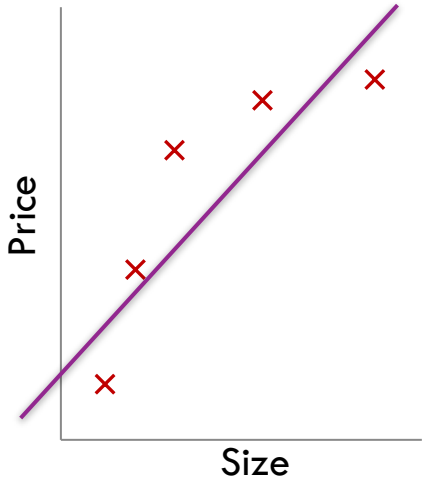


$$\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2 + \theta_5 x_1 x_2$$

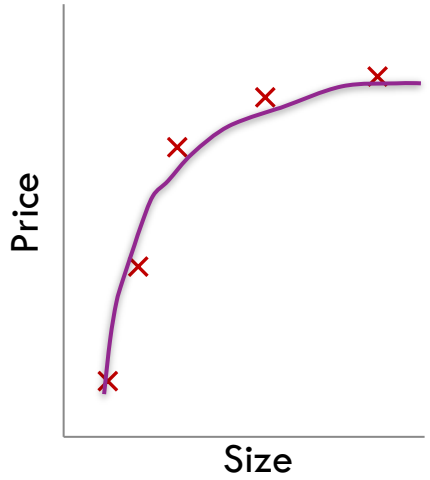


$$\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2 + \theta_5 x_1 x_2 + \theta_6 x_1^2 x_2 + \theta_7 x_1^2 x_2^2 + \theta_8 x_1 x_2^2 + \theta_9 x_1^3 + \cdots$$

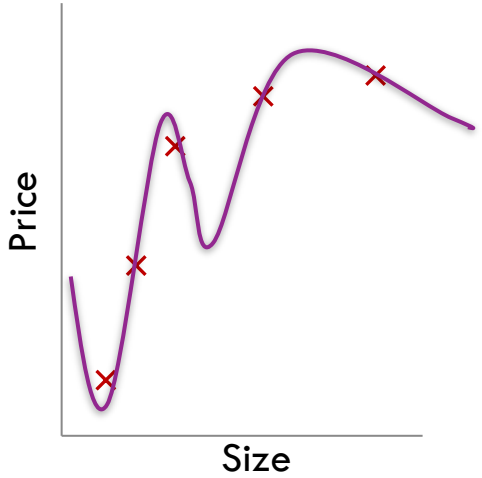
UNDERFITTING AND OVERFITTING



$$\theta_0 + \theta_1 x$$



$$\theta_0 + \theta_1 x + \theta_2 x^2$$



$$\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$$

DEALING WITH OVERFITTING

Feature selection

- Manually or automatically select a subset of the available features

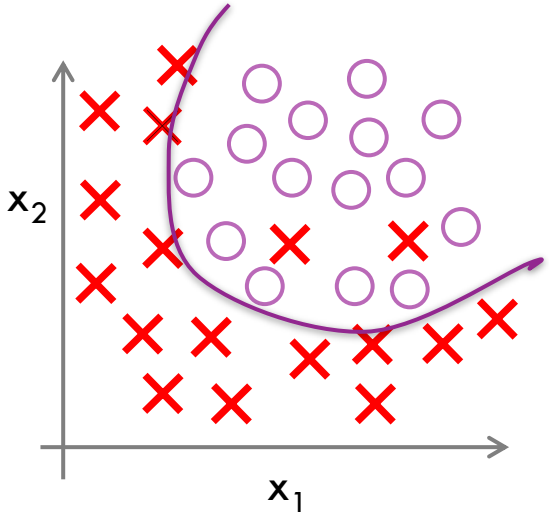
Regularization

- Keep all features, but penalize large parameter values
- Works well when many feature contribute by a little to the prediction

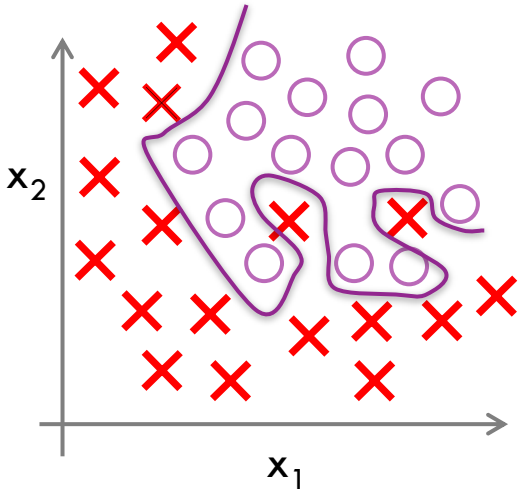
Early stopping

REGULARIZATION

What will happen if we shrink $\theta_6, \theta_7, \ldots, \theta_{12}$?



$$\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2 + \theta_5 x_1 x_2$$



$$\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2 + \theta_5 x_1 x_2 + \theta_6 x_1^2 x_2 + \theta_7 x_1^2 x_2^2 + \theta_8 x_1 x_2^2 + \theta_9 x_1^3 + \cdots$$

REGULARIZATION

$$J_{CE}(\boldsymbol{\theta}) = \frac{1}{m}\sum_{i=1}^{m} L_{CE}\left(h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}), y^{(i)}\right) + \alpha R(\boldsymbol{\theta})$$

L2 regularization, aka ridge (logistic) regression

- $R(\boldsymbol{\theta}) = \sum_{j=1}^{n} \theta_j^2$, $a = \frac{\lambda}{2m}$, λ is the regularization hyper-parameter

L1 regularization, aka lasso (logistic) regression

- $R(\boldsymbol{\theta}) = \sum_{j=1}^{n} |\theta_j|$, $a = \frac{\lambda}{m}$, λ is the regularization hyper-parameter
- Leads to sparse solutions = few non-zero weights

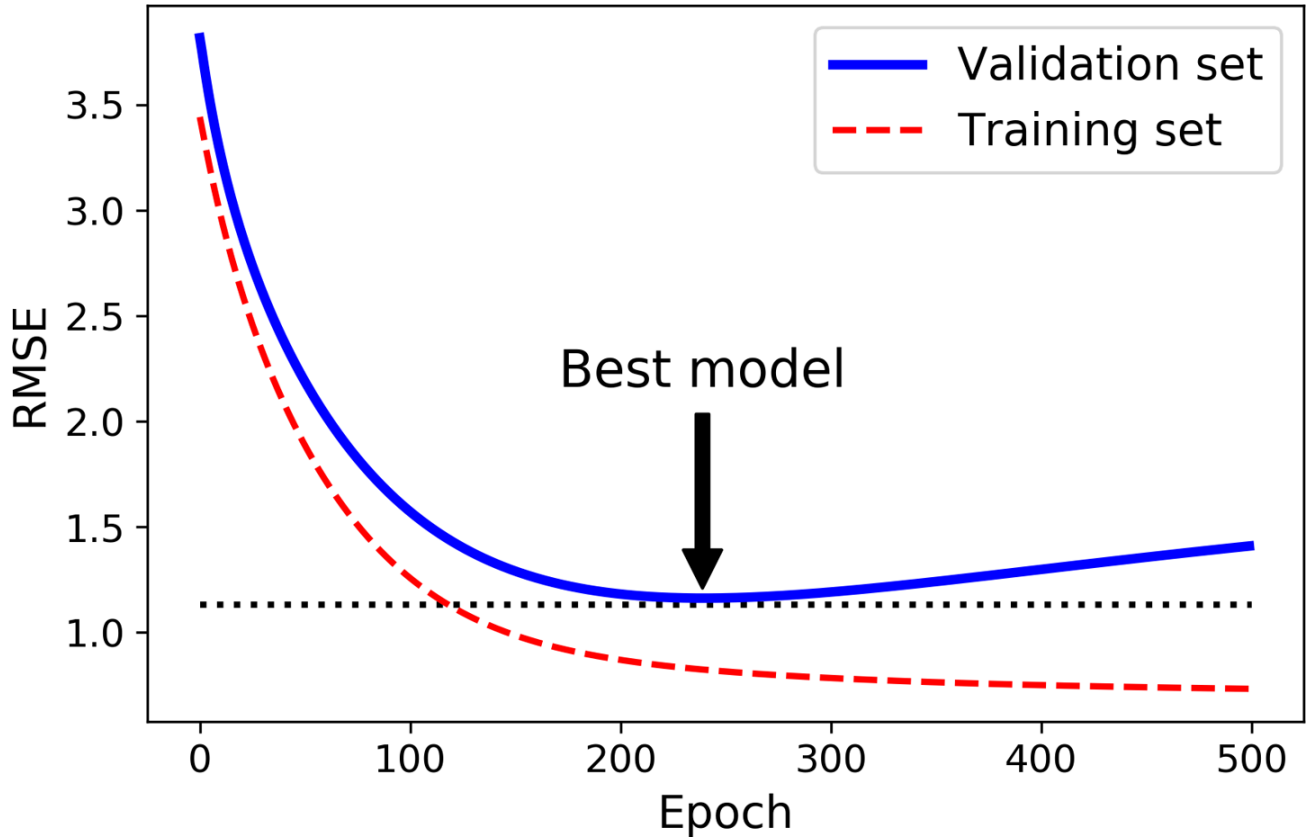
GD IN L2 REGULARIZED LOGISTIC REGRESSION

Without linear algebra notation

- $\boldsymbol{\theta}' = \boldsymbol{\theta}$
- $\theta_0 := \theta_0 - \eta \frac{1}{m} \sum_{i=1}^{m} \left[\sigma\left(\boldsymbol{\theta}' \cdot \boldsymbol{x}^{(i)}\right) - y^{(i)} \right]$
- $\forall j \in \{1, \ldots, n\}$: $\theta_j := \theta_j - \eta \left(\frac{1}{m} \sum_{i=1}^{m} \left[\sigma\left(\boldsymbol{\theta}' \cdot \boldsymbol{x}^{(i)}\right) - y^{(i)} \right] x_j^{(i)} + 2\alpha\theta_j \right)$

EARLY STOPPING

Stop training as soon as the validation error reaches a minimum



INTERPRETABILITY OF LINEAR MODELS

Why does a classifier output particular decisions?

Medicine, autonomous vehicles, GDPR, …

Size of a weight is indicative of the importance of the corresponding feature

READINGS

Lecture notes, Chapter 2