

Announcement

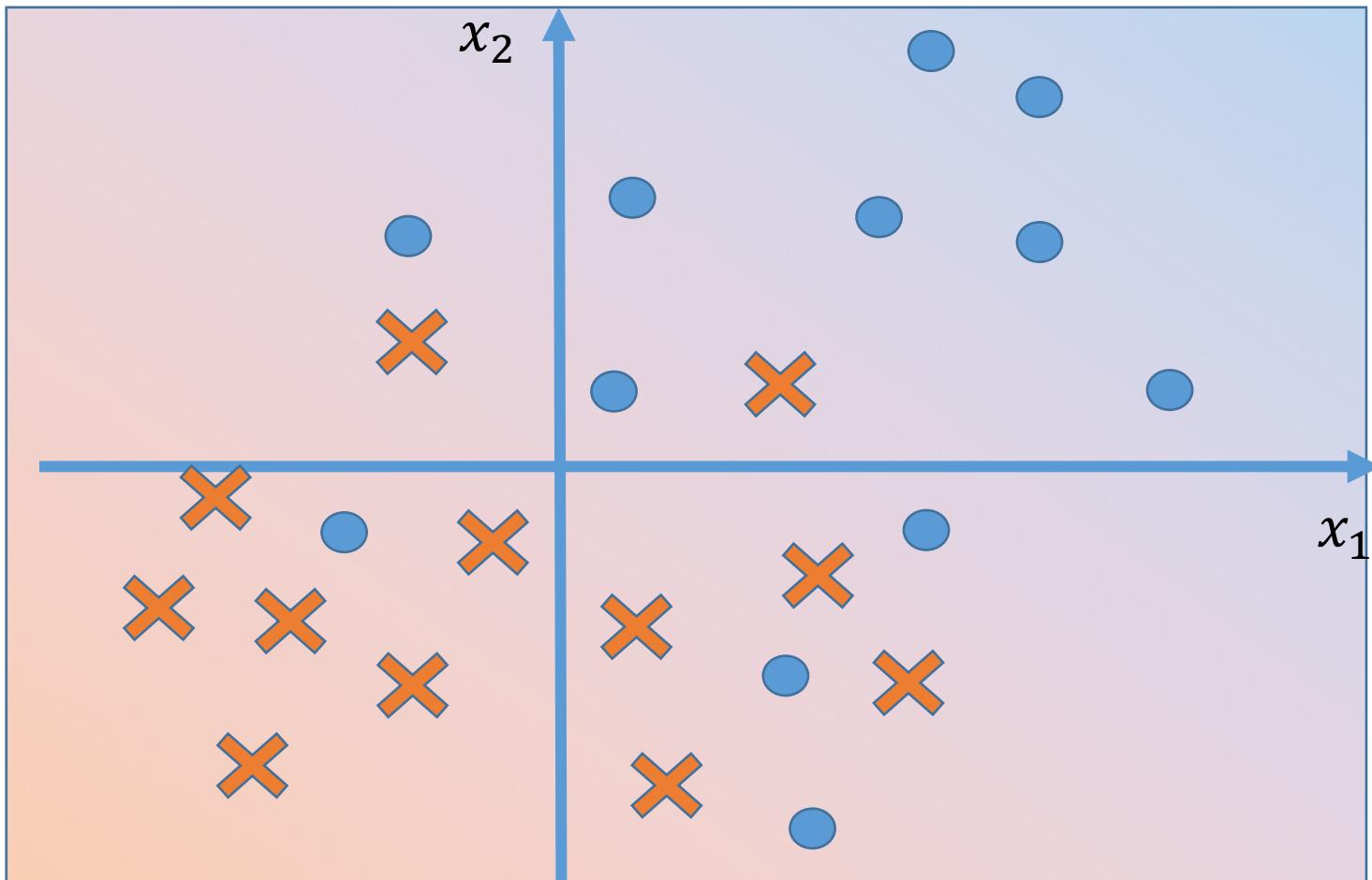
❖ Problem set 2. Problem 2

$$J(\theta) = - \sum_{n=1}^N [y_n \log h_{\theta}(x_n) + (1 - y_n) \log (1 - h_{\theta}(x_n))]$$

$$h_{\theta}(x_n) = \sigma(\theta^T x) = \frac{1}{1 + \exp(-\theta^T x)}$$

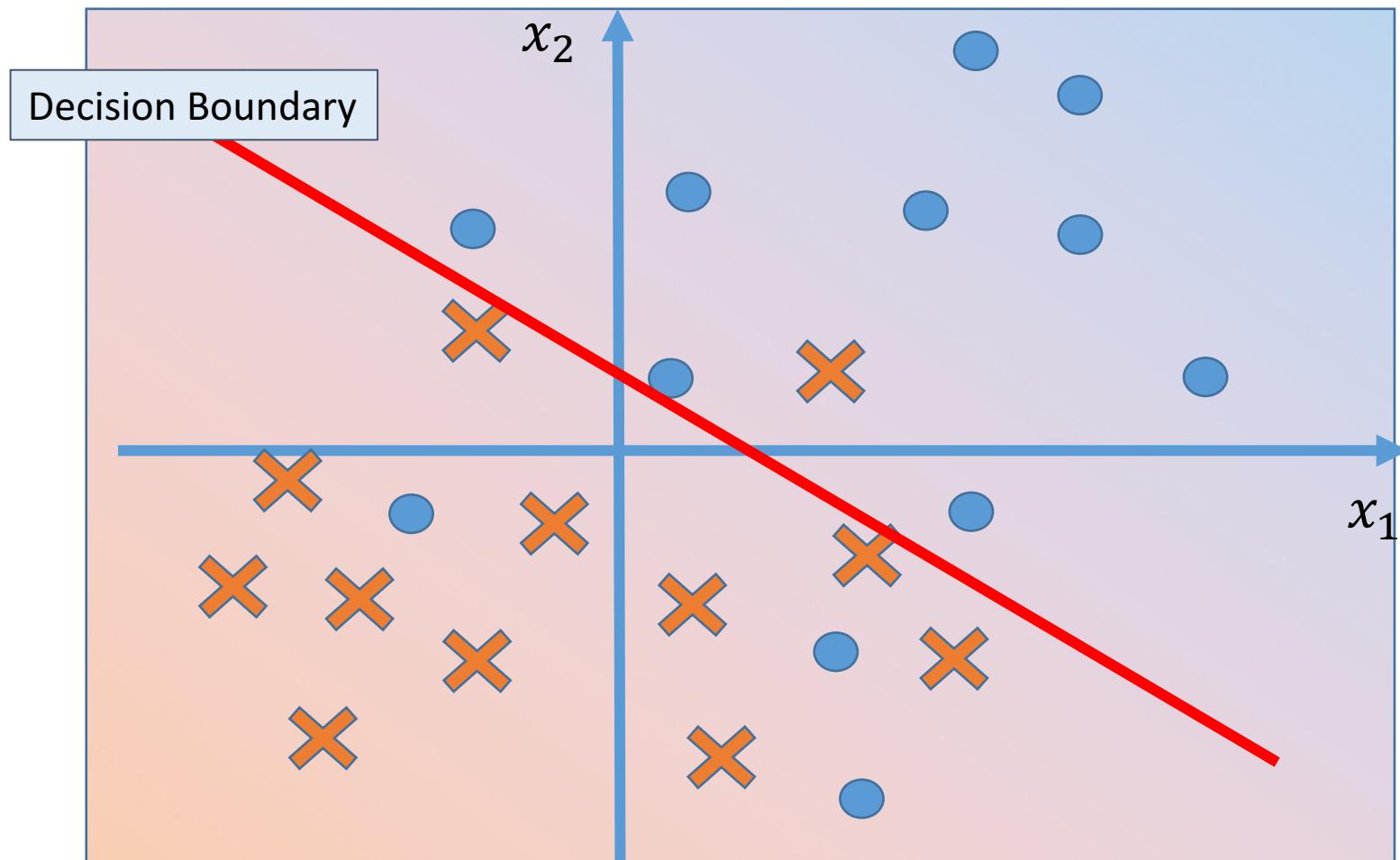
$$-\sum_{i=1}^m y_i \log \sigma(w^T x) + (1 - y_i) \log(1 - \sigma(w^T x))$$

What if data is not linearly separable?



We can consider a probabilistic model (today's lecture)

Prediction by logistic regression



A probabilistic model of modeling $\sigma(w^T x + b) = P(y = 1|x)$

What you will learn today

- ❖ Gradient descent / Stochastic gradient descent
- ❖ Linear regression

The hypothesis space for logistic regression

All functions of the form

Let's ignore b from now by
using the trick in page 11

$$h_{\mathbf{w}}(\mathbf{x}) = \sigma(\mathbf{w}^T \mathbf{x}) = \frac{1}{1 + \exp(-\mathbf{w}^T \mathbf{x})}$$

That is, a linear function, composed with a
sigmoid function (the logistic function)

$$\sigma(z) = \frac{1}{1 + \exp(-z)}$$

We want to find $h_w(x)$ such that

$$h_w(x) \approx P(y = 1 | x)$$

Maximum likelihood estimator for logistic regression

$S = \{(\mathbf{x}_i, y_i)\}$, m examples, Labels: $y \in \{-1, +1\}$

$$\underset{\mathbf{w}}{\operatorname{argmax}} P(S|\mathbf{w}) = \underset{\mathbf{w}}{\operatorname{argmax}} \prod_{i=1}^m P(y_i|\mathbf{x}_i, \mathbf{w})$$

Minimizing Negative Log-likelihood

$$\min_{\mathbf{w}} \sum_i^m \log(1 + \exp(-y_i \mathbf{w}^T \mathbf{x}_i))$$

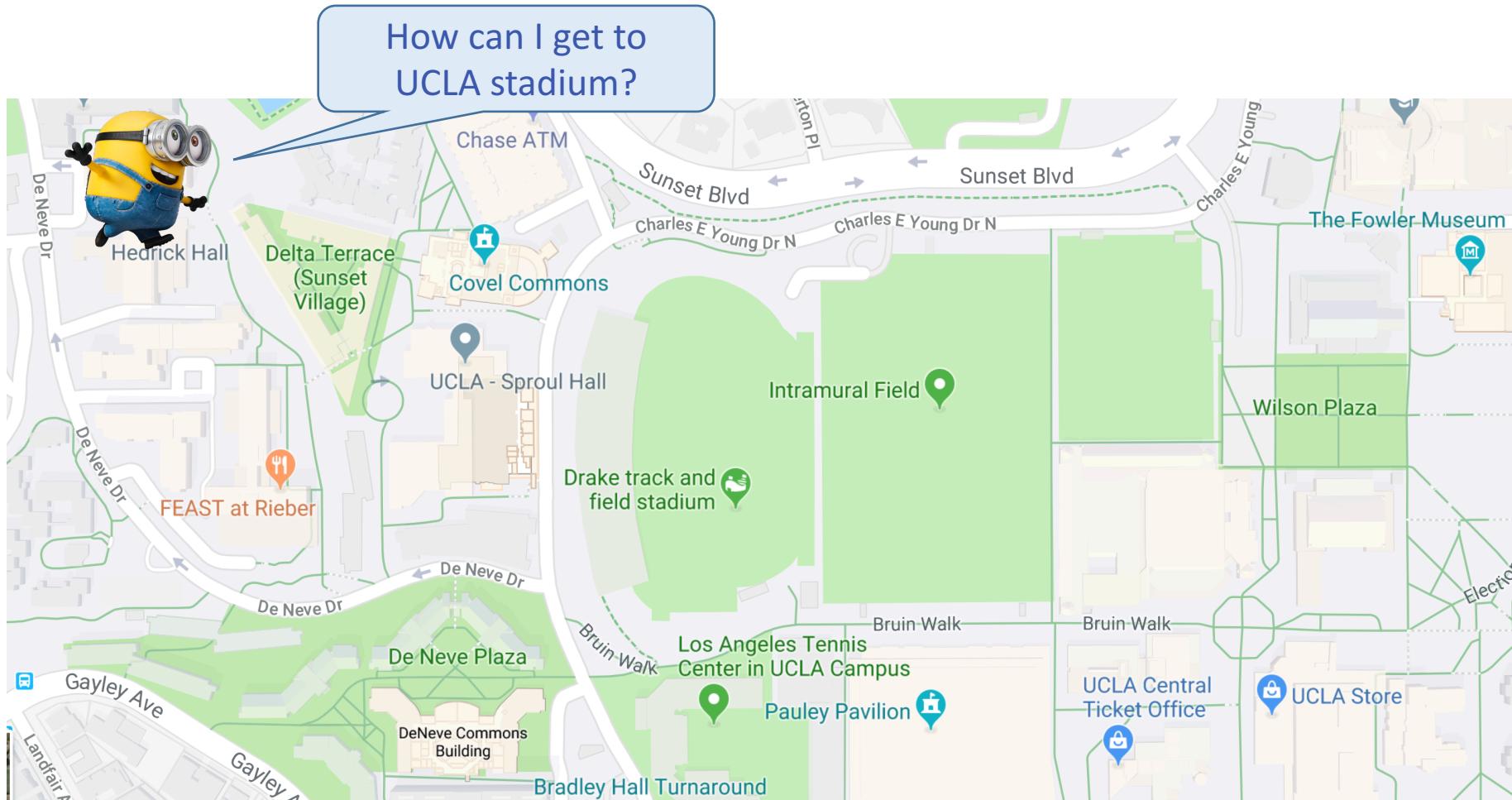
How to solve this?

$$\min_{\mathbf{w}} \sum_i^m \log(1 + \exp(-y_i \mathbf{w}^T \mathbf{x}_i))$$

- ❖ There is no closed-form solution
- ❖ Require iterative updates

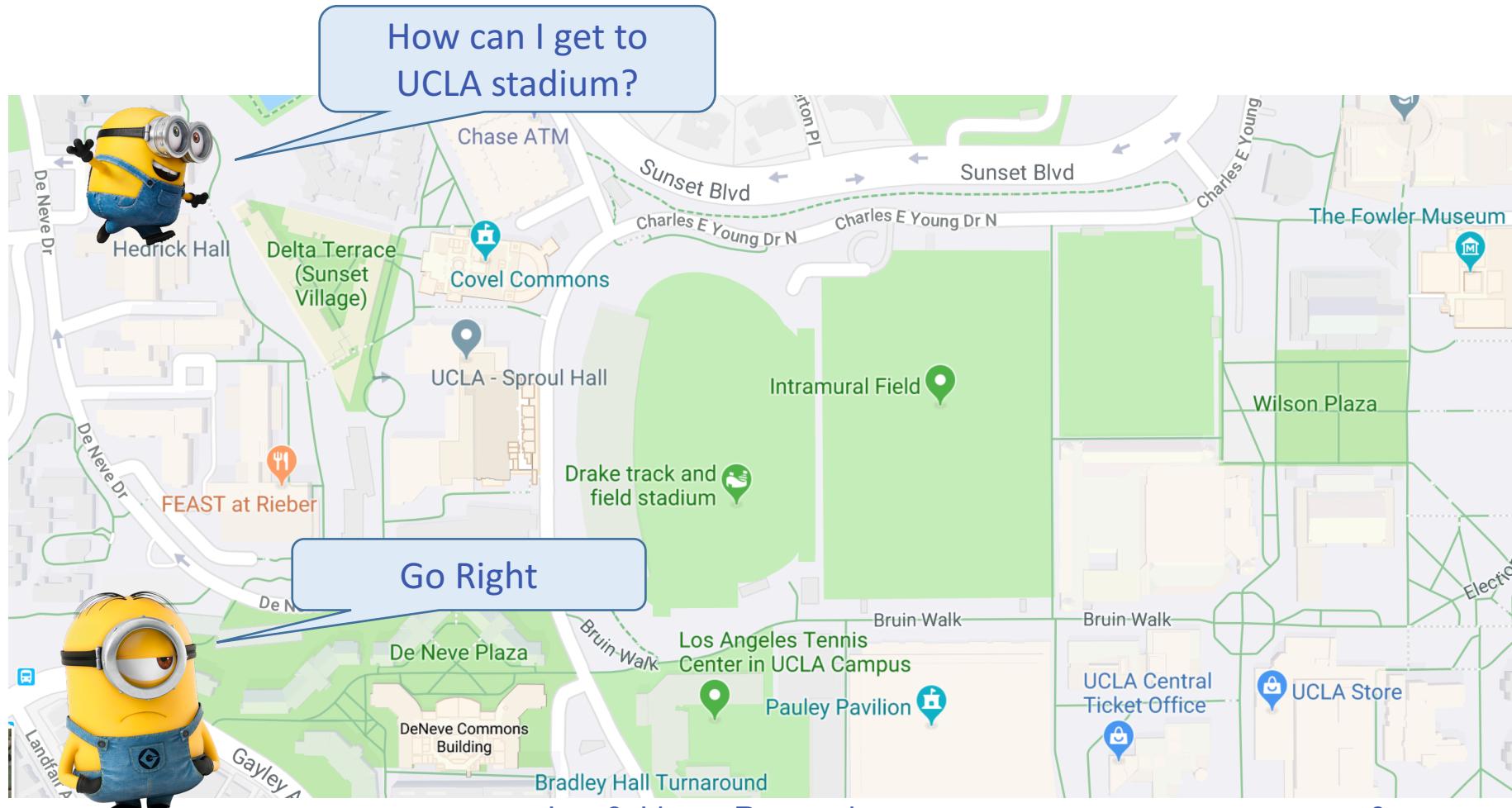
Intuition

❖ Asking direction



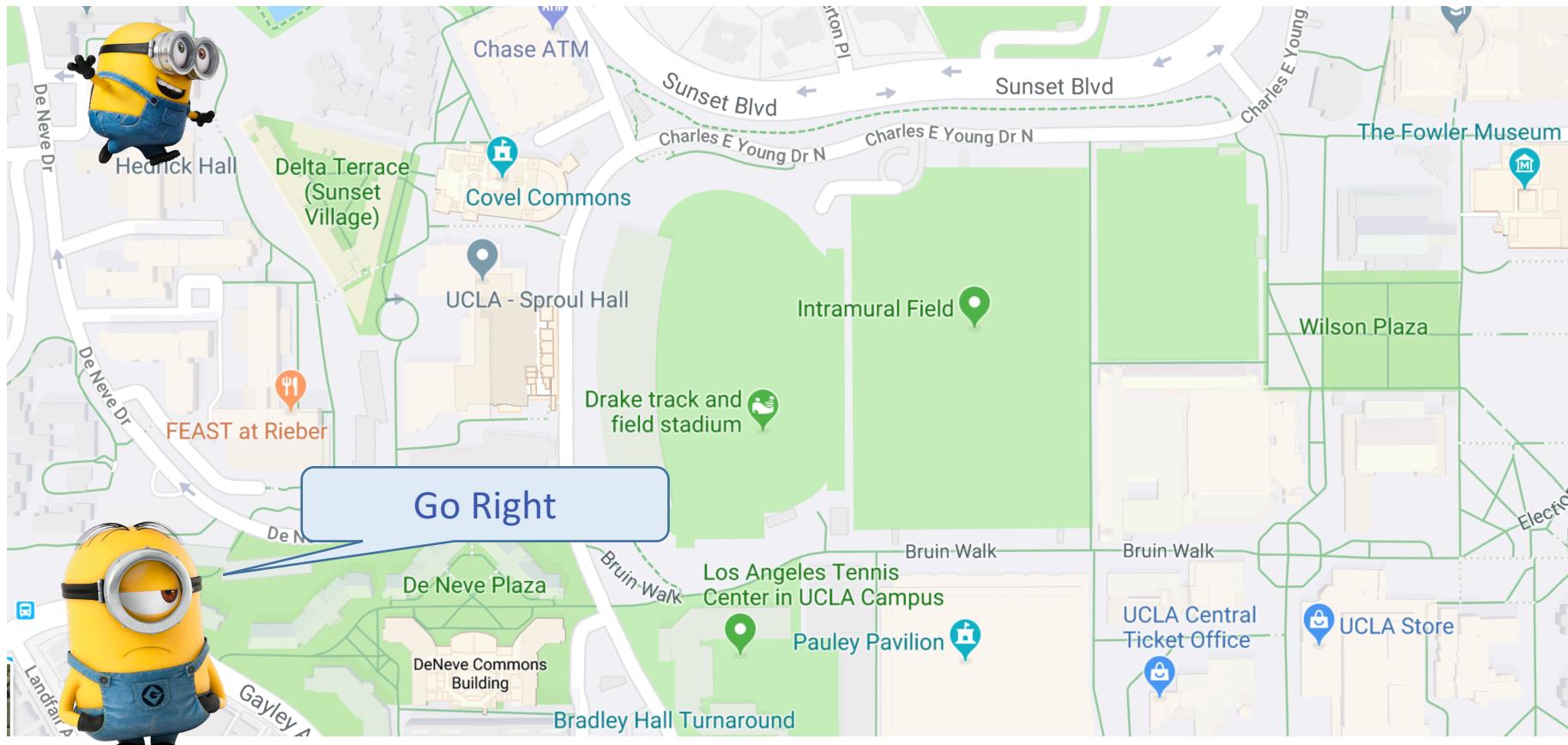
Intuition

❖ Asking direction



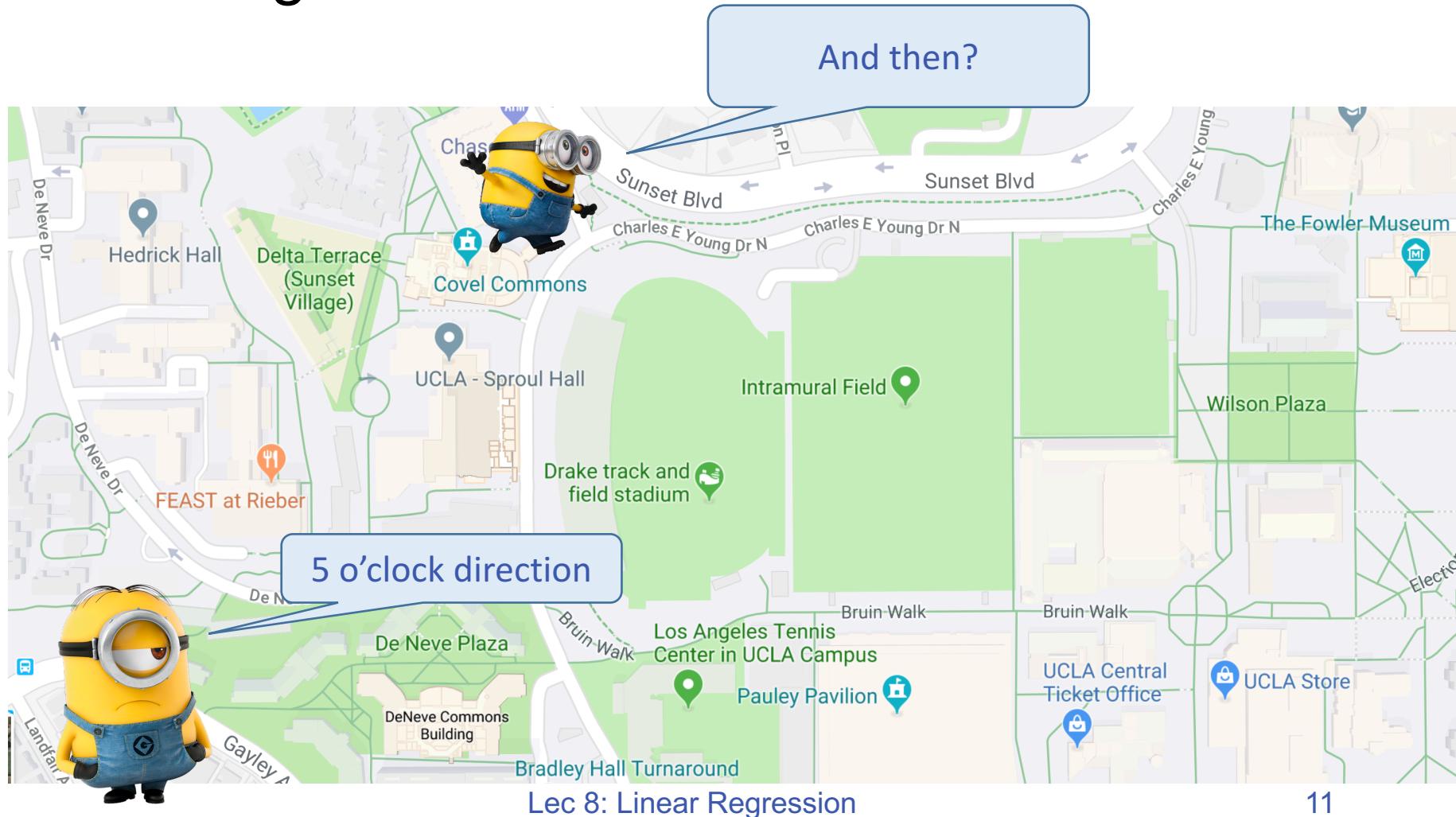
Intuition

❖ Asking direction



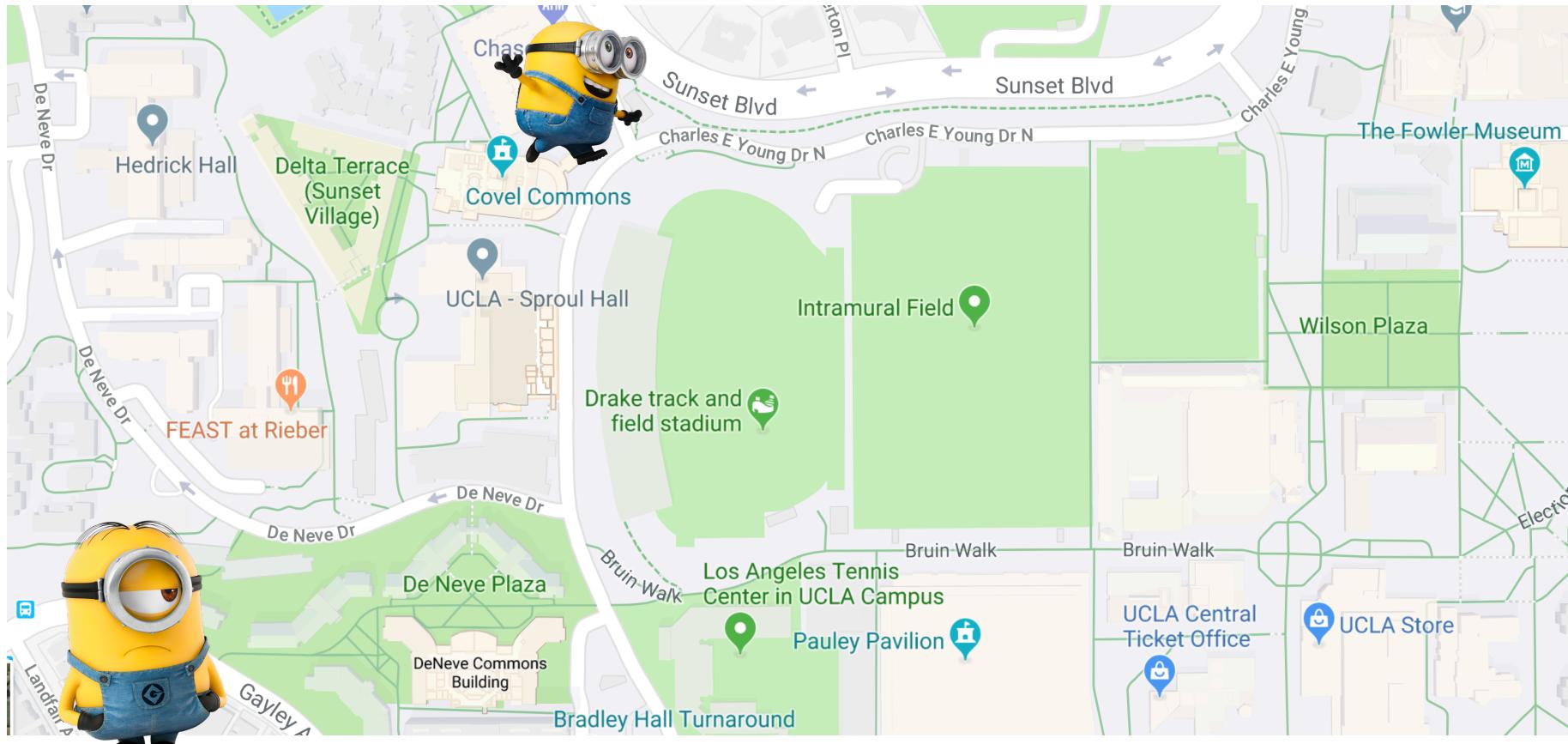
Intuition

❖ Asking direction



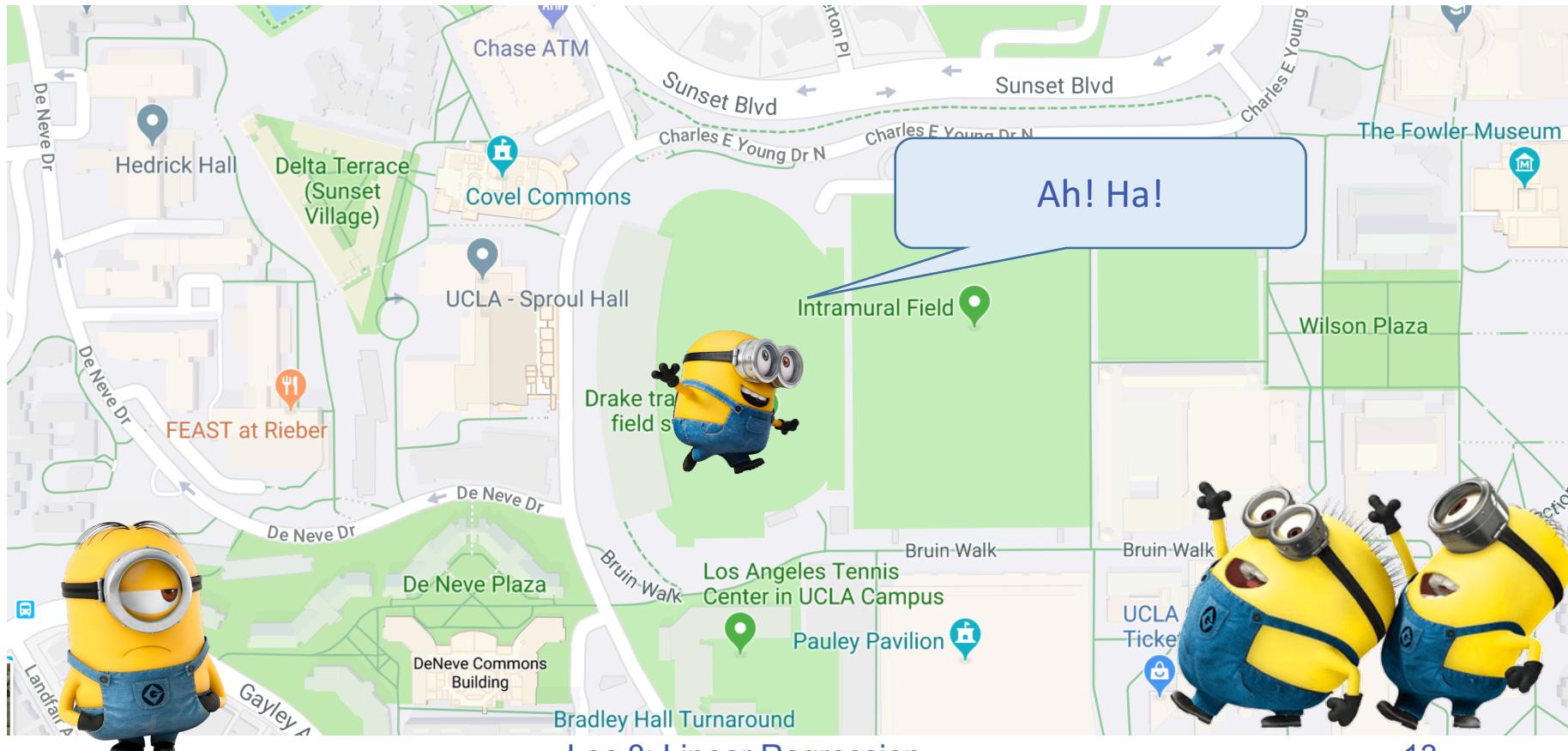
Intuition

❖ Asking direction



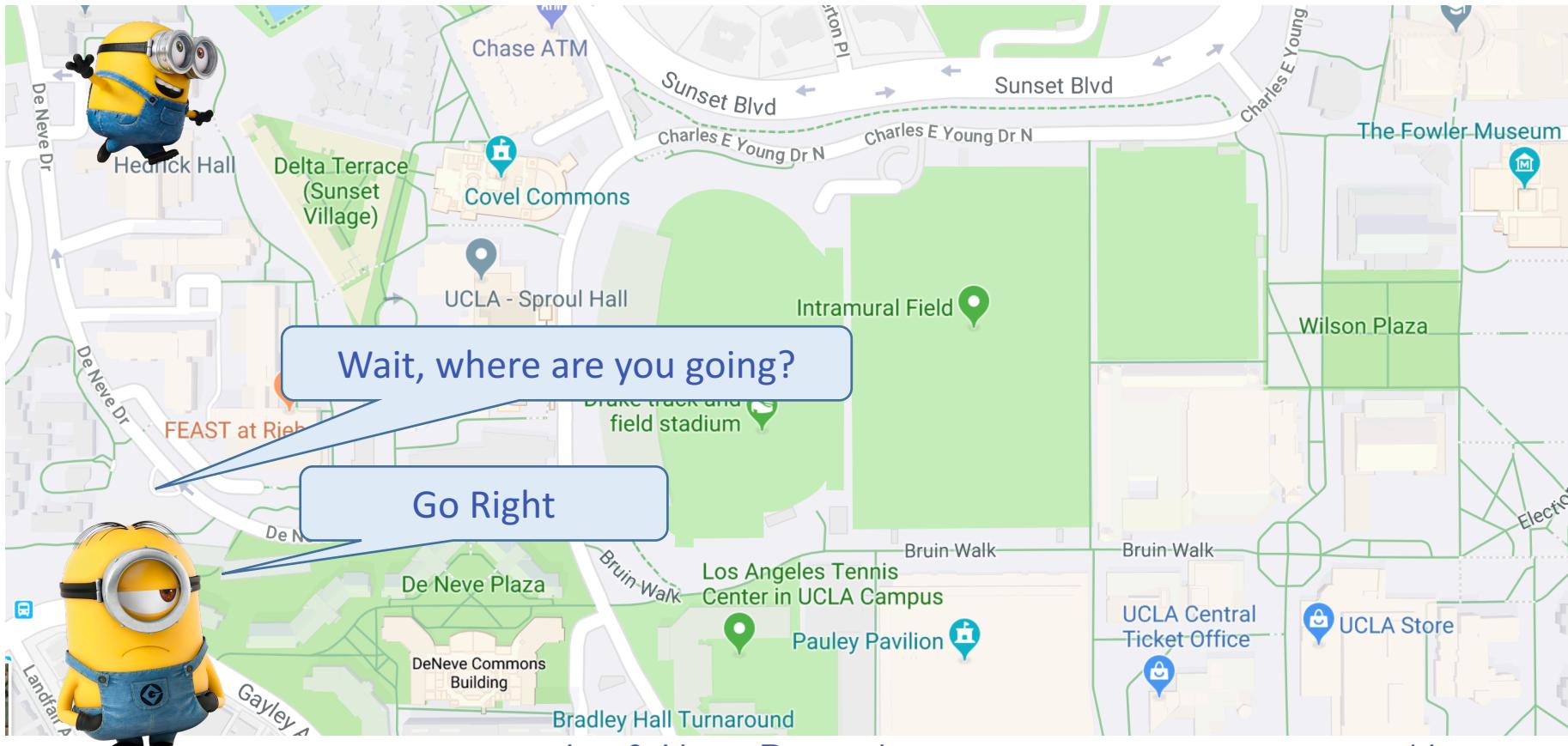
Intuition

❖ Asking direction



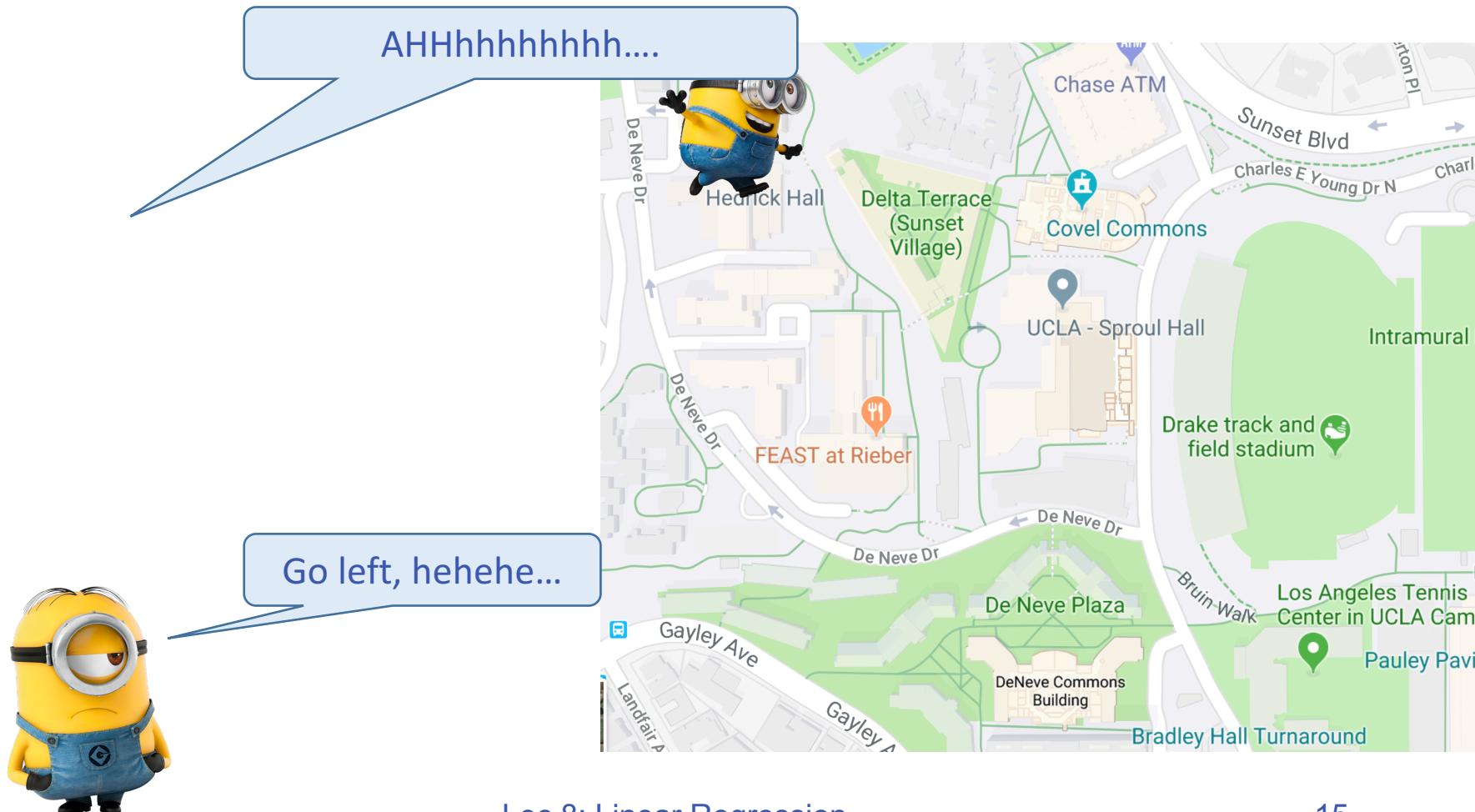
Intuition

- ❖ What may go wrong? Incorrect Step-size



Intuition

- ❖ What may go wrong? Incorrect Direction



Lecture 8: Linear Regression Winter 2018

Kai-Wei Chang

CS @ UCLA

kw+cm146@kwchang.net

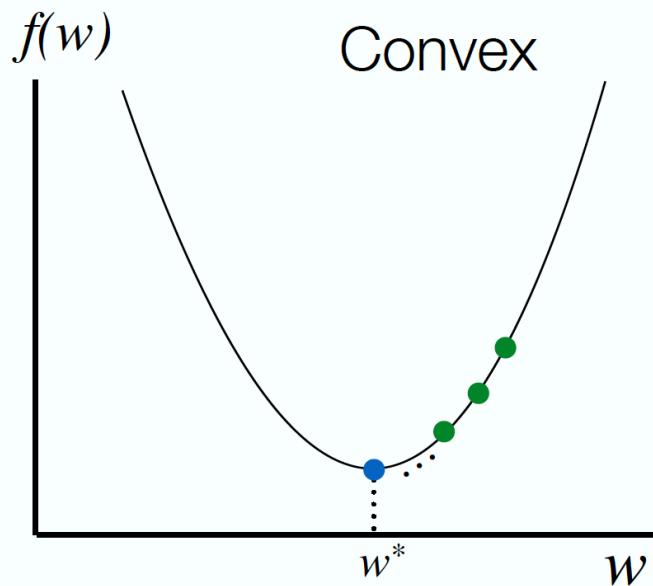
The instructor gratefully acknowledges Dan Roth, Vivek Srikumar, Sriram Sankararaman, Fei Sha, Ameet Talwalkar, Eric Eaton, and Jessica Wu whose slides are heavily used, and the many others who made their course material freely available online.

How to solve this?

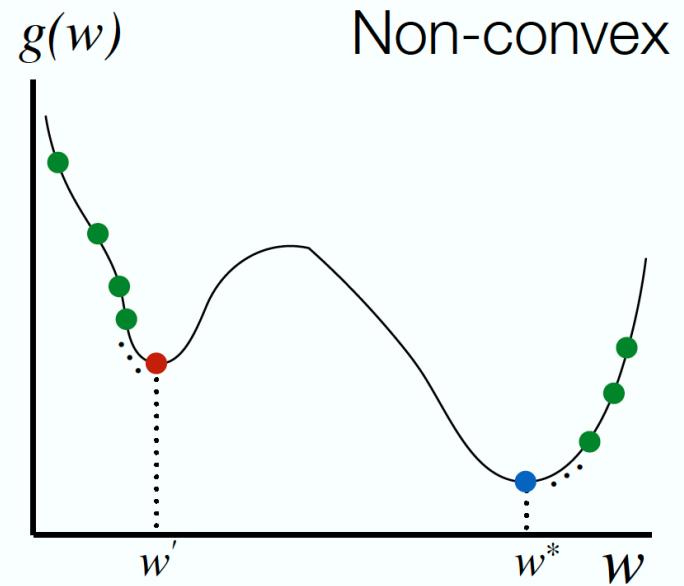
$$\min_{\mathbf{w}} \sum_i^m \log(1 + \exp(-y_i \mathbf{w}^T \mathbf{x}_i))$$

- ❖ This function is **convex**
- ❖ Many convex optimization methods can be used
 - ❖ Gradient descent
 - ❖ Stochastic (sub)-gradient descent
 - ❖ Coordinate descent methods
 - ❖ Newton methods
 - ❖ LBFGS

Convex function



Any local minimum is a global minimum



Multiple local minima may exist

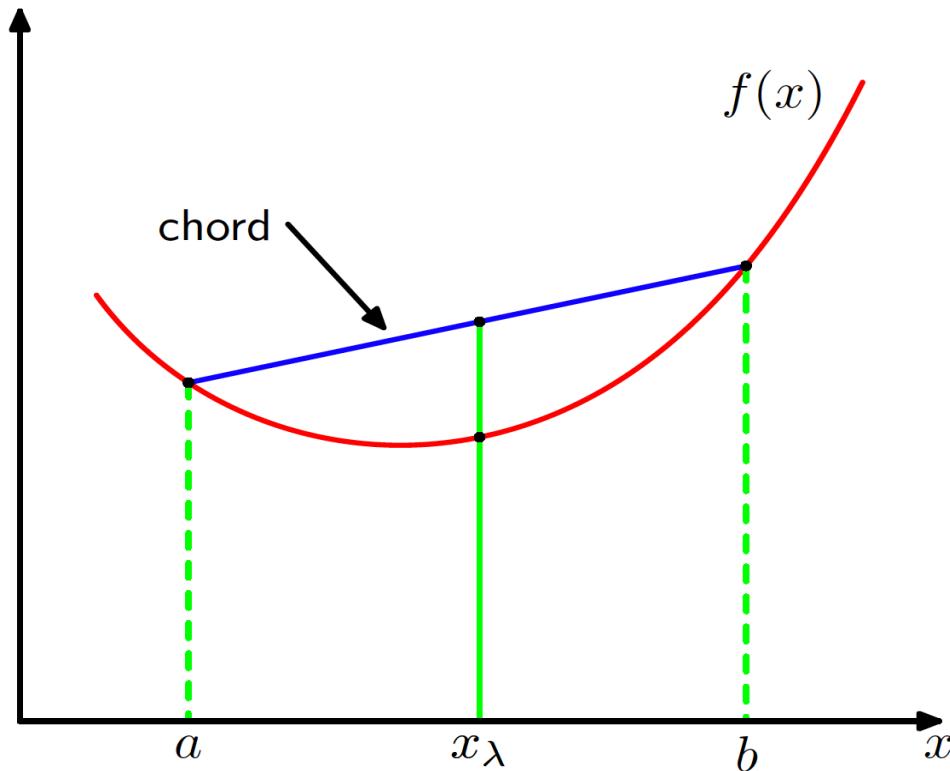
Convex function

A function $f(x)$ is convex if

$$f(\lambda a + (1 - \lambda)b) \leq \lambda f(a) + (1 - \lambda)f(b)$$

for

$$0 \leq \lambda \leq 1$$



How to determine convexity?

$f(x)$ is convex if

$$f''(x) \geq 0$$

Examples:

$$f(x) = x^2, f''(x) = 2 > 0$$

Do you know other convex functions?

Exercise

- ❖ Prove the following functions are convex

$$f(x) = ax + b$$

$$f(x) = x^2$$

$$f(x) = e^x$$

$$f(x) = \frac{1}{x}, x \geq 0$$

Multi-variate function

$f(\mathbf{x})$ is convex

$$f(\lambda \mathbf{a} + (1 - \lambda) \mathbf{b}) \leq \lambda f(\mathbf{a}) + (1 - \lambda) f(\mathbf{b})$$

for all \mathbf{a}, \mathbf{b} , $0 \leq \lambda \leq 1$

Multi-variate functions

- ❖ How to determine convexity in this case?
- ❖ Matrix of second-order derivatives ([Hessian](#))

$$\mathbf{H} = \begin{pmatrix} \frac{\partial^2 f(\mathbf{x})}{\partial x_1^2} & \frac{\partial^2 f(\mathbf{x})}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f(\mathbf{x})}{\partial x_1 \partial x_D} \\ \frac{\partial^2 f(\mathbf{x})}{\partial x_1 \partial x_2} & \frac{\partial^2 f(\mathbf{x})}{\partial x_2^2} & \cdots & \frac{\partial^2 f(\mathbf{x})}{\partial x_2 \partial x_D} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial^2 f(\mathbf{x})}{\partial x_1 \partial x_D} & \frac{\partial^2 f(\mathbf{x})}{\partial x_2 \partial x_D} & \cdots & \frac{\partial^2 f(\mathbf{x})}{\partial x_D^2} \end{pmatrix}$$

Multi-variate functions

- ❖ How to determine convexity in this case?

If the Hessian is positive semi-definite $\mathbf{H} \succeq 0$, then f is convex.

A matrix \mathbf{H} is positive semi-definite if and only if

$$\mathbf{z}^T \mathbf{H} \mathbf{z} = \sum_{j,k} H_{j,k} z_j z_k \geq 0$$

for all \mathbf{z} .

Multi-variate functions

❖ Example

$$f(\mathbf{x}) = x_1^2 + 2x_2^2$$

$$\mathbf{H} = \begin{pmatrix} 2 & 0 \\ 0 & 4 \end{pmatrix}$$

$$\mathbf{z}^T \mathbf{H} \mathbf{z} = 2z_1^2 + 4z_2^2 \geq 0$$

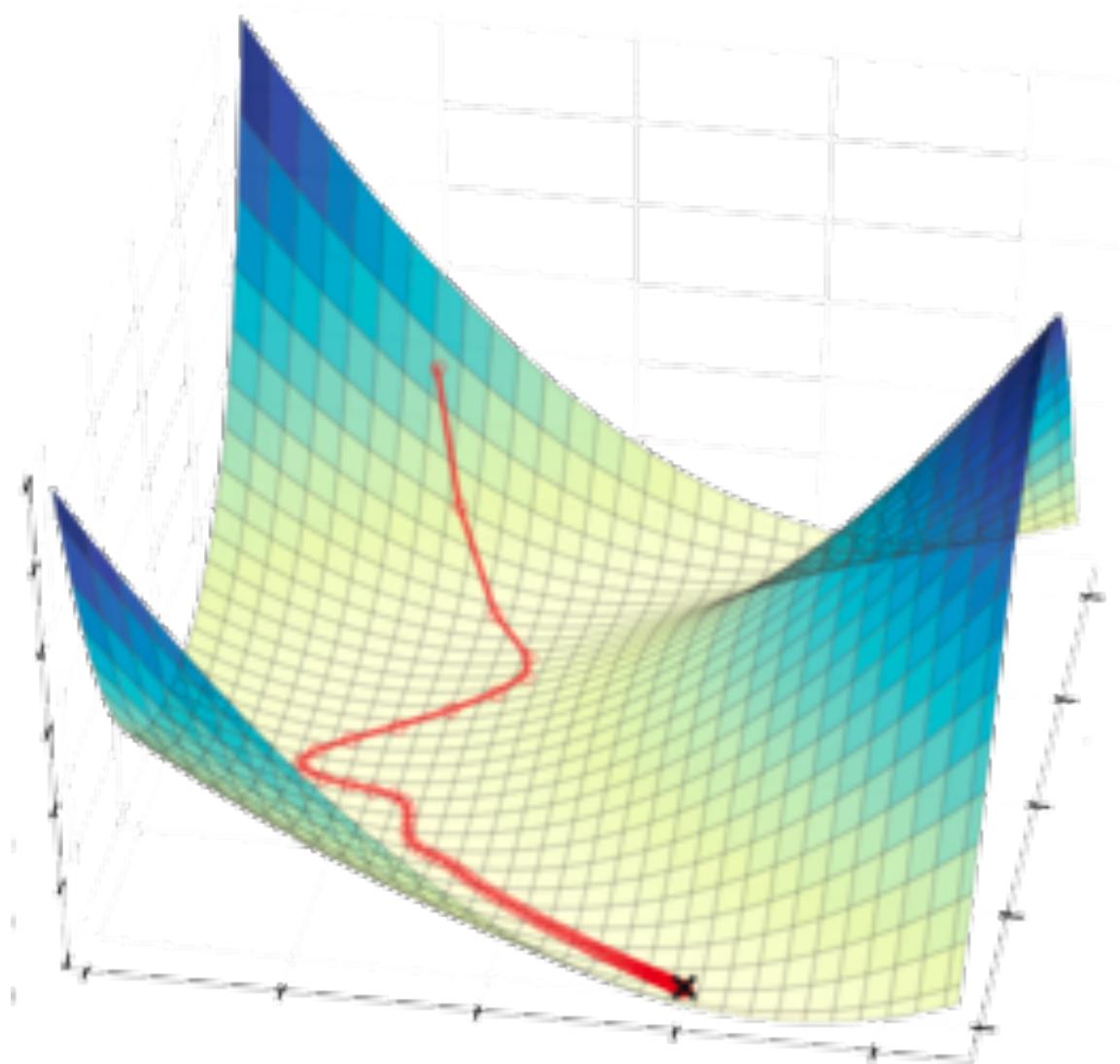
Gradient Descent

Algorithm 1 Gradient Descent (J)

- 1: $t \leftarrow 0$
 - 2: Initialize $\theta^{(0)}$
 - 3: **repeat**
 - 4: $\theta^{(t+1)} \leftarrow \theta^{(t)} - \eta \nabla J(\theta^{(t)})$
 - 5: $t \leftarrow t + 1$
 - 6: **until** convergence
 - 7: Return final value of θ
-

Need to compute the gradient for the negative log likelihood

Gradient Descent



Example

$$\min f(\boldsymbol{\theta}) = 0.5(\theta_1^2 - \theta_2)^2 + 0.5(\theta_1 - 1)^2$$

❖ We compute the gradients

$$\frac{\partial f}{\partial \theta_1} = 2(\theta_1^2 - \theta_2)\theta_1 + \theta_1 - 1$$

$$\frac{\partial f}{\partial \theta_2} = -(\theta_1^2 - \theta_2)$$

Example $\min f(\theta) = 0.5(\theta_1^2 - \theta_2)^2 + 0.5(\theta_1 - 1)^2$

❖ Use the following iterative procedure for gradient descent

$$\nabla f(\theta) = \begin{bmatrix} 2(\theta_1^2 - \theta_2)\theta_1 + \theta_1 - 1 \\ -(\theta_1^2 - \theta_2) \end{bmatrix}$$

- ① Initialize $\theta_1^{(0)}$ and $\theta_2^{(0)}$, and $t = 0$
- ② do

Type equation here.

$$\theta_1^{(t+1)} \leftarrow \theta_1^{(t)} - \eta \left[2(\theta_1^{(t)})^2 - \theta_2^{(t)} \right] \theta_1^{(t)} + \theta_1^{(t)} - 1$$

$$\theta_2^{(t+1)} \leftarrow \theta_2^{(t)} - \eta \left[-(\theta_1^{(t)})^2 - \theta_2^{(t)} \right]$$

$$t \leftarrow t + 1$$

- ③ until $f(\theta^{(t)})$ does not change much

Remarks

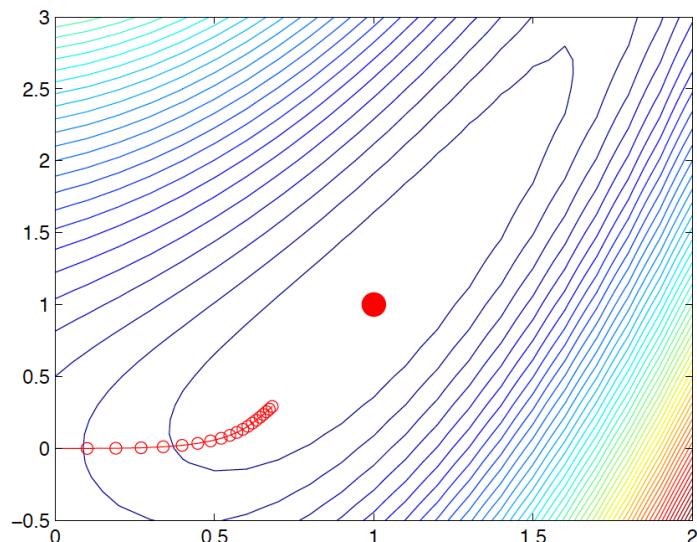
- ❖ η is often called step size or learning rate -- how far our update will go along the the direction of the negative gradient
- ❖ With a **suitable** choice of η , the iterative procedure converges to a stationary point where

$$\frac{\partial f}{\partial \theta} = 0$$

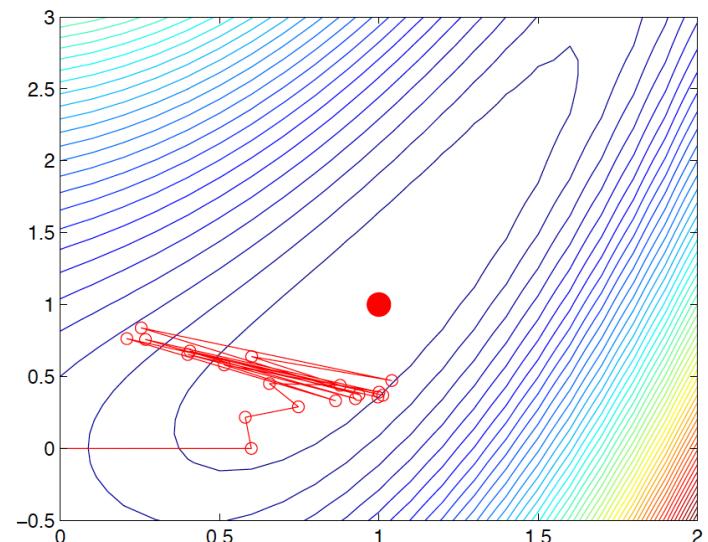
- ❖ A stationary point is only necessary for being the minimum

Choosing the right η is important

small η is too slow?



large η is too unstable?



Linear Regression

Outline

- ❖ Least Squares Method for regression
 - ❖ Examples
 - ❖ The LMS objective
 - ❖ Gradient descent
 - ❖ Incremental/stochastic gradient descent

Regression

- ❖ Predicting a continuous outcome variable
- ❖ Predicting shoe size from height, weight and gender
- ❖ Predicting a company's future stock price using its profit and other financial info
- ❖ Predicting song year from audio features

Regression v.s. Classification

- ❖ We can measure 'closeness' of prediction and labels, leading to different ways to evaluate prediction errors.
 - ❖ Predicting shoe size: better to be off by one size than by 5 sizes
 - ❖ Predicting song year: better to be off by one year than by 20 years
- ❖ This will lead to different learning models and algorithms

Example

The screenshot shows the REDFIN website's search results for Los Angeles. At the top, there is a search bar with "Los Angeles" and a "Search Listings" button. Below the search bar are filters for Price (No min to No max), Beds (No min), and More Options. The main content area displays a map of a specific neighborhood in Los Angeles, with 111 results marked by blue icons. The map includes street names like W Adams Blvd, S Western Ave, S Normandie Ave, W Jefferson Blvd, Exposition Blvd, and S Vermont Ave. Major highways 110 and 20 are also visible. A legend on the left indicates that the blue icons represent the 111 results found.

Feature can be used

3620 South BUDLONG
Los Angeles, CA 90007
Status: Closed

\$1,510,000 | **14** Beds | **6** Baths | **4,418** Sq. Ft.
Last Sold Price | **14** Beds | **6** Baths | **4,418** Sq. Ft.
Built: 1956 | Lot Size: 9,649 Sq. Ft. | Sold On: Jul 26, 2013

Overview Property Details Tour Insights Property History Public Records Activity Schools

1 of 12

Five unit apartment complex within 2 blocks of USC campus, Gate #6. Great for students (most student leases have parents as guarantors). Most USC students live off campus, so housing units like this are always fully leased. Situated on a gated, corner lot, and across from an elementary school, this complex was recently renovated, and has in-unit laundry hook ups, wall -unit AC, and 12 parking spaces. It is within a DPS (Department of Public Safety) and Campus Cruiser patrolled area. This is a great income generating property, not to be missed!

Property Type Multi-Family | Style Two Level, Low Rise
Community Downtown Los Angeles | County [Los Angeles](#)
MLS# 22176741

Property Details for 3620 South BUDLONG, Los Angeles, CA 90007

Details provided by i-Tech MLS and may not match the public record. [Learn More](#)

Interior Features	Kitchen Information	Laundry Information	Heating & Cooling
	<ul style="list-style-type: none"> • Remodeled • Oven, Range 	<ul style="list-style-type: none"> • Inside Laundry 	<ul style="list-style-type: none"> • Wall Cooling Unit(s)
Multi-Unit Information			
Community Features	Unit 2 Information	Unit 3 Information	Unit 4 Information
<ul style="list-style-type: none"> • Units in Complex (Total): 5 	<ul style="list-style-type: none"> • # of Beds: 3 • # of Baths: 1 • Unfurnished • Monthly Rent: \$2,250 	<ul style="list-style-type: none"> • Unfurnished 	<ul style="list-style-type: none"> • Monthly Rent: \$2,350
Multi-Family Information	Unit 1 Information	Unit 6 Information	Unit 5 Information
<ul style="list-style-type: none"> • # Leased: 5 • # of Buildings: 1 • Owner Pays Water • Tenant Pays Electricity, Tenant Pays Gas 	<ul style="list-style-type: none"> • # of Beds: 2 • # of Baths: 1 • Unfurnished • Monthly Rent: \$1,700 	<ul style="list-style-type: none"> • # of Beds: 3 • # of Baths: 1 • Unfurnished • Monthly Rent: \$2,250 	<ul style="list-style-type: none"> • # of Beds: 3 • # of Baths: 2 • Unfurnished • Monthly Rent: \$2,325
Property / Lot Details	Property Features	Property Information	Tax Parcel Number
	<ul style="list-style-type: none"> • Automatic Gate, Card/Code Access 	<ul style="list-style-type: none"> • Updated/Remodeled • Square Footage Source: Public Records 	• Tax Parcel Number: 5040017019
Lot Information	Parking / Garage, Exterior Features, Utilities & Financing		
<ul style="list-style-type: none"> • Lot Size (Sq. Ft.): 9,649 • Lot Size (Acres): 0.2215 • Lot Size Source: Public Records 	Parking Information	Utility Information	Financial Information
	<ul style="list-style-type: none"> • # of Parking Spaces (Total): 12 • Parking Space • Gated 	<ul style="list-style-type: none"> • Green Certification Rating: 0.00 • Green Location: Transportation, Walkability • Green Walk Score: 0 • Green Year Certified: 0 	<ul style="list-style-type: none"> • Capitalization Rate (%): 6.25 • Actual Annual Gross Rent: \$128,331 • Gross Rent Multiplier: 11.29
Building Information	Location Details, Misc. Information & Listing Information		
<ul style="list-style-type: none"> • Total Floors: 2 	Location Information	Expense Information	Listing Information
	<ul style="list-style-type: none"> • Cross Streets: W 36th Pl 	<ul style="list-style-type: none"> • Operating: \$37,664 	<ul style="list-style-type: none"> • Listing Terms: Cash, Cash To Existing Loan • Buyer Financing: Cash

How to learn the unknown parameters?

- ❖ training data (past sales record)

sqft	sale price
2000	800K
2100	907K
1100	312K
5500	2,600K
...	...

What is our model?

Price_per_sqft

- ❖ Sale Price = square_footage $\times w_1 + w_b$

Unexplainable_stuff

How to measure prediction error

- ❖ The classification error (hit or miss) is not appropriate for continuous outcomes
- ❖ How should we evaluate quality of a prediction?
 - ▶ *absolute* difference: $| \text{prediction} - \text{sale price}|$
 - ▶ *squared* difference: $(\text{prediction} - \text{sale price})^2$

sqft	sale price	prediction	error	squared error
2000	810K	720K	90K	90^2
2100	907K	800K	107K	107^2
1100	312K	350K	-38K	38^2
5500	2,600K	2,600K	0	0
...	...			

Minimize squared errors

- ❖ Our model:

Sale Price = square_footage $\times w_1 + w_b$

- ❖ Learning: Using the training data to find the *best* possible value of w
- ❖ Prediction: Predict the sale price given the square_footage

Minimize squared errors

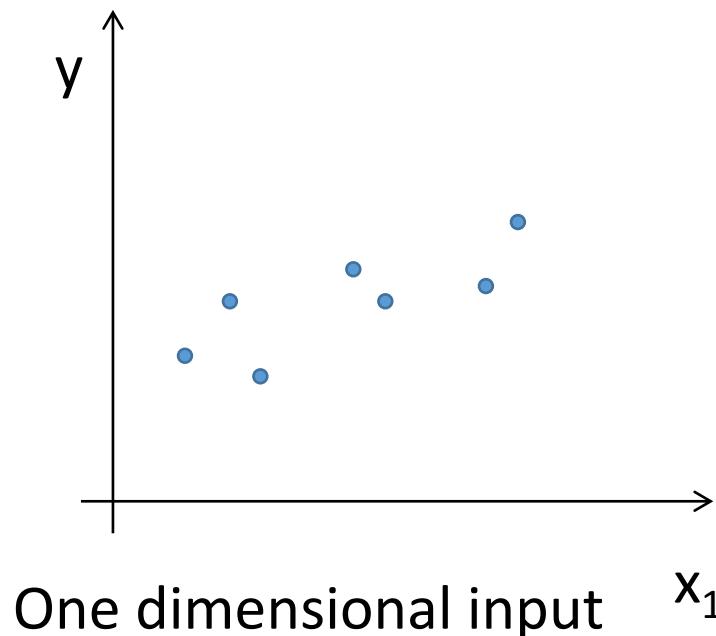
sqft	sale price	prediction	error	squared error
2000	810K	720K	90K	90^2
2100	907K	800K	107K	107^2
1100	312K	350K	38K	38^2
5500	2,600K	2,600K	0	0
...	...			
Total				$90^2 + 107^2 + 38^2 + 0 + \dots$

- ❖ Our model:
Sale Price = square_footage $\times w_1 + w_b$
- ❖ Goal of training: adjust w_1, w_b such that the sum of the squared error is minimized

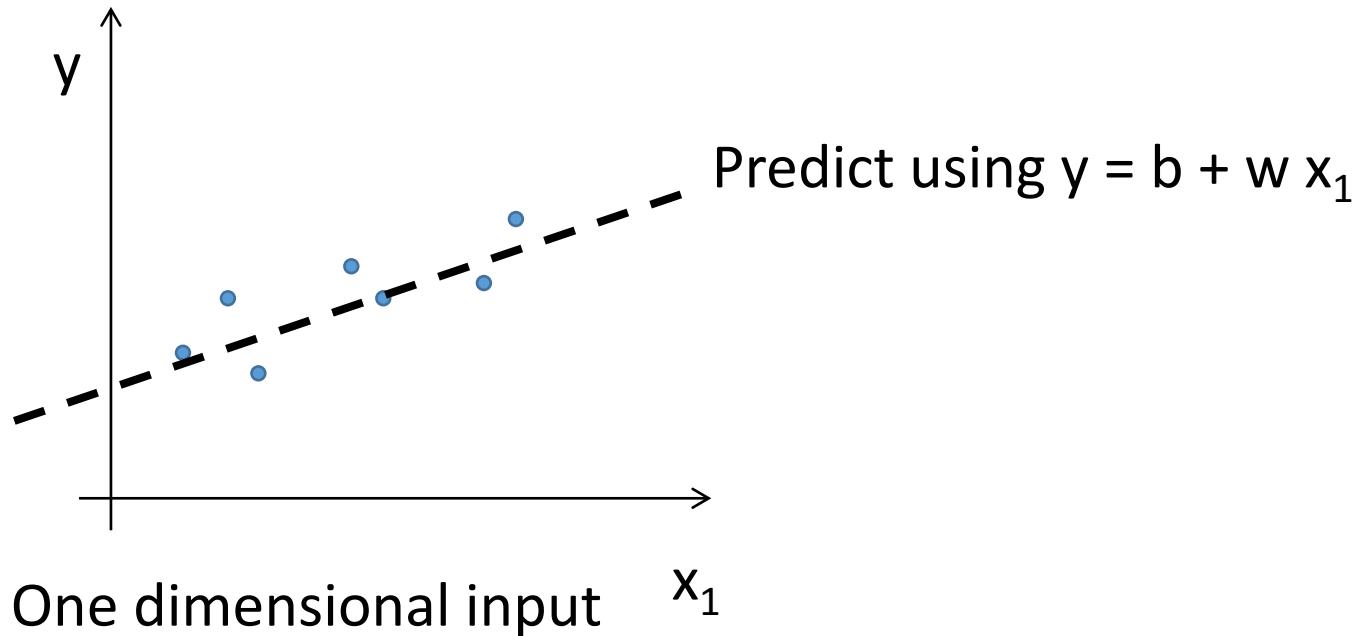
Setup – Linear Regression

- Input: $\mathbf{x} \in \mathbb{R}^D$ (covariates, predictors, features, etc)
- Output: $y \in \mathbb{R}$ (responses, targets, outcomes, outputs, etc)
- Hypotheses/Model: $h_{\mathbf{w}, b}$, with $h_{\mathbf{w}, b}(\mathbf{x}) = b + \sum_d w_d x_d = b + \mathbf{w}^T \mathbf{x}$
 $\mathbf{w} = [w_1 \ w_2 \ \cdots \ w_D]^T$: *weights*
 b is called the **bias or offset or intercept term**.
 $\boldsymbol{\theta} = [b \ w_1 \ w_2 \ \cdots \ w_D]^T$
- Training data: $\mathcal{D} = \{(\mathbf{x}_n, y_n), n = 1, 2, \dots, N\}$

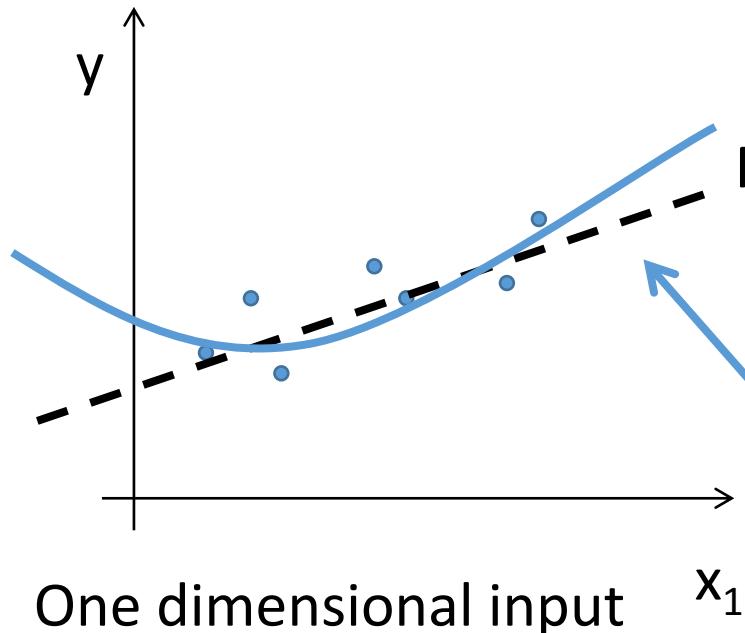
Examples



Examples



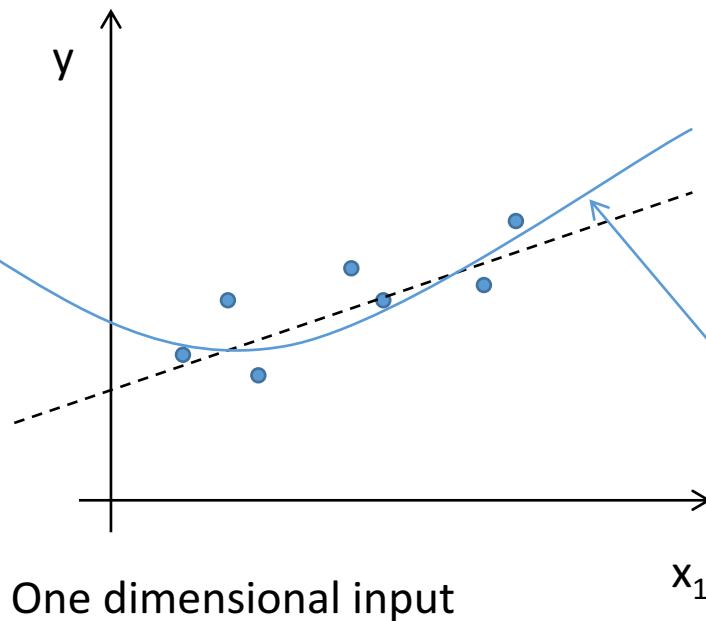
Examples



Predict using $y = b + w x_1$

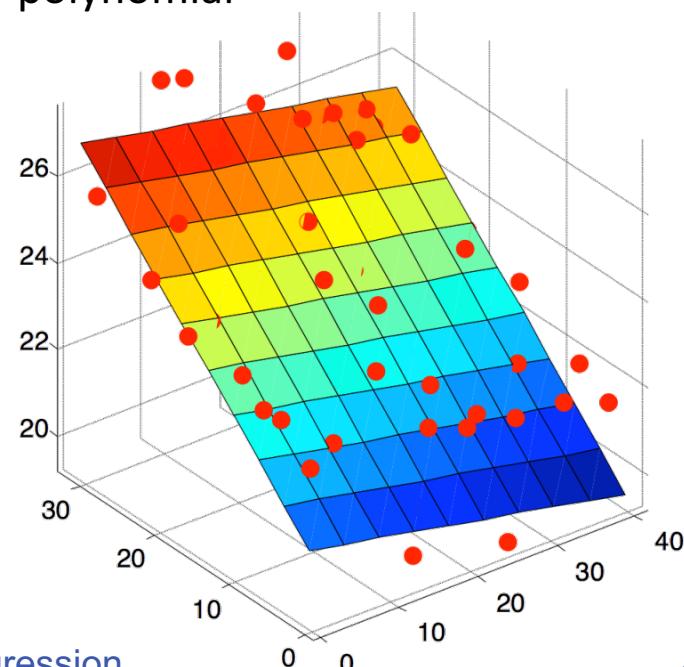
The linear
function is not
our only choice.
We could have
tried to fit the
data as another
polynomial

Examples



Predict using $y = b + w x_1$
The linear function is
not our only choice.
We could have tried
to fit the data as
another polynomial

Two dimensional input
Predict using $y = w_1 x_1 + w_2 x_2 + b$



What is the best weight vector?

Question: How do we know which weight vector is the *best* one for a training set?

Again here, we can use the trick to remove the bias term

For an input (\mathbf{x}_i, y_i) in the training set, the *cost* of a mistake is

$$|y_i - \mathbf{w}^T \mathbf{x}_i|$$

Define the cost (or *loss*) for a particular weight vector \mathbf{w} to be

$$J(\mathbf{w}) = \frac{1}{2} \sum_{i=1}^m (y_i - \mathbf{w}^T \mathbf{x}_i)^2$$

Sum of squared
costs over the
training set

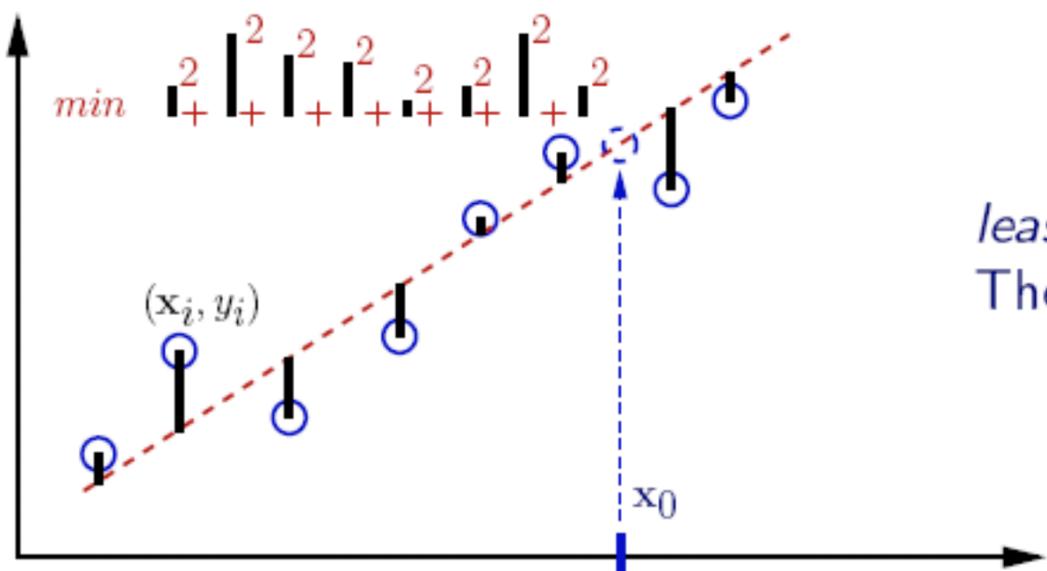
One strategy for learning: *Find the \mathbf{w} with least cost on this data*

Least Mean Squares (LMS) Regression

$$\min_{\mathbf{w}} \frac{1}{2} \sum_{i=1}^m (y_i - \mathbf{w}^T \mathbf{x}_i)^2$$

Learning: minimizing mean squared error

Example

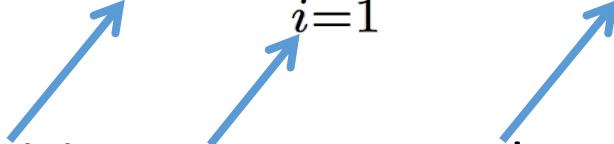


least squares (LSQ)
The fitted line is used as a predictor

Least Mean Squares (LMS) Regression

$$\min_{\mathbf{w}} \frac{1}{2} \sum_{i=1}^m (y_i - \mathbf{w}^T \mathbf{x}_i)^2$$

Learning: minimizing mean squared error



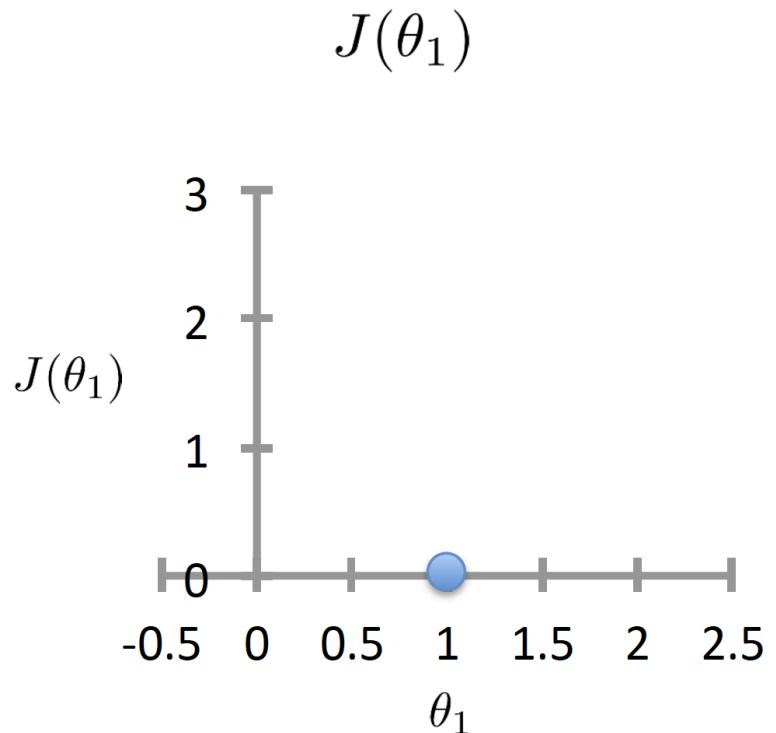
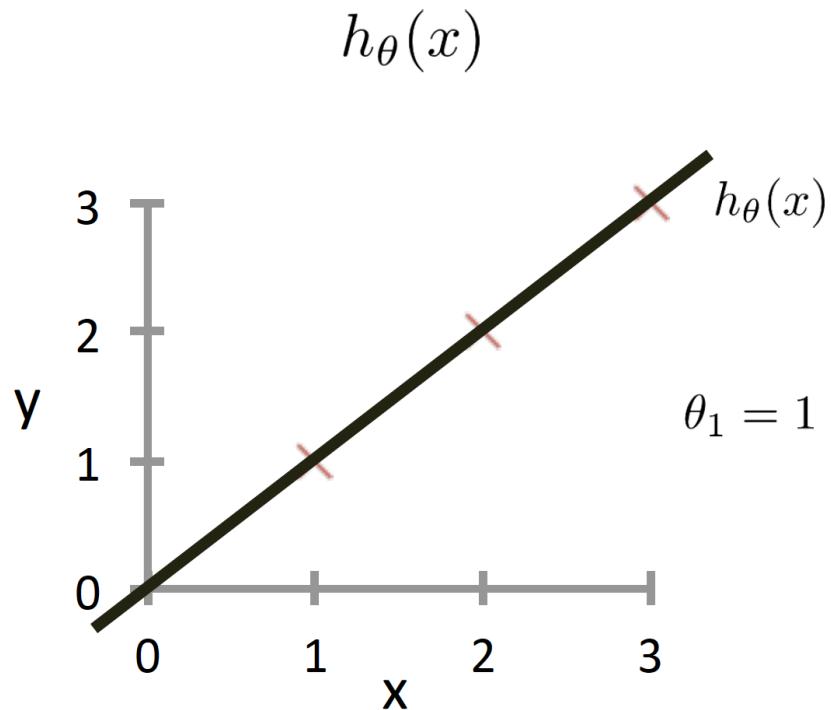
Different strategies exist for *learning by optimization*

- ❖ Gradient descent is a popular algorithm
 - ❖ (For this minimization objective, there is also an analytical solution)

Intuition behind the cost function

Assume $x \in \mathbb{R}$, $\theta_0 = 0$.

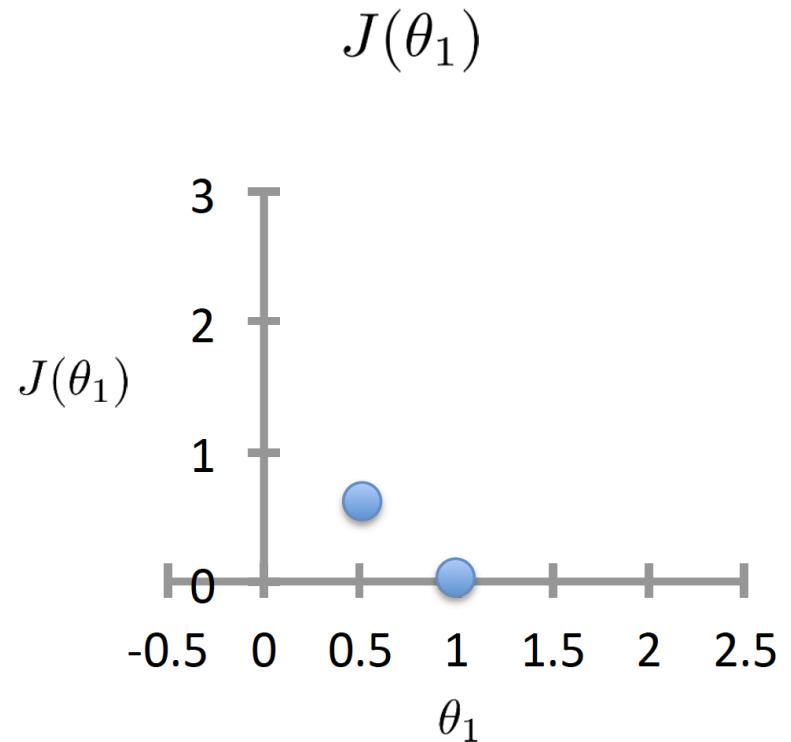
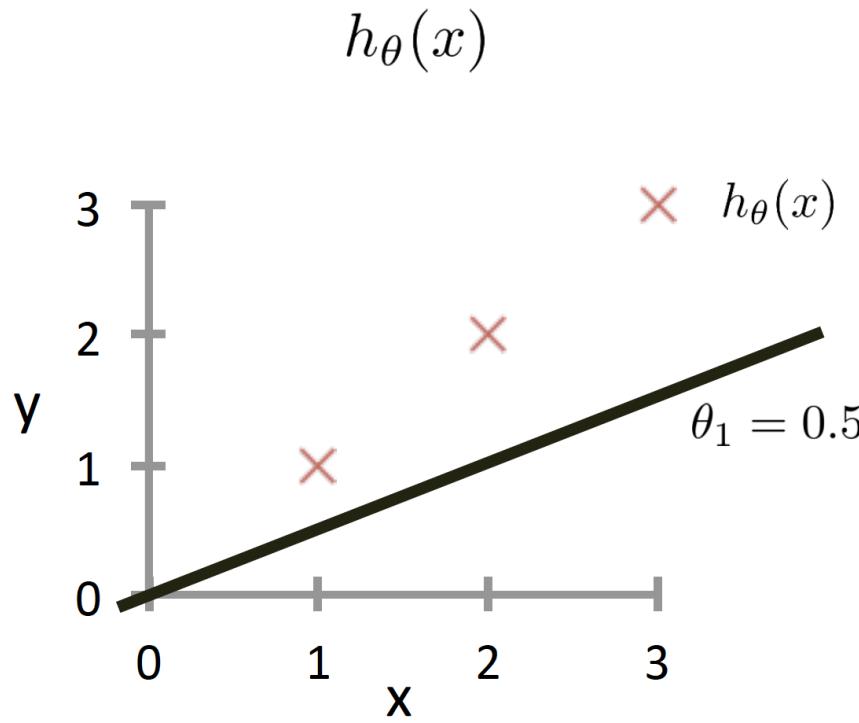
$$h_{\theta}(x) = \theta_0 + \theta_1 x = \theta_1 x$$



Intuition behind the cost function

Assume $x \in \mathbb{R}$, $\theta_0 = 0$.

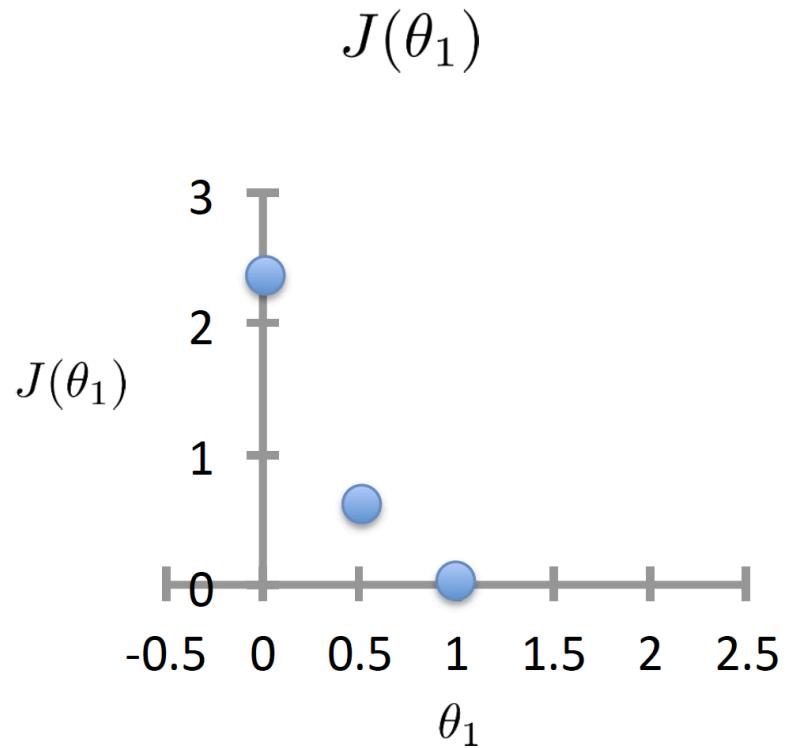
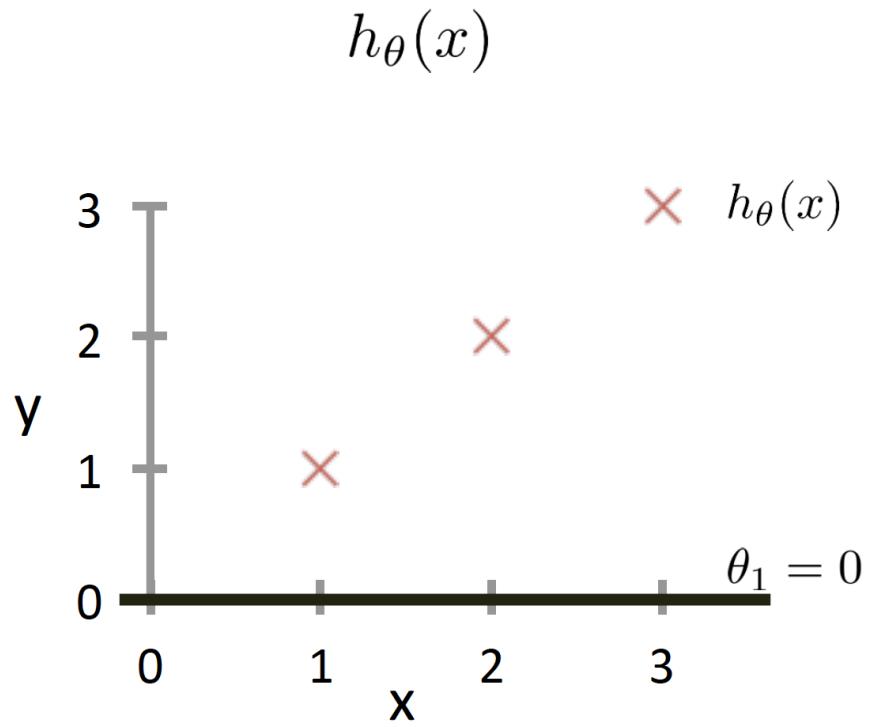
$$h_{\theta}(x) = \theta_0 + \theta_1 x = \theta_1 x$$



Intuition behind cost function

Assume $x \in \mathbb{R}$, $\theta_0 = 0$.

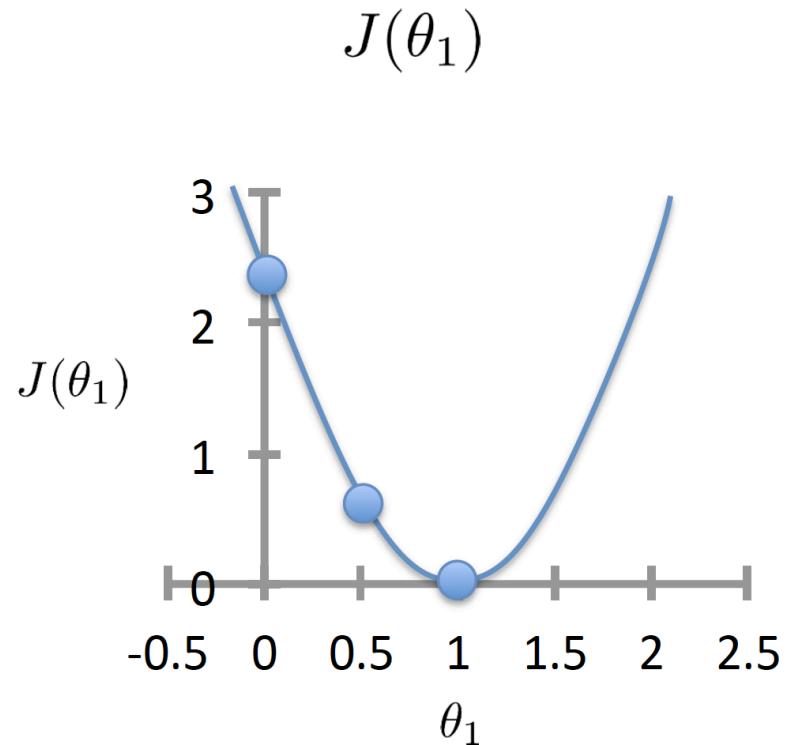
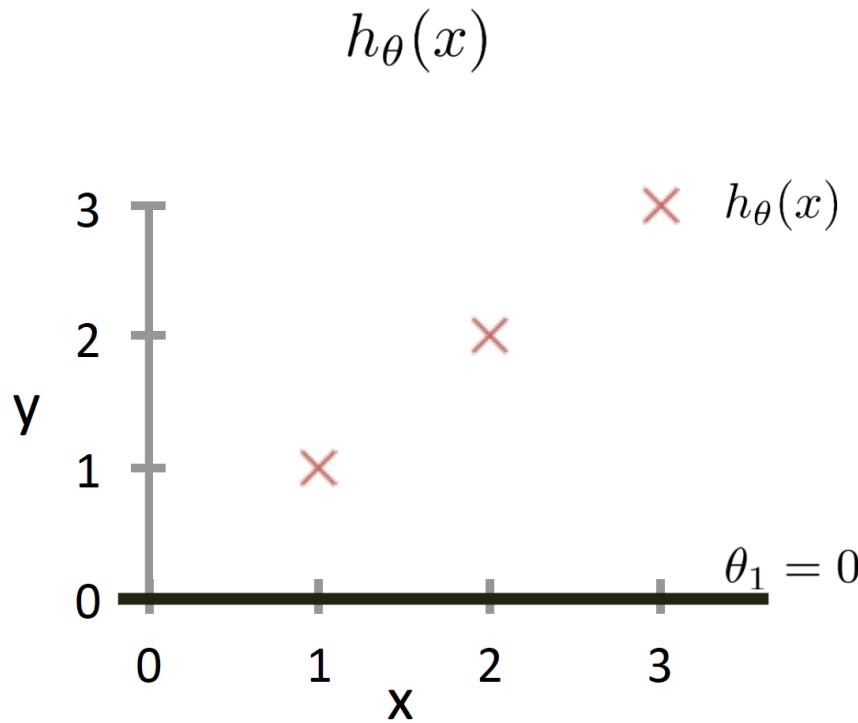
$$h_{\theta}(x) = \theta_0 + \theta_1 x = \theta_1 x$$



Intuition behind cost function

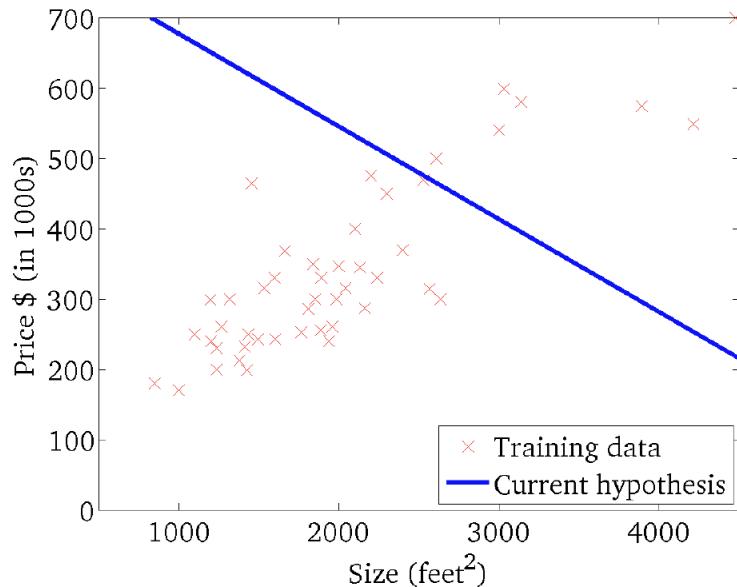
Assume $x \in \mathbb{R}$, $\theta_0 = 0$.

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

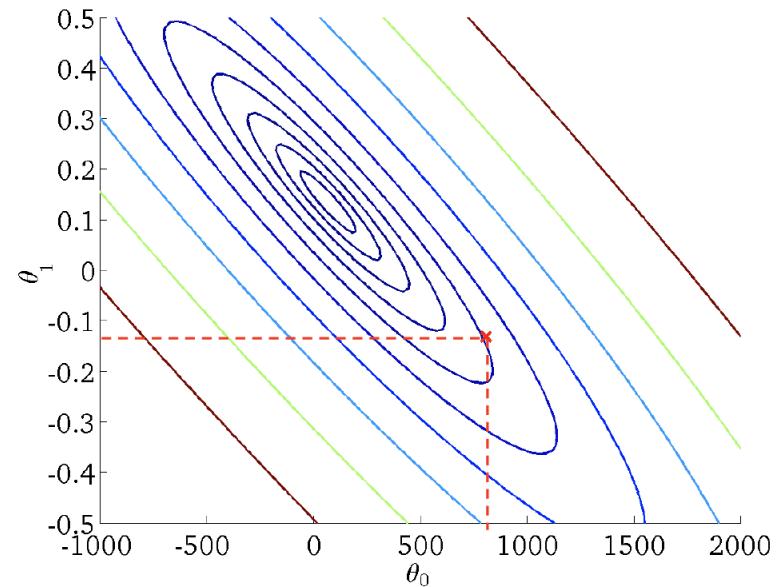


Intuition behind optimization

$$h_{\theta}(x)$$

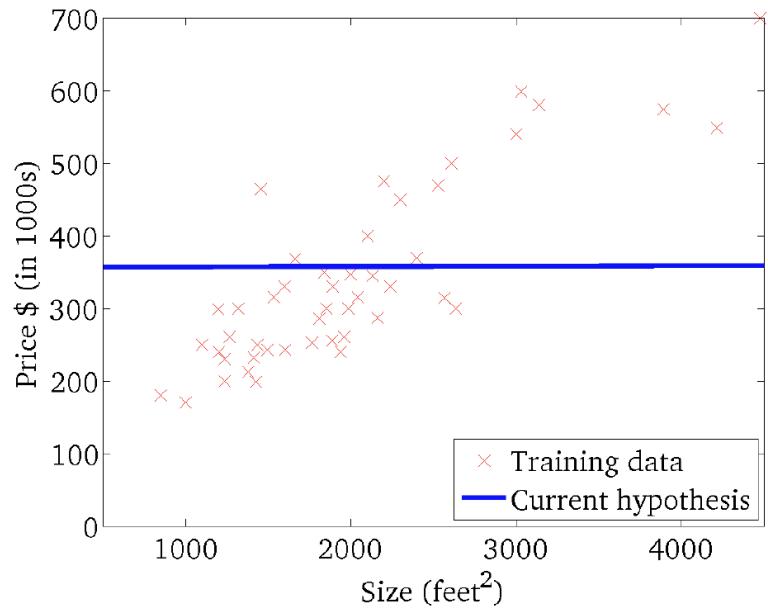


$$J(\theta_0, \theta_1)$$

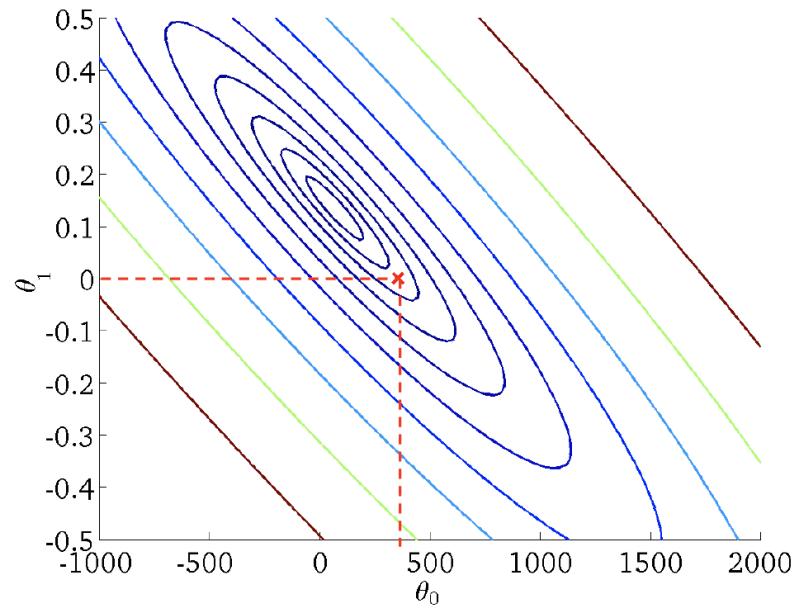


Intuition behind optimization

$$h_{\theta}(x)$$

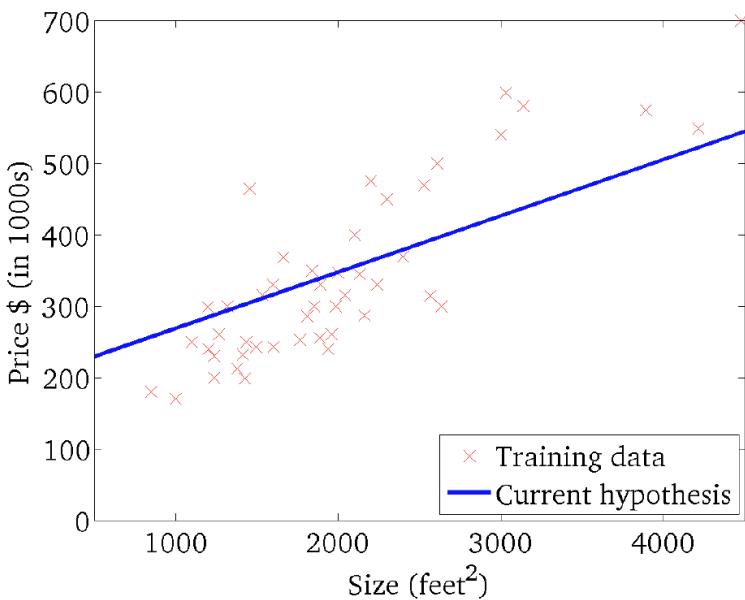


$$J(\theta_0, \theta_1)$$

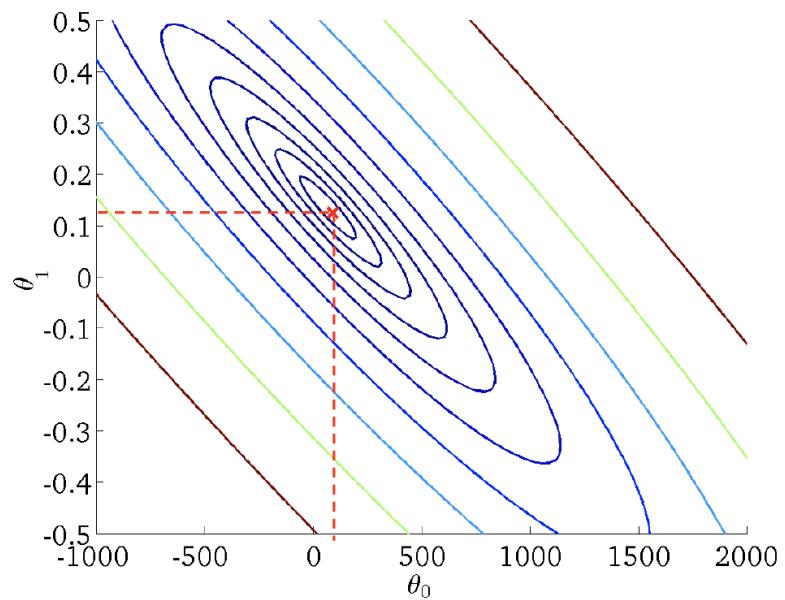


Intuition behind optimization

$$h_{\theta}(x)$$



$$J(\theta_0, \theta_1)$$



How to minimizing the loss

- ❖ Optimization methods
 - ❖ Gradient Descent
 - ❖ Stochastic Gradient Descent
 - ❖ Analytic solution
- ❖ ...many other approaches

Iterative optimization

Algorithm 2 Gradient Descent (J)

- 1: $t \leftarrow 0$
 - 2: Initialize $\theta^{(0)}$
 - 3: **repeat**
 - 4: $\theta^{(t+1)} \leftarrow \theta^{(t)} - \boxed{\eta \nabla J(\theta^{(t)})}$ **How to compute the gradient?**
 - 5: $t \leftarrow t + 1$
 - 6: **until** convergence
 - 7: Return final value of θ
-

Need to compute the gradient for the linear regression cost function (residual sum of squares RSS)

Compute the gradient of $J(\theta)$

- ❖ What is our objective function?

$$J(\mathbf{w}) = \frac{1}{2} \sum_{i=1}^m (y_i - \mathbf{w}^T \mathbf{x}_i)^2$$

- ❖ Gradient of $J(\mathbf{w})$ is:

$$\nabla J(\mathbf{w}^t) = \left[\frac{\partial J}{\partial w_1}, \frac{\partial J}{\partial w_2}, \dots, \frac{\partial J}{\partial w_d} \right]$$

Gradient of the cost

We are trying to minimize

$$J(\mathbf{w}) = \frac{1}{2} \sum_{i=1}^m (y_i - \mathbf{w}^T \mathbf{x}_i)^2$$

- ❖ The gradient is of the form

$$\nabla J(\mathbf{w}^t) = \left[\frac{\partial J}{\partial w_1}, \frac{\partial J}{\partial w_2}, \dots, \frac{\partial J}{\partial w_d} \right]$$

$$\begin{aligned}\frac{\partial J}{\partial w_j} &= \frac{\partial}{\partial w_j} \frac{1}{2} \sum_{i=1}^m (y_i - \mathbf{w}^T \mathbf{x}_i)^2 \\ &= \frac{1}{2} \sum_{i=1}^m \frac{\partial}{\partial w_j} (y_i - \mathbf{w}^T \mathbf{x}_i)^2 \\ &= \frac{1}{2} \sum_{i=1}^m 2(y_i - \mathbf{w}^T \mathbf{x}_i) \frac{\partial}{\partial w_j} (y_i - w_1 x_{i1} - \dots - w_j x_{ij} - \dots) \\ &= \frac{1}{2} \sum_{i=1}^m 2(y_i - \mathbf{w}^T \mathbf{x}_i)(-x_{ij}) \\ &= -\sum_{i=1}^m (y_i - \mathbf{w}^T \mathbf{x}_i)x_{ij}\end{aligned}$$

Sum of Error f Lec 8 Linear Regression

One element
of the gradient
vector

We are trying to minimize

$$J(\mathbf{w}) = \frac{1}{2} \sum_{i=1}^m (y_i - \mathbf{w}^T \mathbf{x}_i)^2$$

Gradient descent for LMS

1. Initialize \mathbf{w}^0
2. For $t = 0, 1, 2, \dots$ (*until total error is below a threshold*)
 1. Compute gradient of $J(\mathbf{w})$ at \mathbf{w}^T . Call it $\nabla J(\mathbf{w}^t)$

Evaluate the function for **each** training example to compute the error and construct the gradient vector

$$\frac{\partial J}{\partial w_j} = - \sum_{i=1}^m (y_i - \mathbf{w}^T \mathbf{x}_i) x_{ij}$$

One element of
 $\nabla J(\mathbf{w}^t)$



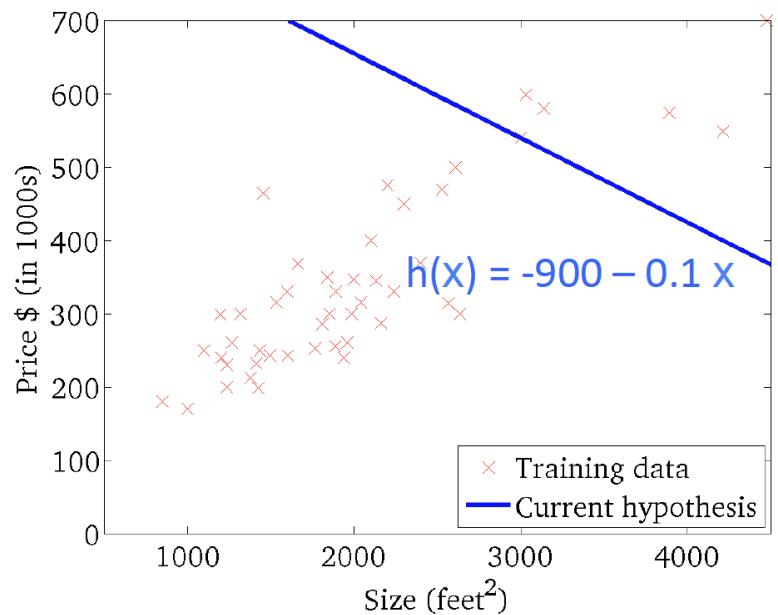
2. Update \mathbf{w} as follows:

$$\mathbf{w}^{t+1} = \mathbf{w}^t - r \nabla J(\mathbf{w}^t)$$

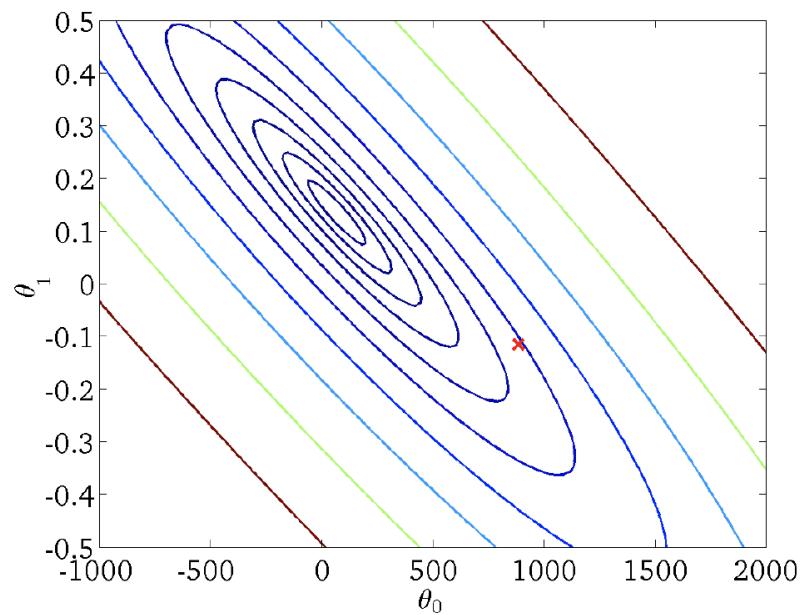
r : Called the **learning rate**

(For now, a small constant. We will get to this later)

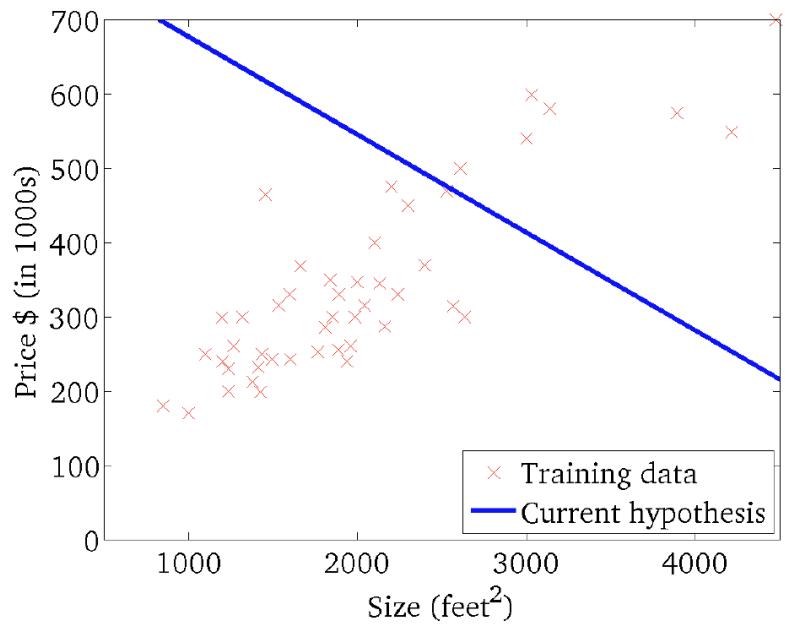
$$h_{\theta}(x)$$



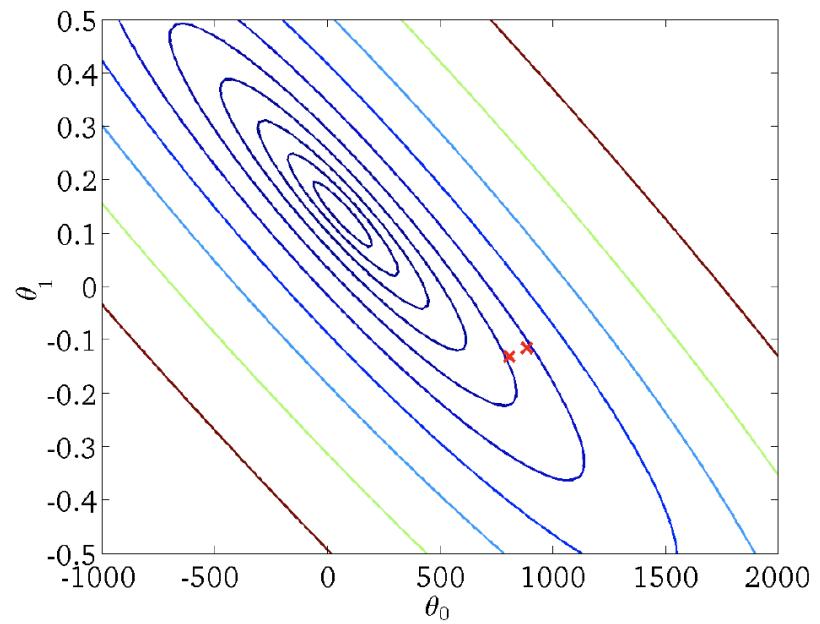
$$J(\theta_0, \theta_1)$$



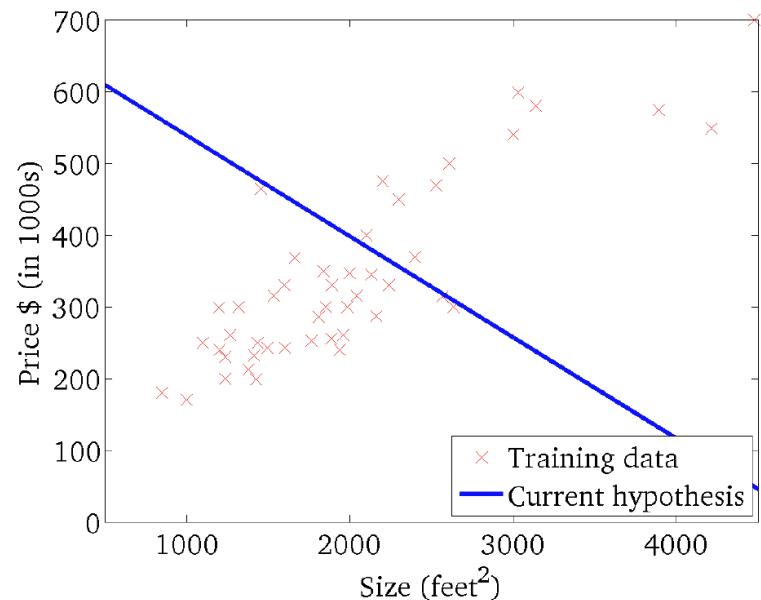
$$h_{\theta}(x)$$



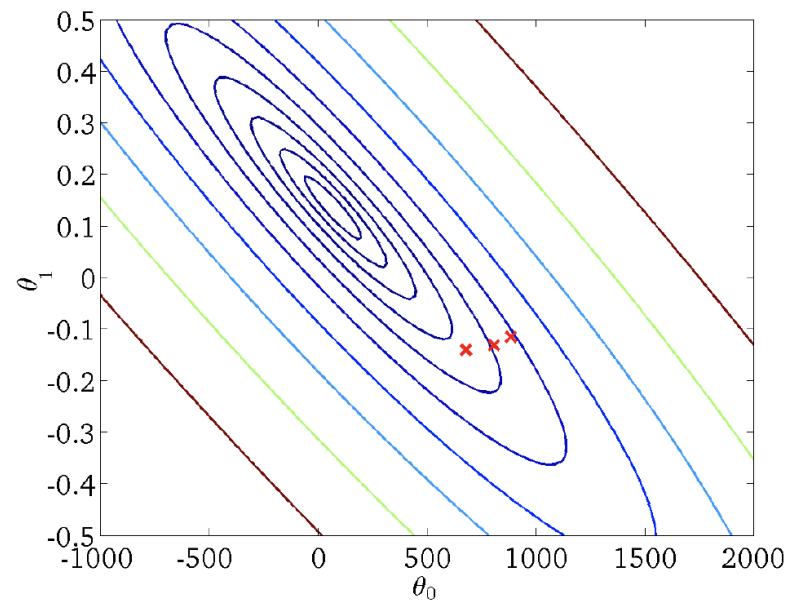
$$J(\theta_0, \theta_1)$$



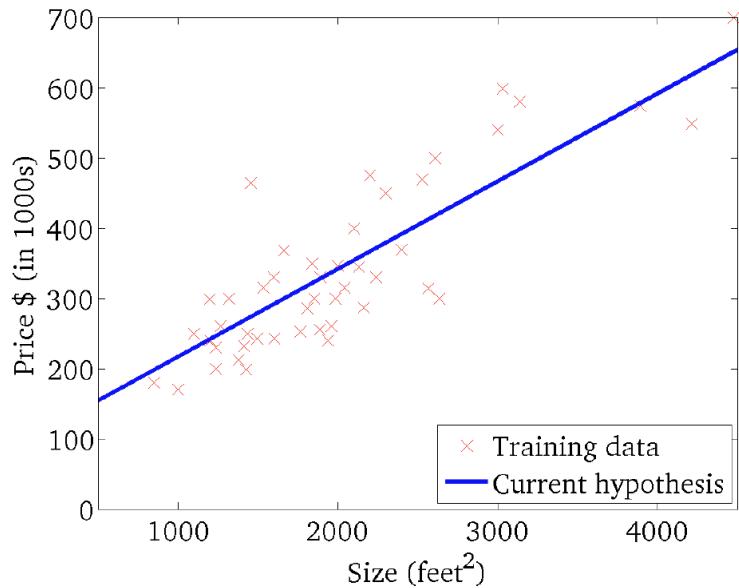
$$h_{\theta}(x)$$



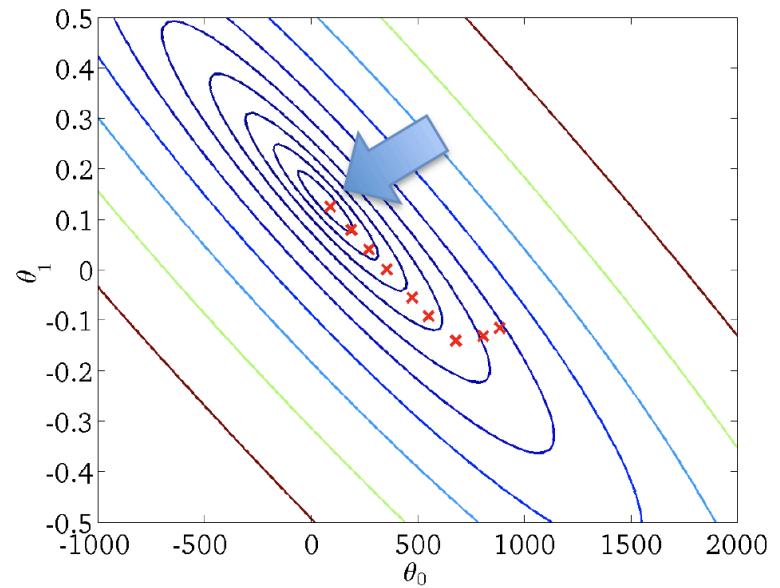
$$J(\theta_0, \theta_1)$$



$$h_{\theta}(x)$$



$$J(\theta_0, \theta_1)$$



We are trying to minimize

$$J(\mathbf{w}) = \frac{1}{2} \sum_{i=1}^m (y_i - \mathbf{w}^T \mathbf{x}_i)^2$$

Gradient descent for LMS

1 Initialize \mathbf{w}^0

The weight vector is not updated until *all errors are calculated*

Why not make early updates to the weight vector as soon as we encounter errors instead of waiting for a full pass over the data?

compute the error and construct the gradient vector

$$\frac{\partial J}{\partial w_j} = - \sum_{i=1}^m (y_i - \mathbf{w}^T \mathbf{x}_i) x_{ij}$$

One element
of ∇J

2. Update \mathbf{w} as follows:

$$\mathbf{w}^{t+1} = \mathbf{w}^t - r \nabla J(\mathbf{w}^t)$$

r : Called the *learning rate*

(For now, a small constant. We will get to this later)

Incremental/Stochastic gradient descent

- ❖ Repeat for each example (\mathbf{x}_i, y_i)
 - ❖ Pretend that the entire training set is represented by this single example
 - ❖ Use this example to calculate the gradient and update the model
- ❖ Contrast with *batch gradient descent* which makes one update to the weight vector for every pass over the data

Stochastic gradient descent

$$\text{If } f(w) = \frac{1}{|D|} \sum_i^{|D|} f_i(w)$$

$$\nabla f(w) = \frac{1}{|D|} \sum_i \nabla f_i(w) = E_{i \sim D} \nabla f_i(w)$$

- ❖ Approximate the true gradient by a gradient at a single example at a time

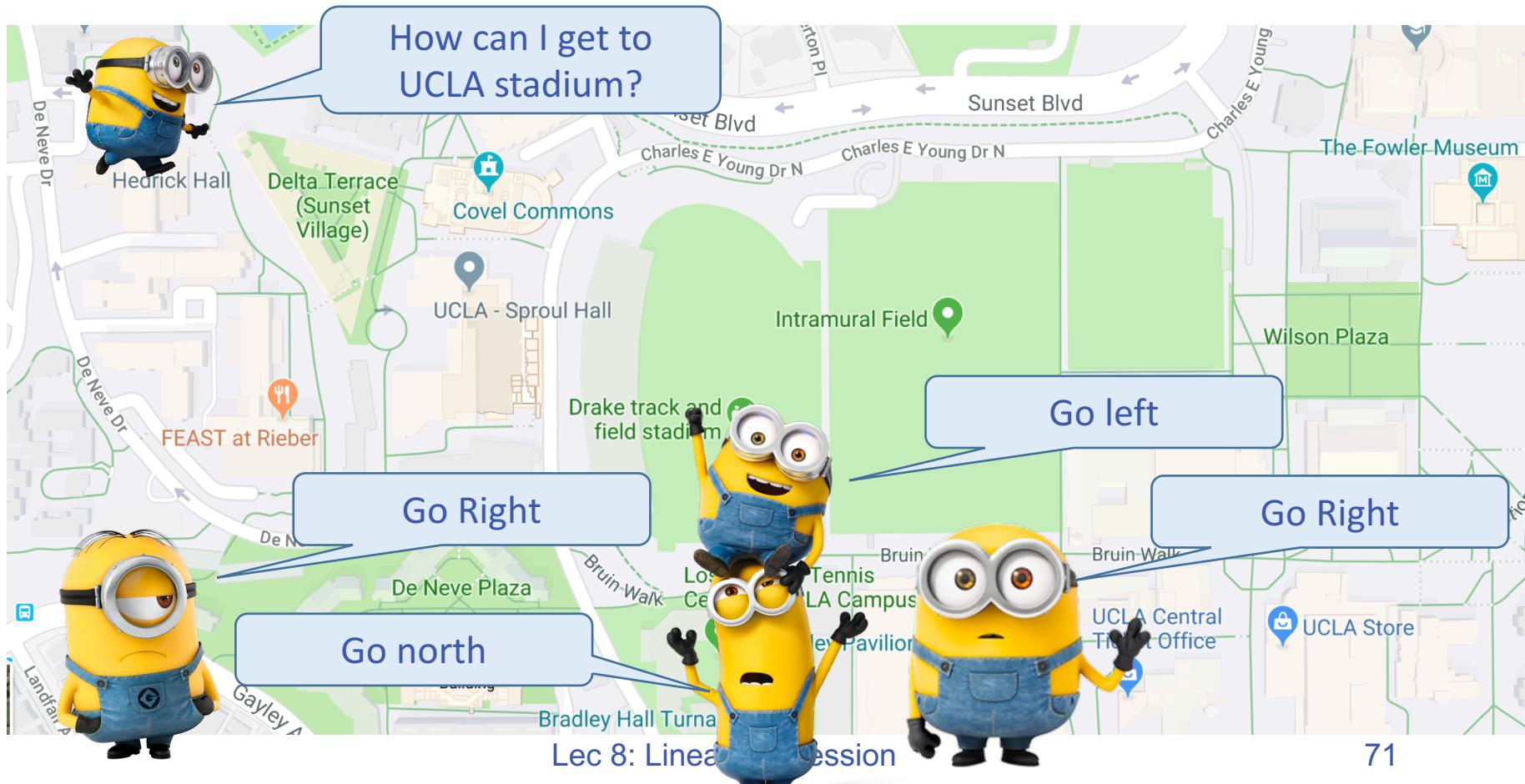
Repeat until converge:

Randomly pick one sample (x_i, y_i)

Update $w \leftarrow w - \eta \nabla f_i(w)$

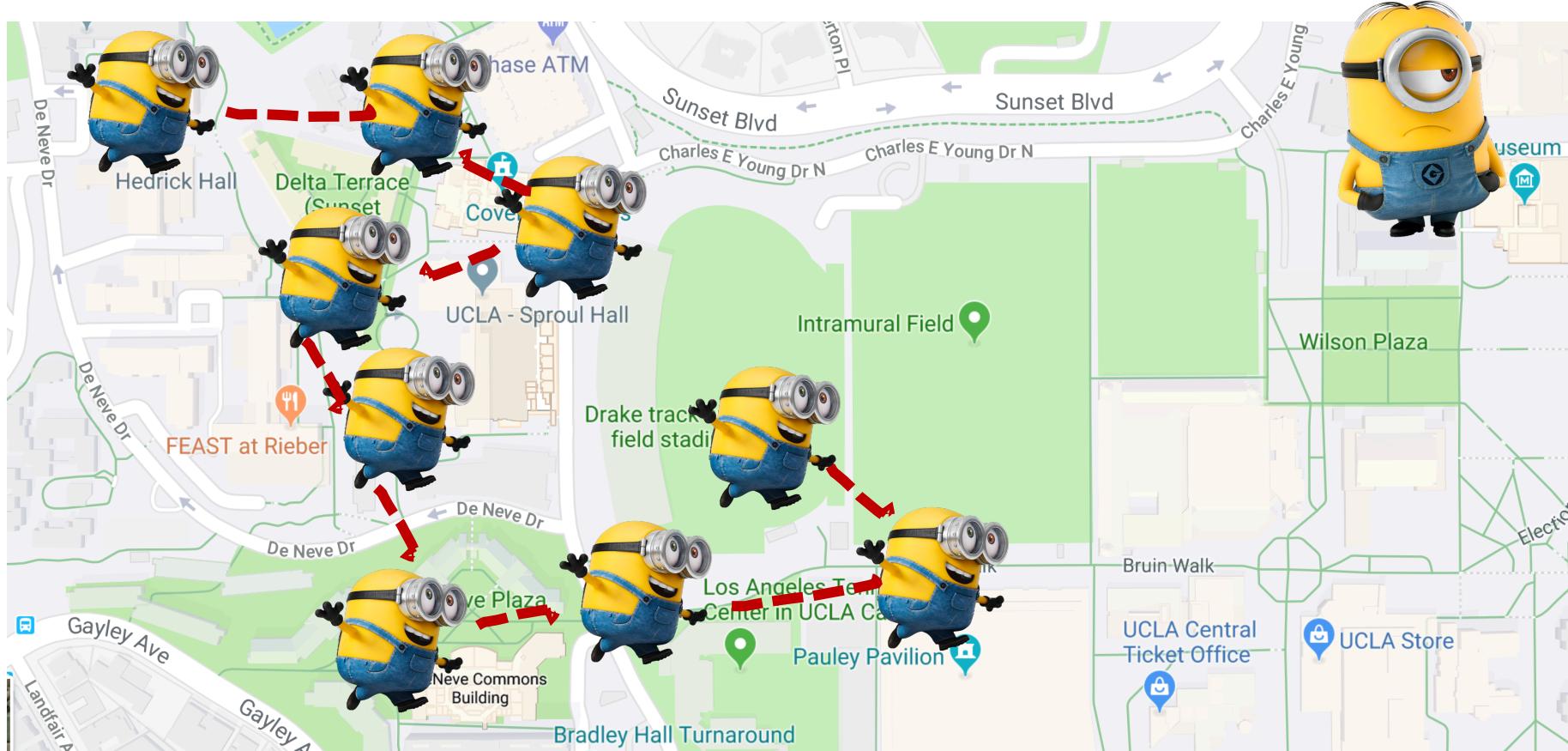
Intuition

- ❖ Asking direction. Gradient descent: compute gradient of all instances.



Intuition

- ❖ Asking direction. Stochastic Gradient descent: compute approximate gradient by one instance



Stochastic gradient Descent

Given a training set $\mathcal{D} = \{(x, y)\}$

1. Initialize $w \leftarrow \mathbf{0} \in \mathbb{R}^n$
2. For epoch 1 ... T :
3. For (x, y) in \mathcal{D} :
4. Update $w \leftarrow w - \eta \nabla f(w)$
5. Return w

We will see more example later in this lecture

The Perceptron Algorithm [Rosenblatt 1958]

Given a training set $\mathcal{D} = \{(x, y)\}$

1. Initialize $w \leftarrow \mathbf{0} \in \mathbb{R}^n$
2. For epoch 1 ... T :
3. For (x, y) in \mathcal{D} :
4. if $y(w^\top x) < 0$
5. $w \leftarrow w + \eta yx$
6. Return w

Prediction: $y^{\text{test}} \leftarrow \text{sgn}(w^\top x^{\text{test}})$

The Perceptron Algorithm [Rosenblatt 1958]

Given a training set $\mathcal{D} = \{(x, y)\}$

1. Initialize $w \leftarrow \mathbf{0} \in \mathbb{R}^n$
2. For epoch 1 ... T :
3. For (x, y) in \mathcal{D} :
4. if $y(w^\top x) < 0$
5. $w \leftarrow w + \eta yx$
6. Return w

Prediction: y^{test}

Perceptron effectively minimizing:

$$\sum_i \max(0, 1 - y_i(w^\top x_i))$$

Stochastic gradient descent for linear regression

1. Initialize \mathbf{w}
2. For $t = 0, 1, 2, \dots$ (until error below some threshold)
 - ❖ For each training example (\mathbf{x}_i, y_i) :
 - ❖ Update \mathbf{w} . For each element of the weight vector (w_j):

$$w_j^{t+1} = w_j^t + r(y_i - \mathbf{w}^T \mathbf{x}_i)x_{ij}$$

Incremental/Stochastic gradient descent

1. Initialize \mathbf{w}
2. For $t = 0, 1, 2, \dots$ (until error below some threshold)
 - ❖ For each training example (\mathbf{x}_i, y_i) :
 - ❖ Update \mathbf{w} . For each element of the weight vector (w_j):

$$w_j^{t+1} = w_j^t + r(y_i - \mathbf{w}^T \mathbf{x}_i)x_{ij}$$

Contrast with the previous method, where the weights are updated only after all examples are processed once

Online/Incremental algorithms are often preferred when the training set is very large

Learning Rates and Convergence Remarks

- ❖ In the general (non-separable) case the learning rate r must decrease to approach zero to guarantee convergence
- ❖ The learning rate is called the *step size*.
 - ❖ More sophisticated algorithms choose the step size automatically and converge faster
- ❖ Choosing a better starting point can also have impact
- ❖ Gradient descent and its stochastic version are very simple algorithms
 - ❖ Yet, almost all the algorithms we will learn in the class can be traced back to gradient decent algorithms for different loss functions and different hypotheses spaces

Linear regression: Summary

- ❖ **What we want:** Predict a real valued output using a feature representation of the input
- ❖ **Assumption:** Output is a linear function of the inputs
- ❖ Learning by minimizing total cost
 - ❖ Gradient descent and stochastic gradient descent to find the *best* weight vector
 - ❖ This particular optimization can be computed directly by framing the problem as a matrix problem

Exercises

We are trying to minimize

$$J(\mathbf{w}) = \frac{1}{2} \sum_{i=1}^m (y_i - \mathbf{w}^T \mathbf{x}_i)^2$$

1. LMS regression can be solved analytically. Given a dataset

$D = \{(x_1, y_1), (x_2, y_2), \dots, (x_m, y_m)\}$, define matrix X and vector Y as follows:

$$X = \begin{bmatrix} \mathbf{x}_1 & \mathbf{x}_2 & \cdots \mathbf{x}_m \end{bmatrix}_{d \times m} \quad Y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}_{m \times 1}$$

Show that the optimization problem we saw earlier is equivalent to

$$\min_{\mathbf{w}} (X^T \mathbf{w} - Y)^T (X^T \mathbf{w} - Y)$$

This can be solved analytically. Show that the solution \mathbf{w}^* is

$$\mathbf{w}^* = (X X^T)^{-1} X Y$$

Probabilistic interpretation

(advanced topic: not covered in the exam)

- Noisy observation model

$$Y = \theta_0 + \theta_1 X + \eta$$

where $\eta \sim \mathcal{N}(0, \sigma^2)$ is a Gaussian random variable

- Likelihood of one training sample (x_n, y_n)

$$p(y_n|x_n; \boldsymbol{\theta}) = \mathcal{N}(\theta_0 + \theta_1 x_n, \sigma^2) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{[y_n - (\theta_0 + \theta_1 x_n)]^2}{2\sigma^2}}$$

[Advanced topic] **Probabilistic interpretation**

Log-likelihood of the training data \mathcal{D} (assuming i.i.d)

$$\mathcal{LL}(\theta) = \log P(\mathcal{D})$$

$$\begin{aligned} &= \log \prod_{n=1}^N p(y_n | x_n) = \sum_n \log p(y_n | x_n) \\ &= \sum_n \left\{ -\frac{[y_n - (\theta_0 + \theta_1 x_n)]^2}{2\sigma^2} - \log \sqrt{2\pi}\sigma \right\} \\ &= -\frac{1}{2\sigma^2} \sum_n [y_n - (\theta_0 + \theta_1 x_n)]^2 - \frac{N}{2} \log \sigma^2 - N \log \sqrt{2\pi} \\ &= -\frac{1}{2} \left\{ \frac{1}{\sigma^2} \sum_n [y_n - (\theta_0 + \theta_1 x_n)]^2 + N \log \sigma^2 \right\} + \text{const} \end{aligned}$$

What is the relationship between minimizing J and maximizing the log-likelihood?

[Advanced topic] **Probabilistic interpretation**

Estimating σ , θ_0 and θ_1 can be done in two steps

- Maximize over θ_0 and θ_1

$$\max \log P(\mathcal{D}) \Leftrightarrow \min \sum_n [y_n - (\theta_0 + \theta_1 x_n)]^2 \leftarrow \text{That is } J(\boldsymbol{\theta})!$$

- Maximize over $s = \sigma^2$ (we could estimate σ directly)

$$\begin{aligned} \frac{\partial \log P(\mathcal{D})}{\partial s} &= -\frac{1}{2} \left\{ -\frac{1}{s^2} \sum_n [y_n - (\theta_0 + \theta_1 x_n)]^2 + N \frac{1}{s} \right\} = 0 \\ \rightarrow \sigma^{*2} &= s^* = \frac{1}{N} \sum_n [y_n - (\theta_0 + \theta_1 x_n)]^2 \end{aligned}$$