Lecture 8: Linear Regression
Winter 2018

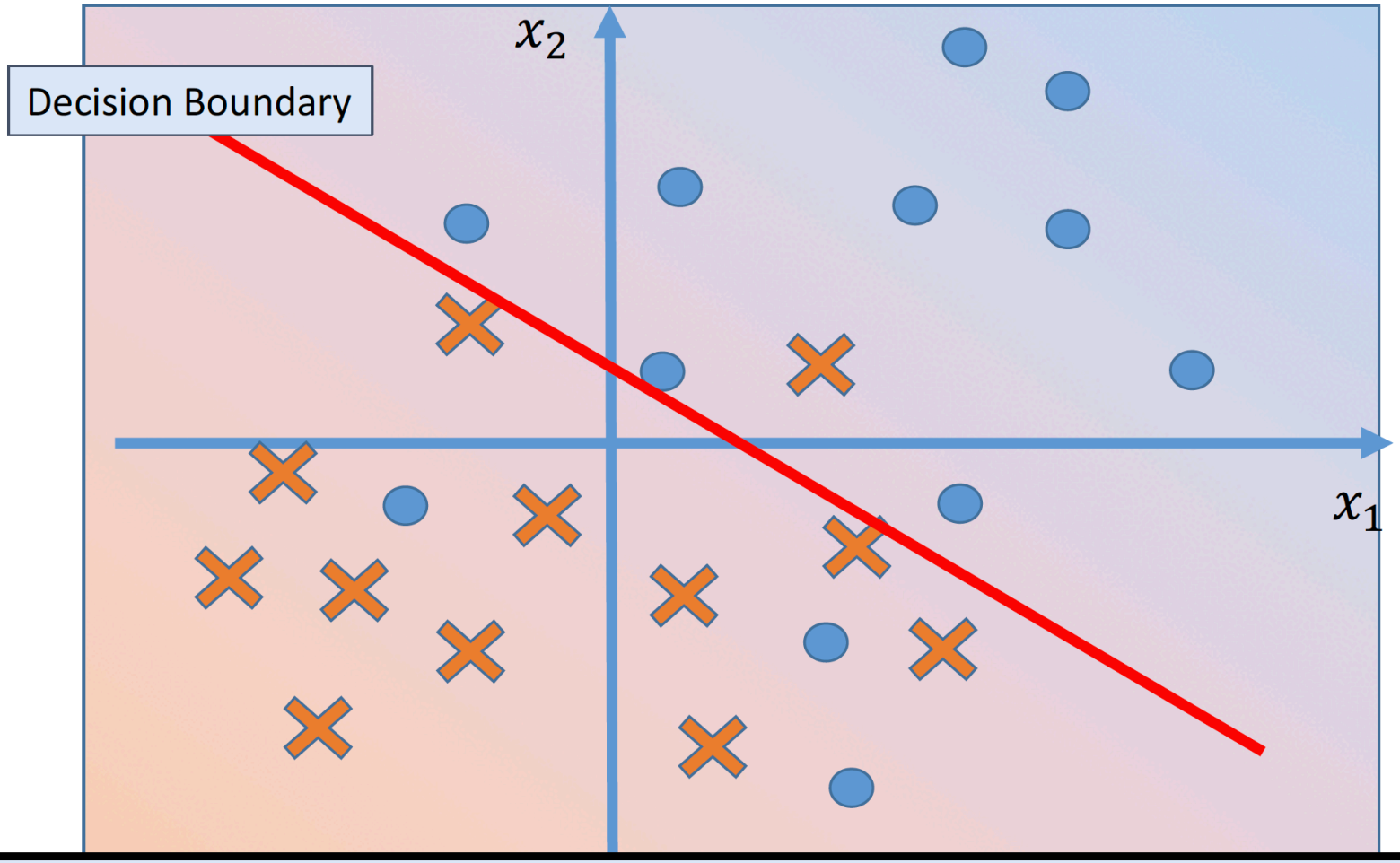
Kai-Wei Chang

CS @ UCLA

kw+cm146@kwchang.net

The instructor gratefully acknowledges Dan Roth, Vivek Srikuar, 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.

Prediction by logistic regression



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

The hypothesis space for logistic regression

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

All functions of the form

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

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

$$\mathrm{h}_w(x) \approx P(y=1\,|x)$$

Maximum likelihood estimator for logistic regression

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

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

Minimizing Negative Log-likelihood

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

Alternative Formulation

- Sometimes, we define
  - Inputs: Feature vectors $x \in R^N$
  - Labels: $y \in \{0, 1\}$ (or $y \in [0, 1]$)
- Training data
  - $S = \{(x_i, y_i)\}$, m examples
- Hypothesis space

$$H = \{ h \mid h : X \to Y, h(x) = \sigma(w^T x + b) \}$$

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

Alternative representation

$$P(y|\mathbf{w},\mathbf{x}) = \boldsymbol{\sigma}\left(\boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{x}\right) = \frac{1}{1+\exp(-\mathrm{y}\quad \mathbf{w}^T\mathbf{x}_i)}$$

Maximum Likelihood estimator:

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

$$P(y_i|\mathbf{x}_i,\mathbf{w}) = \begin{cases} \sigma(w^T x) & if\ y_i = 1 \\ 1-\sigma(w^T x) & if\ y_i = 0 \end{cases}$$

Alternative representation

$$P(y|\mathbf{w},\mathbf{x}) = \boldsymbol{\sigma}\left(\boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{x}\right) = \frac{1}{1+\exp(-\mathrm{y}\,\mathbf{w}^T\mathbf{x}_i)}$$

Maximum Likelihood estimator:

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

Equivalent to solving

$$P(y|\mathbf{x}_i,\mathbf{w}) = \begin{cases} \sigma(w^T x) & if\ y=1 \\ 1-\sigma(w^T x) & if\ y=0 \end{cases}$$

$$\underset{\mathbf{w}}{\operatorname{argmax}} \prod_{i=1}^{m} P(y=1|\mathbf{x}_i,\mathbf{w})^{y_i} P(y=0|\mathbf{x}_i,\mathbf{w})^{1-y_i}$$

Alternative representation

$$P(y|\mathbf{w},\mathbf{x}) = \boldsymbol{\sigma}\left(\boldsymbol{w}^T\boldsymbol{x}\right) = \frac{1}{1+\exp(-\mathrm{y}\,\mathbf{w}^T\mathbf{x}_i)}$$

Maximum Log-Likelihood estimator:

$$\underset{\mathbf{w}}{\operatorname{argmax}} \prod_{i=1}^{m} P(y=1|\mathbf{x}_i,\mathbf{w})^{y_i} P(y=0|\mathbf{x}_i,\mathbf{w})^{1-y_i}$$

Equivalent to solving

$$P(y|\mathbf{x}_i,\mathbf{w}) = \begin{cases} \sigma(w^T x) & if\ y=1 \\ 1-\sigma(w^T x) & if\ y=0 \end{cases}$$

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

Linear Regression

Outline

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

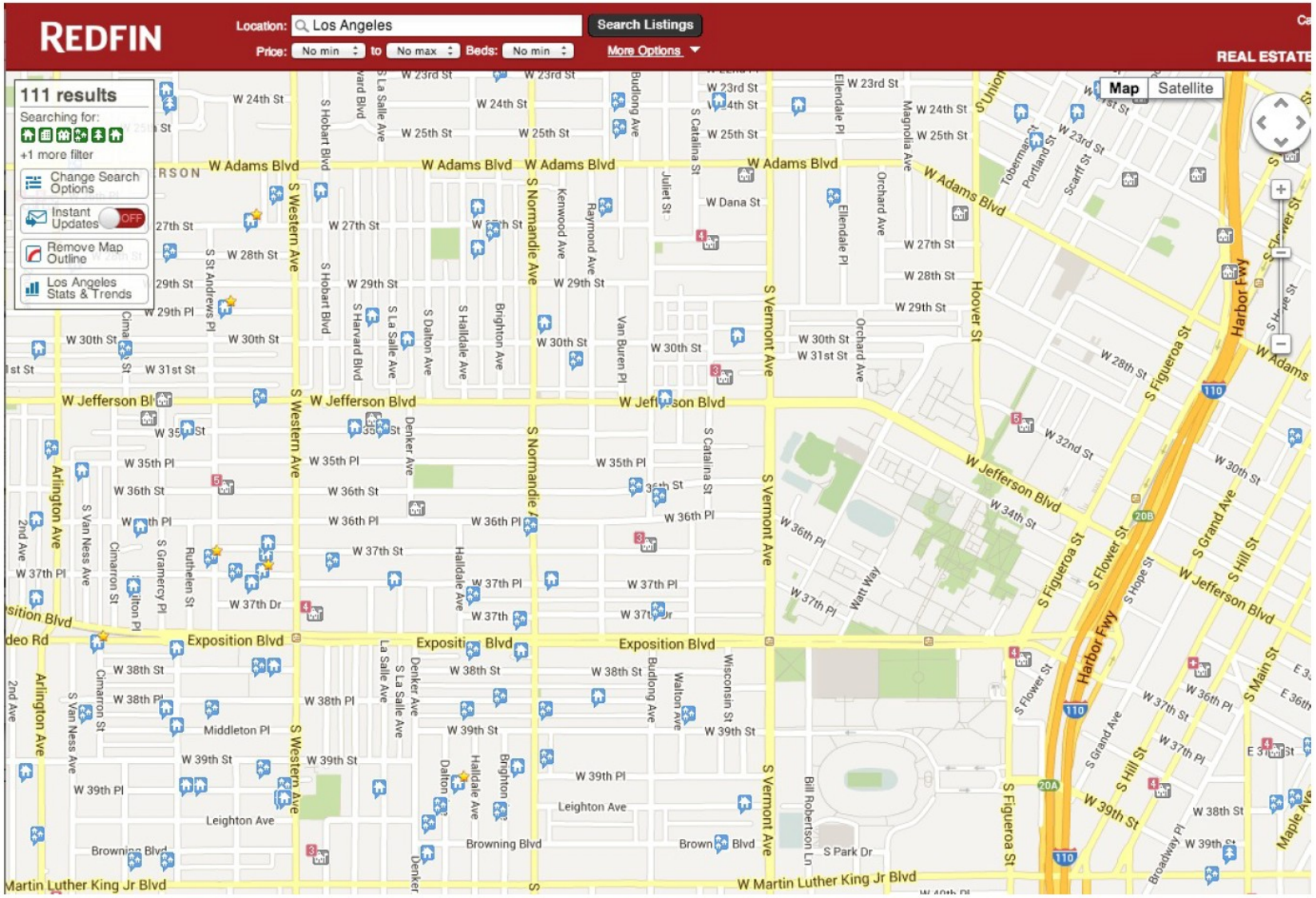
Regression

- 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

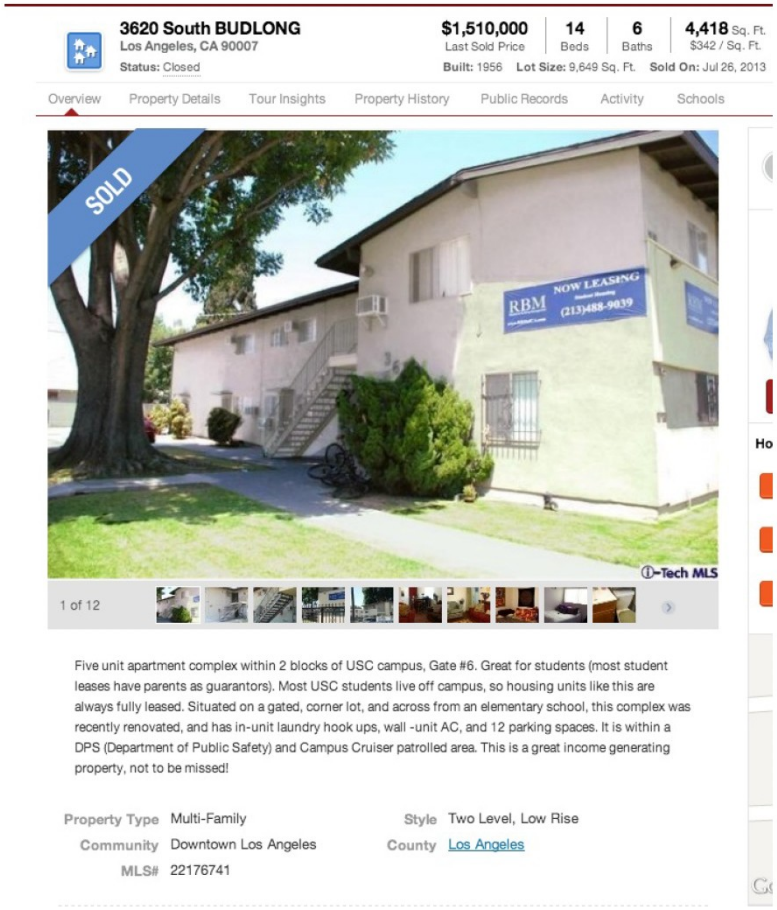
Regression v.s. Classification

- 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


REDFIN
Location: Los Angeles
Search Listings
Price: No min to No max Beds: No min
More Options
REAL ESTATE
111 results
Searching for:
+1 more filter
Change Search Options
Instant Updates OFF
Remove Map Outline
Los Angeles Stats & Trends
Map
Satellite
W Adams Blvd
W Jefferson Blvd
Exposition Blvd
W Martin Luther King Jr Blvd
S Western Ave
S Normandie Ave
S Vermont Ave
S Figueroa St
Harbor Fwy
Hoover St
Arlington Ave
Leighton Ave
Browning Blvd
Middleton Pl
S Park Dr
Bill Robertson Ln
S Grand Ave
S Hill St
S Main St
S Flower St
S Hope St
W Jefferson Blvd
W 24th St
W 25th St
W 27th St
W 28th St
W 29th St
W 30th St
W 31st St
W 32nd St
W 34th St
W 35th St
W 35th Pl
W 36th St
W 36th Pl
W 37th St
W 37th Pl
W 38th St
W 38th Pl
W 39th St
W 39th Pl
W 23rd St
W Dana St

Feature can be used



3620 South BUDLONG
Los Angeles, CA 90007
Status: Closed

$1,510,000 Last Sold Price | **14** Beds | **6** Baths | **4,418** Sq. Ft. $342 / 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

SOLD

RBM NOW LEASING (213)488-9039

i-Tech MLS

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
- Remodeled
- Oven, Range

Laundry Information
- Inside Laundry

Heating & Cooling
- Wall Cooling Unit(s)

Multi-Unit Information

Community Features
- Units in Complex (Total): 5

Multi-Family Information
- # Leased: 5
- # of Buildings: 1
- Owner Pays Water
- Tenant Pays Electricity, Tenant Pays Gas

Unit 1 Information
- # of Beds: 2
- # of Baths: 1
- Unfurnished
- Monthly Rent: $1,700

Unit 2 Information
- # of Beds: 3
- # of Baths: 1
- Unfurnished
- Monthly Rent: $2,250

Unit 3 Information
- Unfurnished

Unit 4 Information
- # of Beds: 3
- # of Baths: 1
- Unfurnished
- Monthly Rent: $2,350

Unit 5 Information
- # of Beds: 3
- # of Baths: 2
- Unfurnished
- Monthly Rent: $2,325

Unit 6 Information
- # of Beds: 3
- # of Baths: 1
- Monthly Rent: $2,250

Property / Lot Details

Property Features
- Automatic Gate, Card/Code Access

Lot Information
- Lot Size (Sq. Ft.): 9,649
- Lot Size (Acres): 0.2215
- Lot Size Source: Public Records

- Automatic Gate, Lawn, Sidewalks
- Corner Lot, Near Public Transit

Property Information
- Updated/Remodeled
- Square Footage Source: Public Records

- Tax Parcel Number: 5040017019

Parking / Garage, Exterior Features, Utilities & Financing

Parking Information
- # of Parking Spaces (Total): 12
- Parking Space
- Gated

Building Information
- Total Floors: 2

Utility Information
- Green Certification Rating: 0.00
- Green Location: Transportation, Walkability
- Green Walk Score: 0
- Green Year Certified: 0

Financial Information
- Capitalization Rate (%): 6.25
- Actual Annual Gross Rent: $128,331
- Gross Rent Multiplier: 11.29

Location Details, Misc. Information & Listing Information

Location Information
- Cross Streets: W 36th Pl

Expense Information
- Operating: $37,664

Listing Information
- 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: | prediction - sale price|
  - *squared* difference: (prediction - 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				$90^2 + 107^2 + 38^2 + 0 + \cdots$

- 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: $\boldsymbol{x} \in \mathbb{R}^{\mathsf{D}}$ (covariates, predictors, features, etc)
- Output: $y \in \mathbb{R}$ (responses, targets, outcomes, outputs, etc)
- Hypotheses/Model: $h_{\boldsymbol{w},b}$, with $h_{\boldsymbol{w},b}(\boldsymbol{x}) = b + \sum_d w_d x_d = b + \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}$

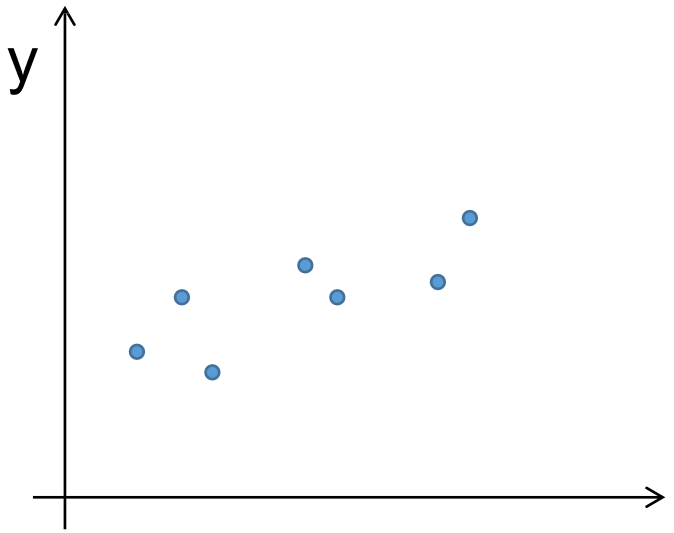
  $\boldsymbol{w} = [w_1\ w_2\ \cdots\ w_{\mathsf{D}}]^{\mathrm{T}}$: *weights*

  b is called the bias or offset or intercept term.

  $\boldsymbol{\theta} = [b\ w_1\ w_2\ \cdots\ w_{\mathsf{D}}]^{\mathrm{T}}$

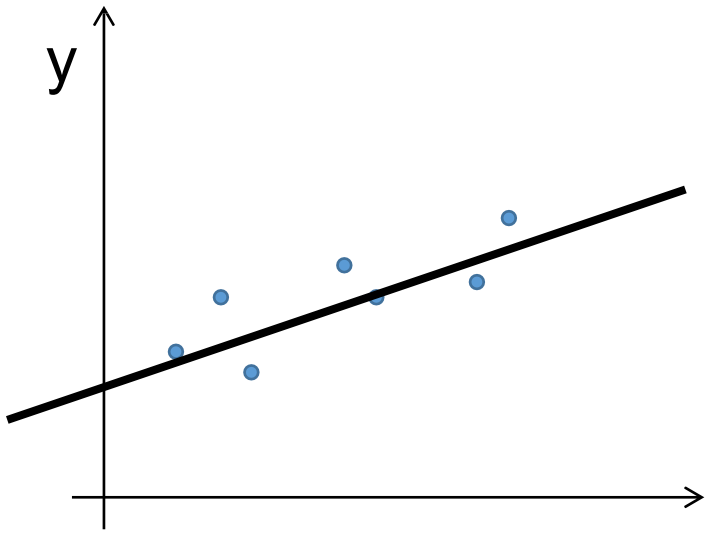
- Training data: $\mathcal{D} = \{(\boldsymbol{x}_n, y_n), n = 1, 2, \ldots, \mathsf{N}\}$

Examples



One dimensional input

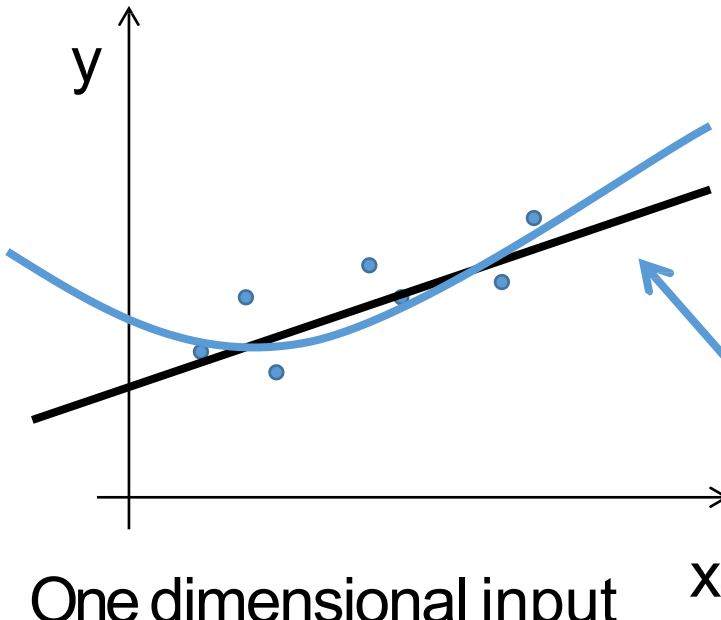
Examples



Predict using $y = b + w\, x_1$

One dimensional input

Examples

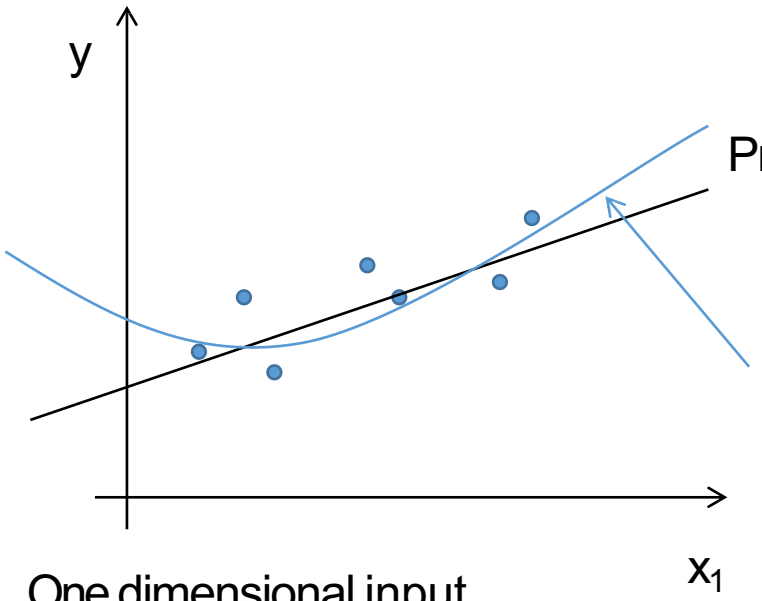


One dimensional input

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

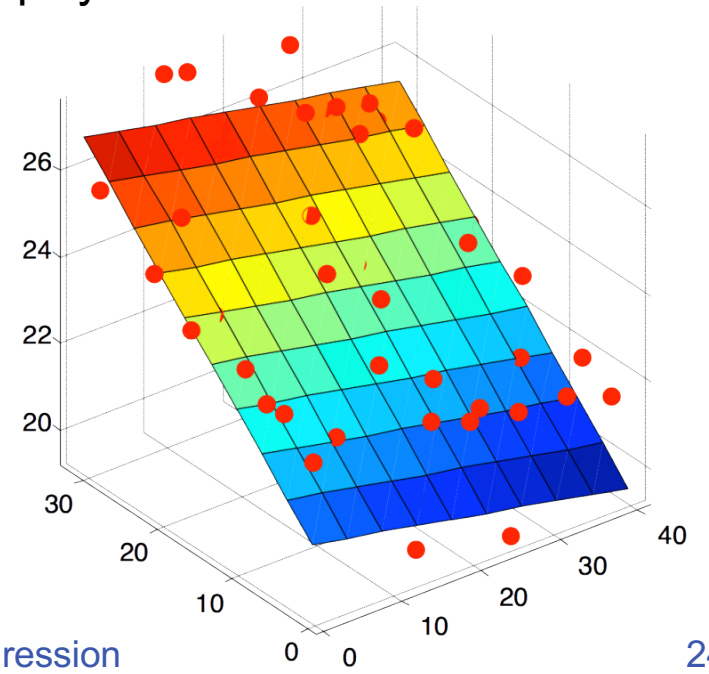
Examples



One dimensional input

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 $y = w_1x_1 + w_2x_2 + b$

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 biasterm

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

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

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

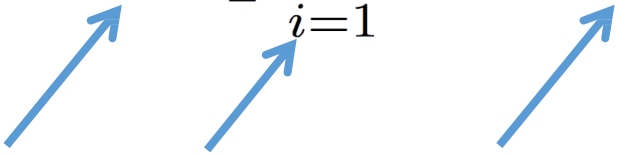
$$J(\mathbf{w}) = \frac{1}{2} \sum_{i=1}^{m} \left(y_i - \mathbf{w}^T \mathbf{x}_i \right)^2$$

Sum of squared costs over the training set

One strategy for learning: *Find the **w** with least cost on this data*

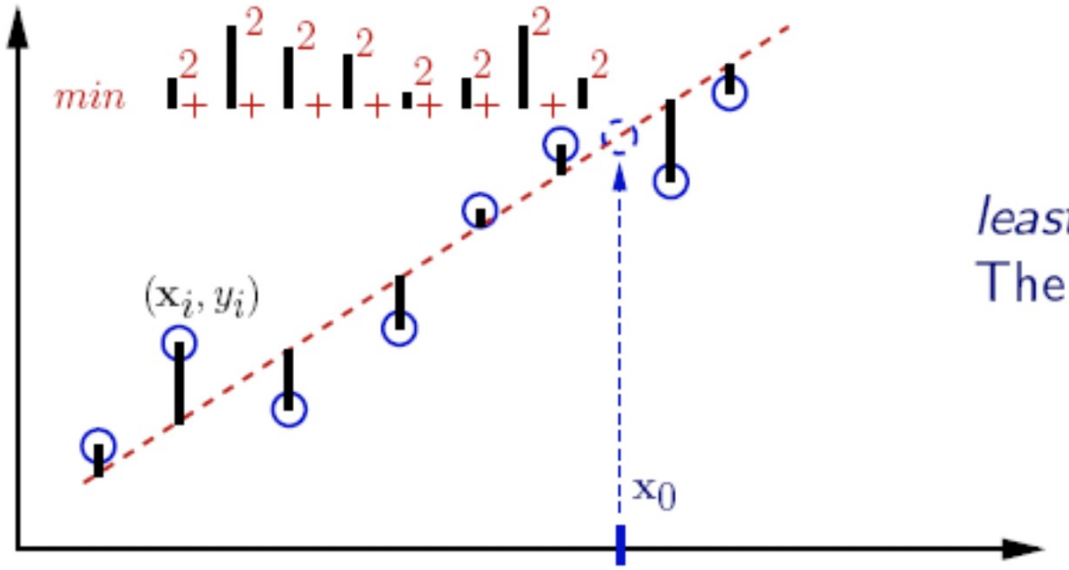
Least Mean Squares (LMS) Regression

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



Learning: minimizing mean squared error

Example



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

Least Mean Squares (LMS) Regression

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

Learning: minimizing mean squared error

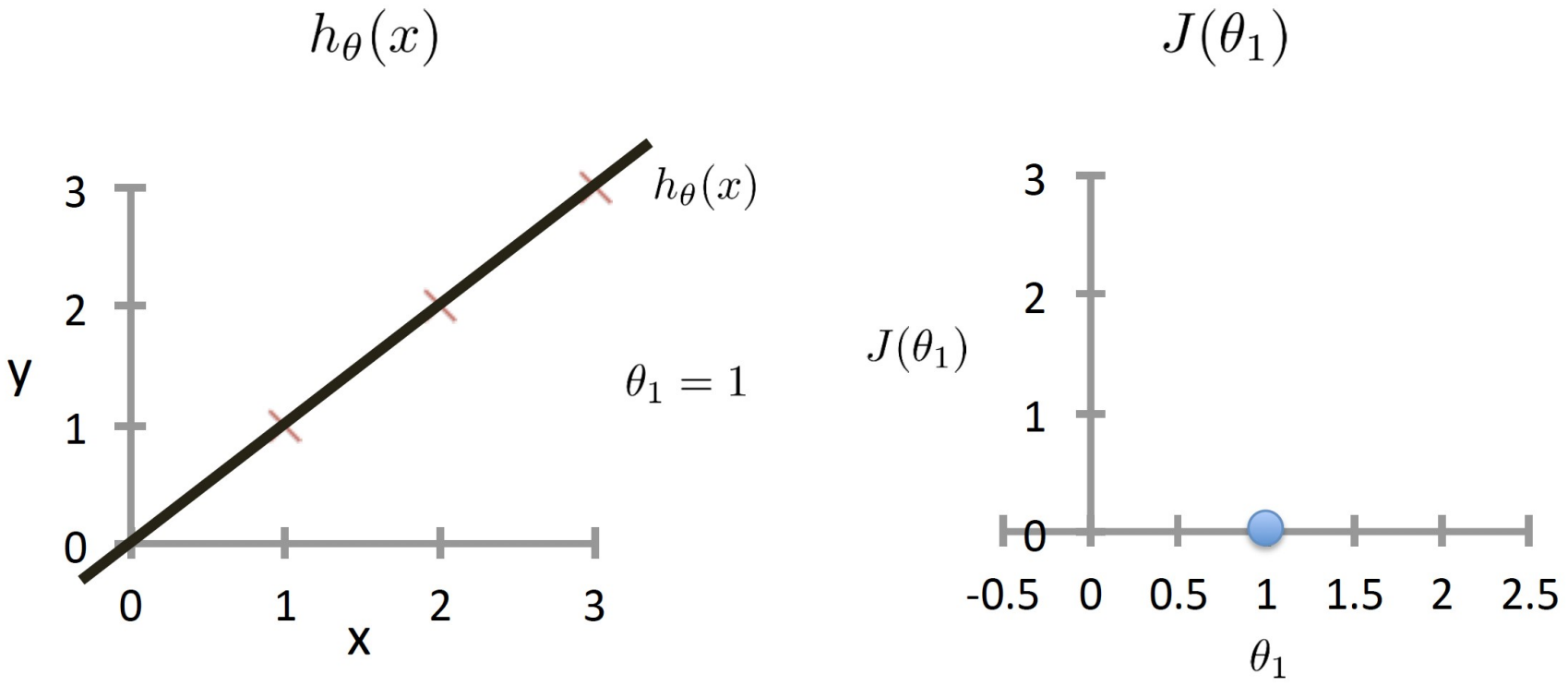
Different strategies exist for *learning by optimization*

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

Intuition behind the cost function

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

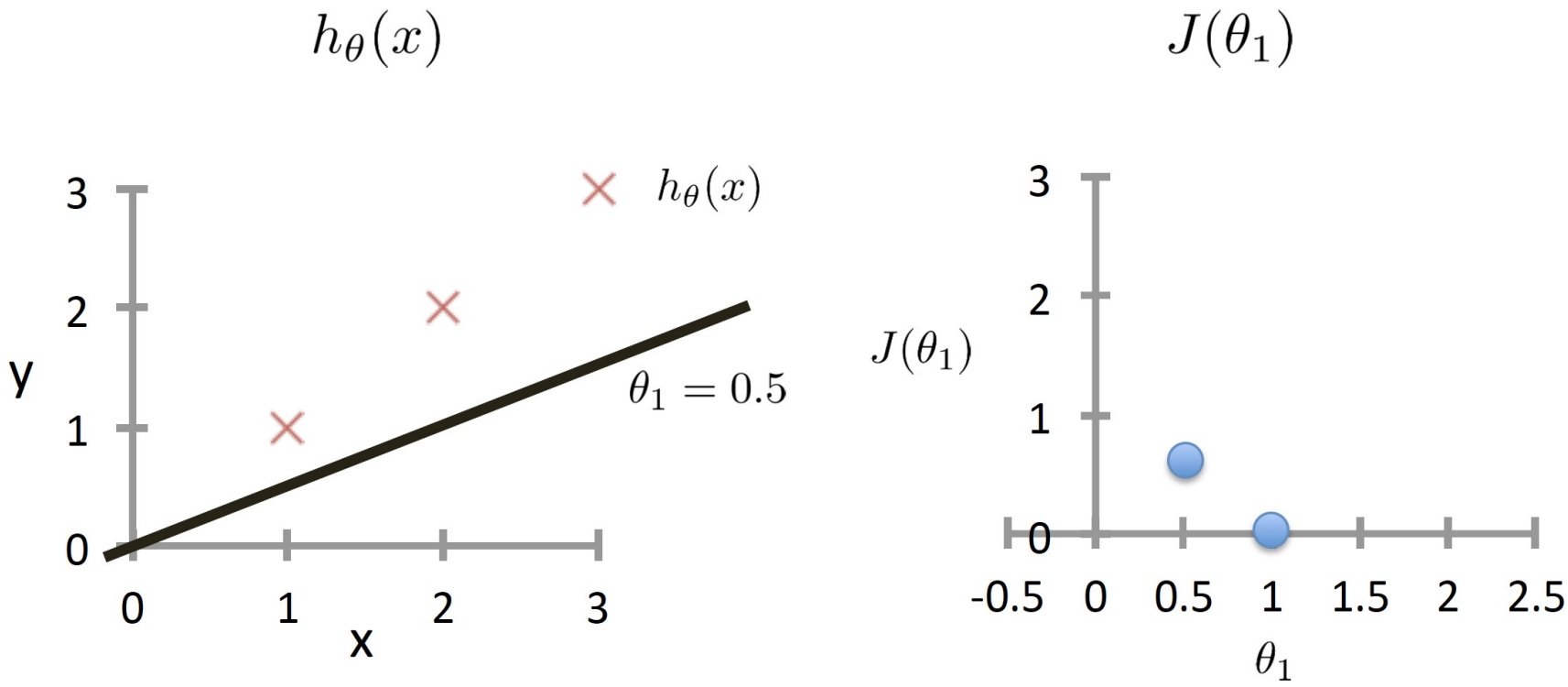
$h_{\boldsymbol{\theta}}(x) = \theta_0 + \theta_1 x = \theta_1 x$



Intuition behind the cost function

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

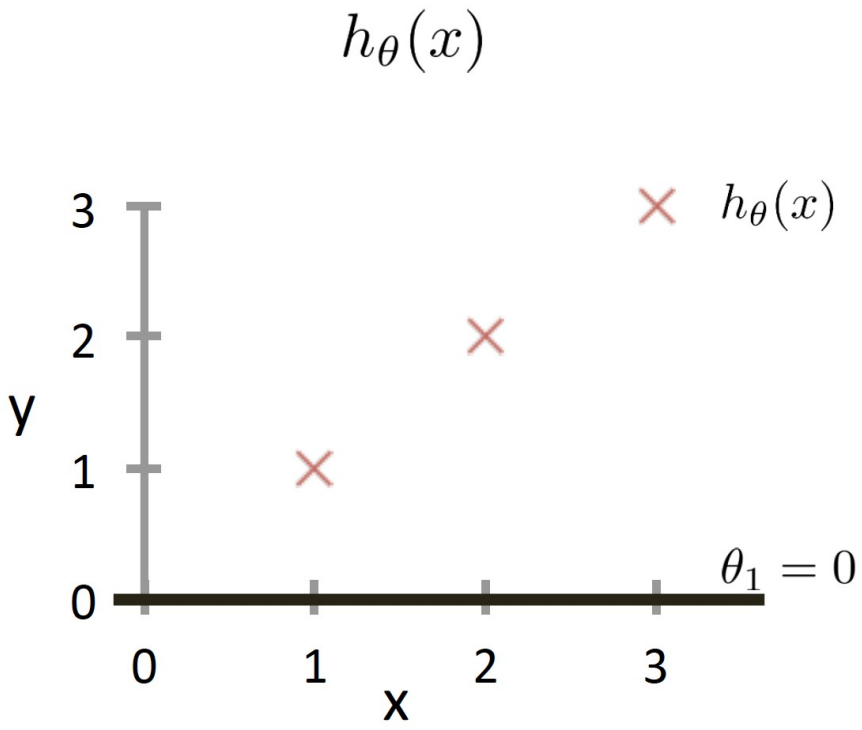
$h_{\boldsymbol{\theta}}(x) = \theta_0 + \theta_1 x = \theta_1 x$

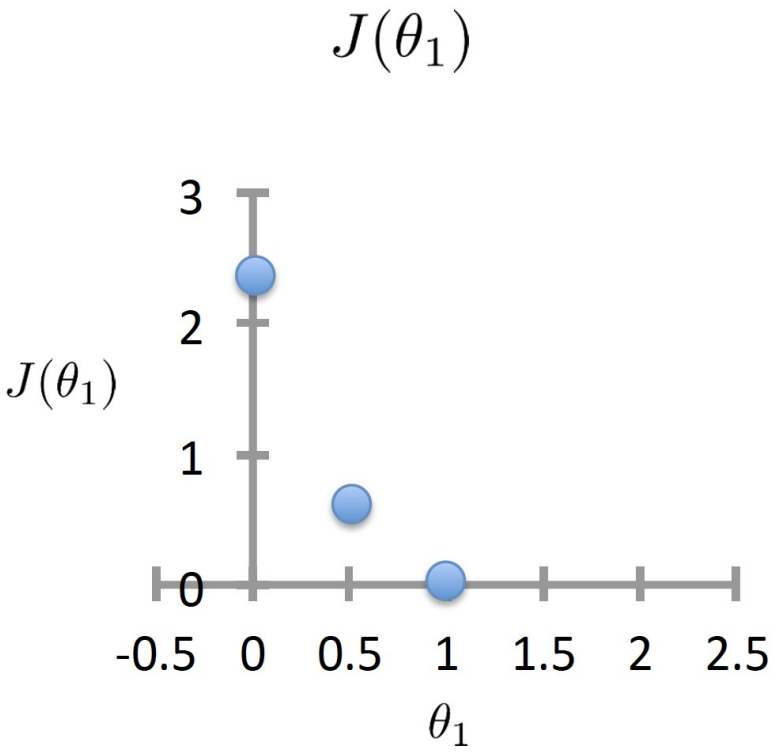


Intuition behind cost function

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

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

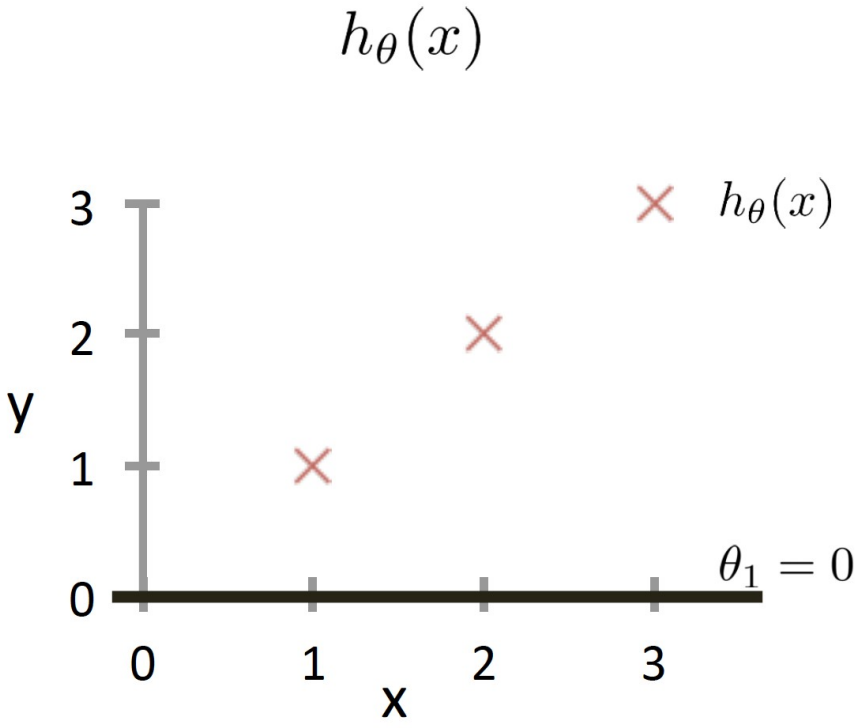


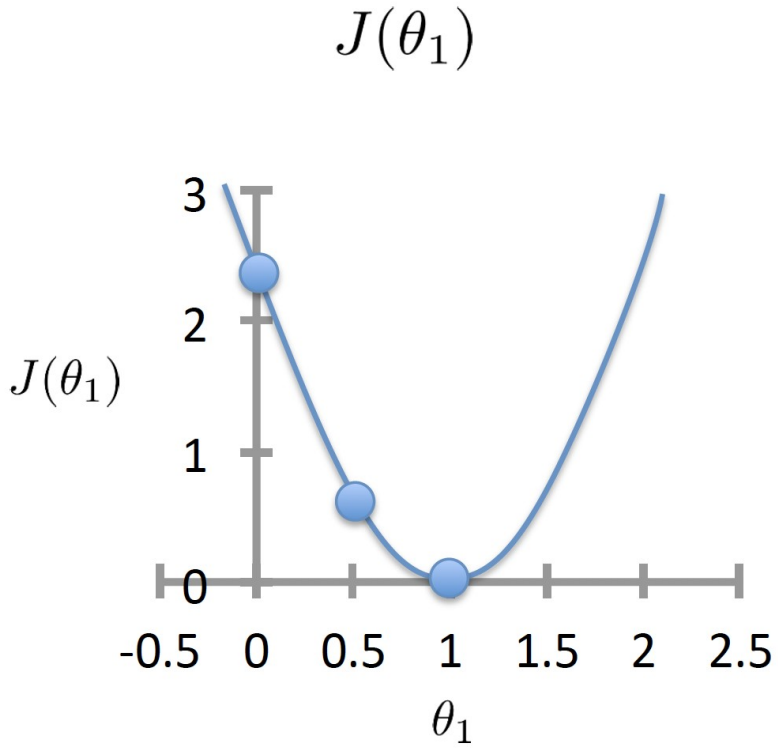


Intuition behind cost function

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

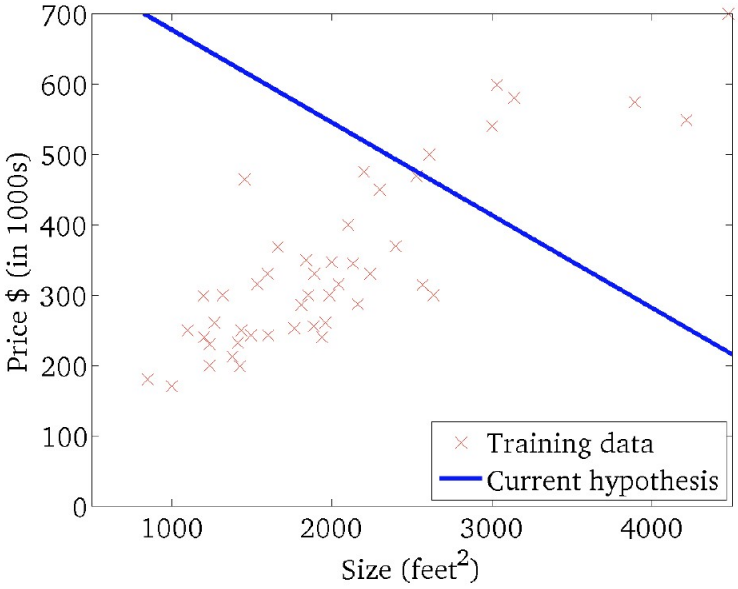
$h_{\boldsymbol{\theta}}(x) = \theta_0 + \theta_1 x = \theta_1 x$



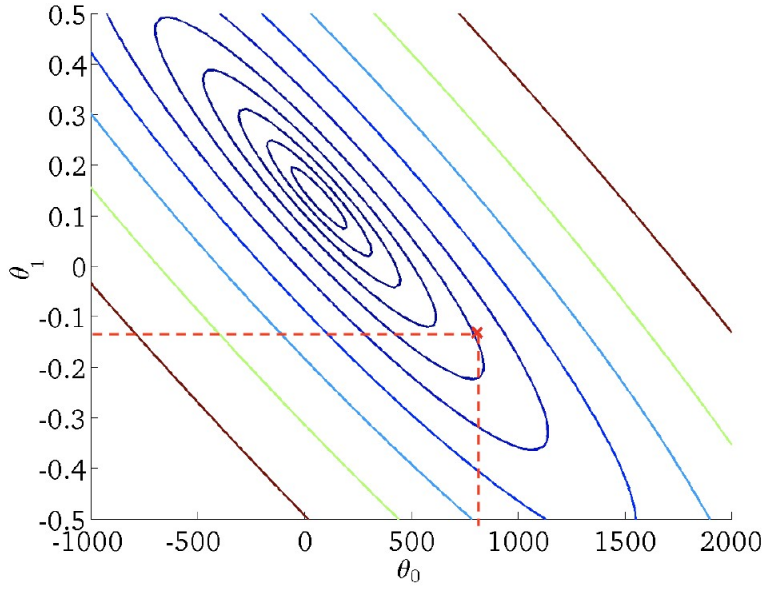


Intuition behind optimization

$h_\theta(x)$

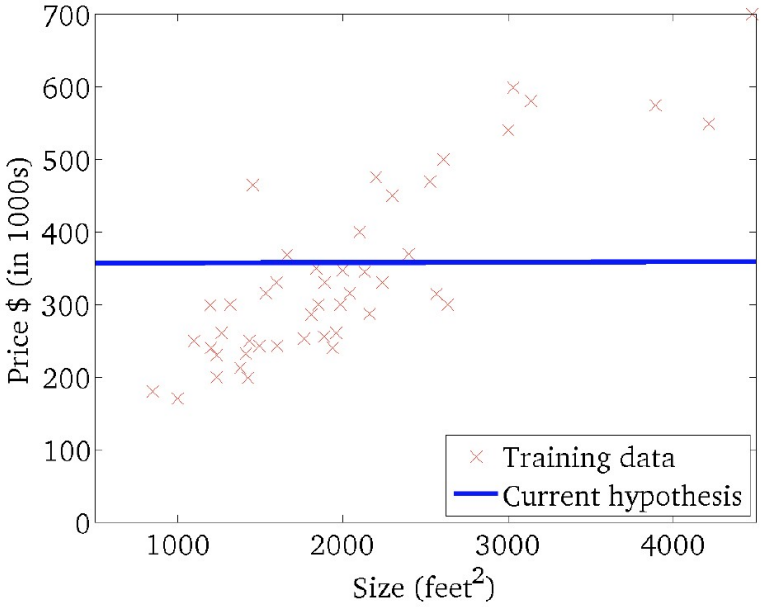


$J(\theta_0, \theta_1)$

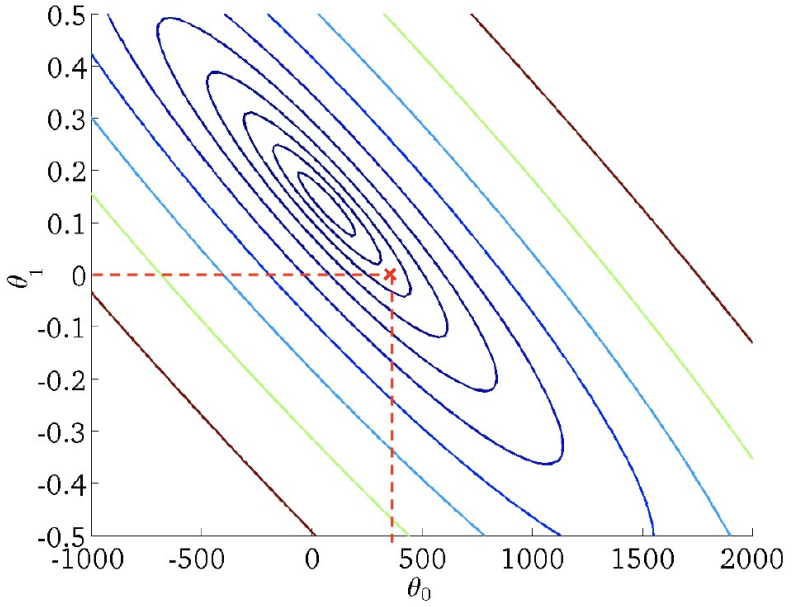


Intuition behind optimization

$h_\theta(x)$

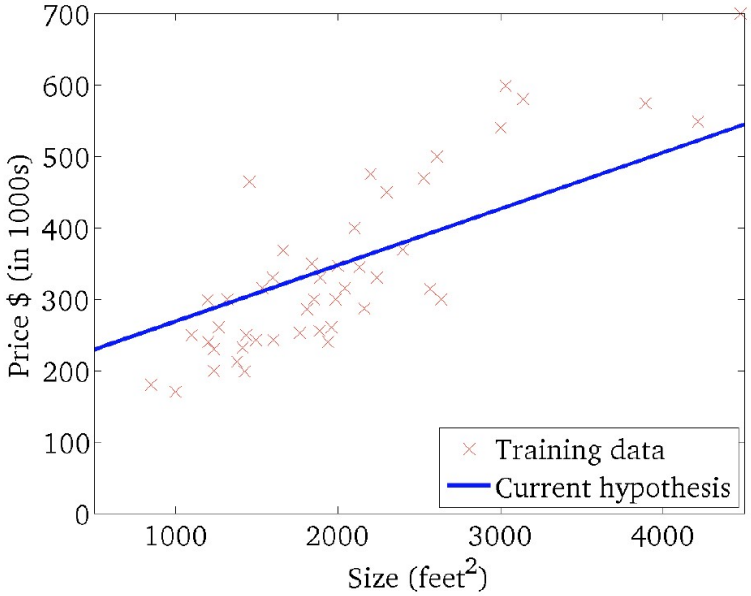


$J(\theta_0, \theta_1)$

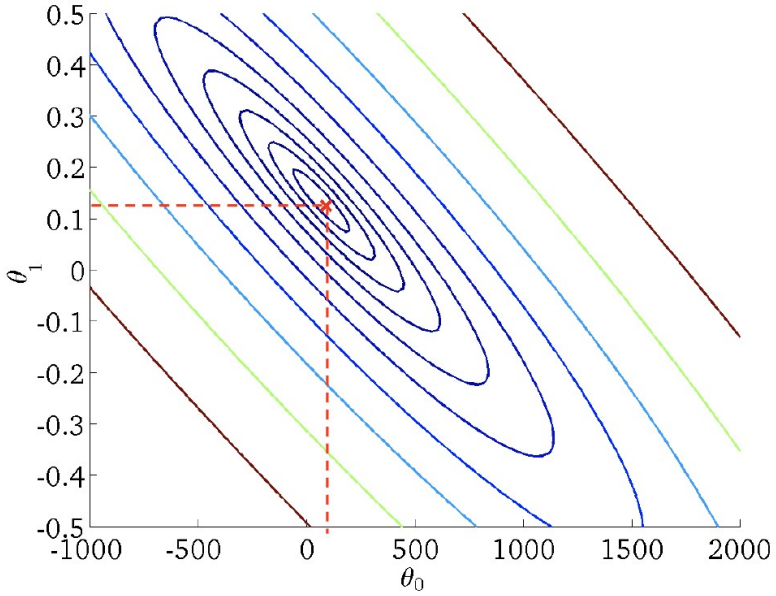


Intuition behind optimization

$h_\theta(x)$



$J(\theta_0, \theta_1)$



How to minimizing the loss

- Optimization methods
  - Gradient Descent
  - Stochastic Gradient Descent
  - Analytic solution

  - …many other approaches

How to solve these?

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

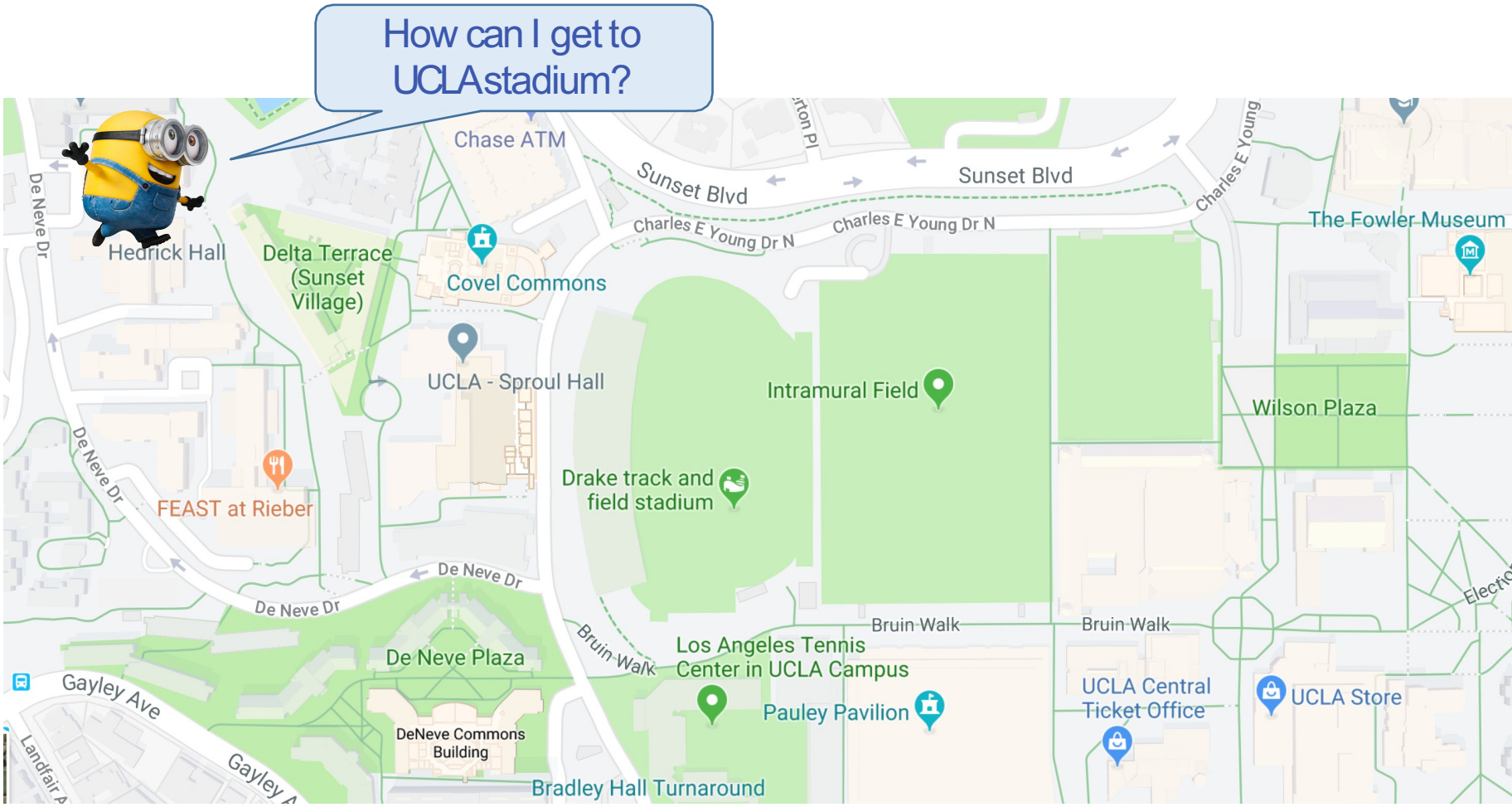
Or $$\min_{\mathbf{w}} \frac{1}{2} \sum_{i=1}^{m} \left(y_i - \mathbf{w}^T \mathbf{x}_i\right)^2$$

There is no closed-form solution

One way to solve it is by gradient descent

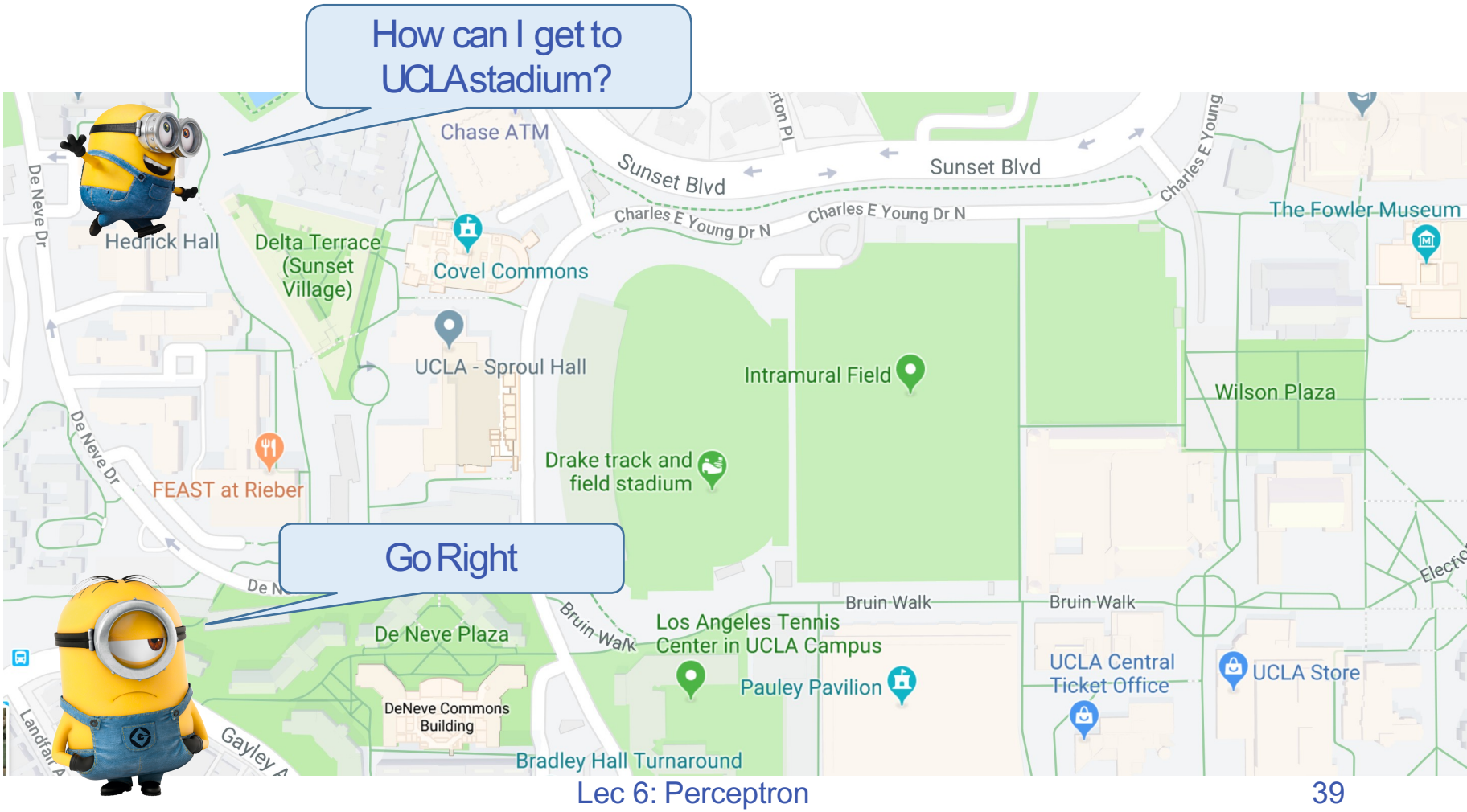
Intuition

- Asking direction



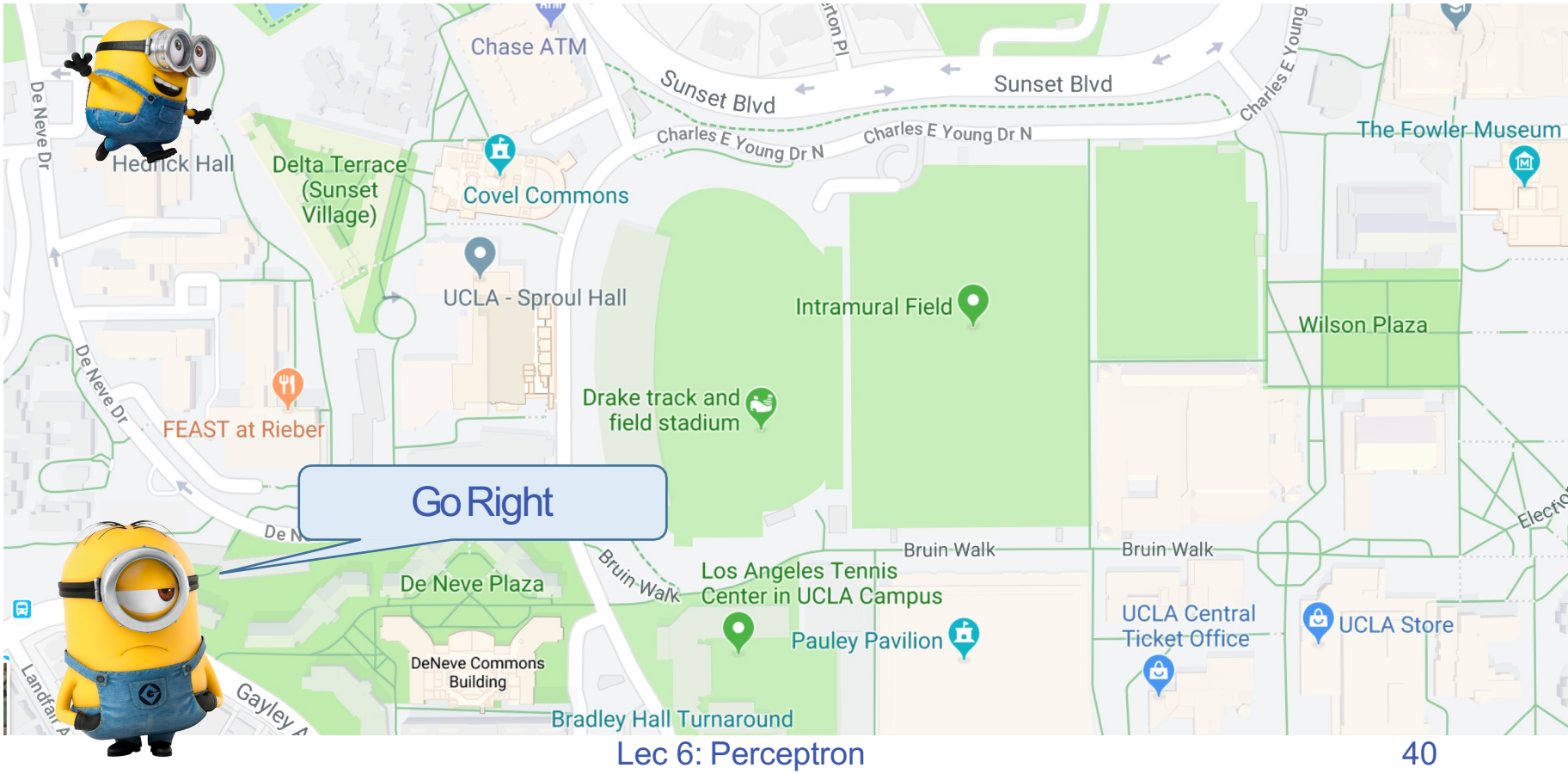
Intuition

- Asking direction



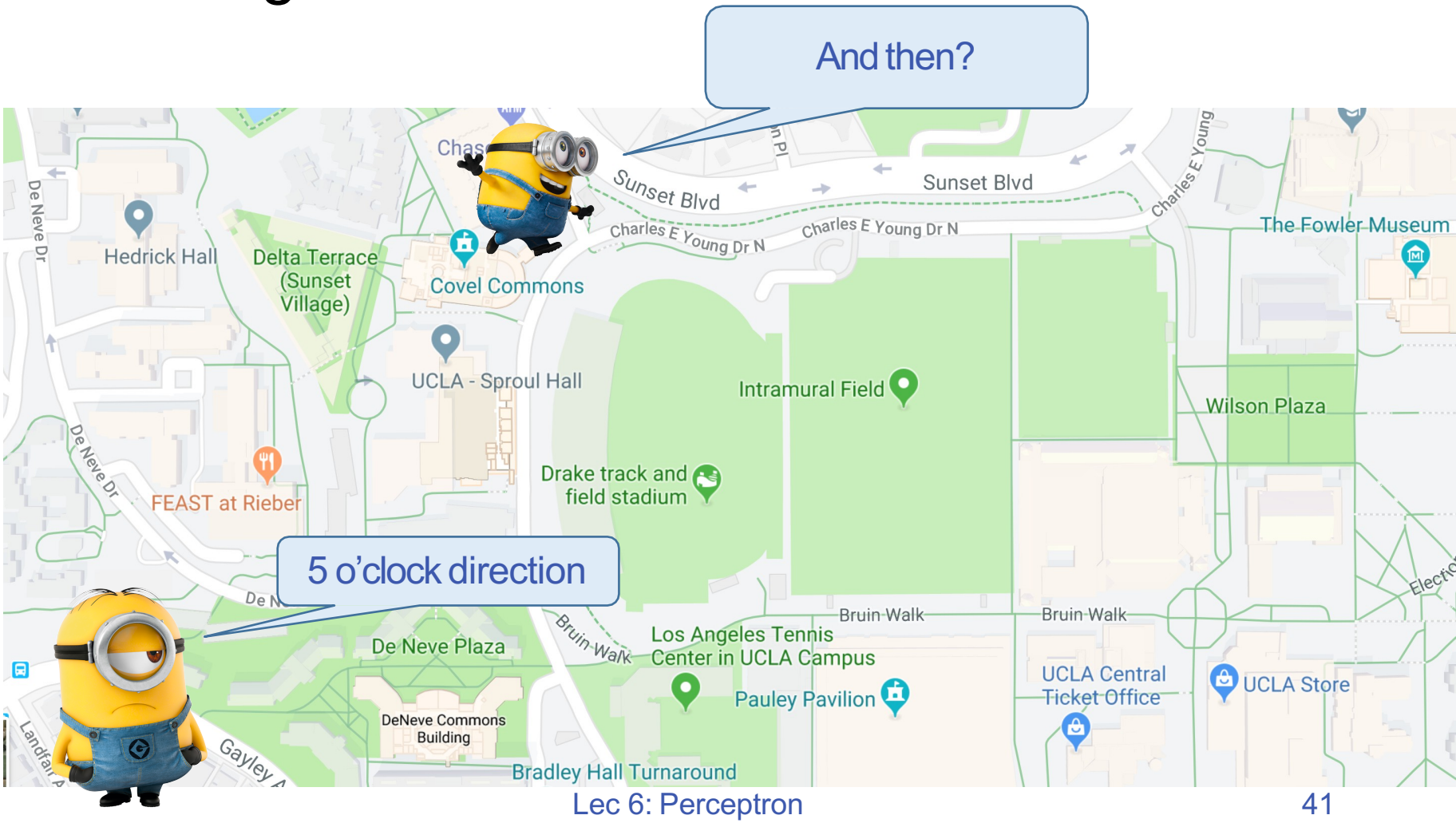
Intuition

- Asking direction



Intuition

- Asking direction

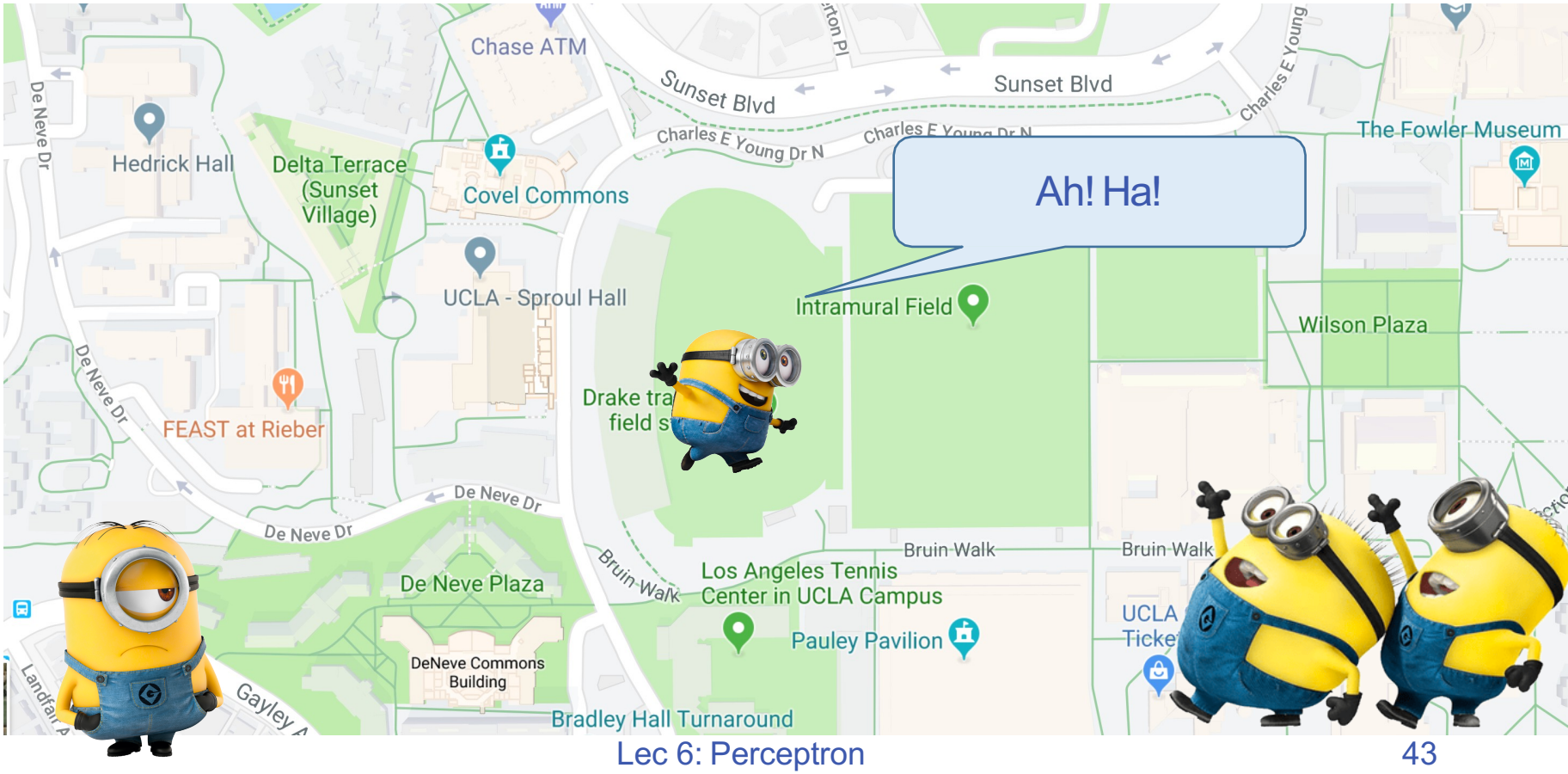


Intuition

- Asking direction

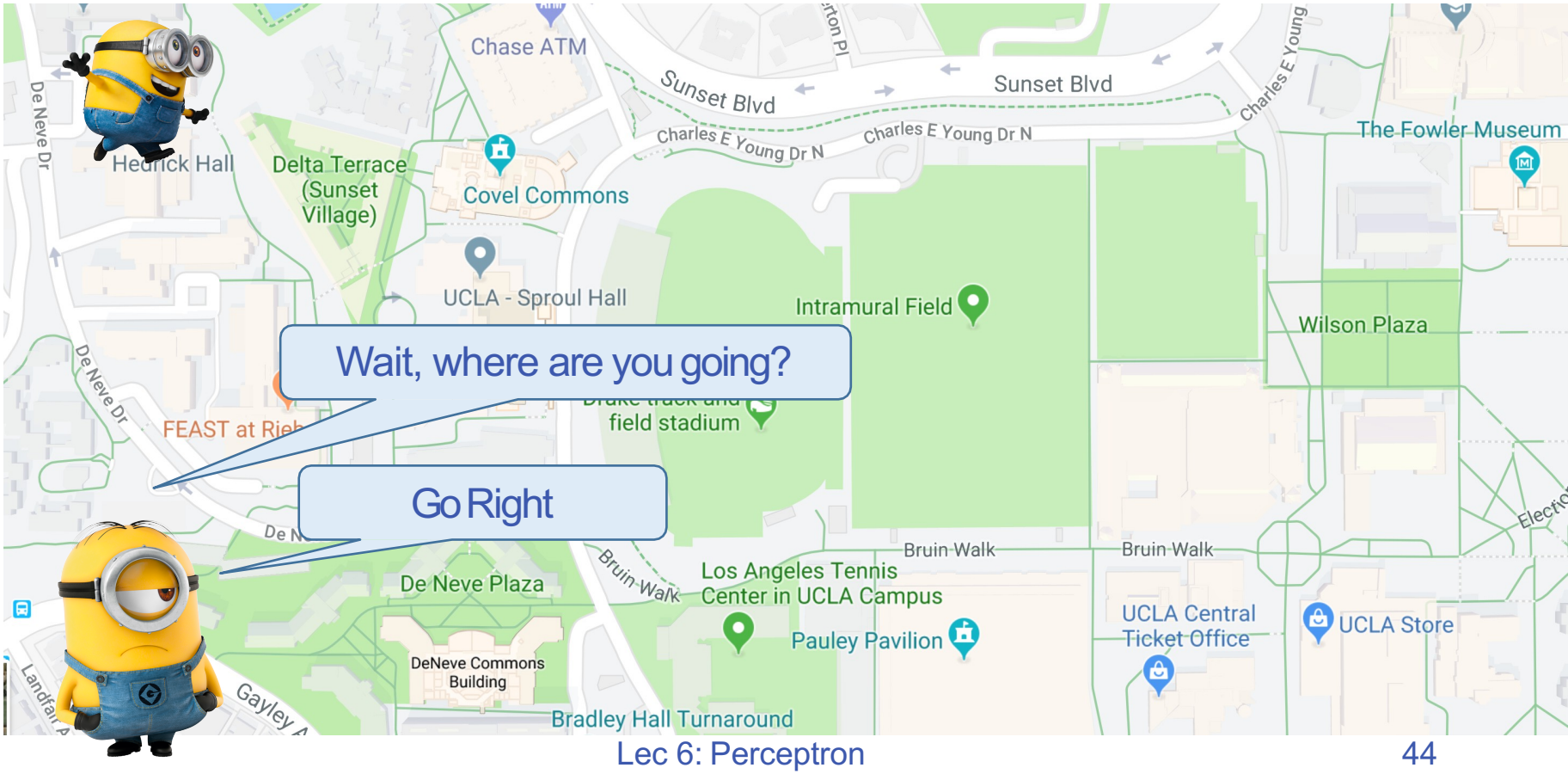
Intuition

- Asking direction



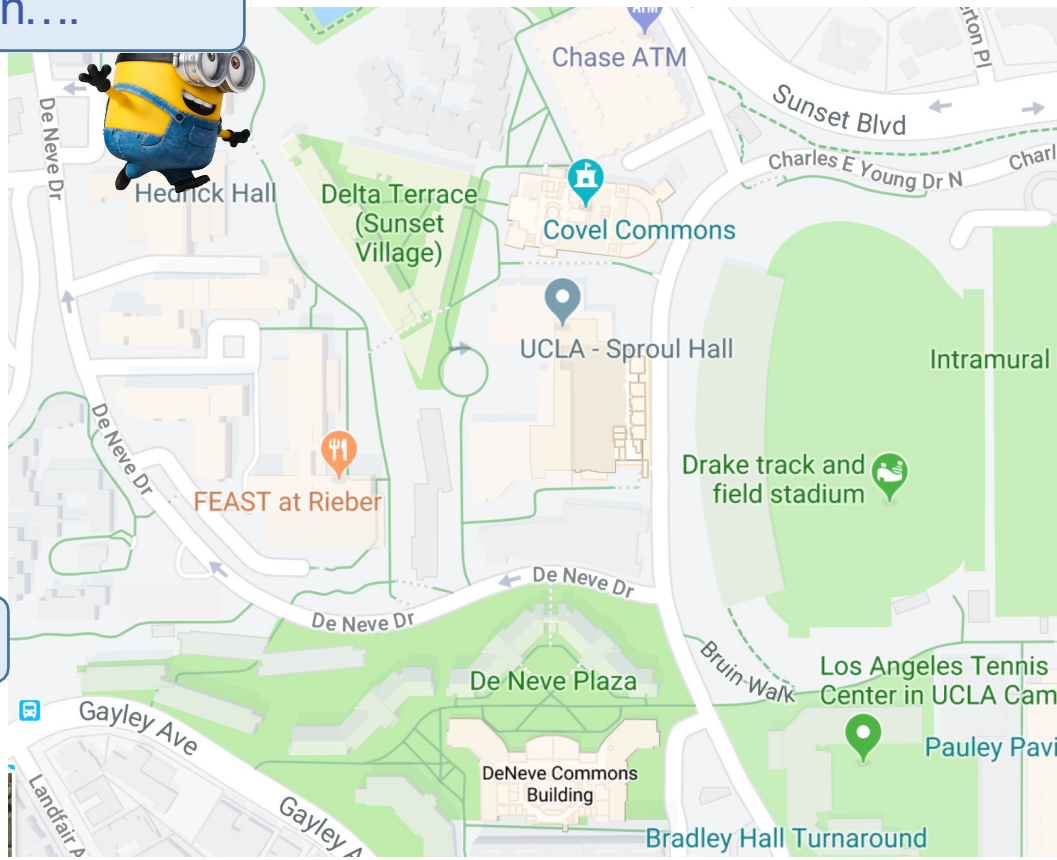
Intuition

- What may go wrong? Incorrect Step-size



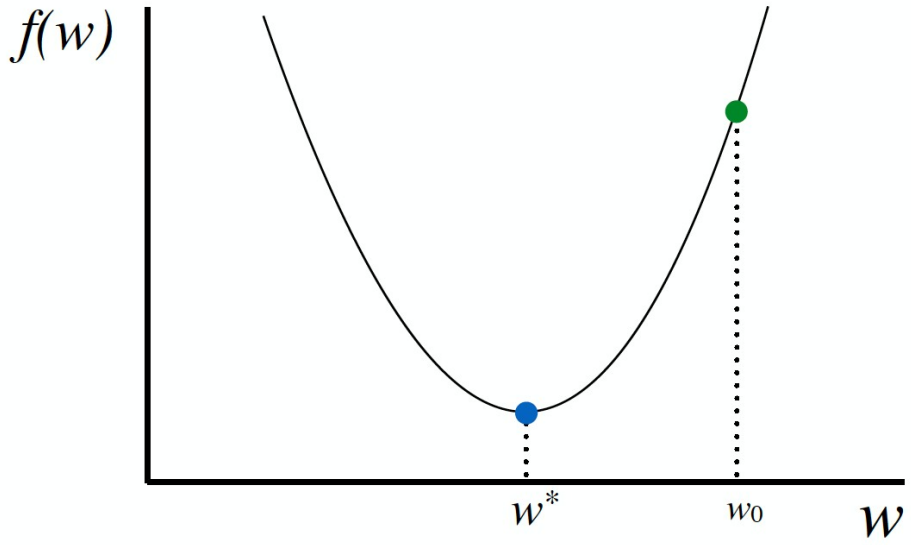
Intuition

- What may go wrong? Incorrect Direction



Gradient descent

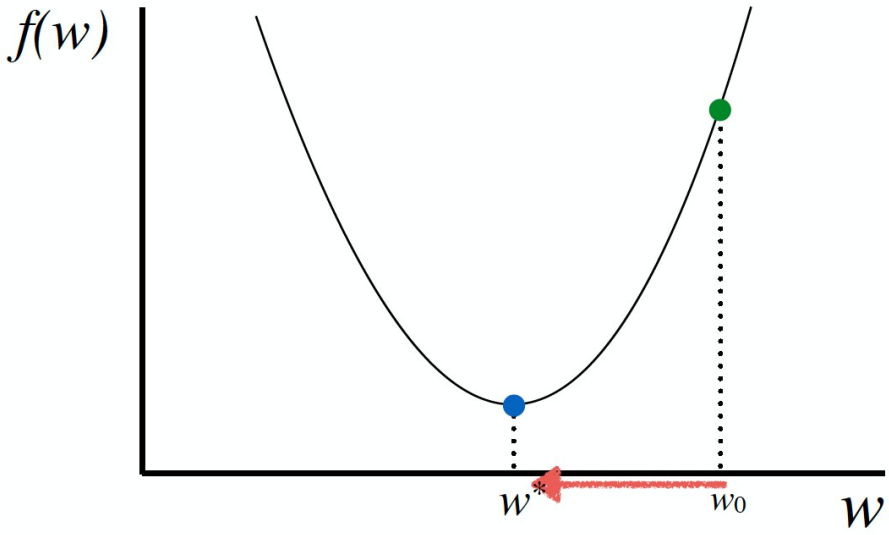
Start at a random point



Gradient descent

Start at a random point

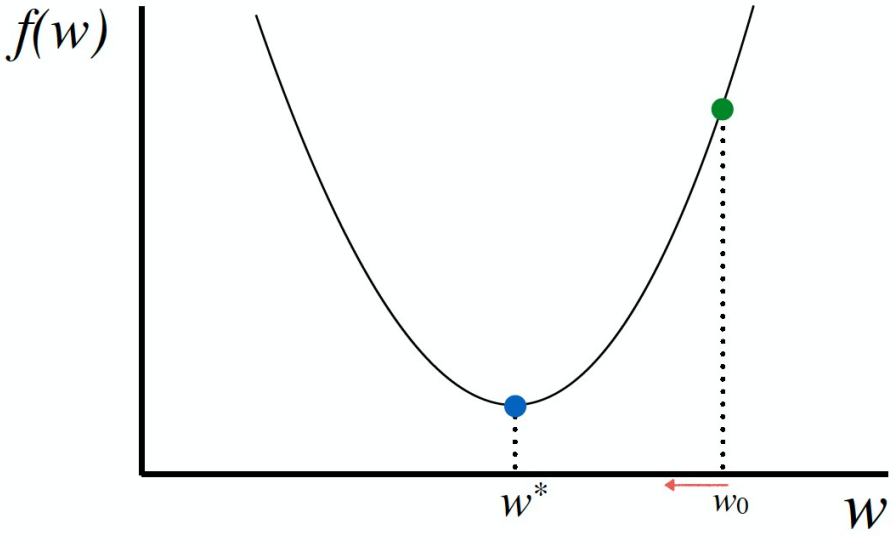
Determine a descent direction



Gradient descent

Start at a random point

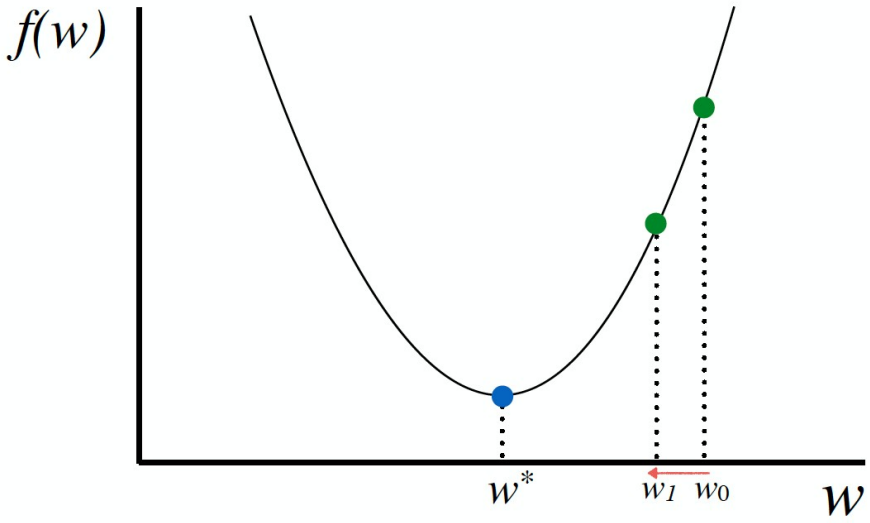
Determine a descent direction
Choose a step size



Gradient descent

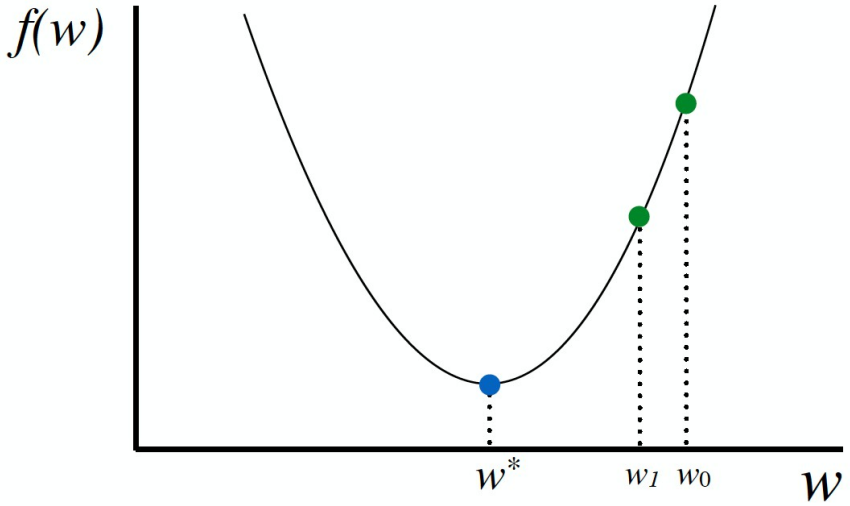
Start at a random point

Determine a descent direction
Choose a step size
Update



Gradient descent

Start at a random point
Repeat
    Determine a descent direction
    Choose a step size
    Update
Until stopping criterion is satisfied



Gradient descent

Start at a random point

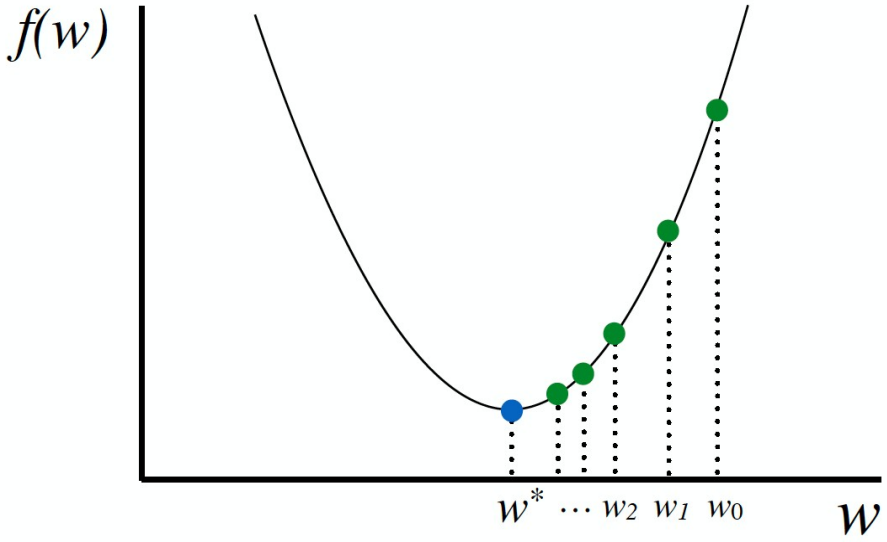
Repeat

Determine a descent direction

Choose a step size

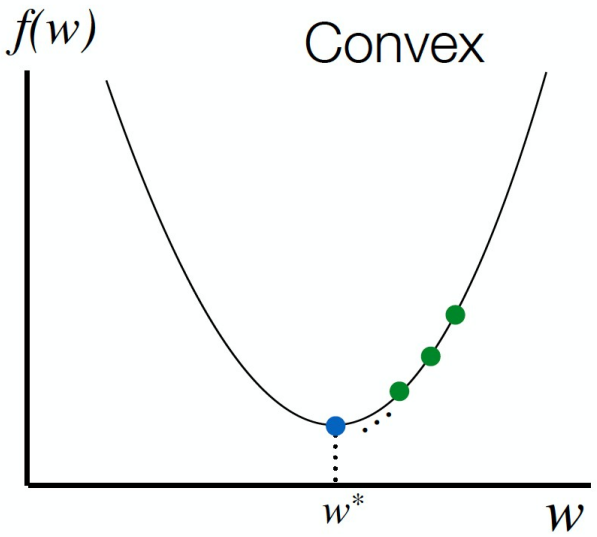
Update

Until stopping criterion is satisfied

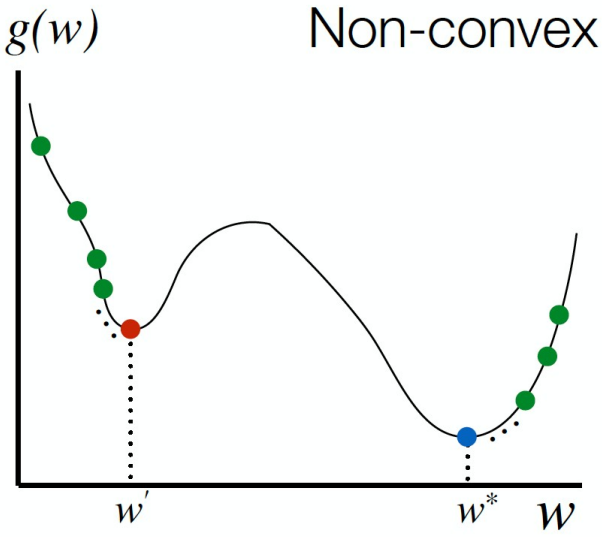


Where will we converge?

- If function is convex, it converges to the global optimum (need proper choice of step-size)



Any local minimum is a global minimum



Multiple local minima may exist

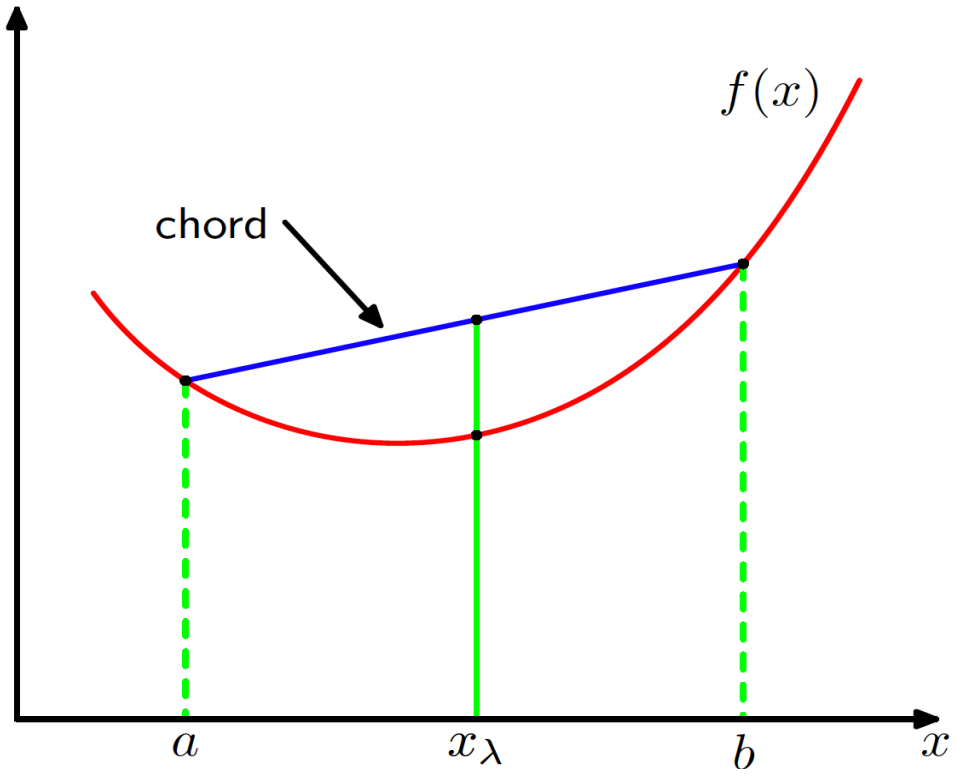
Convex function

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

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

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

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

Multi-variate functions

How to determine convexity in this case?

Matrix of second-order derivatives (Hessian)

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

Multi-variate functions

How to determine convexity in this case?

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

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

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

for all $\boldsymbol{z}$.

Multi-variate functions

- Example

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

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

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

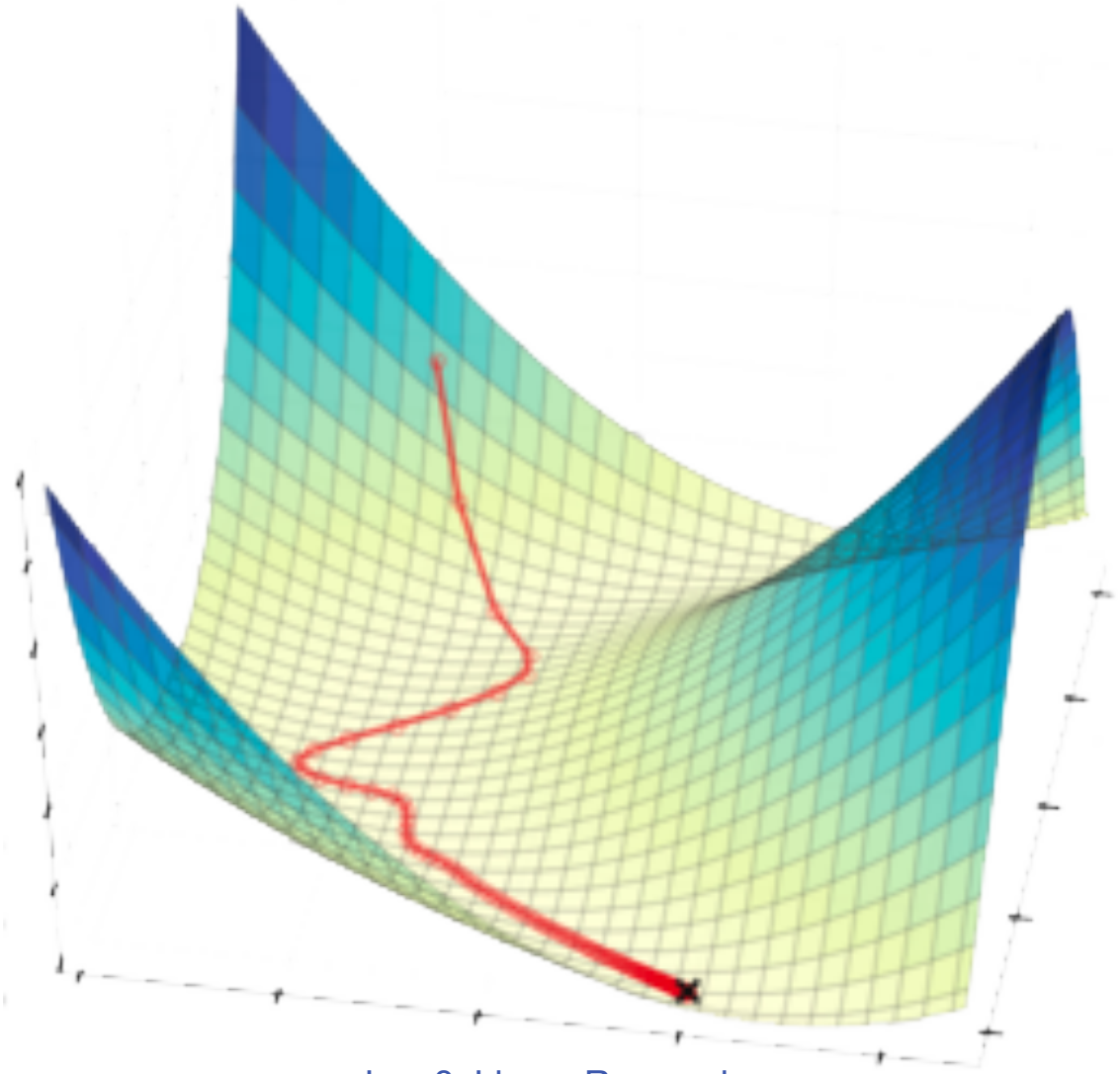
Gradient Descent

Algorithm 1 Gradient Descent (J)

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

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

- 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.

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

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

$$t \leftarrow t + 1$$

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

Remarks

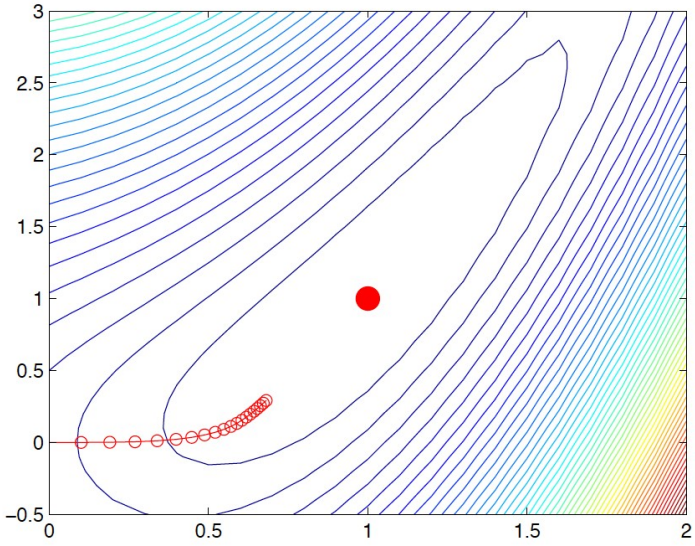
- η is often called step size or learning rate -- how far our update will go along the the direction of the negative gradient
- With a suitable choice of η, the iterative procedure converges to a stationary point where

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

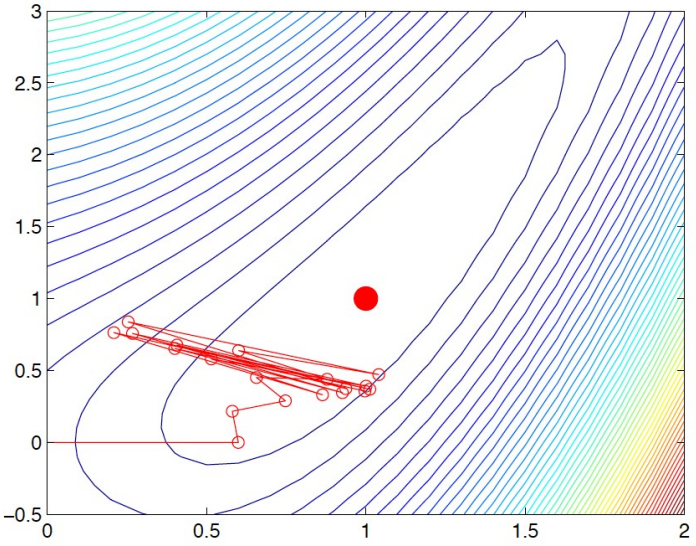
- A stationary point is only necessary for being the minimum

Choosing the right η is important

small η is too slow?



large η is too unstable?



Iterative optimization

Algorithm 2 Gradient Descent (J)

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

How to compute the gradient?

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?

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

- Gradient of J(w) is:

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

Gradient of the cost

We are trying to minimize

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

- The gradient is of the form

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

$$\begin{aligned}
\frac{\partial J}{\partial w_j} &= \frac{\partial}{\partial w_j}\frac{1}{2}\sum_{i=1}^{m}\left(y_i - \mathbf{w}^T\mathbf{x}_i\right)^2 \\
&= \frac{1}{2}\sum_{i=1}^{m}\frac{\partial}{\partial w_j}\left(y_i - \mathbf{w}^T\mathbf{x}_i\right)^2 \\
&= \frac{1}{2}\sum_{i=1}^{m}2(y_i - \mathbf{w}^T\mathbf{x}_i)\frac{\partial}{\partial w_j}\left(y_i - w_1x_{i1} - \cdots w_jx_{ij} - \cdots\right) \\
&= \frac{1}{2}\sum_{i=1}^{m}2(y_i - \mathbf{w}^T\mathbf{x}_i)(-x_{ij}) \\
&= -\sum_{i=1}^{m}(y_i - \mathbf{w}^T\mathbf{x}_i)x_{ij}
\end{aligned}$$

One element of the gradient vector

Sum of £ Error £ Input

Gradient descent for LMS

We are trying to minimize

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

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

      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 $\nabla J(\boldsymbol{w}^t)$

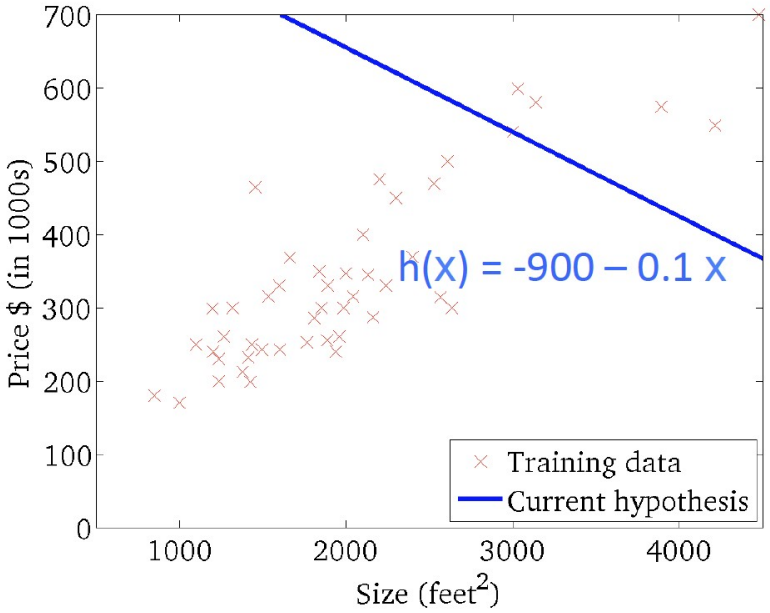
   2. Update **w** as follows:

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

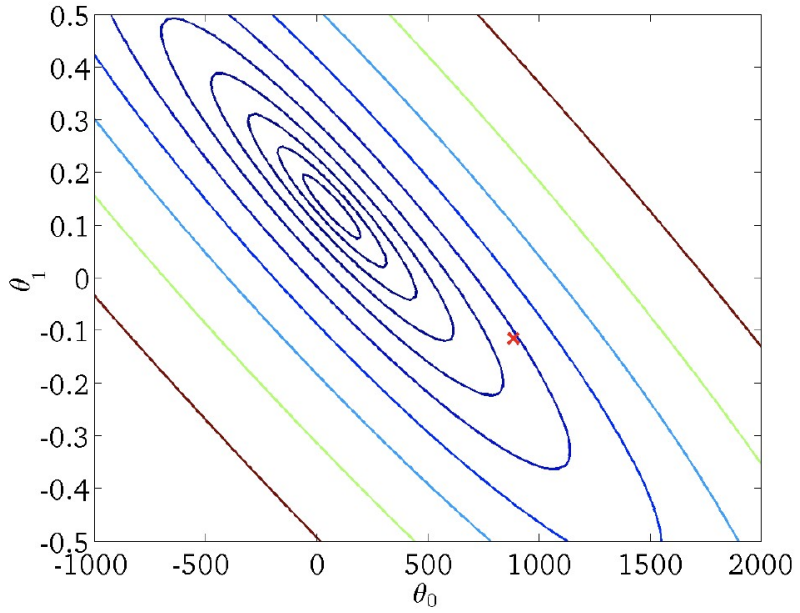
r: Called the learning rate

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

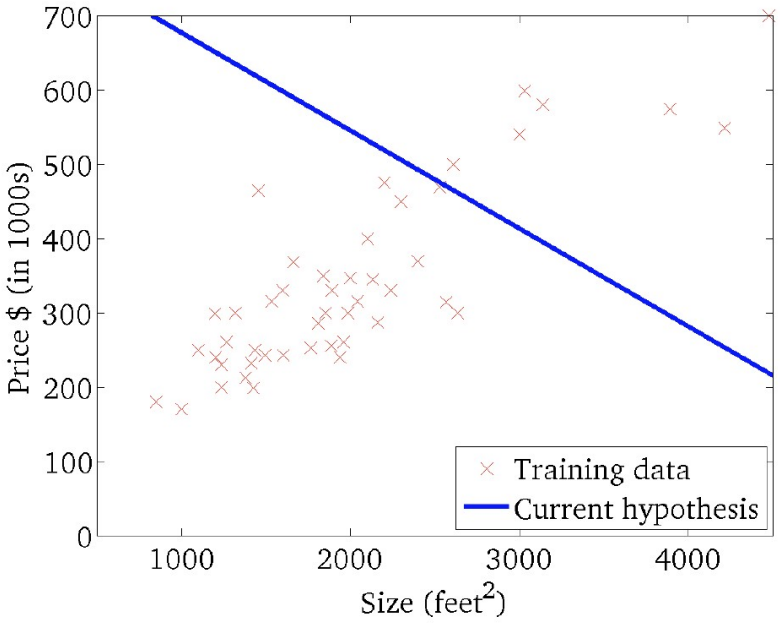
$h_\theta(x)$



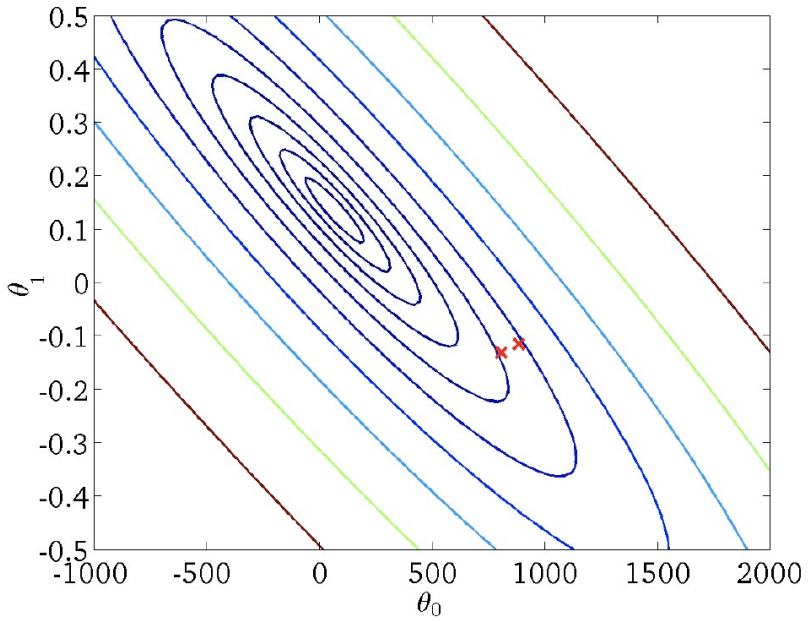
$J(\theta_0, \theta_1)$



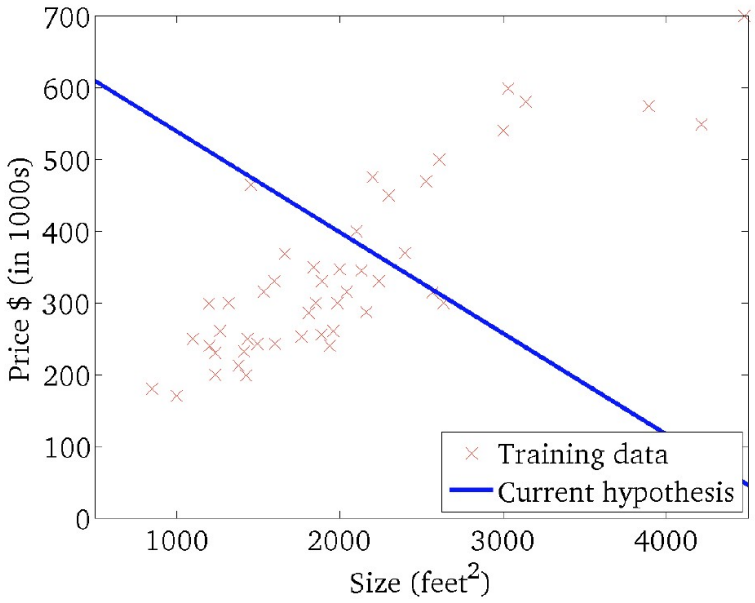
$h_\theta(x)$



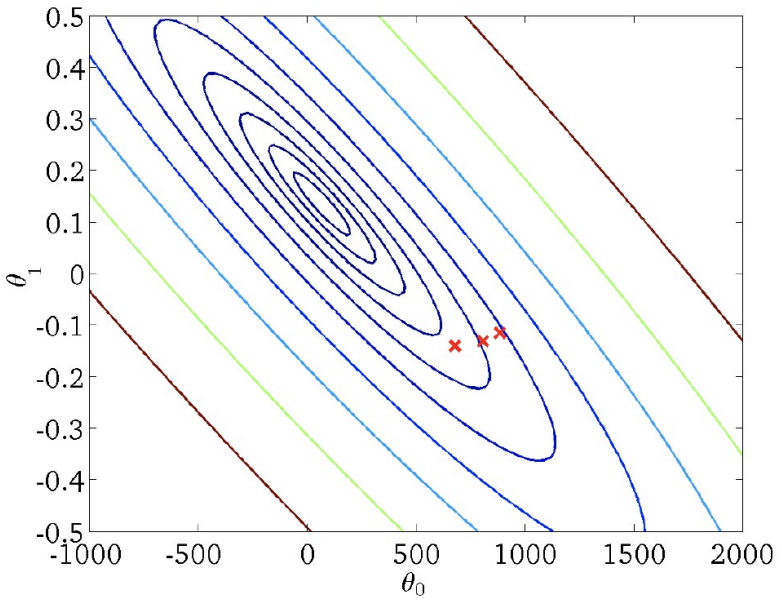
$J(\theta_0, \theta_1)$



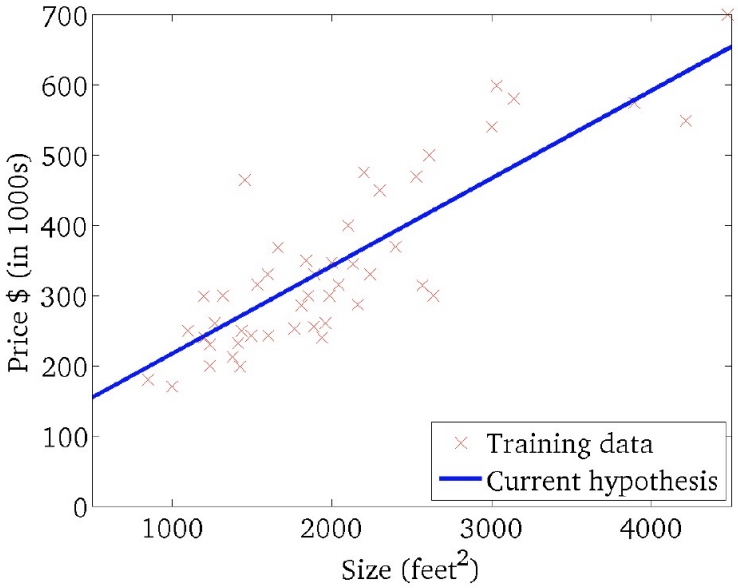
$h_\theta(x)$



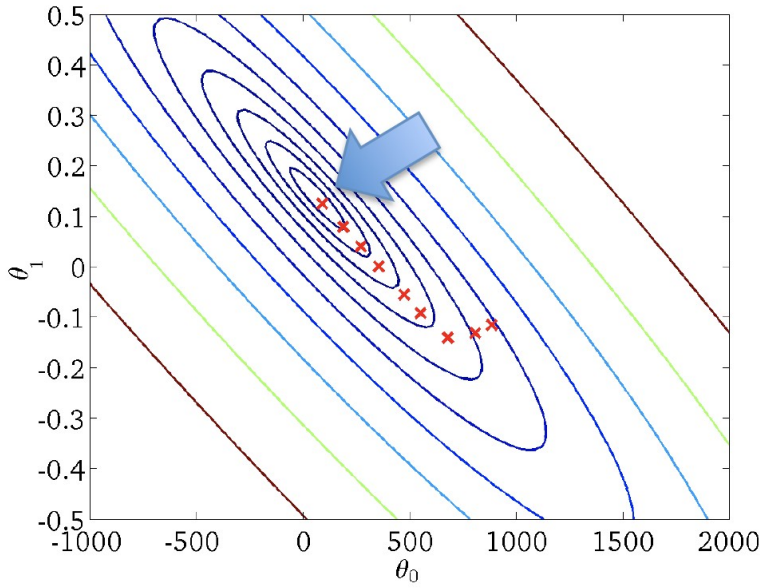
$J(\theta_0, \theta_1)$



$h_\theta(x)$



$J(\theta_0, \theta_1)$



Gradient descent for LMS

We are trying to minimize

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

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 r J

2. Update **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

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

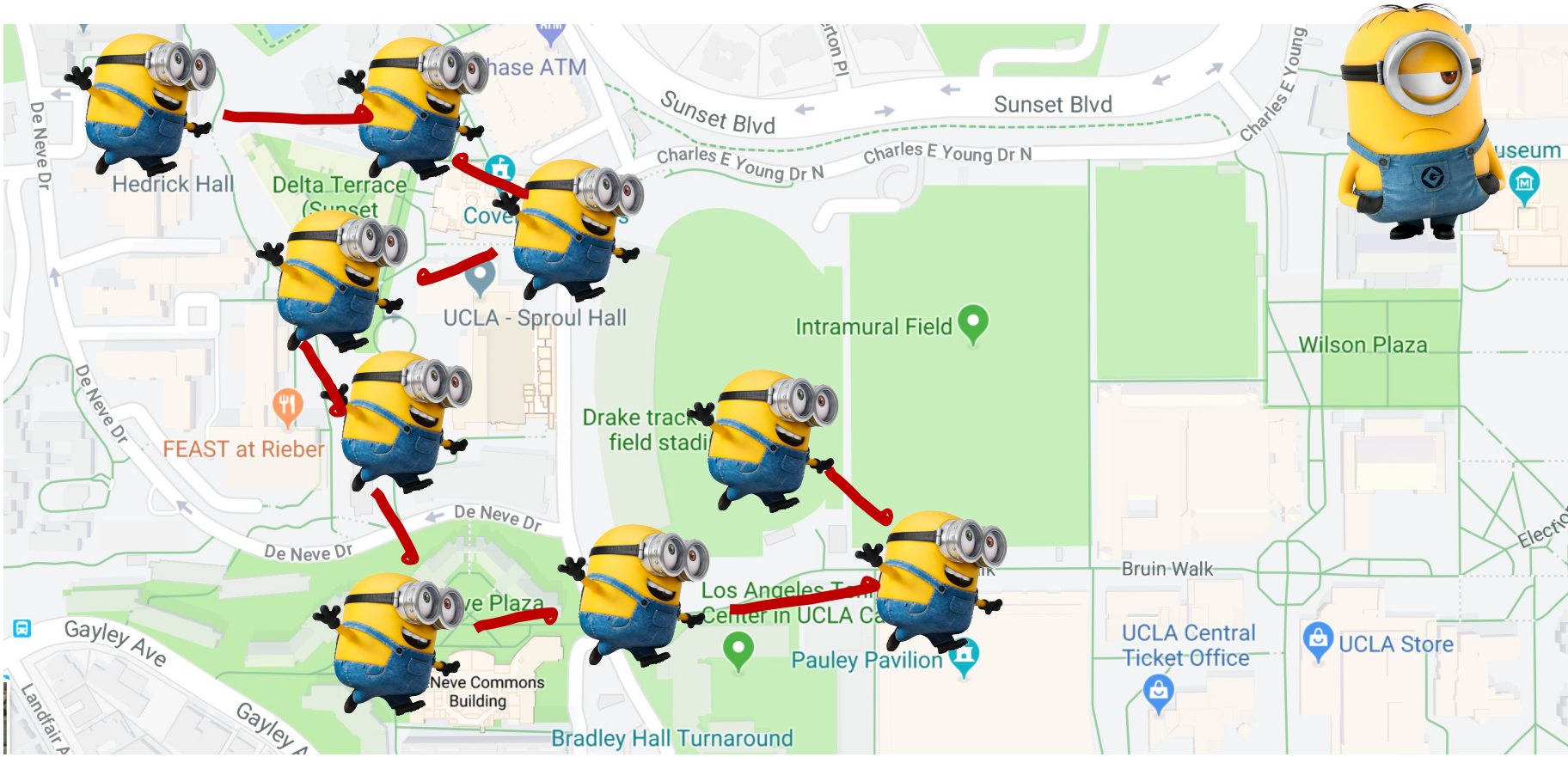
Repeat until converge:
  Randomly pick one sample (x_i, y_i)
  Update $w \leftarrow w - \eta \nabla f_i(w)$

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 $\mathcal{D} = \{(\boldsymbol{x}, y)\}$

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

The Perceptron Algorithm [Rosenblatt 1958]

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

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

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

The Perceptron Algorithm [Rosenblatt 1958]

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

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

Prediction: y^{test}

Perceptron effectively minimizing:

$$\sum_i \max(0,\ 1 - \ y_i(\mathbf{w}^\mathrm{T}\mathbf{x}_i))$$

Stochastic gradient descent for linear regression

1. Initialize $\mathbf{w}$
2. For t = 0, 1, 2, …. (until error below some threshold)
   - For each training example $(\mathbf{x}_i, y_i)$:
     - Update **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, …. (until error below some threshold)
   - For each training example $(\mathbf{x}_i, y_i)$:
     - Update **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 examplesare processed once

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

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

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

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

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

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

Show that the optimization problem we saw earlier is equivalent to

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

This can be solved analytically. Show that the solution w* is

$$\mathbf{w}^* = \left(XX^T\right)^{-1} XY$$

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)

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

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)

$$\frac{\partial \log P(\mathcal{D})}{\partial s} = -\frac{1}{2}\left\{-\frac{1}{s^2}\sum_n [y_n - (\theta_0 + \theta_1 x_n)]^2 + \mathsf{N}\frac{1}{s}\right\} = 0$$

$$\rightarrow \sigma^{*2} = s^* = \frac{1}{\mathsf{N}}\sum_n [y_n - (\theta_0 + \theta_1 x_n)]^2$$