CS 6375
Linear Regression

Rishabh Iyer
University of Texas at Dallas

Slides adapted from Nick Rouzzi and Sriraam Natarajan

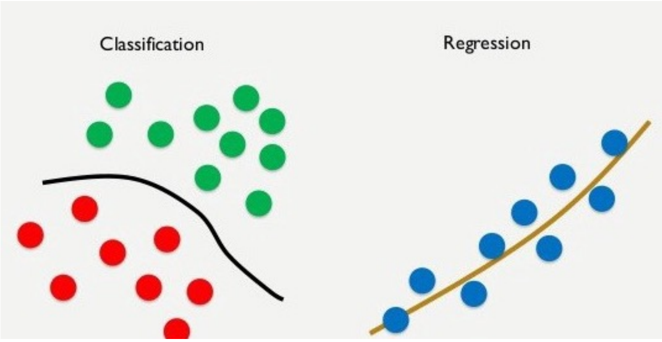
Recap: Course Topics

- **Supervised Learning**
  - SVMs & kernel methods
  - Decision trees, Random Forests, Gradient Boosted Trees
  - Nearest Neighbor: KNN Classifiers
  - Logistic Regression
  - Neural networks
  - Probabilistic models: Bayesian networks, Naïve Bayes
- **Unsupervised Learning**
  - Clustering: k-means & spectral clustering
  - Dimensionality reduction
  - PCA
  - Matrix Factorizations
- **Parameter estimation**
  - Bayesian methods, MAP estimation, maximum likelihood estimation, expectation maximization, …
- **Evaluation**
  - AOC, cross-validation, precision/recall
- **Statistical Methods**
  - Boosting, bagging, bootstrapping
  - Sampling
- **Reinforcement Learning, Semi-supervised Learning, Active Learning, ….**

Part I: Recap of Supervised Learning and Linear Regression Setup

Recap: Supervised Learning

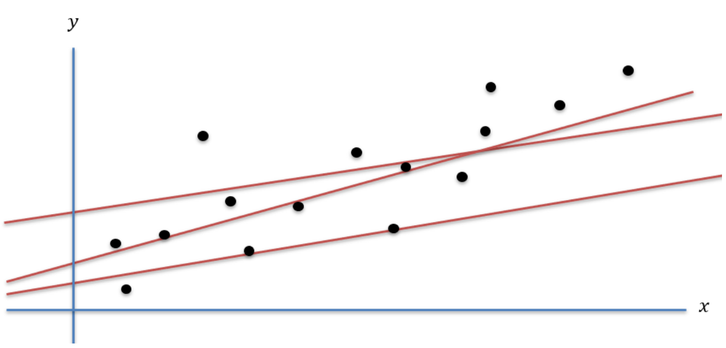
- **Input:** $(x^{(1)}, y^{(1)}), \dots, (x^{(M)}, y^{(M)})$
  - $x^{(m)}$ is the m^{th} data item and $y^{(m)}$ is the m^{th} **label**
- **Goal:** find a function f such that $f(x^{(m)})$ is a "good approximation" to $y^{(m)}$
  - Can use it to predict y values for previously unseen x values

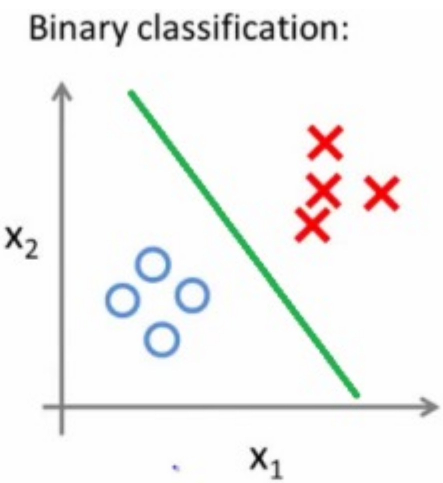


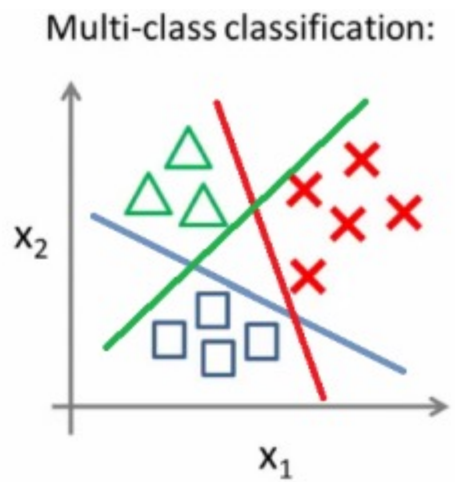
Recap: Classification vs Regression

Classification vs Regression

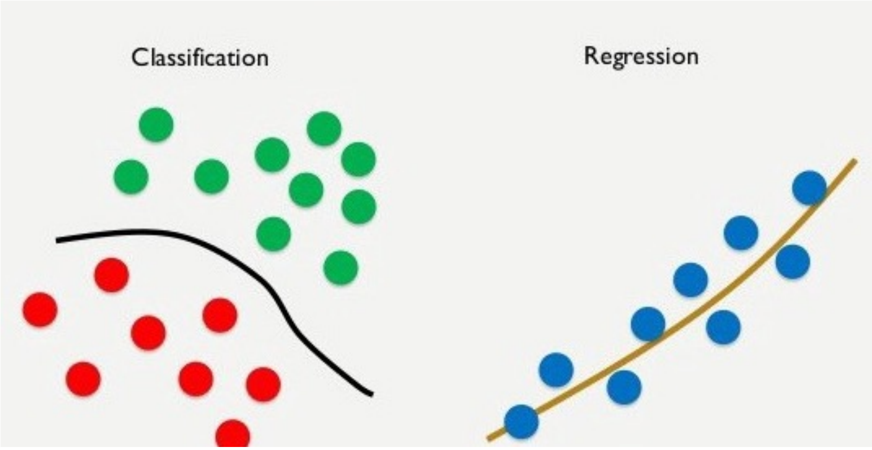
- Input: pairs of points $(x^{(1)}, y^{(1)}), \ldots, (x^{(M)}, y^{(M)})$ with $x^{(m)} \in \mathbb{R}^n$
- Regression case: $y^{(m)} \in \mathbb{R}$
- Classification case: $y^{(m)} \in [0, k-1]$ [k-class classification]
- If k = 2, we get Binary classification







Recap: Examples of Supervised Learning



Classification

- Spam email detection
- Handwritten digit recognition
- Medical Diagnosis
- Fraud Detection
- Face Recognition

Regression

- Housing Price Prediction
- Stock Market Prediction
- Weather Prediction
- Market Analysis and Business Trends

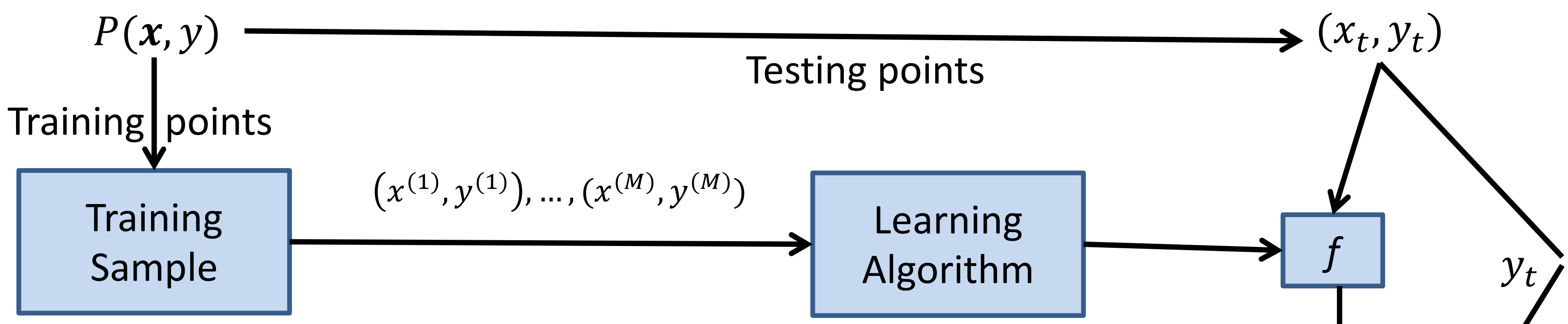
Recap: Hypothesis Space

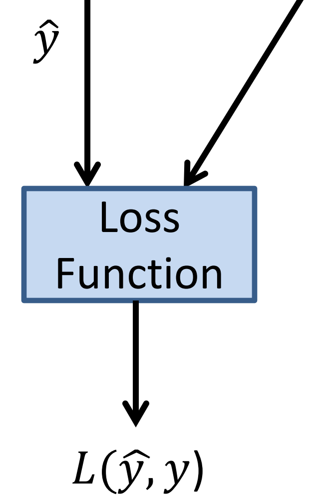
- Hypothesis space (Aka Model): set of allowable functions $f: X \to Y$

- Goal: find the "best" element of the hypothesis space

  - How do we measure the quality of f?

Recap: Supervised Learning Workflow





- **Step 1:** Collect Training Data
  $(x^{(1)}, y^{(1)}), \dots, (x^{(M)}, y^{(M)})$
- **Step 2:** Select Model f and Loss function L
- **Step 3**: Solve Learning Problem:

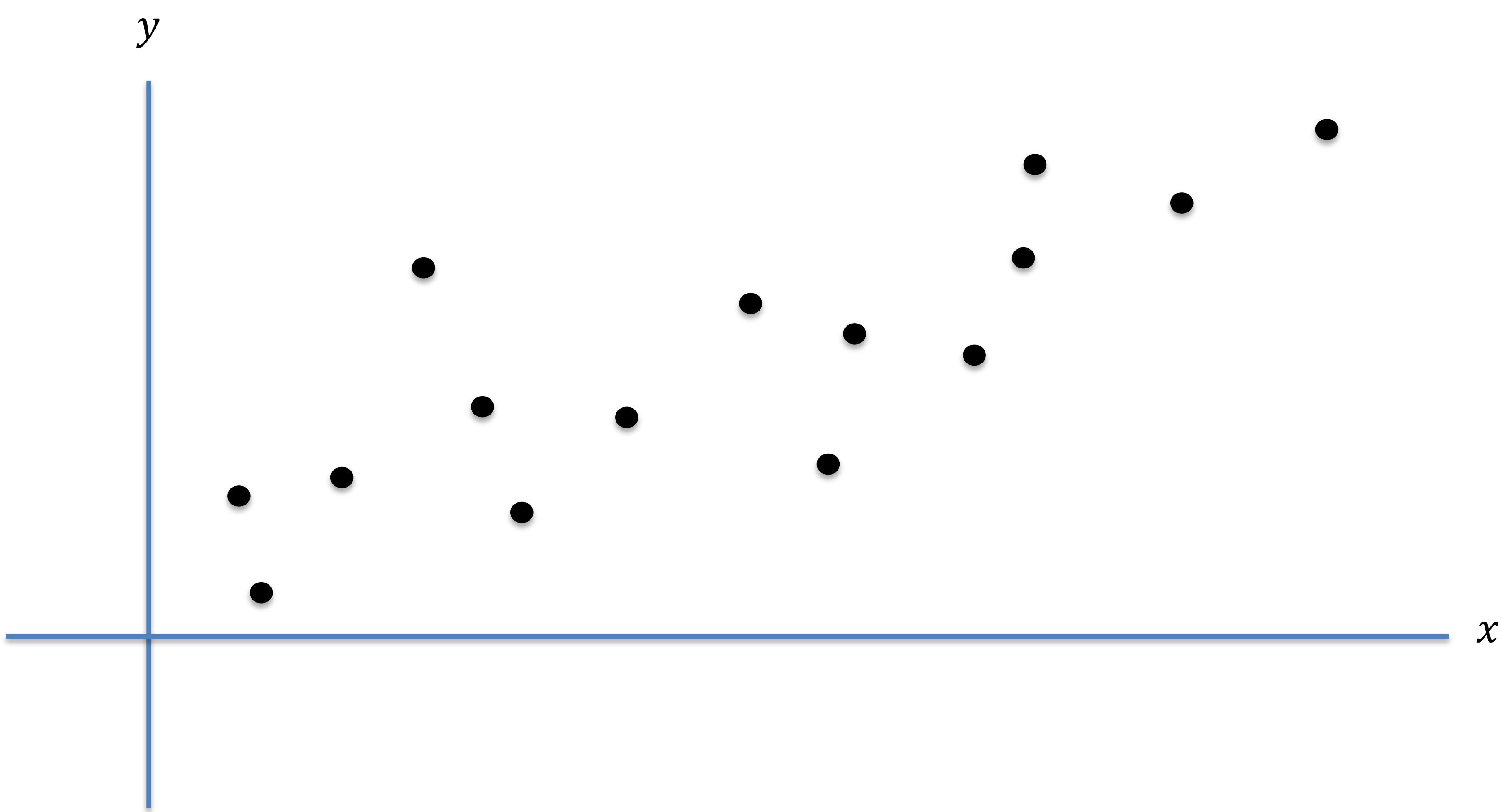
$$\min_f \sum_{i=1:m} L(f(x^i), y^i)$$

- **Step 4:** Obtain Predictions $\hat{y}_t = f(x_t)$ on all **Test Data**
- **Step 5:** Evaluation -- Measure the error $Err(\hat{y}_t, y_t)$

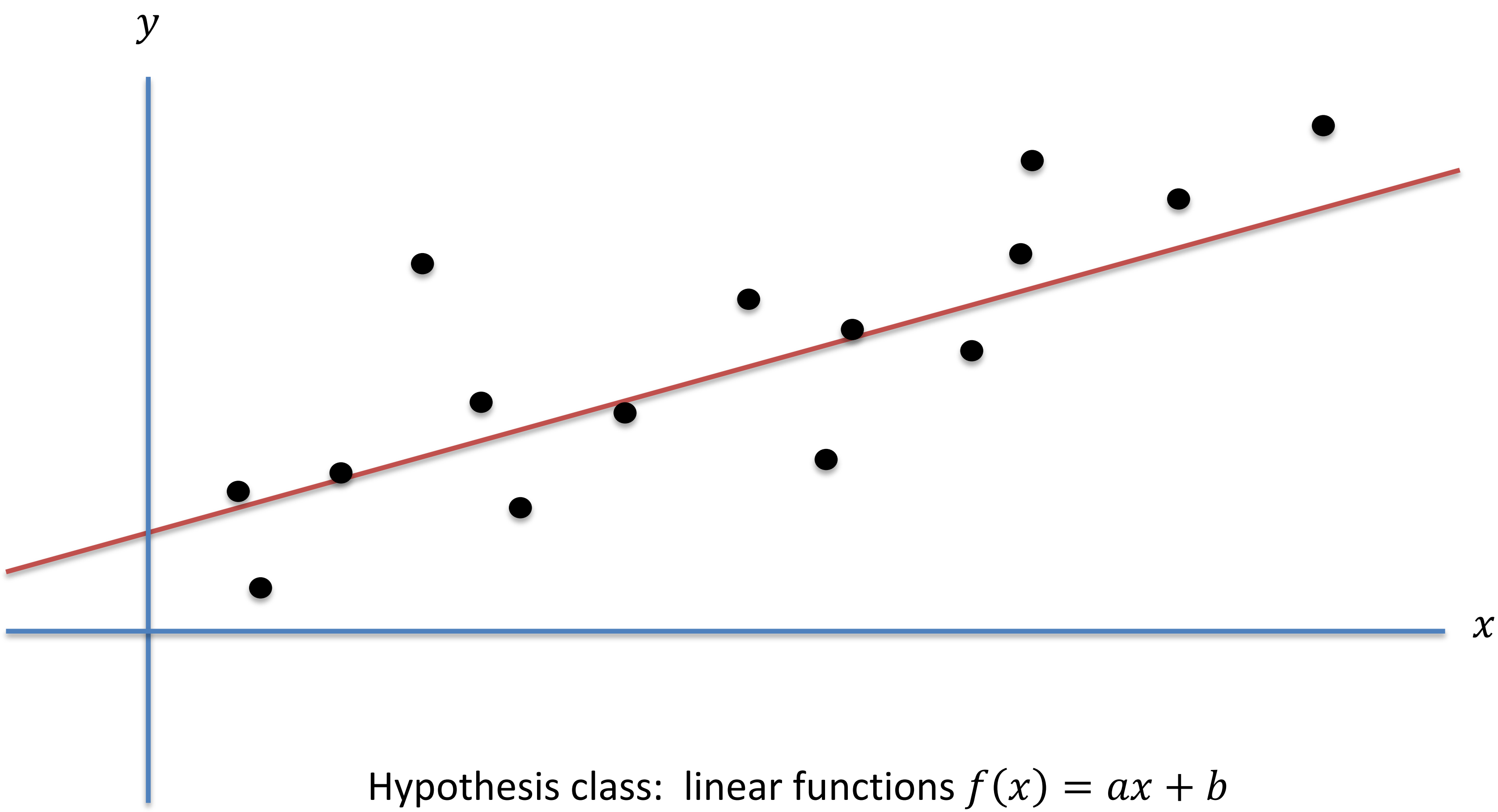
Linear Regression

- Simple linear regression
  - Input: pairs of points $\left(x^{(1)}, y^{(1)}\right), \ldots, (x^{(M)}, y^{(M)})$ with $x^{(m)} \in \mathbb{R}^d$ and $y^{(m)} \in \mathbb{R}$ (Regression)
  - Hypothesis space: set of linear functions $f(x) = a^T x + b$ with $a \in \mathbb{R}^n, b \in \mathbb{R}$
  - In one dimension, $a, b \in \mathbb{R}$ and $f(x) = ax + b$
  - Error metric and Loss Function: squared difference between the predicted value and the actual value

Regression



Regression



Hypothesis class: linear functions $f(x) = ax + b$

How do we compute the error of a specific hypothesis?

Linear Regression

- For any data point, x, the learning algorithm predicts $f(x)$
- In typical regression applications, measure the fit using a squared loss function

$$L(f) = \frac{1}{M}\sum_m \left(f\left(x^{(m)}\right) - y^{(m)}\right)^2$$

- Want to minimize the average loss on the training data
- The optimal linear hypothesis is then given by

$$\min_{a,b} \frac{1}{M}\sum_m \left(ax^{(m)} + b - y^{(m)}\right)^2$$

Linear Regression

$$\min_{a,b} \frac{1}{M} \sum_{m} \left(ax^{(m)} + b - y^{(m)}\right)^2$$

- How do we find the optimal a and b?

Linear Regression

$$\min_{a,b} \frac{1}{M} \sum_m \left(ax^{(m)} + b - y^{(m)}\right)^2$$

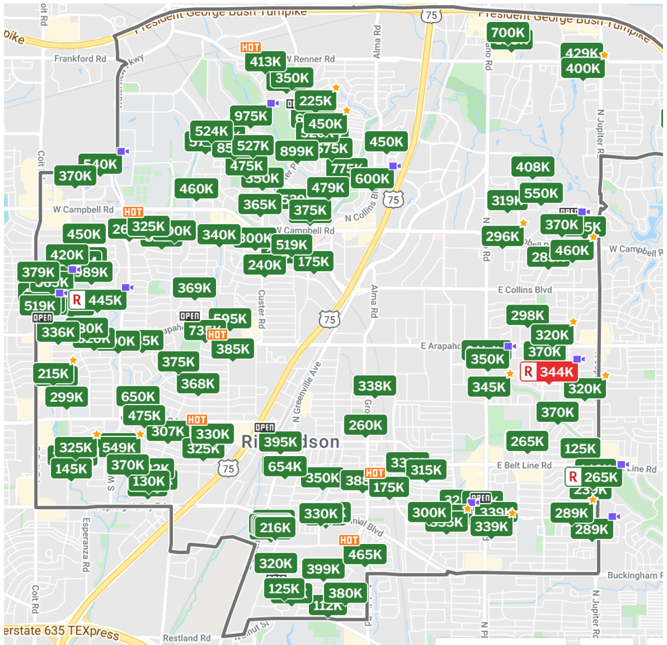
- How do we find the optimal a and b?
  - Solution 1: take derivatives and solve
    (there is a closed form solution!)
  - Solution 2: use gradient descent

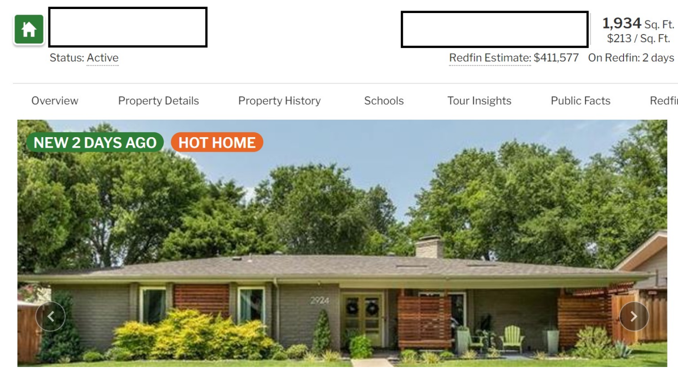
Linear Regression

$$\min_{a,b} \frac{1}{M} \sum_{m} \left(a x^{(m)} + b - y^{(m)}\right)^2$$

- How do we find the optimal a and b?
  - Solution 1: take derivatives and solve
    (there is a closed form solution!)
  - Solution 2: use gradient descent
    - This approach is much more likely to be useful for general loss functions

Recap – Housing Price Prediction Application





Home Facts

Status	Active	Time on Redfin	2 days
Property Type	Residential, Single Family	HOA Dues	$4/month
Year Built	1969	Style	Single Detached, Mid-Century Modern, Ranch, Traditional
Community	Canyon Creek Country Club 9	Lot Size	10,019 Sq. Ft.
MLS#	14375892		

Part II: Gradient Descent and Optimization

Gradient Descent

Iterative method to minimize a (convex) differentiable function f

A function $f: \mathbb{R}^n \to \mathbb{R}$ is convex if

$$\lambda f(x) + (1-\lambda) f(y) \geq f(\lambda x + (1-\lambda) y)$$

for all $\lambda \in [0,1]$ and all $x, y \in \mathbb{R}^n$

Gradient Descent

Iterative method to minimize a (convex) differentiable function f

- Pick an initial point x_0
- Iterate until convergence

$$x_{t+1} = x_t - \gamma_t \nabla f(x_t)$$

where γ_t is the t^{th} step size (sometimes called learning rate)

Basics of Convexity and Gradient Desc

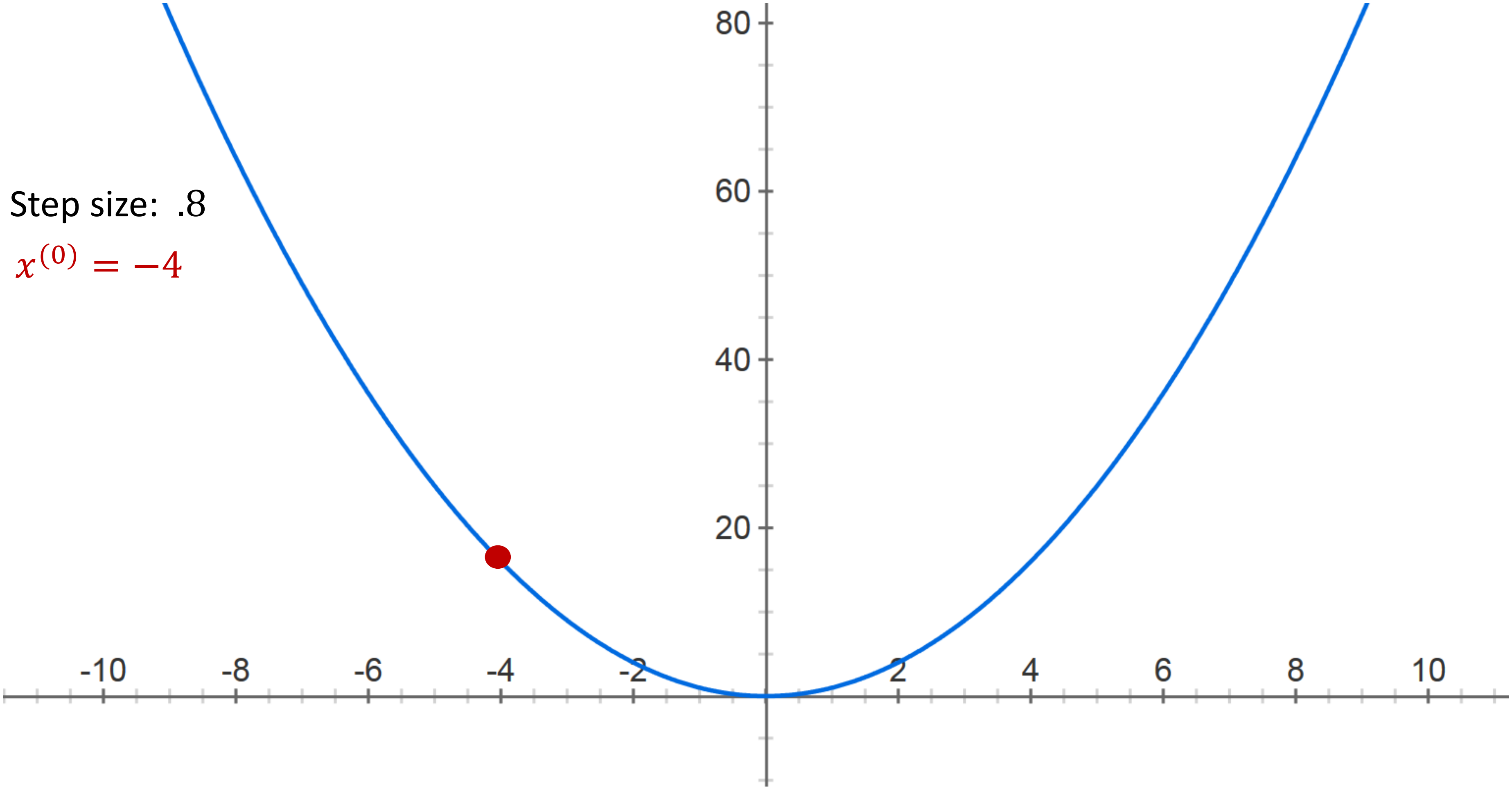
- For additional reading, please see some of my slides from my Spring 2020 Class "Optimization in Machine Learning"
- Github Location for Lecture Notes and Slides: https://github.com/rishabhk108/OptimizationML
- Please skim through:
  - Lectures 1 and 2 for basics
  - Lectures 3-5 for convex functions
  - Lectures 6-8 on Gradient Descent
  - This includes slightly more mathematical details like convergence analysis and proofs for convergence etc.

Gradient Descent

$f(x) = x^2$

Step size: .8

$x^{(0)} = -4$



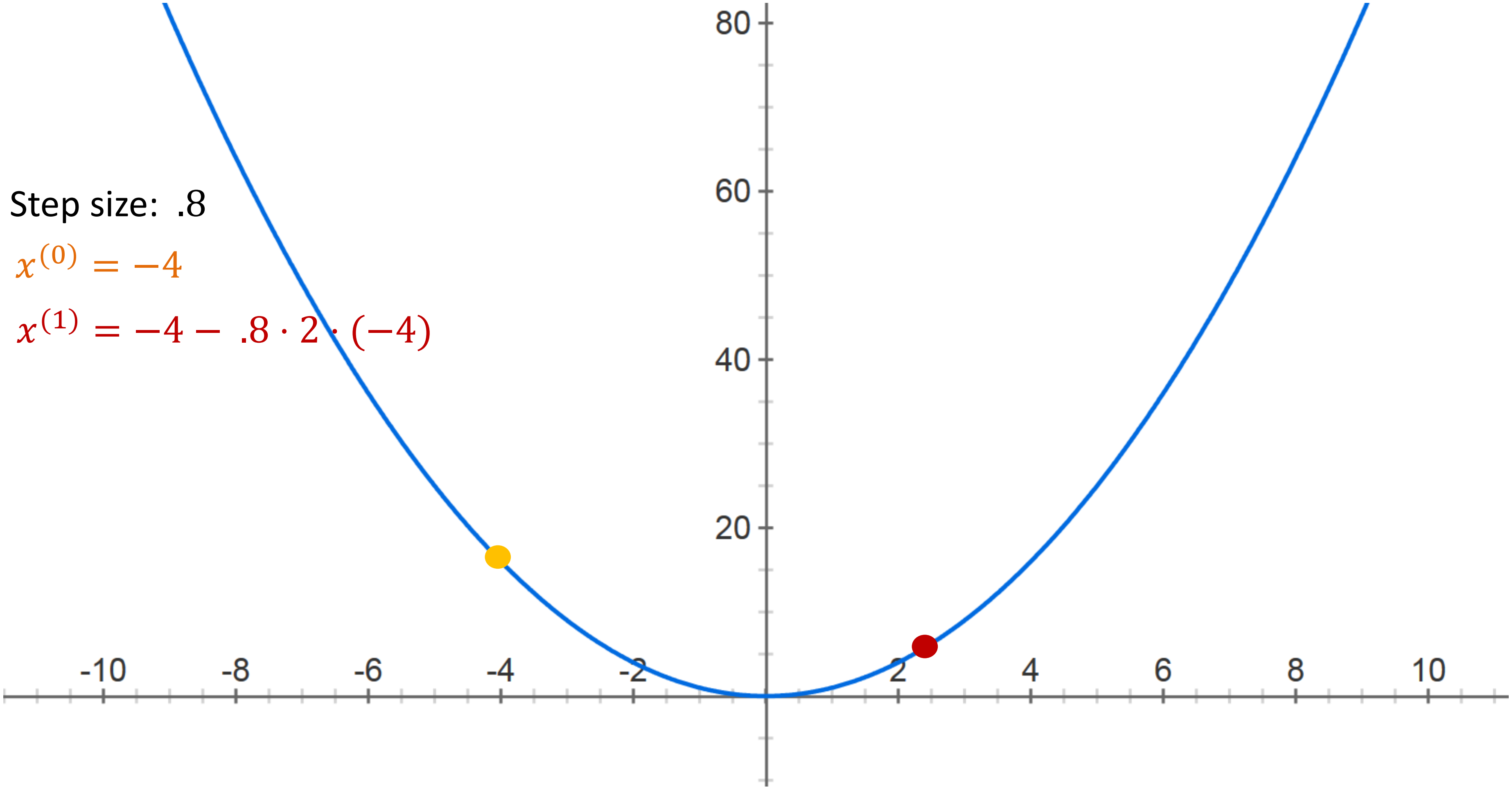
Gradient Descent

$f(x) = x^2$

Step size: .8

$x^{(0)} = -4$

$x^{(1)} = -4 - .8 \cdot 2 \cdot (-4)$



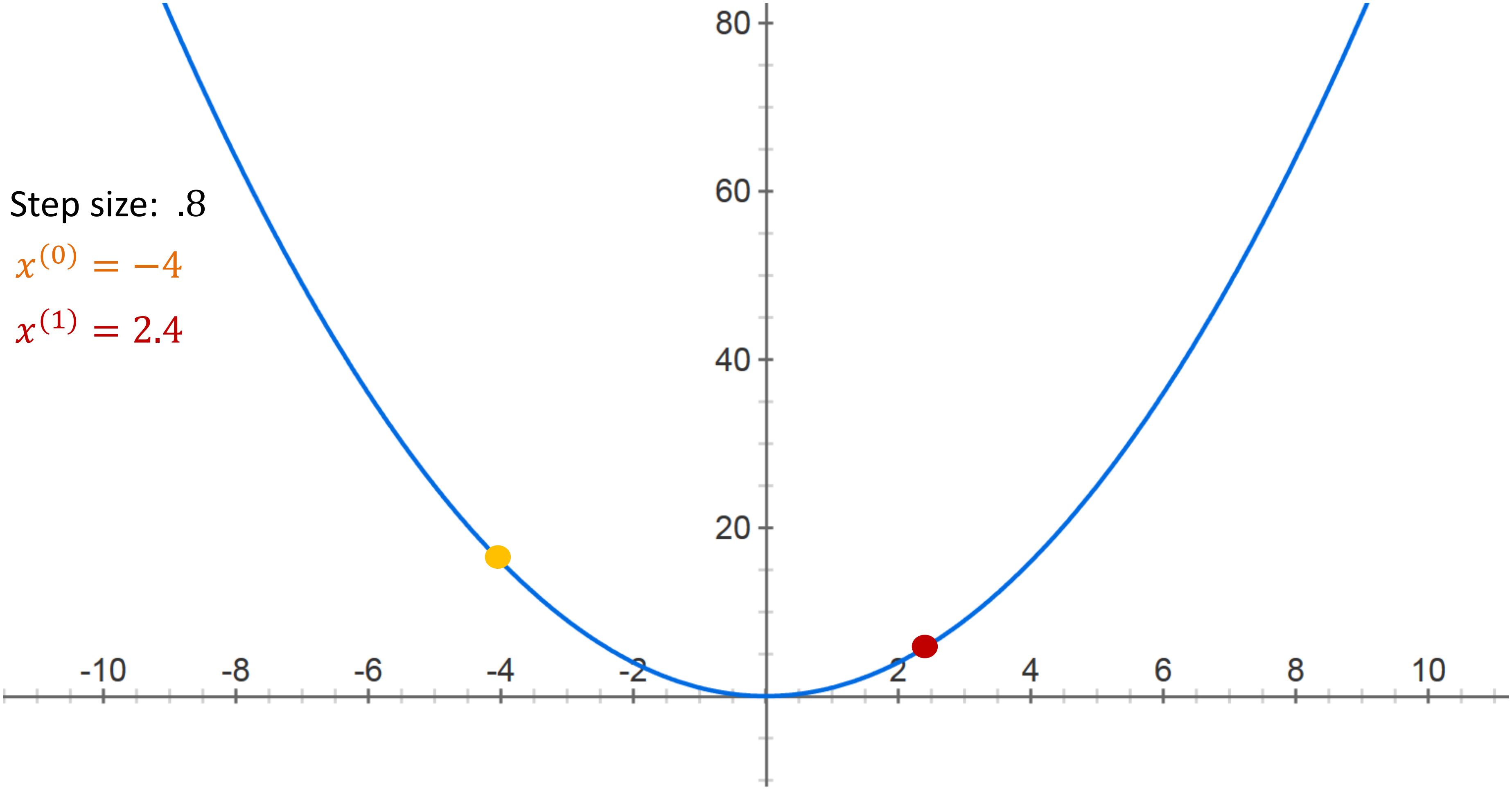
Gradient Descent

$f(x) = x^2$

Step size: .8

$x^{(0)} = -4$

$x^{(1)} = 2.4$



Gradient Descent

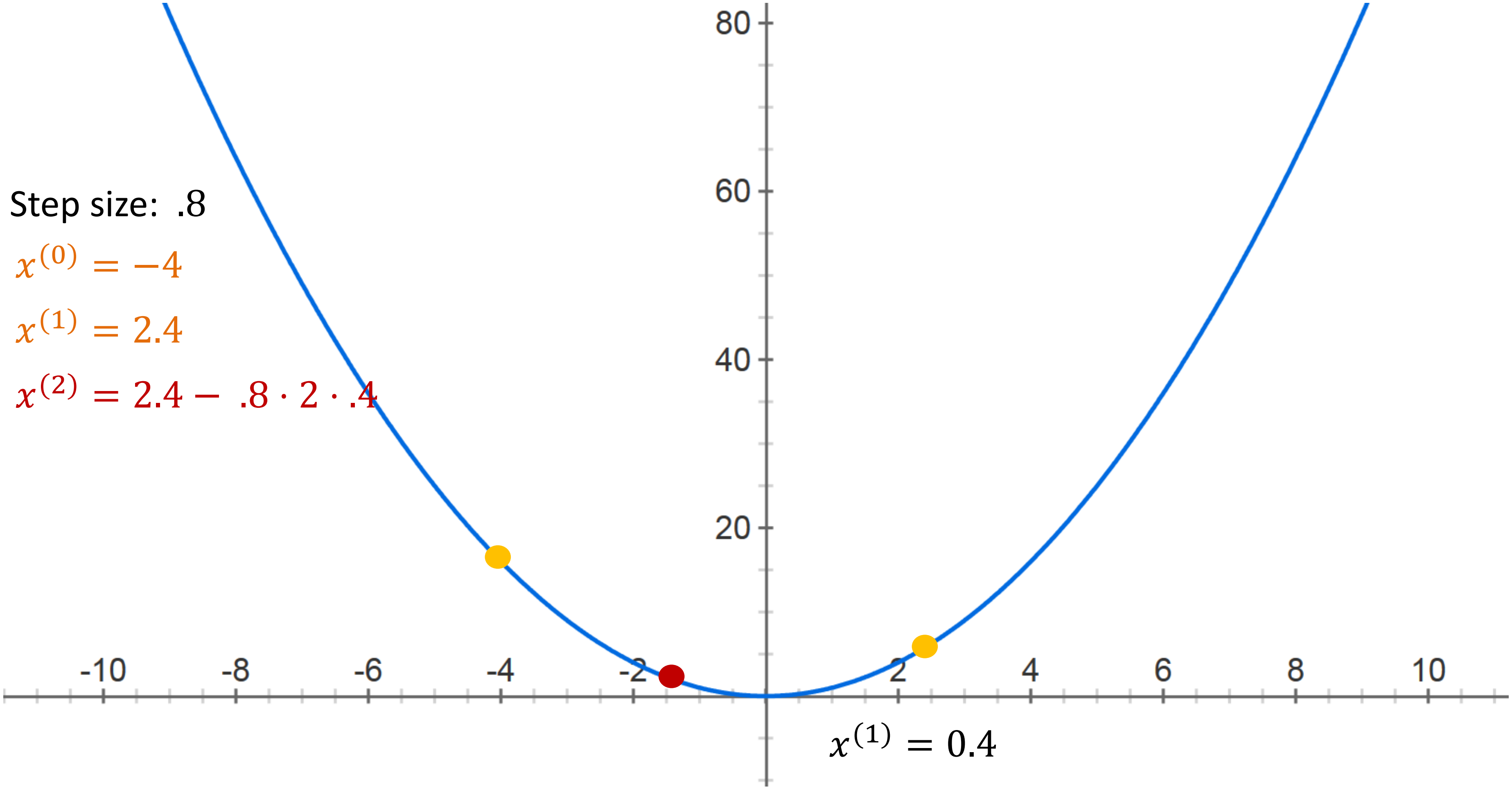
$f(x) = x^2$

Step size: .8

$x^{(0)} = -4$

$x^{(1)} = 2.4$

$x^{(2)} = 2.4 - .8 \cdot 2 \cdot .4$



$x^{(1)} = 0.4$

Gradient Descent

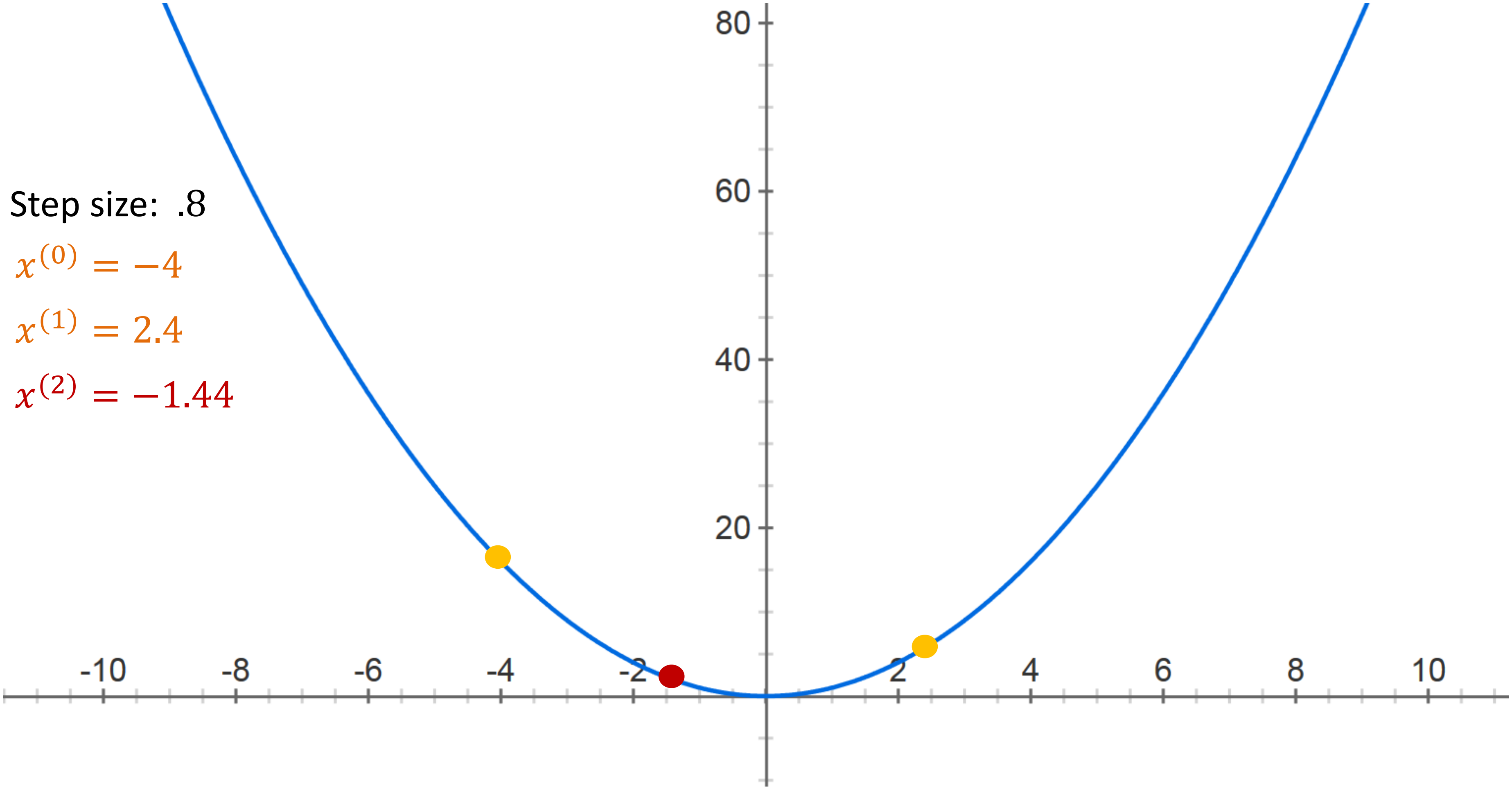
$f(x) = x^2$

Step size: .8

$x^{(0)} = -4$

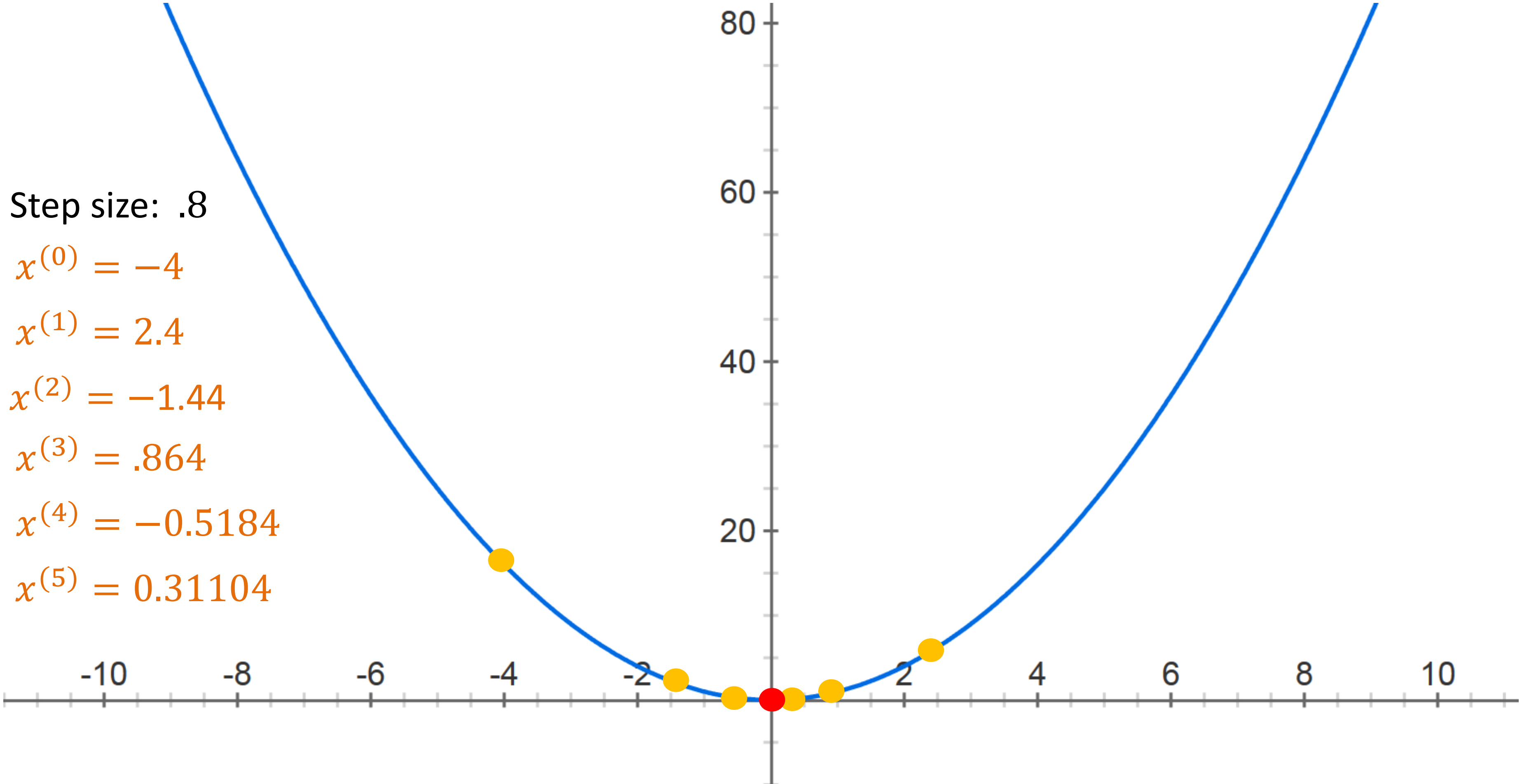
$x^{(1)} = 2.4$

$x^{(2)} = -1.44$



Gradient Descent

$f(x) = x^2$



Step size: .8

$x^{(0)} = -4$

$x^{(1)} = 2.4$

$x^{(2)} = -1.44$

$x^{(3)} = .864$

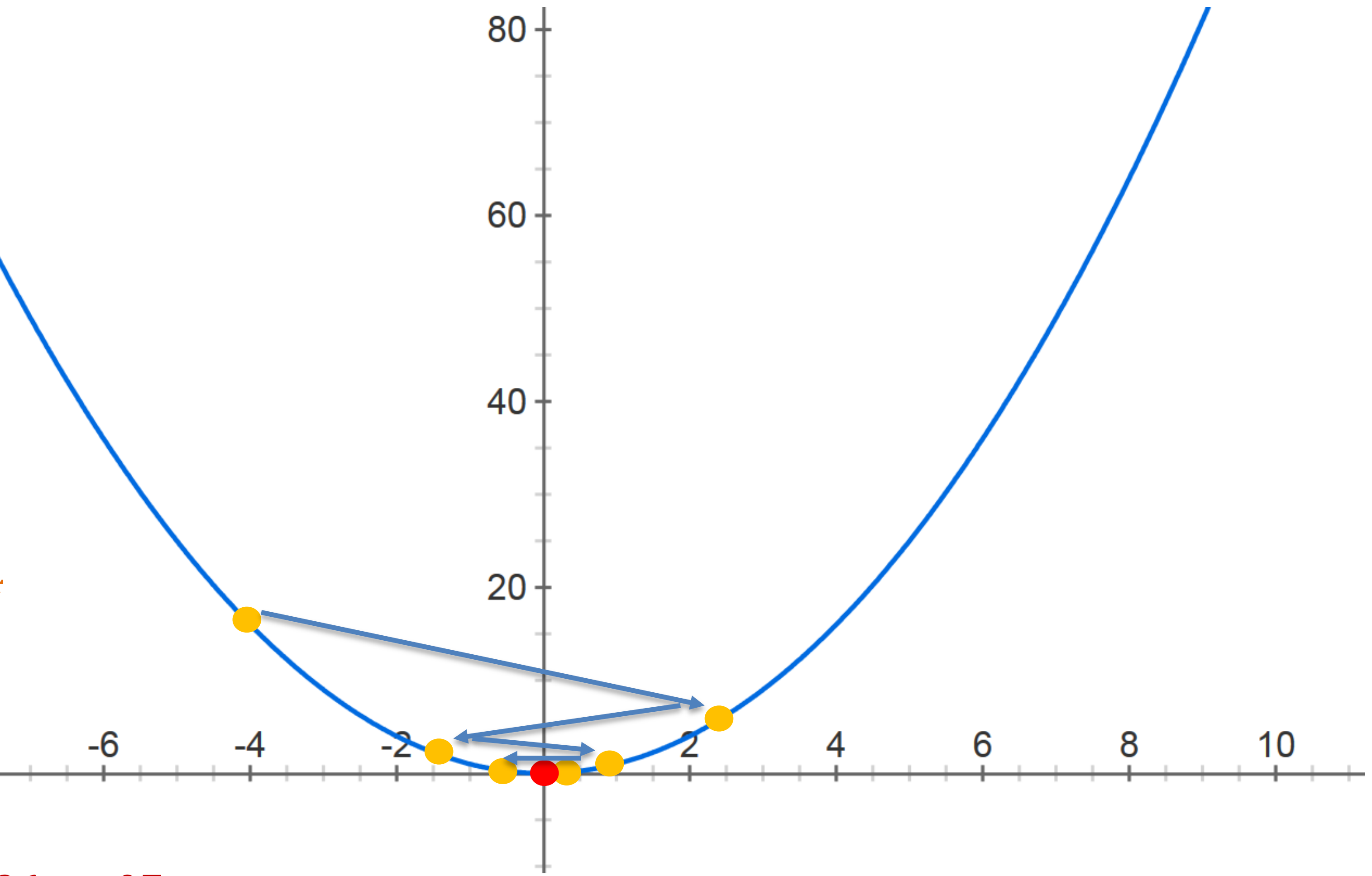
$x^{(4)} = -0.5184$

$x^{(5)} = 0.31104$

$x^{(30)} = -8.84296e - 07$

Gradient Descent: Good Convergence

$f(x) = x^2$



Step size: 0.8

$x^{(0)} = -4$

$x^{(1)} = 2.4$

$x^{(2)} = -1.44$

$x^{(3)} = .864$

$x^{(4)} = -0.5184$

$x^{(5)} = 0.31104$

$x^{(30)} = -8.84296e - 07$

Gradient Descent: Slow Convergence

$f(x) = x^2$

Step size: 0.08

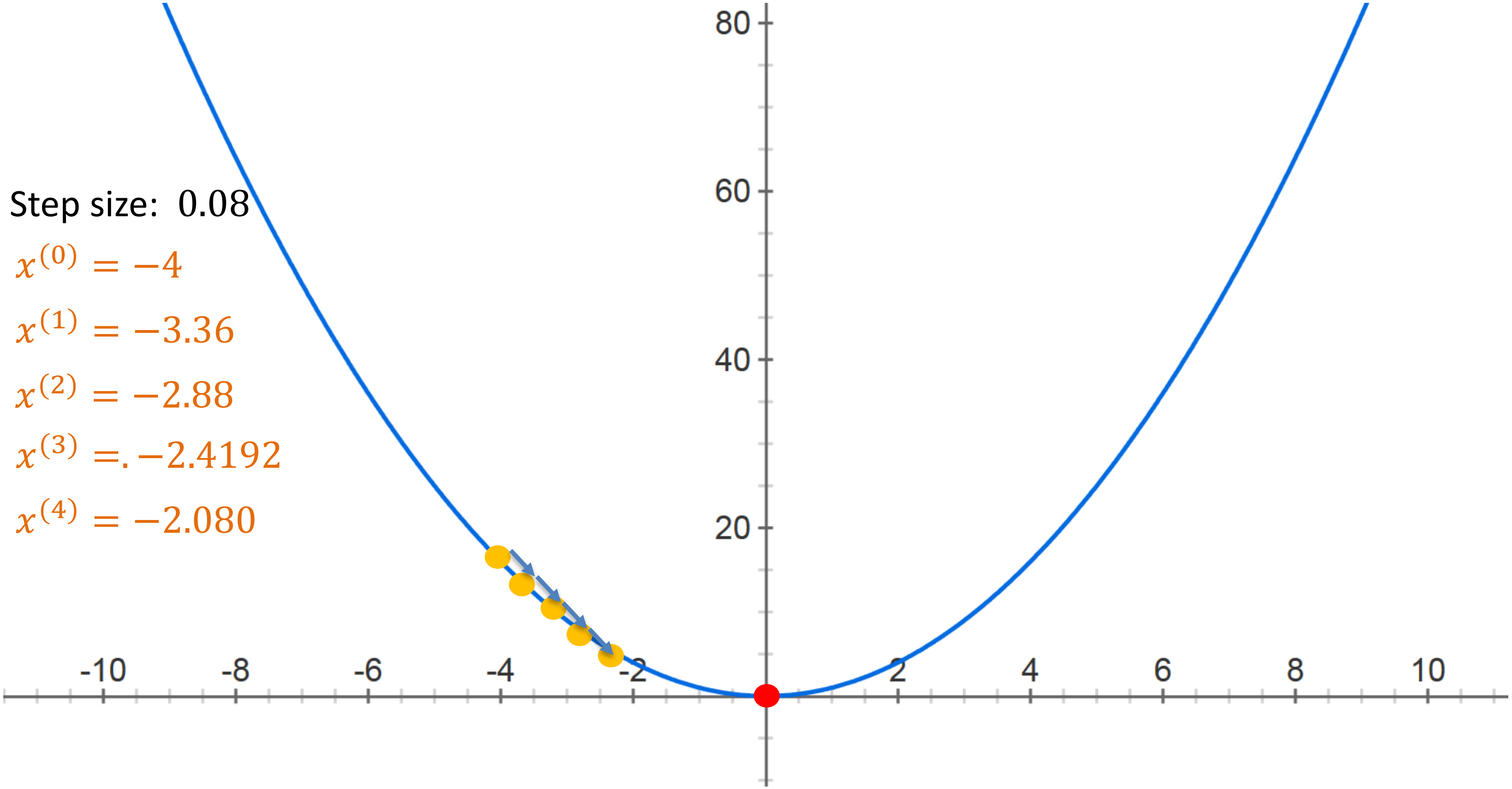
$x^{(0)} = -4$

$x^{(1)} = -3.36$

$x^{(2)} = -2.88$

$x^{(3)} =. -2.4192$

$x^{(4)} = -2.080$



Gradient Descent: Divergence

$f(x) = x^2$

Step size: 1.5

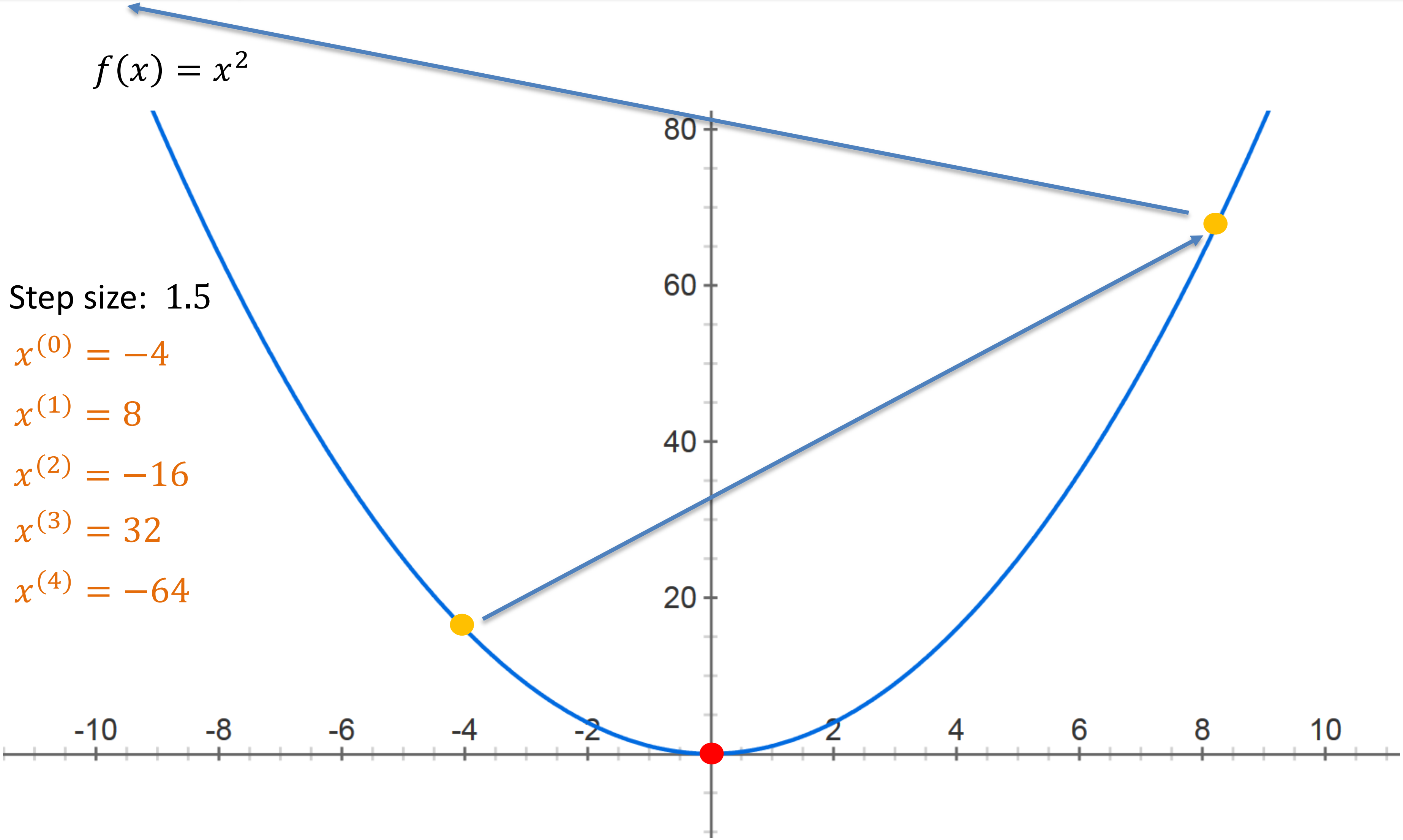
$x^{(0)} = -4$

$x^{(1)} = 8$

$x^{(2)} = -16$

$x^{(3)} = 32$

$x^{(4)} = -64$



Gradient Descent

$$\min_{a,b} \frac{1}{M} \sum_m \left(ax^{(m)} + b\ - y^{(m)}\right)^2$$

- What is the gradient of this function?

- What does a gradient descent iteration look like for this simple regression problem?

Gradients for Linear Regression

- The Loss Function for Linear Regression is:

$$L(a,b) = \frac{1}{M}\sum_{m}\left(ax^{(m)} + b - y^{(m)}\right)^2$$

- The gradients with respect to a and b are:

$$\nabla L_a(a,b) = \frac{1}{M}\sum_m 2(ax^{(m)} + b - y^m)\, x^{(m)}$$

$$\nabla L_b(a,b) = \frac{1}{M}\sum_m 2(ax^{(m)} + b - y^m)$$

- The gradients can be obtained by using the chain rule

Linear Regression

- In higher dimensions, the linear regression problem is essentially the same with $x^{(m)} \in \mathbb{R}^n$

$$\min_{a \in \mathbb{R}^n, b} \frac{1}{M} \sum_m \left(a^T x^{(m)} + b - y^{(m)}\right)^2$$

- Can still use gradient descent to minimize this
  - Not much more difficult than the $n = 1$ case

Gradient Descent

- Gradient descent converges under certain technical conditions on the function f and the step size γ_t
  - If f is convex, then any fixed point of gradient descent must correspond to a global minimum of f
  - In general, for a nonconvex function, may only converge to a local optimum
- Very fast convergence because the Linear Regression is *smooth* (loosely, think *differentiable*) and strongly convex (*loosely, bounded below by a quadratic function*)*

* See Lectures 6-8 for better understanding of smooth and strongly convex

Part III: Polynomial Regression

Polynomial Regression

- What if we enlarge the hypothesis class?
  - Quadratic functions: $ax^2 + bx + c$
  - k-degree polynomials: $a_k x^k + a_{k-1} x^{k-1} + \cdots + a_1 x + a_0$

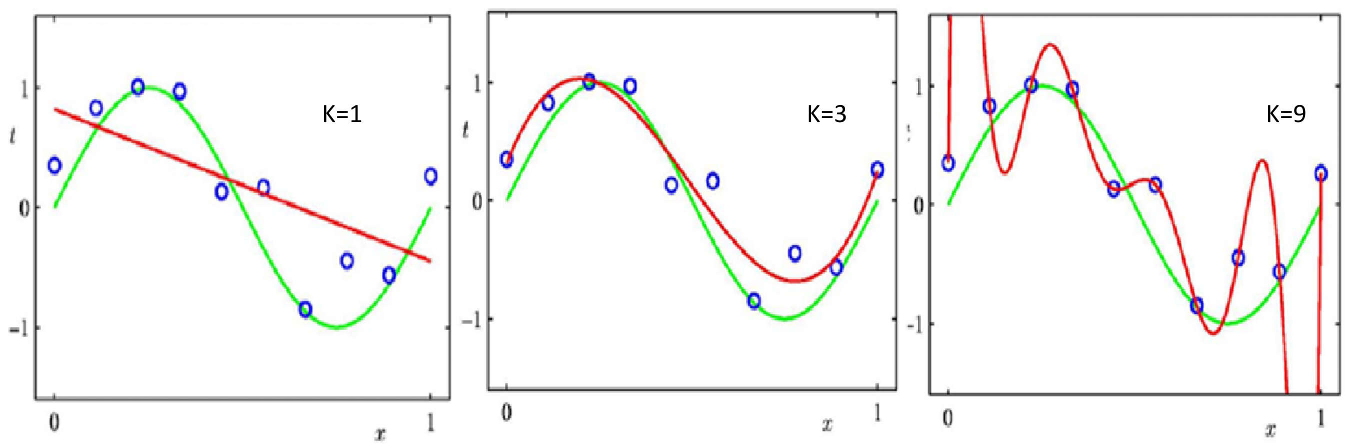
$$\min_{a_k,\ldots,a_0} \frac{1}{M} \sum_m \left(a_k \left(x^{(m)} \right)^k + \cdots + a_1 x^{(m)} + a_0 - y^{(m)} \right)^2$$

Polynomial Regression

- What if we enlarge the hypothesis class?
  - Quadratic functions: $ax^2 + bx + c$
  - k-degree polynomials: $a_k x^k + a_{k-1} x^{k-1} + \cdots + a_1 x + a_0$
- Can we always learn "better" with a larger hypothesis class?

Polynomial Regression

- What if we enlarge the hypothesis class?
  - Quadratic functions: $ax^2 + bx + c$
  - k-degree polynomials: $a_k x^k + a_{k-1} x^{k-1} + \cdots + a_1 x + a_0$
- Can we always learn "better" with a larger hypothesis class?

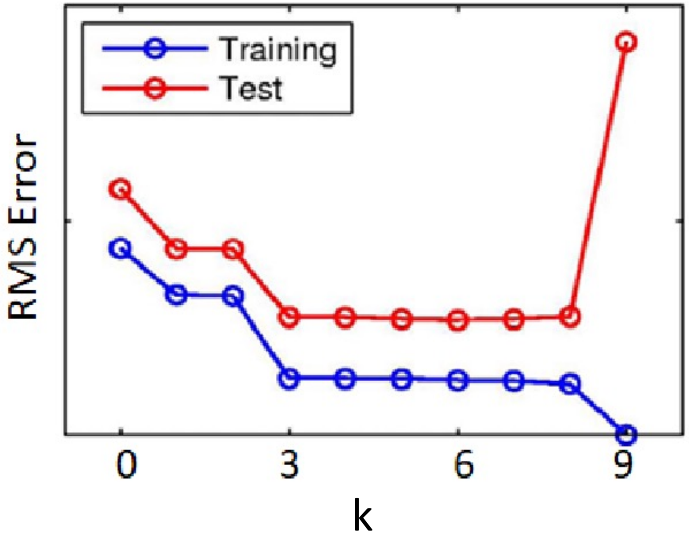


Caveats with Polynomial Regression

- Larger hypothesis space always decreases the cost function, but this does NOT necessarily mean better predictive performance
  - This phenomenon is known as overfitting
  - Ideally, we would select the simplest hypothesis consistent with the observed data
- In practice, we cannot simply evaluate our learned hypothesis on the training data, we want it to perform well on unseen data (otherwise, we can just memorize the training data!)
  - Report the loss on some held-out test data (i.e., data not used as part of the training process)

Overfitting

- As *the degree of the polynomial (k)* increases, training error decreases monotonically
- As *k* increases test error can increase
- Test error can decrease at first, but increases
- Overfitting can occur
  - When the model is too complex and trivially fits the data (i.e., too many parameters)
  - When the data is not enough to estimate the parameters
  - Model captures the noise (or the chance)



Part IV: Hands On

House Price Prediction

- Boston House Price Dataset

CRIM: Per capita crime rate by town
ZN: Proportion of residential land zoned for lots over 25,000 sq. ft
INDUS: Proportion of non-retail business acres per town
CHAS: Charles River dummy variable (= 1 if tract bounds river; 0 otherwise)
NOX: Nitric oxide concentration (parts per 10 million)
RM: Average number of rooms per dwelling
AGE: Proportion of owner-occupied units built prior to 1940
DIS: Weighted distances to five Boston employment centers
RAD: Index of accessibility to radial highways
TAX: Full-value property tax rate per $10,000
PTRATIO: Pupil-teacher ratio by town
B: $1000(Bk - 0.63)^2$, where Bk is the proportion of [people of African American descent] by town
LSTAT: Percentage of lower status of the population
MEDV: Median value of owner-occupied homes in $1000s

Summary of the Hands On Portion

- Load the Dataset
- Exploratory Data Analysis
- Training a Linear Regression Model
- Training a Polynomial Regression Model
- Training a Linear/Poly Regression from scratch using gradient descent