Week 1

Friday, July 9, 2021 9:55 PM

Machine Learning

- Grew out of work in AI
- New capability for computers

Ex Data mining
- Large datasets from growth of automation/web
  - web-click data, medical records, biology, engineering

Applications unprogramable by hand
- Autonomous helicopter, handwriting recognition, most of NLP, Computer vision ← Skill set

Self-customizing Programs
- Amazon, Netflix recommendations

Understanding human learning (brain, real AI)

Lecture # 2

Definition: Arthur Samuel (1959) Machine learning: Field of study that gives computers the ability to learn without being explicity programmed

= wrote a program for checkers, got more experience than Samuel, computer became better.

Definition: Tom Mitchell (1998) Well-posed Learning Problem: A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T as measured by P improves with experice E.

Ex: Checkers

E = playing checkers 10 thousand times

T = ~~learn which~~ is the winning ~~placements~~ (Playing checkers)

P = ~~more games won~~ (Probability of winning the next game of checkers)

Email Question:

T ~~E~~ = Classifying emails as spam or not spam

E ~~T~~ = Watching you label emails as spam or not spam

P = the # or fraction of emails getting correctly classifed

Machine Learning
- Supervised vs Unsupervised

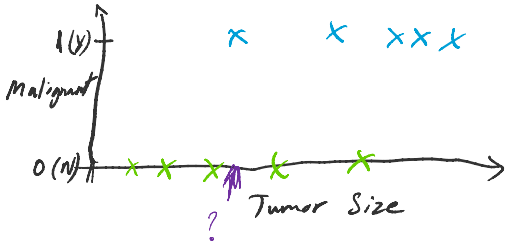
Machine Learning

- Supervised vs Unsupervised

Lecture 3 Supervised Learning

- Housing Prices prediction
  - Learning Algorithm
    - Straight line? Quadratic?
- "right answers" given
- Regression: Predict continous valued output → Ex: given picture, predict age of person in pic

Ex Breast cancer (malignant, benign)



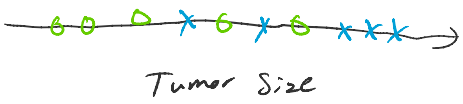
Classification: Discrete values output (1 or 0)

0, 1, 2, 3

/ | | |

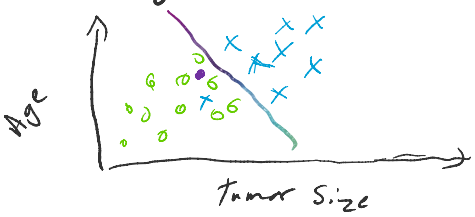
n. Type1 Type2 Type3

cancer



Different way of plotting

Lets say we have more information: Age



- Clump Thickness
- Uniformity of Cell Size
- Uniformity of Cell Shape
- ⋮

What if there are multiple amount of features

- computer will run out of space, Ram issue, heating... blah blah

↳ Support vector Machine

↳ math trick? needed?

Problems: 1. Predict when product sells

↳ regression

1. Predict [illegible]
   ↳ regression

2. hacked / comprimized accounts?
   ↳ classification

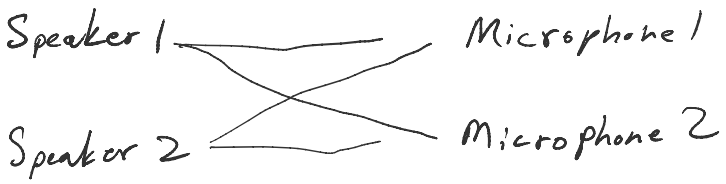
Lecture 4: Unsupervised Learning

- given data and find structure

- Clustering algorithm
  ↳ google news: same topic but different news sources
  ↳ Genes: How much a certain gene shows up

- Used to organize computing cluster
- Social network analysis
- market segmentations
- Astronomical data analysis

Cocktail party problem (non-clustering)



- overlapping sounds
  ↳ find structure
    ↳ seperates the two recording: makes each speaker much clearer
- finding structure in a chaotic environment.

One line:

```
[W,s,v] = svd((repmat(sum(x.*x,1), size(x,1),1).*x)*x');
```

Octive: Prototype your model

Question:

- Checking email is spam/not spam
  ↳ supervised: classification

[illegible] articles about same stories

↳ supervised: classification

- group articles about same stories
  ↳ unsupervised: cluster
- dividing customers into different marketing segments
  ↳ unsupervised: cluster
- Diabetes or not
  ↳ supervised: classification

Lecture 5: Model-Representation

Supervised: given the "right answer" for each example in the data

↳ training set:

Notation: m = # of training examples

x's = "input" variable / features

y's = "output" variable / "target" variable

(x, y) — one training example

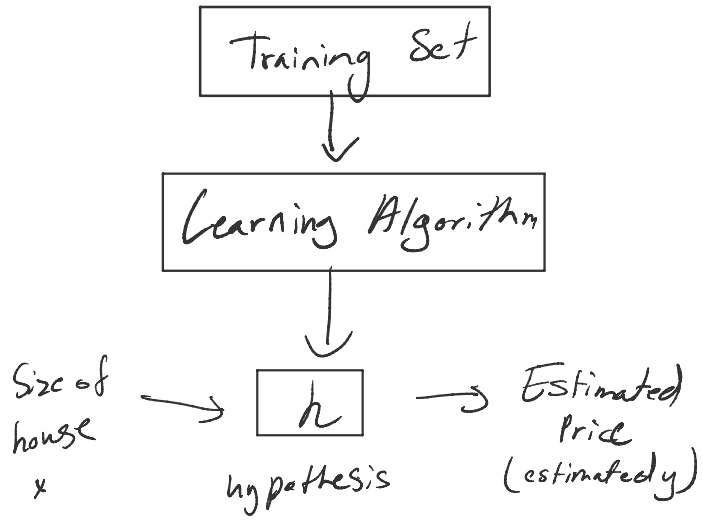
$(x^{(i)}, y^{(i)})$ — i^{th} training example

Size in ft^2 (x)	Price ($) ($y$)
2104	260
1416	332
1534	315
852	178
...	...

Ex $x^{(1)} = 2104$

$x^{(2)} = 1416$

$y^{(1)} = 460$



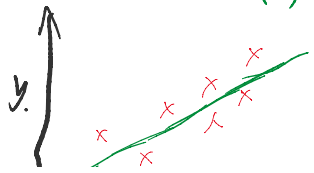
h maps from x's to y's

How do we represent h?

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

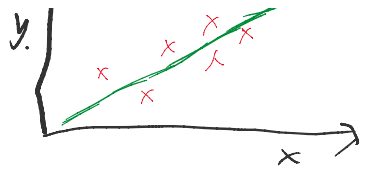
Shorthand: $h(x)$

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



h maps from x's to y's

- the target variable is continuous



Linear regression with one variable
Univariate linear regression

Lecture 6: Cost Function

θ_i's: Parameters in $h(\theta) = \theta_0 + \theta_1(x)$

How to choose?

$h(x) = 1 + .5(x)$

$\theta_0 = 1$

$\theta_1 = 0.5$

Idea: Choose θ_0, θ_1 so that $h_\theta(x)$ is close to y for our training examples (x, y).

(x, y)

- minimize θ_0, θ_1, so $\frac{1}{2m}\sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$

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

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

minimize $J(\theta_0, \theta_1)$

Cost function

Cost function = squared error function

- pretty reasonable for most regression problems

To sum, cost functions can be used to measure the accuracy

To sum, cost functions can be used to measure the accuracy of hypothesis function

- takes an average difference actually a fancier version of an average of all the results of the hypothesis with the training set (inputs from x's and actual output y's)

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (\hat{y}_i - y_i)^2 = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x_i) - y_i)^2$$

where

$\frac{1}{2}\bar{x}$ is the mean of the squares of $h_\theta(x_i) - y_i$, or the difference between the predicted value and the actual value

- Why is the mean ½-ed?
  - convenience for the computation of the gradient descent
    ↳ derivative form of the square function will cancel out the ½ term.

Lecture 7: Cost Function – Intuition 1

Hypothesis: $h_\theta(x) = \theta_0 + \theta_1 x$

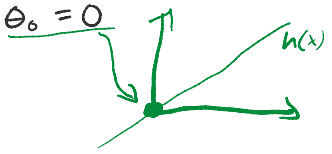
Parameters:
θ_0, θ_1

Cost function:
$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$$

Goal: minimize $J(\theta_0, \theta_1)$

Simplified Version

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



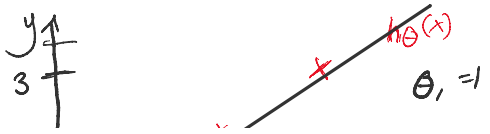
$$J(\theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$$

$h_\theta(x^{(i)}) = \theta_1 x^{(i)}$

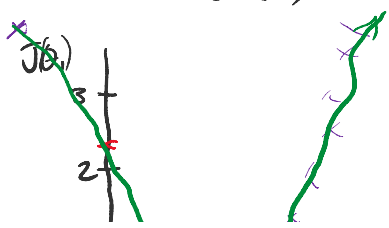
Minimize $J(\theta_1)$
θ_1

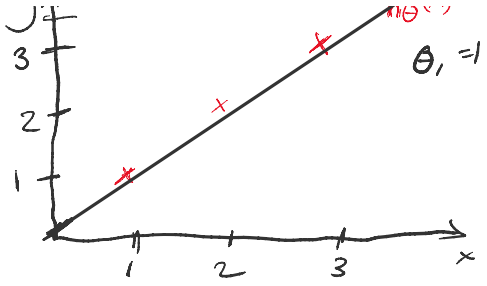
$h_\theta(x)$

for fixed θ_1, this is a function of x



$J(\theta_1)$





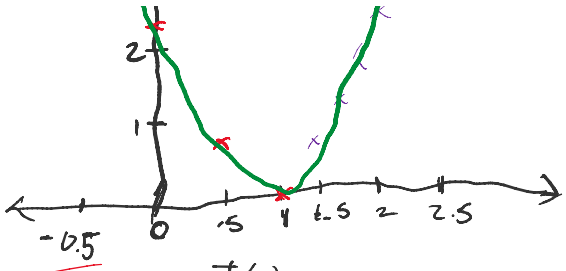
$\Leftarrow m = 3$

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

$$= \frac{1}{2(3)}\sum_{i=1}^{3}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$$

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

$$J(1) = 0$$

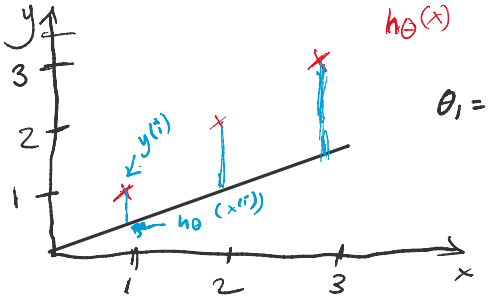


$J(1) = 0$

5.75 error

Minimize the cost function

In this situation, $\theta_1 = 1$ is our global minimum



$m = 3$

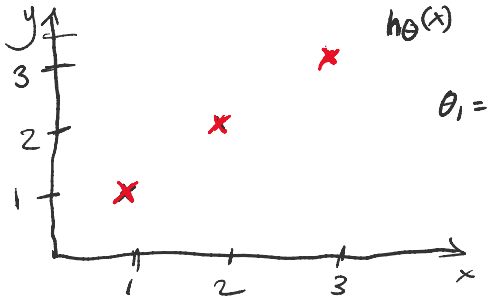
$$J(0.5) = \frac{1}{2(3)}\sum_{i=1}^{3}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$$

$$= \frac{1}{2(3)}\left[(.5-1)^2 + (1-2)^2 + (1.5-3)^2\right]$$

$$= \frac{1}{2(3)}\left[.25 + 1 + 2.25\right]$$

$$= \frac{1}{6}\left[\frac{14}{4}\right]$$

$$= \frac{7}{12} \approx 0.58333\ldots$$



$$J(0) = \frac{1}{\cdot}\left[(0-1)^2 + (0-2)^2 + (0-3)^2\right]$$

$$J(0) = \frac{1}{2(3)}\left[(0-1)^2 + (0-2)^2 + (0-3)^2\right]$$
$$= \frac{1}{6}\left[1 + 4 + 9\right]$$
$$= \frac{1}{6}\left[14\right]$$
$$= 2.333\ldots$$

Optimization Objective: $\min_{\theta_1} J(\theta_1)$

$$J'(\theta_1) = h_\theta(x^{(i)})$$

Lecture 8: Cost function: Intuition II

*Contour plots

IDK ... Redo later

Lecture 9: Parameter Learning: Gradient Descent

Linear regression with one variable

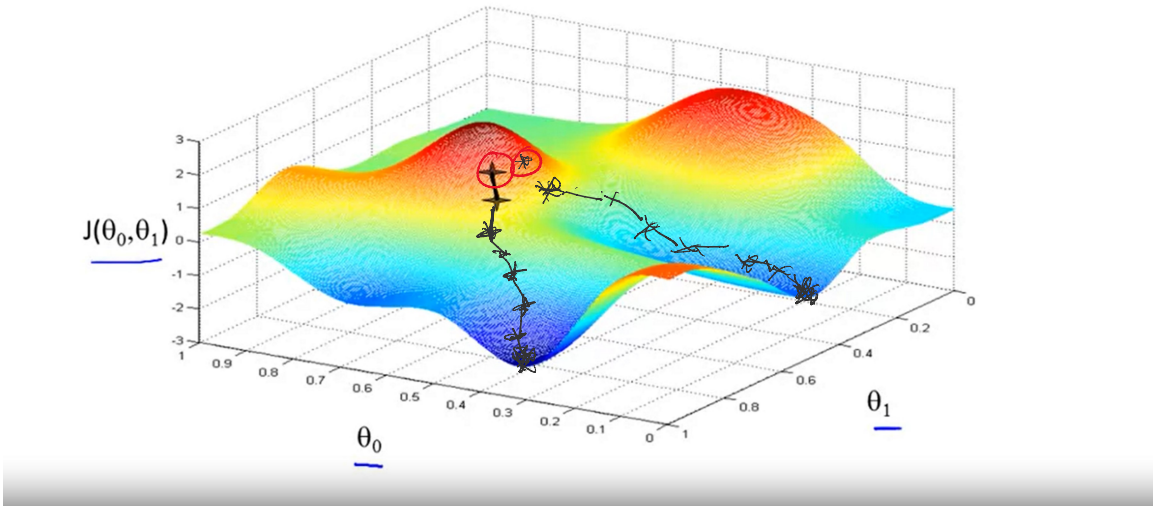
Set up:

Have some function $J(\theta_0, \theta_1)$

Want $\min_{\theta_0, \theta_1} J(\theta_0, \theta_1)$

Outline:

- Start with some θ_0, θ_1 (say $\theta_0 = 0$, $\theta_1 = 0$)
- Keep changing θ_0, θ_1 to reduce $J(\theta_0, \theta_1)$ until we end up at minimum



Gradient descent algorithm

repeat until convergence {

$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$ (for j = 0 and j = 1)

Learning Rate

Simultaneously update θ_0 and θ_1

}

θ_0, θ_1

Assignment

$a := b$

$a := a + 1$

Truth assertion

$a = b$ ✓

$a = a + 1$ ✗

Correct: Simultaneous update

$\text{temp0} := \theta_0 - \alpha \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1)$

$\text{temp1} := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1)$

$\theta_0 := \text{temp0}$

$\theta_1 := \text{temp1}$

Incorrect:

$\text{temp0} := \theta_0 - \alpha \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1)$

$\theta_0 := \text{temp0}$

$\text{temp1} := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1)$

$\theta_1 := \text{temp1}$

updates θ_0 so temp1 has different value

Question

Suppose $\theta_0 = 1, \theta_1 = 2$, and we simultaneously update θ_0 and θ_1 using the rule: $\theta_j := \theta_j + \sqrt{\theta_0 \theta_1}$ (for j = 0 and j=1) What are the resulting values of θ_0 and θ_1?

- ○ $\theta_0 = 1, \theta_1 = 2$
- ○ $\theta_0 = 1 + \sqrt{2}, \theta_1 = 2 + \sqrt{2}$
- ○ $\theta_0 = 2 + \sqrt{2}, \theta_1 = 1 + \sqrt{2}$
- ○ $\theta_0 = 1 + \sqrt{2}, \theta_1 = 2 + \sqrt{(1+\sqrt{2}) \cdot 2}$

$\theta_j := \theta_j + \sqrt{\theta_0 \theta_1}$

$\text{temp0} := \theta_0 + \sqrt{\theta_0 \theta_1}$

$\text{temp1} := \theta_1 + \sqrt{\theta_0 \theta_1}$

$\theta_0 := \text{temp0} := 1 + \sqrt{2}$

$\theta_1 := \text{temp1} := 2 + \sqrt{2}$

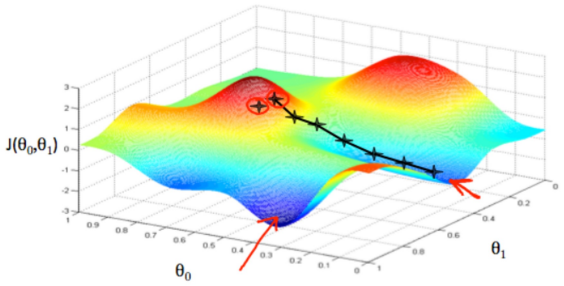
Read:

So we have our hypothesis function and we have a way of measuring how well it fits into the data. Now we need to estimate the parameters in the hypothesis function. That's where gradient descent comes in.

Imagine that we graph our hypothesis function based on its fields \theta_0∂0and \theta_1∂1(actually we are graphing the cost function as a function of the parameter estimates). We are not graphing x and y itself, but the parameter range of our hypothesis function and the cost resulting from selecting a particular set of parameters.

We put \theta_0∂0on the x axis and \theta_1∂1on the y axis, with the cost function on the vertical z axis. The points on our graph will be the result of the cost function using our hypothesis with those specific theta parameters. The graph below depicts such a setup.

From <https://www.coursera.org/learn/machine-learning/supplement/2GnUg/gradient-descent>



My understanding: this is $\int h(x) = J(x)$

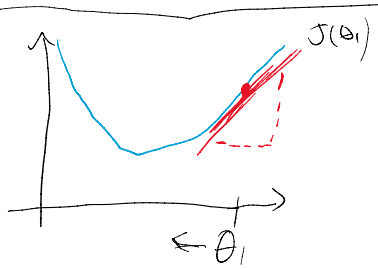
We will know that we have succeeded when our cost function is at the very bottom of the pits in our graph, i.e. when its value is the minimum. The red arrows show the minimum points in the graph.

The way we do this is by taking the derivative (the tangential line to a function) of our cost function. The slope of the tangent is the derivative at that point and it will give us a direction to move towards. We make steps down the cost function in the direction with the steepest descent. The size of each step is determined by the parameter α, which is called the learning rate.

For example, the distance between each 'star' in the graph above represents a step determined by our parameter α. A smaller α would result in a smaller step and a larger α results in a larger step. The direction in which the step is taken is determined by the partial derivative of J(\theta_0,\theta_1)/(∂0,∂1). Depending on where one starts on the graph, one could end up at different points. The image above shows us two different starting points that end up in two different places.

From <https://www.coursera.org/learn/machine-learning/supplement/2GnUg/gradient-descent>

Lecture 10: Gradient descent algorithm Intuition

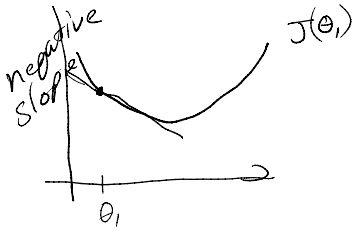


$(\theta_1 \in \mathbb{R})$

$$\theta_1 := \theta_1 - \alpha \frac{d}{d\theta} J(\theta_1)$$

$$\geq 0$$

$\theta_1 := \theta_1 - \alpha \cdot$ (positive number)



$$\frac{d}{d\theta} J(\theta_1)$$

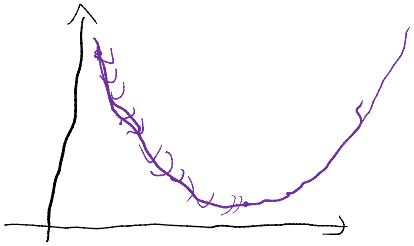
$$\leq 0$$

$\theta_1 := \theta_1 - \alpha$ (negative number)

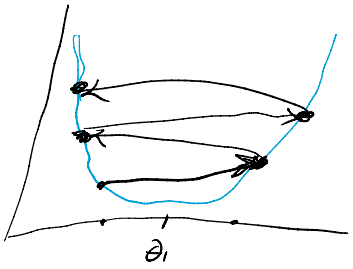
↑ Increase theta to minimize $J(\theta_1)$

$$\theta_1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_1)$$

If α is too small, gradient descent can be slow



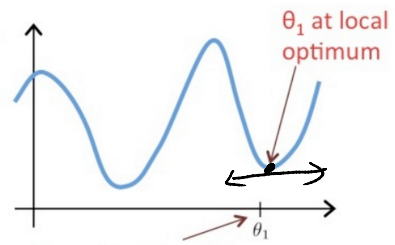
If α is too large, gradient descent can overshoot the minimum. It may fail to converge, or even diverge



overshoot the minimum

Suppose θ_1 is at a local optimum of $J(\theta_1)$, such as shown in the figure.

What will one step of gradient descent $\theta_1 := \theta_1 - \alpha \frac{d}{d\theta_1} J(\theta_1)$ do?



$$\frac{d}{d\theta_1} J(\theta_1) = 0$$

$$\theta_1 := \theta_1 - \alpha(0)$$

$\theta_1 := \theta_1$ Leave it unchanged

step (1) ↘

$$\theta_1 := \theta_1 - \alpha \frac{d^2}{d^2\theta_1} J(\theta_1)$$

then we find zeros, and then

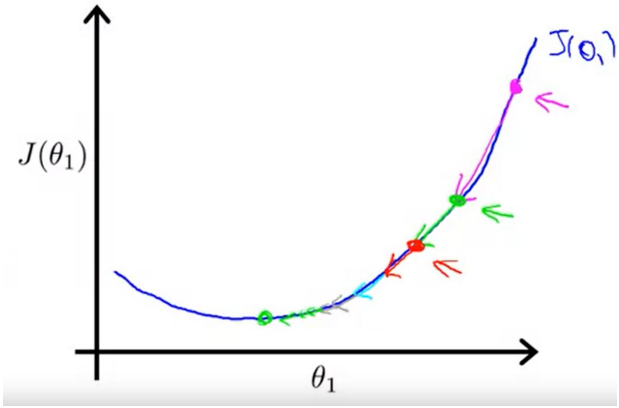
$$\theta_1 := \theta_1 - \alpha \int \frac{d^2}{d^2\theta} J(\theta_1)$$

$$\theta_1 := \theta_1 - \alpha \frac{d}{d\theta} J(\theta_1)$$

Gradient descent can converge to a local minimum, even with the learning rate α fixed.

$$\theta_1 := \theta_1 - \alpha \frac{d}{d\theta_1} J(\theta_1)$$

As we approach a local minimum, gradient descent will automatically take smaller steps. So, we need to decrease α over time



Lecture 11: Gradient Descent for Linear Regression

Gradient Descent | Linear Regression Model

Gradient Descent

repeat until convergence {

$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$

(for $j = 1$ and $j = 0$)

}

Linear Regression Model

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

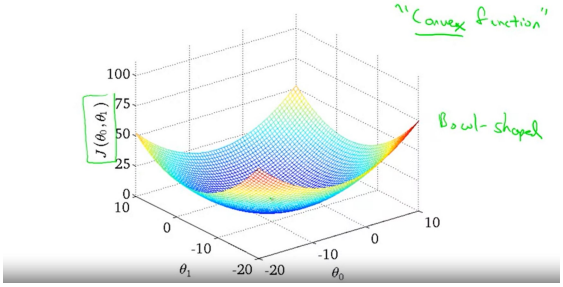
$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right)^2$

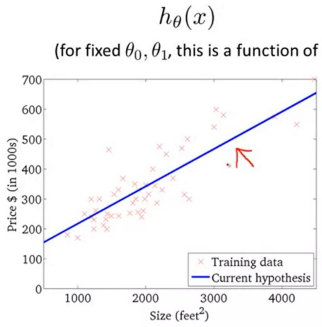
$$\frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1) = \frac{\partial}{\partial \theta_j} \cdot \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right)^2$$

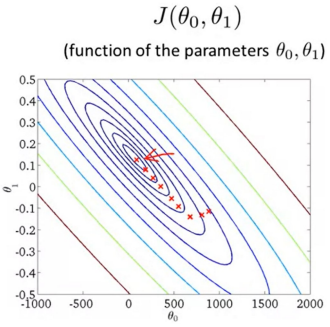
$$= \frac{\partial}{\partial \theta_j} \frac{1}{2m} \sum_{i=1}^{m} \left(\theta_0 + \theta_1 x^{(i)} - y^{(i)} \right)^2$$

θ_0 $j = 0$: $\frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right)$

θ_1 $j = 1$: $\frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right) \cdot x^{(i)}$ chain rule!







"Batch" Gradient Descent

"Batch": Each step of gradient descent uses all the training examples. → $\sum_{i=1}^{m}$

Which of the following are true statements? Select all that apply.

- [ ] To make gradient descent converge, we must slowly decrease α over time.
- [ ] Gradient descent is guaranteed to find the global minimum for any function $J(\theta_0, \theta_1)$.
- [x] Gradient descent can converge even if α is kept fixed. (But α cannot be too large, or else it may fail to converge.)

✓ **Correct**

- [x] For the specific choice of cost function $J(\theta_0, \theta_1)$ used in linear regression, there are no local optima (other than the global optimum).