CS2109S: Introduction to AI and Machine Learning

Lecture 5: Linear Regression

15 September 2023

Admin

- **Midterm (Confirmed)**
  - Date & Time: **6 October, 10:00 – 12:00**
  - Venue: **MPSH 1A**
  - Notes:
    - Please come at 9:30. You are **allowed into hall at 9:45**
    - Materials: all topics covered before recess week **until Lecture 4**
    - Cheatsheet: 1 x A4 paper, both sides

Recap

- **Machine Learning**
  - What is ML? – machine that learns through data
  - Types of Feedback: supervised, unsupervised, semi-supervised, reinforcement
  - Supervised Learning
- **Performance Measure**
  - Regression: mean squared error, mean absolute error
  - Classification: correctness, accuracy, confusion matrix, precision, recall, F1
- **Decision Trees**
  - Decision Tree Learning (DTL): greedy, top-down, recursive algorithm
  - Entropy and Information Gain
  - Different types of attributes: many values, differing costs, missing values
  - Pruning: min-sample, max-depth
  - Ensemble Methods: bagging, boosting

ERRATA

Decision Tree Learning

with Information Gain

```
def DTL(examples, attributes, default):
    if examples is empty: return default
    if examples have the same classification:
        return classification
    if attributes is empty:
        return mode(examples)
    best = choose_attribute(attributes, examples)
    tree = a new decision tree with root best
    for each value v_i of best:
        examples_i = {rows in examples with best = v_i}
        subtree = DTL(examples_i, attributes – best, mode(examples))
        add a branch to tree with label v_i and subtree subtree
```

Example	Attributes										Target
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*Wait*
X_1	T	F	F	T	Some	\$\$\$	F	T	French	0–10	T
X_2	T	F	F	T	Full	\$	F	F	Thai	30–60	F
X_3	F	T	F	F	Some	\$	F	F	Burger	0–10	T
X_4	T	F	T	T	Full	\$	F	F	Thai	10–30	T
X_5	T	F	T	F	Full	\$\$\$	F	T	French	>60	F
X_6	F	T	F	T	Some	\$\$	T	T	Italian	0–10	T
X_7	F	T	F	F	None	\$	T	F	Burger	0–10	F
X_8	F	F	F	T	Some	\$\$	T	T	Thai	0–10	T
X_9	F	T	T	F	Full	\$	T	F	Burger	>60	F
X_{10}	T	T	T	T	Full	\$\$\$	F	T	Italian	10–30	F
X_{11}	F	F	F	F	None	\$	F	F	Thai	0–10	F
X_{12}	T	T	T	T	Full	\$	F	F	Burger	30–60	T

(2,4,6) (2,2,4,4)

Patrons: (2,4,6)

2 x F 4 x T 2 x T, 4 x F

$$IG(Patrons) = 1 - \left[\frac{2}{12} I(0,1) + \frac{4}{12} I(1,0) + \frac{6}{12} I\left(\frac{2}{6}, \frac{4}{6}\right)\right] = 0.541\ bits$$

Type: (2,2,4,4)

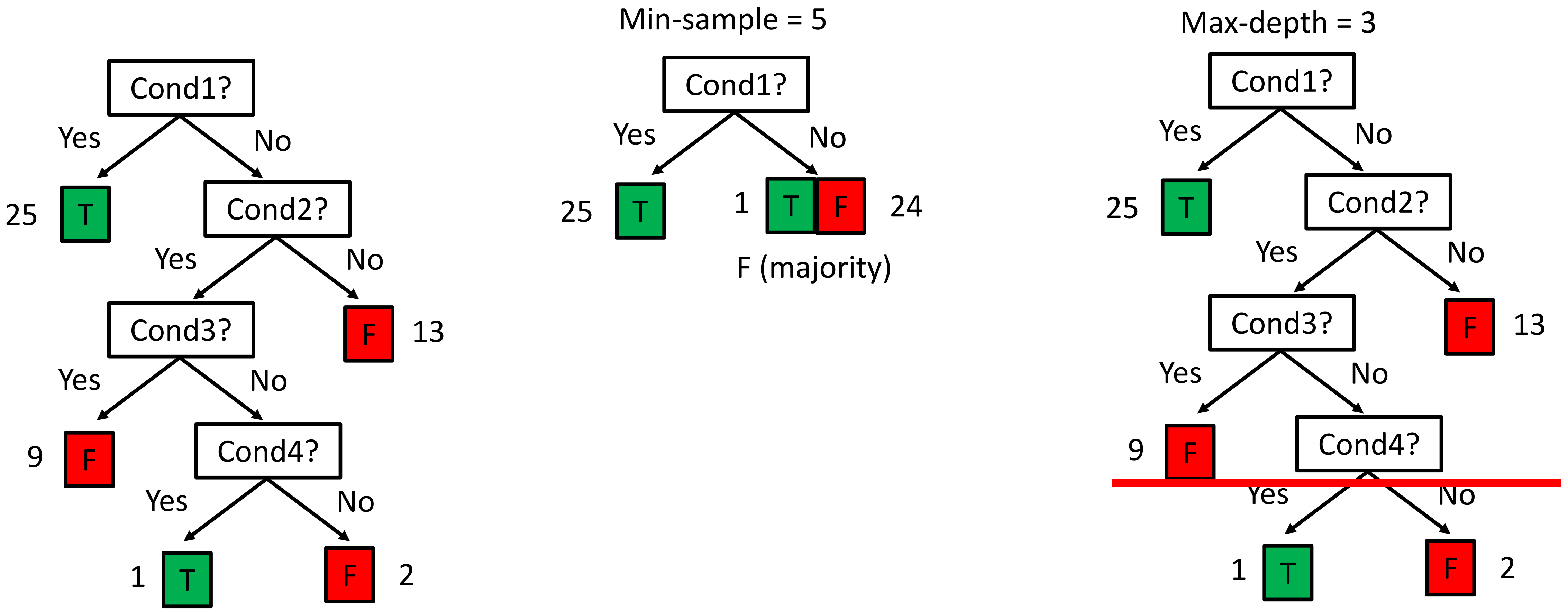
T+F T+F 2T, 2F 2T, 2 F

$$IG(Type) = 1 - [\frac{2}{12} I(\frac{1}{2}, \frac{1}{2}) + \frac{2}{12} I(\frac{1}{2}, \frac{1}{2}) + \frac{4}{12} I(\frac{2}{4}, \frac{2}{4}) + \frac{4}{12} I(\frac{2}{4}, \frac{2}{4})] = 0 \text{ bits}$$

ERRATA

Pruning

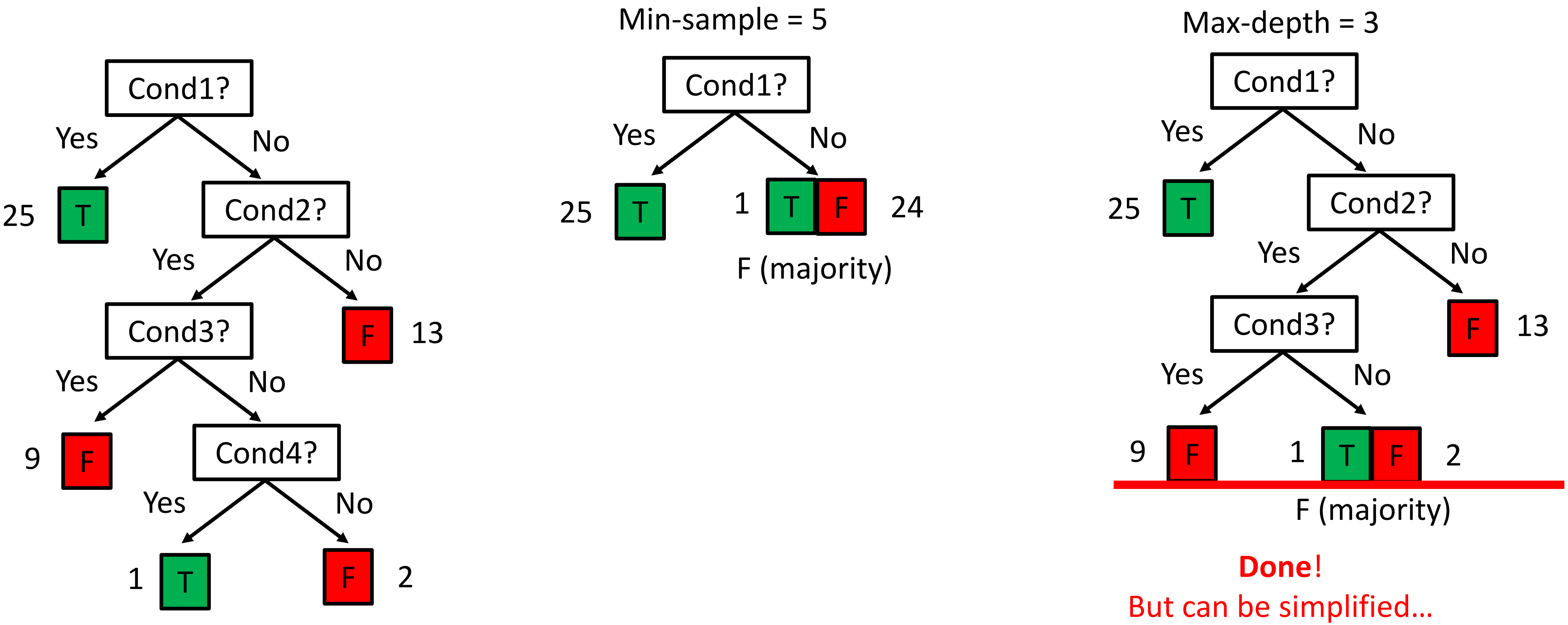
Prevent nodes from being split even when if fails to cleanly separate examples.



ERRATA

Pruning

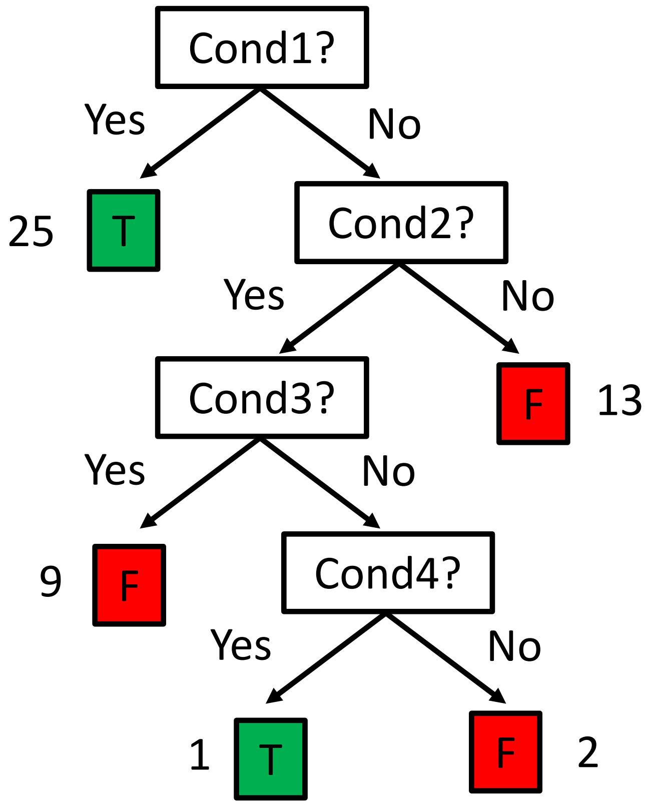
Prevent nodes from being split even when if fails to cleanly separate examples.

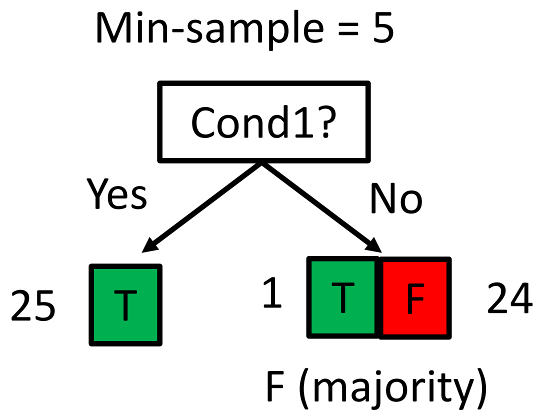


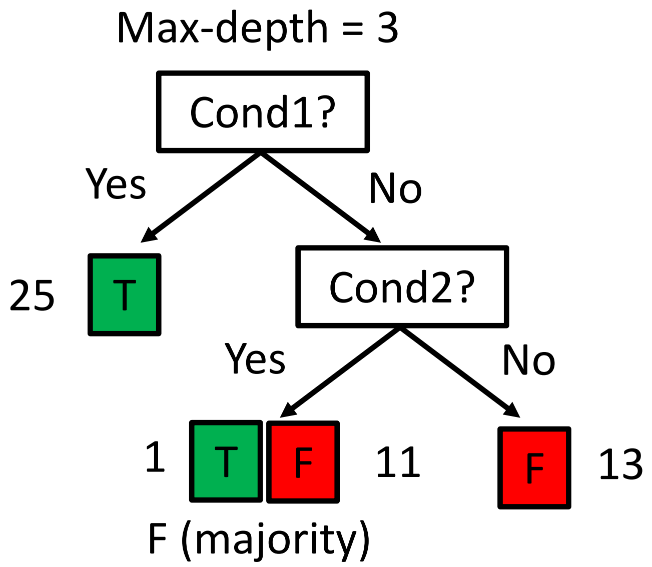
ERRATA

Pruning

Prevent nodes from being split even when if fails to cleanly separate examples.



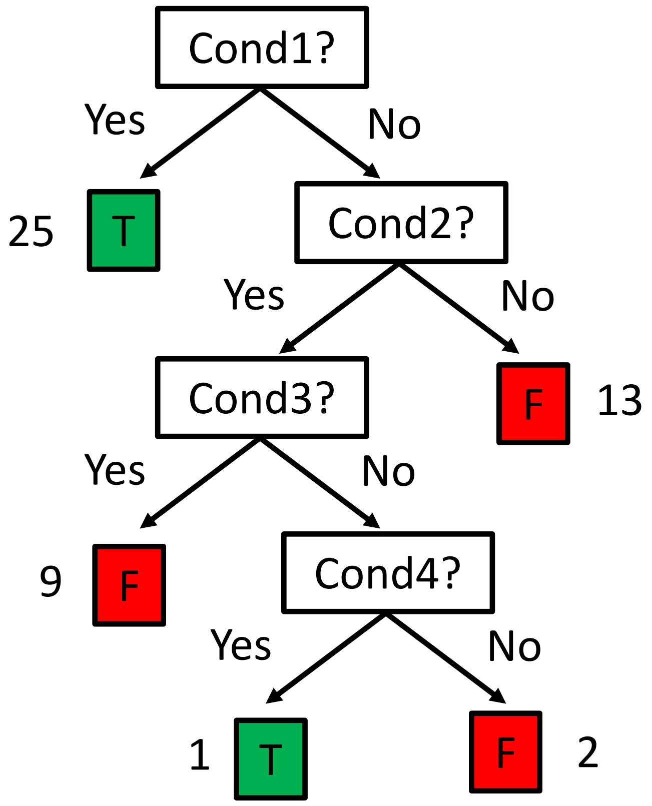




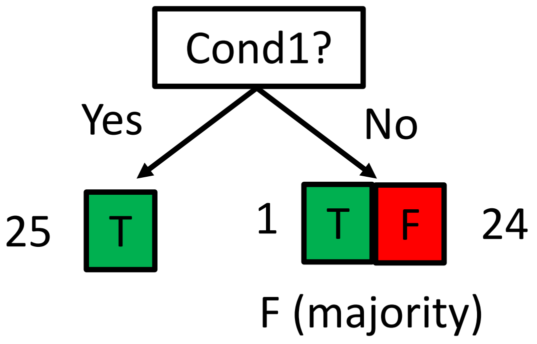
ERRATA

Pruning

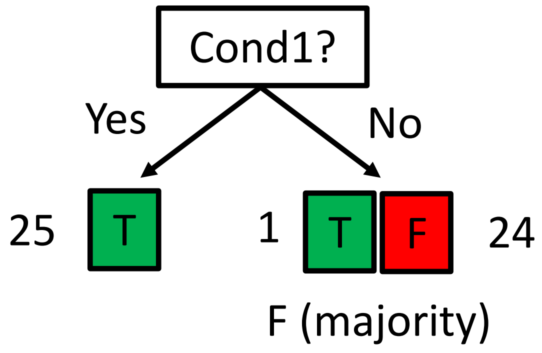
Prevent nodes from being split even when if fails to cleanly separate examples.



Min-sample = 5



Max-depth = 3



Outline

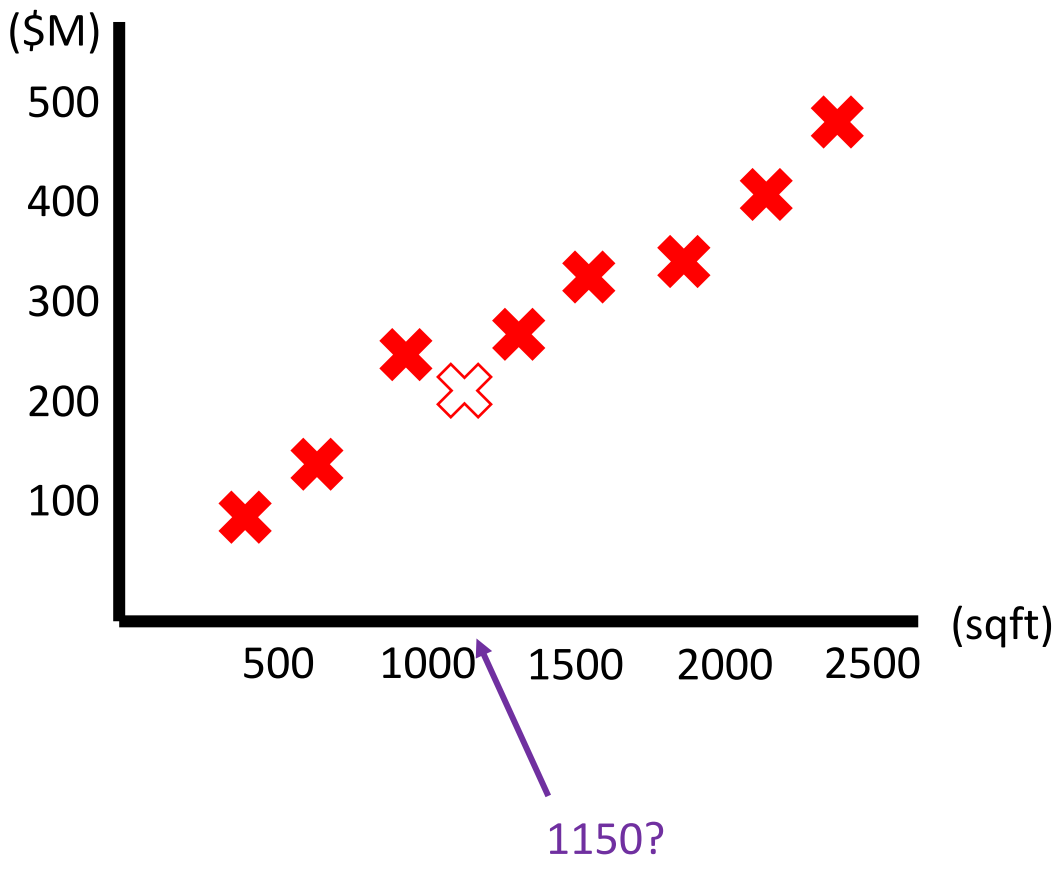
- Linear Regression
- Gradient Descent
  - Gradient Descent Algorithm
  - Linear Regression with Gradient Descent
  - Variants of Gradient Descent
- Linear Regression: Challenges and Solutions
  - Linear Regression with Many Attributes
  - Dealing with Features of Different Scales
  - Dealing with Non-Linear Relationship
- Normal Equation

Outline

- **Linear Regression**
- Gradient Descent
  - Gradient Descent Algorithm
  - Linear Regression with Gradient Descent
  - Variants of Gradient Descent
- Linear Regression: Challenges and Solutions
  - Linear Regression with Many Attributes
  - Dealing with Features of Different Scales
  - Dealing with Non-Linear Relationship
- Normal Equation

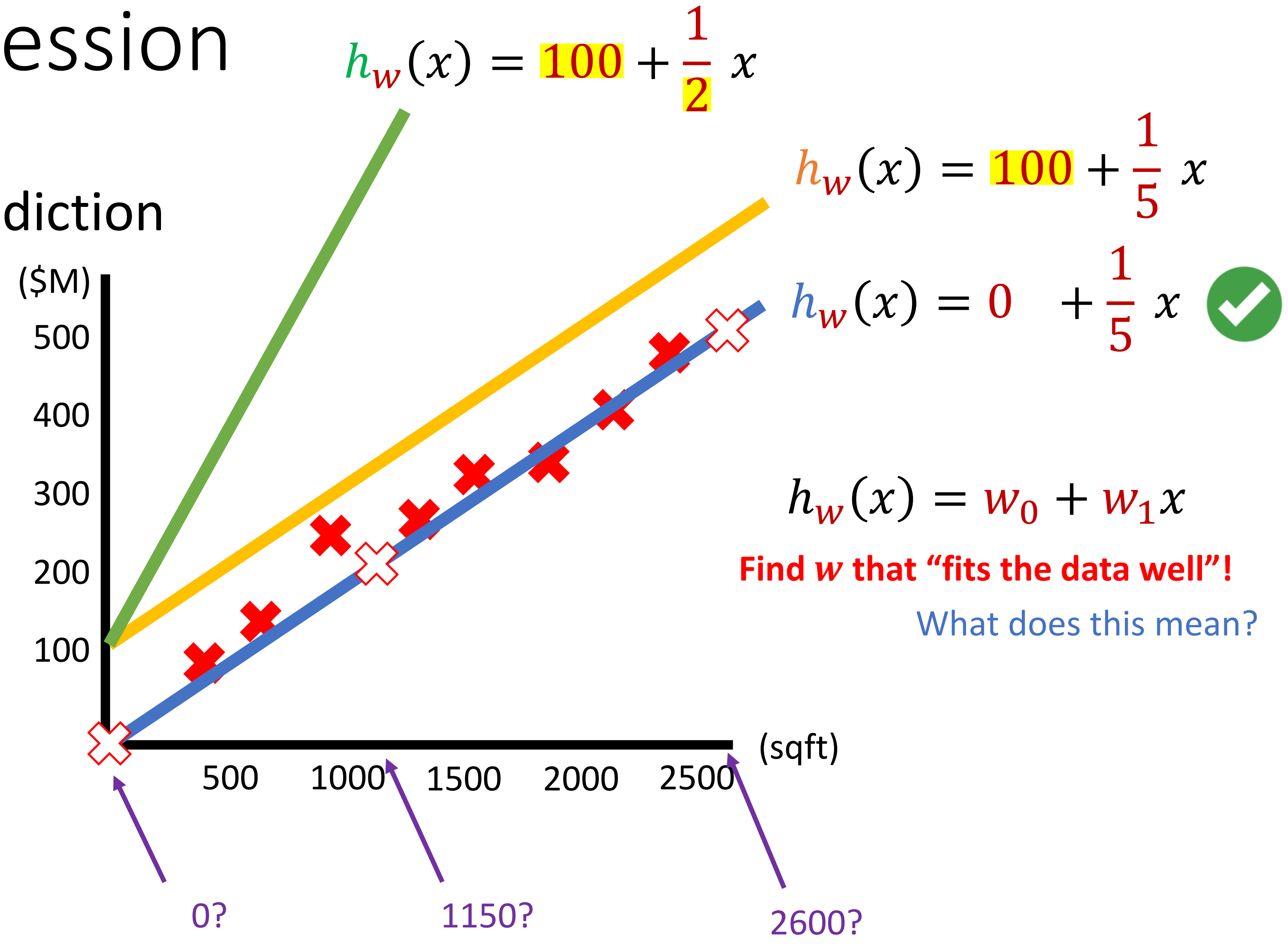
Regression

Housing price prediction



Linear Regression

Housing price prediction



$$h_w(x) = w_0 + w_1 x$$

Find w that "fits the data well"!

What does this mean?

Linear Regression: Measuring Fit

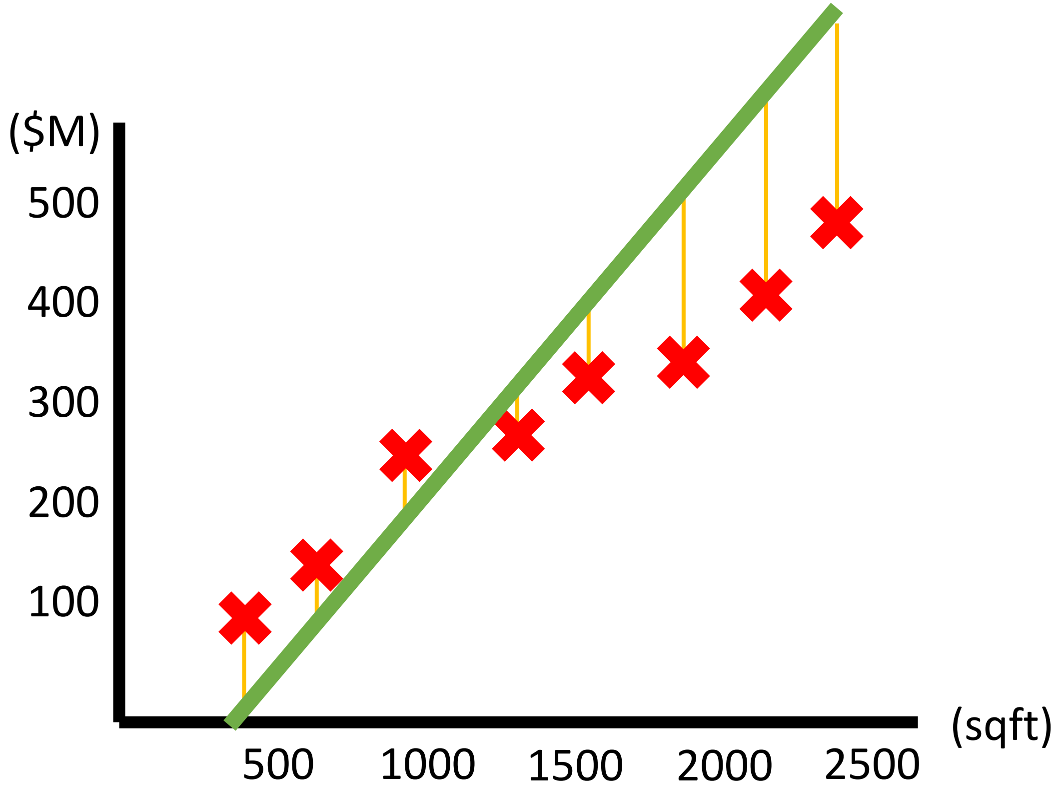
For a set of m examples
$\{(x^{(1)}, y^{(1)}), \ldots, (x^{(m)}, y^{(m)})\}$
we can compute the average (mean) **squared error** as follows.

$$J_{MSE}(w) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_w(x^{(i)}) - y^{(i)}\right)^2$$

Loss function

$\hat{y}^{(i)}$

Want to minimize the loss/error!

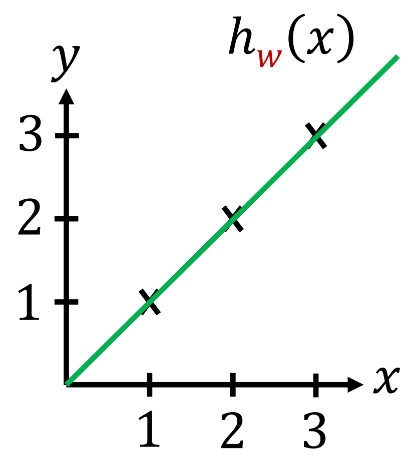


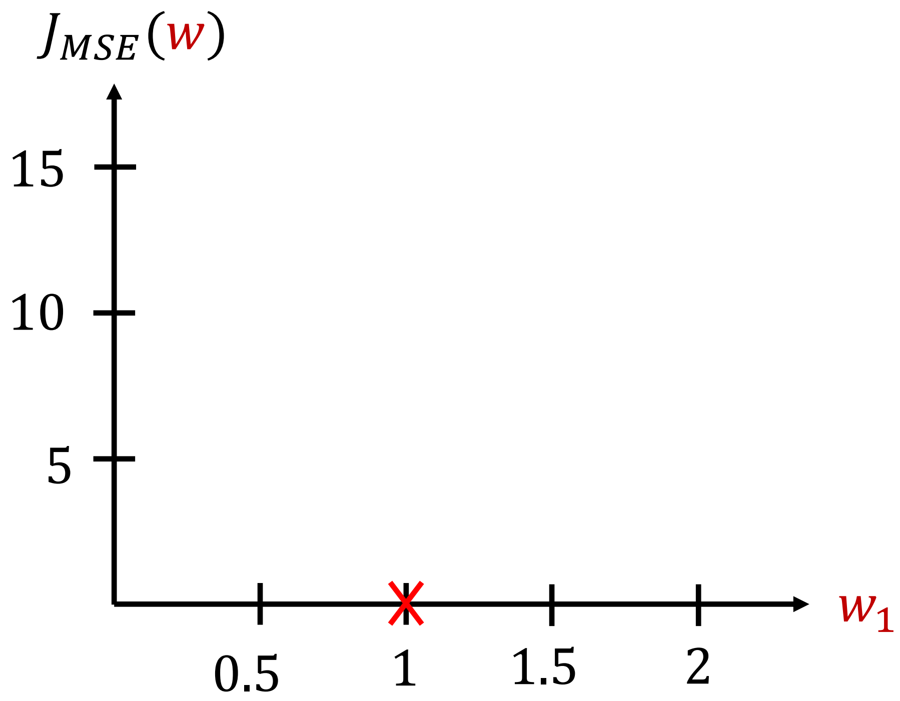
Linear Regression: Loss Landscape

Simplify: fix to 0!

Hypothesis:

$$h_w(x) = \boxed{0} + w_1 x$$





Loss Function:

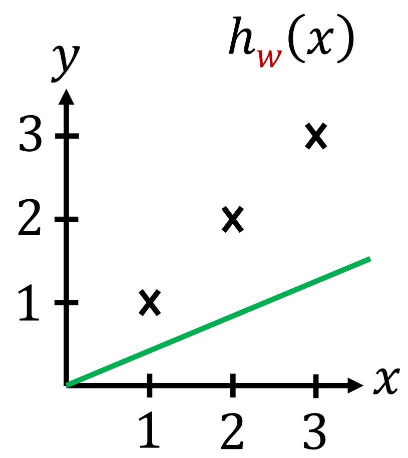
$$J_{MSE}(w) = \frac{1}{2m}\sum_{i=1}^{m}\left(w_1 x^{(i)} - y^{(i)}\right)^2$$

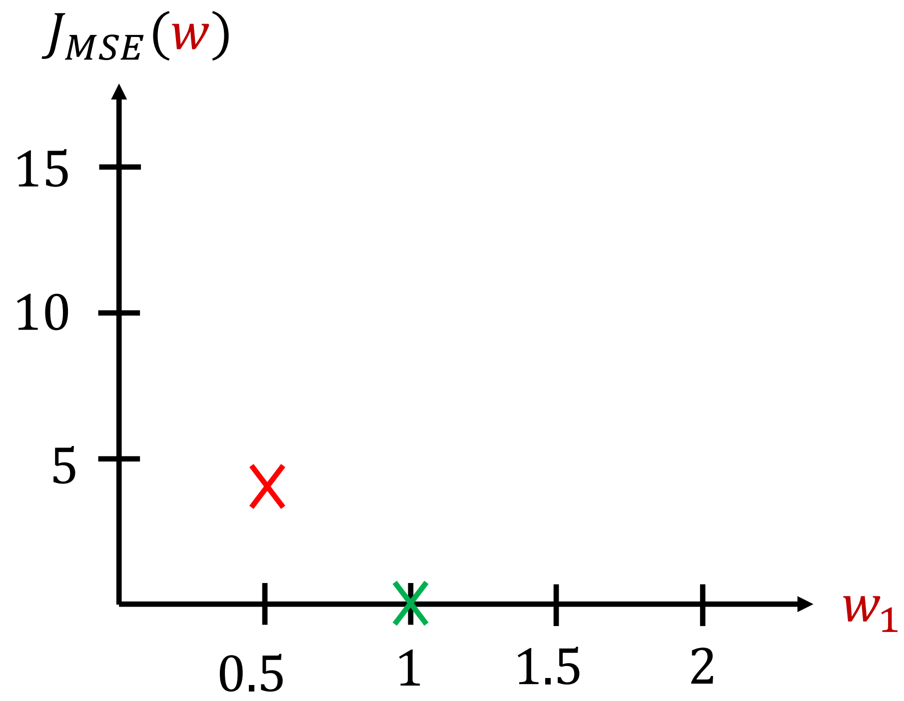
Linear Regression: Loss Landscape

Simplify: fix to 0!

Hypothesis:

$$h_w(x) = \boxed{0} + w_1 x$$





Loss Function:

$$J_{MSE}(w) = \frac{1}{2m}\sum_{i=1}^{m}\left(w_1 x^{(i)} - y^{(i)}\right)^2$$

$= (1\text{-}0.5)^2 + (2\text{-}1)^2 + (3\text{-}1.5)^2$

$= 0.5^2 + 1 + (1.5)^2$

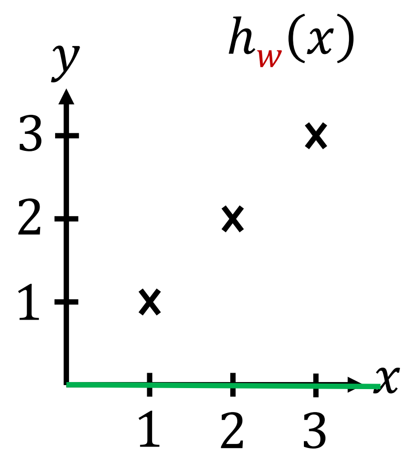
$= 3.5$

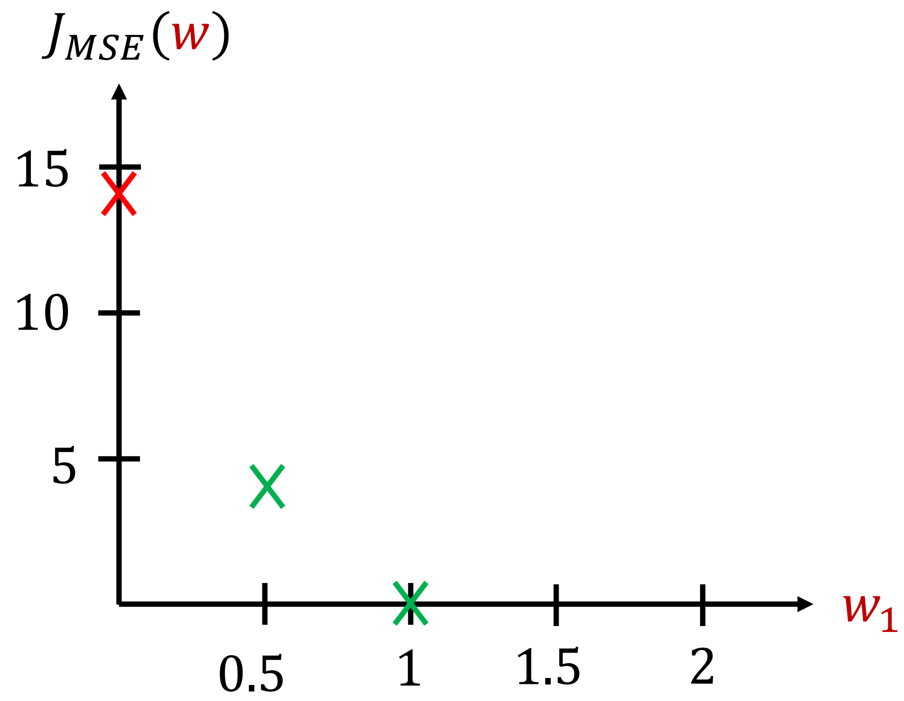
Linear Regression: Loss Landscape

Simplify: fix to 0!

Hypothesis:

$$h_w(x) = \boxed{0} + w_1 x$$





Loss Function:

$$J_{MSE}(w) = \frac{1}{2m}\sum_{i=1}^{m}\left(w_1 x^{(i)} - y^{(i)}\right)^2$$

$= 1^2 + 2^2 + 3^2$

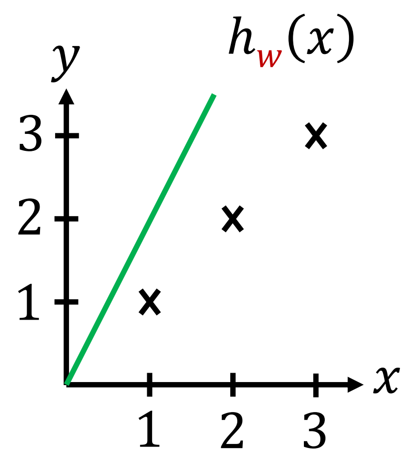
$= 14$

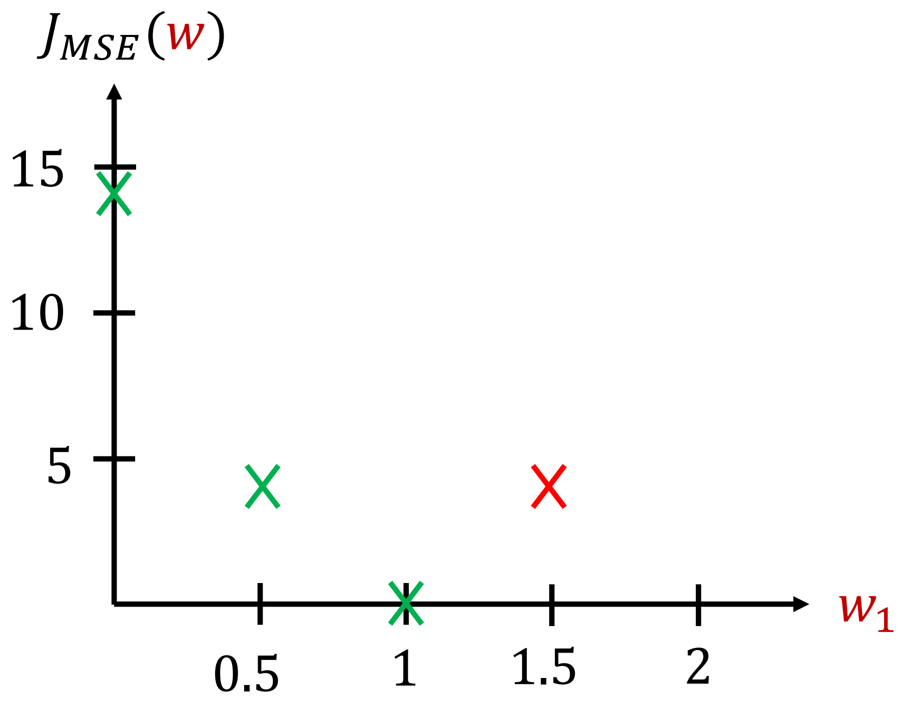
Linear Regression: Loss Landscape

Simplify: fix to 0!

Hypothesis:

$$h_w(x) = \boxed{0} + w_1 x$$





Loss Function:

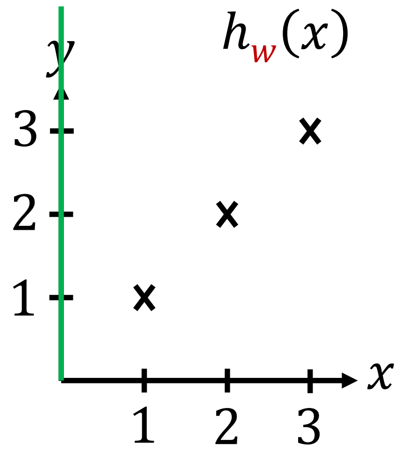
$$J_{MSE}(w) = \frac{1}{2m} \sum_{i=1}^{m} \left(w_1 x^{(i)} - y^{(i)} \right)^2$$

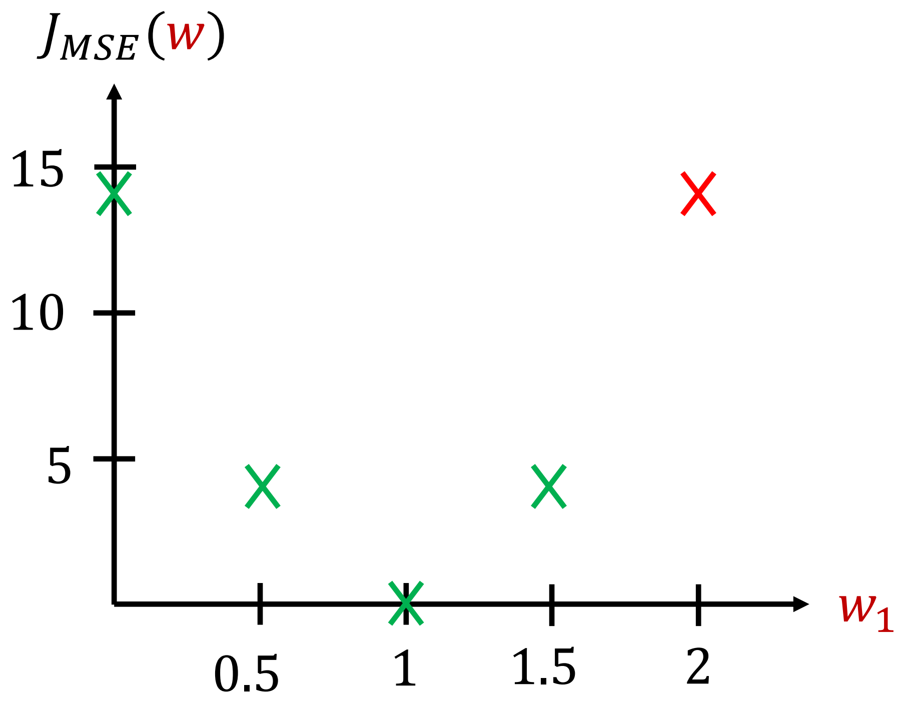
Linear Regression: Loss Landscape

Hypothesis:

Simplify: fix to 0!

$$h_w(x) = \boxed{0} + w_1 x$$





Loss Function:

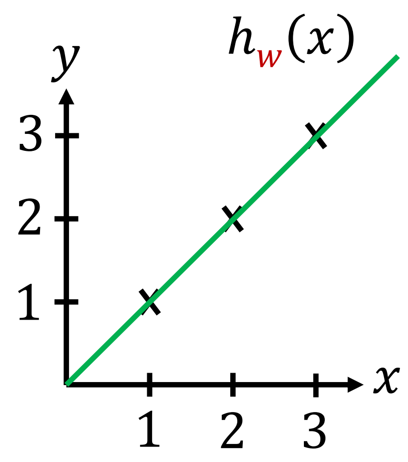
$$J_{MSE}(w) = \frac{1}{2m}\sum_{i=1}^{m}\left(w_1 x^{(i)} - y^{(i)}\right)^2$$

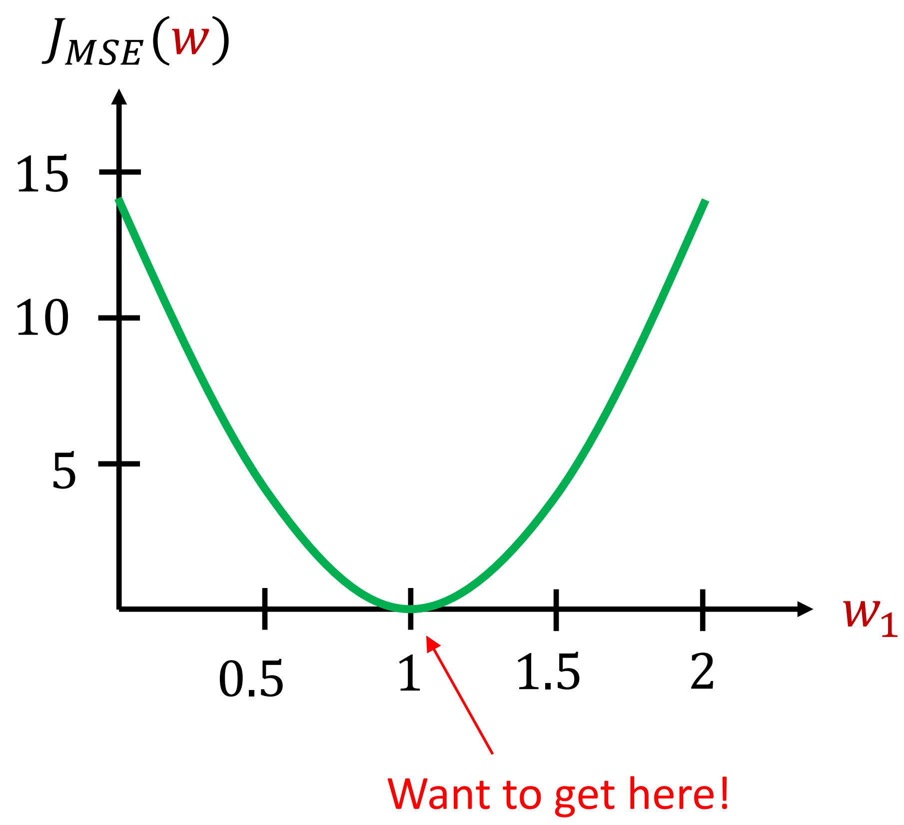
Linear Regression: Loss Landscape

Simplify: fix to 0!

Hypothesis:

$$h_w(x) = \boxed{0} + w_1 x$$





Loss Function:

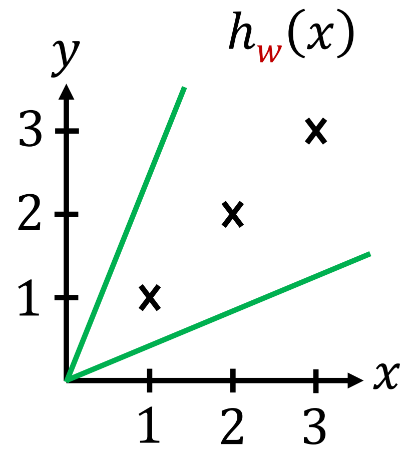
$$J_{MSE}(w) = \frac{1}{2m}\sum_{i=1}^{m}\left(w_1 x^{(i)} - y^{(i)}\right)^2$$

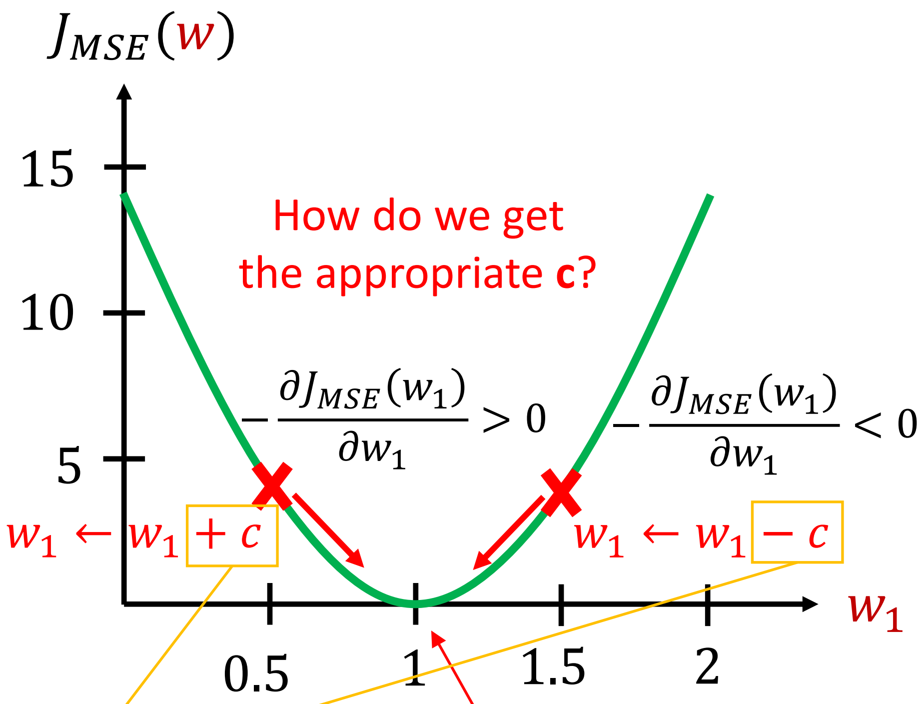
Linear Regression: Loss Landscape

Simplify: fix to 0!

Hypothesis:

$$h_w(x) = \boxed{0} + w_1 x$$





Want to get here!

Loss Function:

$$J_{MSE}(w) = \frac{1}{2m} \sum_{i=1}^{m} \left(w_1 x^{(i)} - y^{(i)} \right)^2$$

$$\frac{\partial J_{MSE}(w)}{\partial w} = \frac{1}{m} \sum_{i=1}^{m} (w_1 x^{(i)} - y^{(i)}) x^{(i)}$$

(0.5*1 - 1)*1 = -0.5

(1.5*1 - 1)*1 = +0.5

$$w_1 \leftarrow w_1 \boxed{- \gamma \frac{\partial J_{MSE}(w_1)}{\partial w_1}}$$

+ve constant

Outline

- Linear Regression
- **Gradient Descent**
  - Gradient Descent Algorithm
  - Linear Regression with Gradient Descent
  - Variants of Gradient Descent
- Linear Regression: Challenges and Solutions
  - Linear Regression with Many Attributes
  - Dealing with Features of Different Scales
  - Dealing with Non-Linear Relationship
- Normal Equation

Gradient Descent

Remember Hill-climbing?

- Start at some w
- Pick a nearby point that reduces $J(w)$

$$w_j \leftarrow w_j - \gamma \frac{\partial J(w_0, w_1, \dots)}{\partial w_j}$$

- Repeat until minimum is reached

Learning Rate

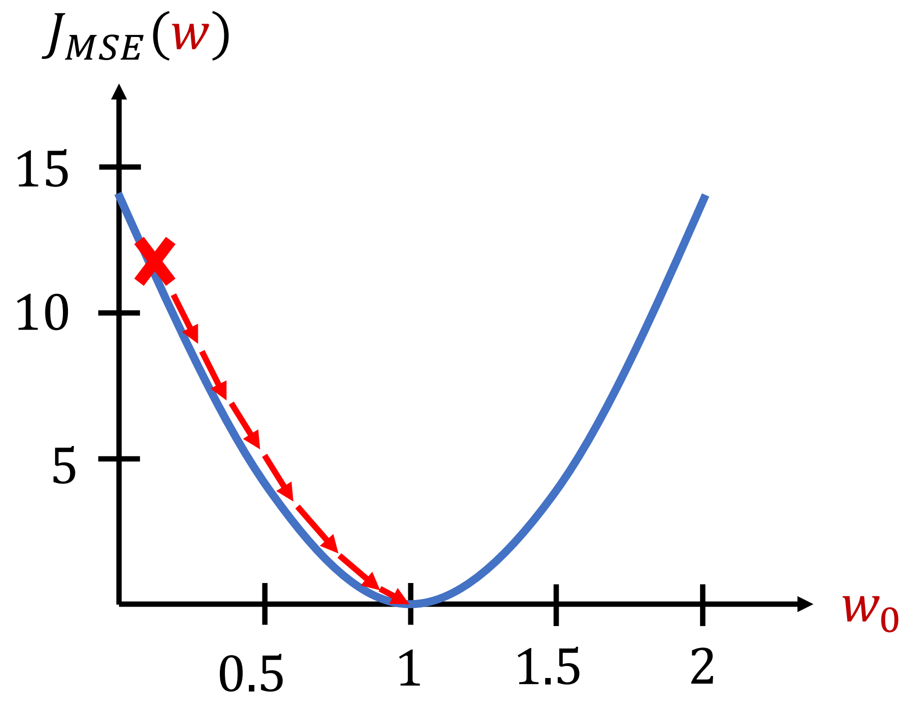
Gradient Descent: 1 Parameter

- Start at some w_0
- Pick a nearby point that reduces $J(w_0)$

$$w_0 \leftarrow w_0 - \gamma \boxed{\frac{\partial J(w_0)}{\partial w_0}}$$

- Repeat until minimum is reached

Learning Rate



As it gets closer to a minimum,
- The **gradient** becomes smaller
- The **steps** becomes smaller

Can we use mean absolute error (MAE) for our J?

MAE is <u>not</u> **fully differentiable** (its derivative is undefined at 0)

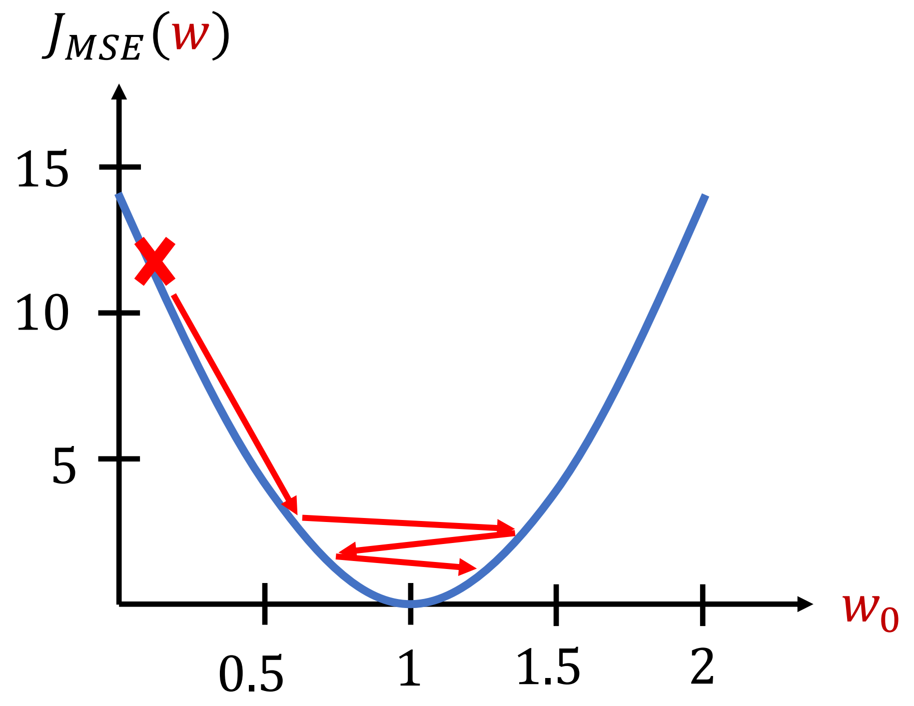
Gradient Descent: 1 Parameter

- Start at some w_0
- Pick a nearby point that reduces $J(w_0)$

$$w_0 \leftarrow w_0 - \gamma \frac{\partial J(w_0)}{\partial w_0}$$

- Repeat until minimum is reached

Learning Rate



γ too large

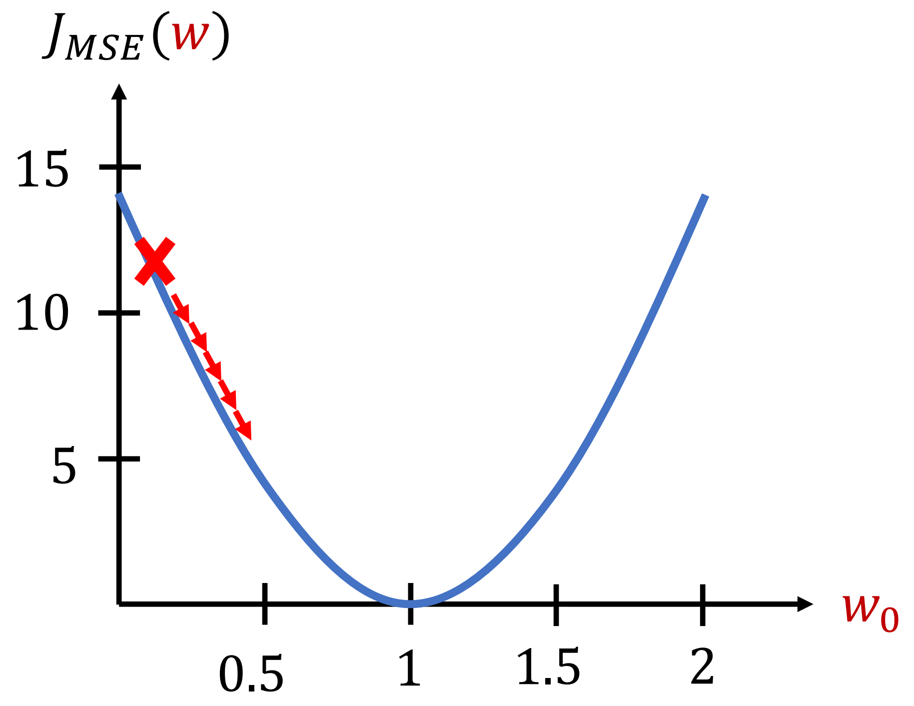
Gradient Descent: 1 Parameter

- Start at some w_0
- Pick a nearby point that reduces $J(w_0)$

$$w_0 \leftarrow w_0 - \gamma \frac{\partial J(w_0)}{\partial w_0}$$

Learning Rate

- Repeat until minimum is reached



γ too small

Gradient Descent: 2 Parameters

- Start at some $w = (w_0, w_1)$
- Pick a nearby point that reduces $J(w)$

$$w_j \leftarrow w_j - \gamma \frac{\partial J(w_0, w_1)}{\partial w_j}$$

- Repeat until minimum is reached

Learning Rate

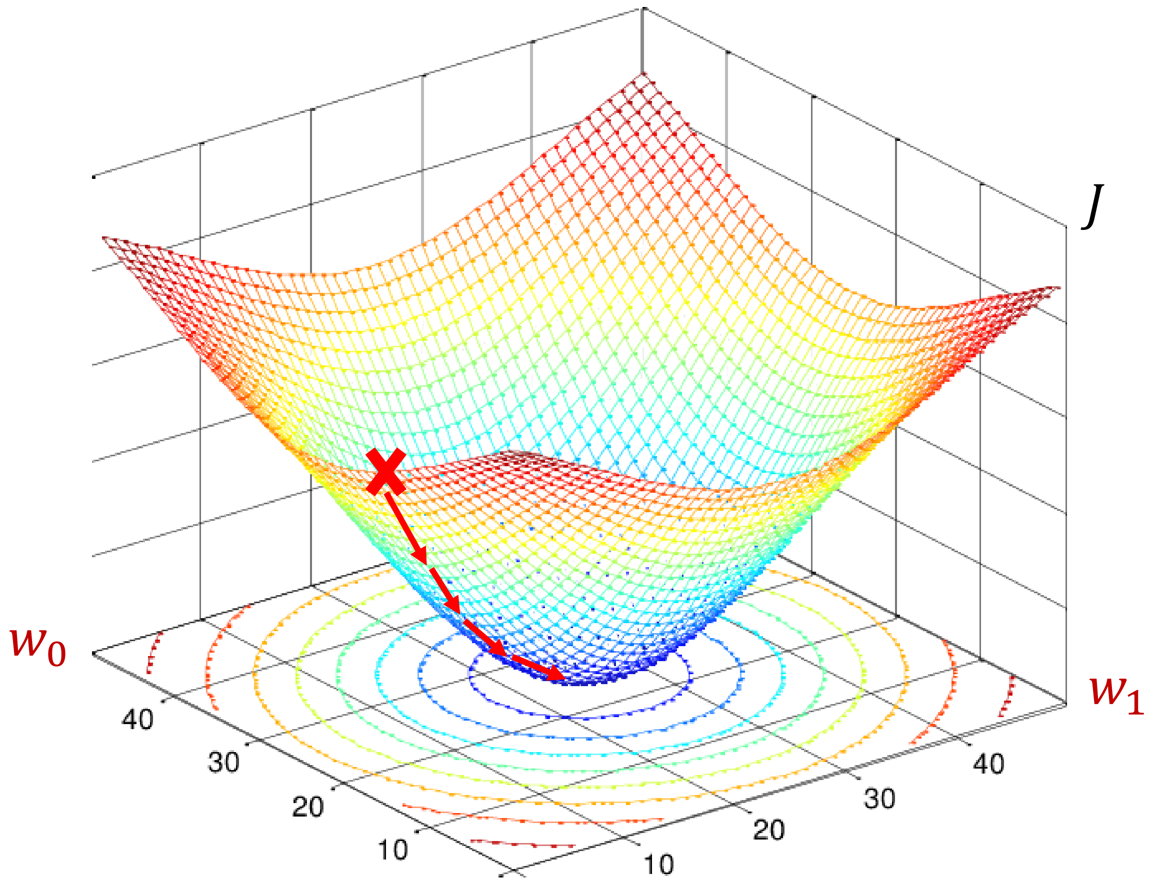


Image Credit: https://www.researchgate.net/figure/The-2D-Caussian-function-of-Example-2-Example-1-Consider-the-following-strongly-convex_fig2_230787652

Gradient Descent

- Start at some w
- Pick a nearby point that reduces $J(w)$

$$w_j \leftarrow w_j - \gamma \frac{\partial J(w_0, w_1, \dots)}{\partial w_j}$$

Learning Rate

- Repeat until minimum is reached

Common Mistakes

w_0 changed!

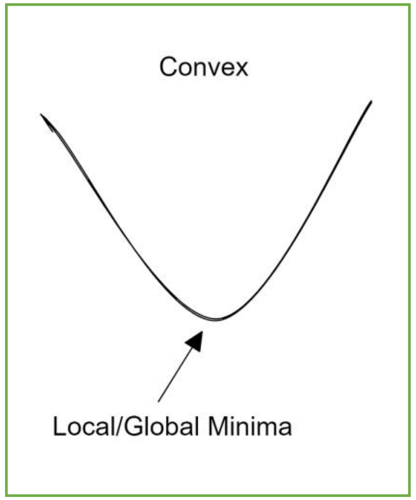
$$w_0 = w_0 + \gamma \frac{\partial J(w_0, w_1)}{\partial w_0}$$
$$w_1 = w_1 + \gamma \frac{\partial J(w_0, w_1)}{\partial w_1}$$

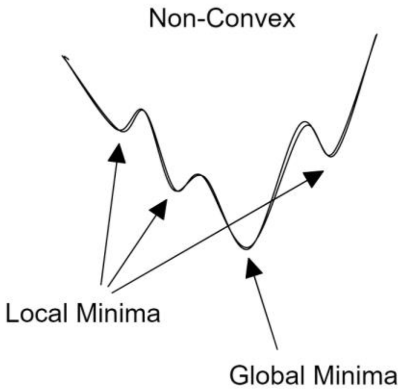
$$a = \frac{\partial J(w_0, w_1)}{\partial w_0}$$
$$b = \frac{\partial J(w_0, w_1)}{\partial w_1}$$
$$w_0 = w_0 + \gamma a$$
$$w_1 = w_1 + \gamma b$$

Linear Regression with Gradient Descent

Hypothesis:

$h_w(x) = w_0 + w_1 x$





Loss Function:

$$J_{MSE}(w_0, w_1) = \frac{1}{2m}\sum_{i=1}^{m}(w_0 + w_1 x^{(i)} - y^{(i)})^2$$

$$\frac{\partial J_{MSE}(w_0, w_1)}{\partial w_j} = \frac{\partial}{\partial w_j}\frac{1}{2m}\sum_{i=1}^{m}(w_0 + w_1 x^{(i)} - y^{(i)})^2$$

$$\frac{\partial J_{MSE}(w_0, w_1)}{\partial w_0} = \frac{1}{m}\sum_{i=1}^{m}(w_0 + w_1 x^{(i)} - y^{(i)})$$

$$\frac{\partial J_{MSE}(w_0, w_1)}{\partial w_1} = \frac{1}{m}\sum_{i=1}^{m}(w_0 + w_1 x^{(i)} - y^{(i)}).x^{(i)}$$

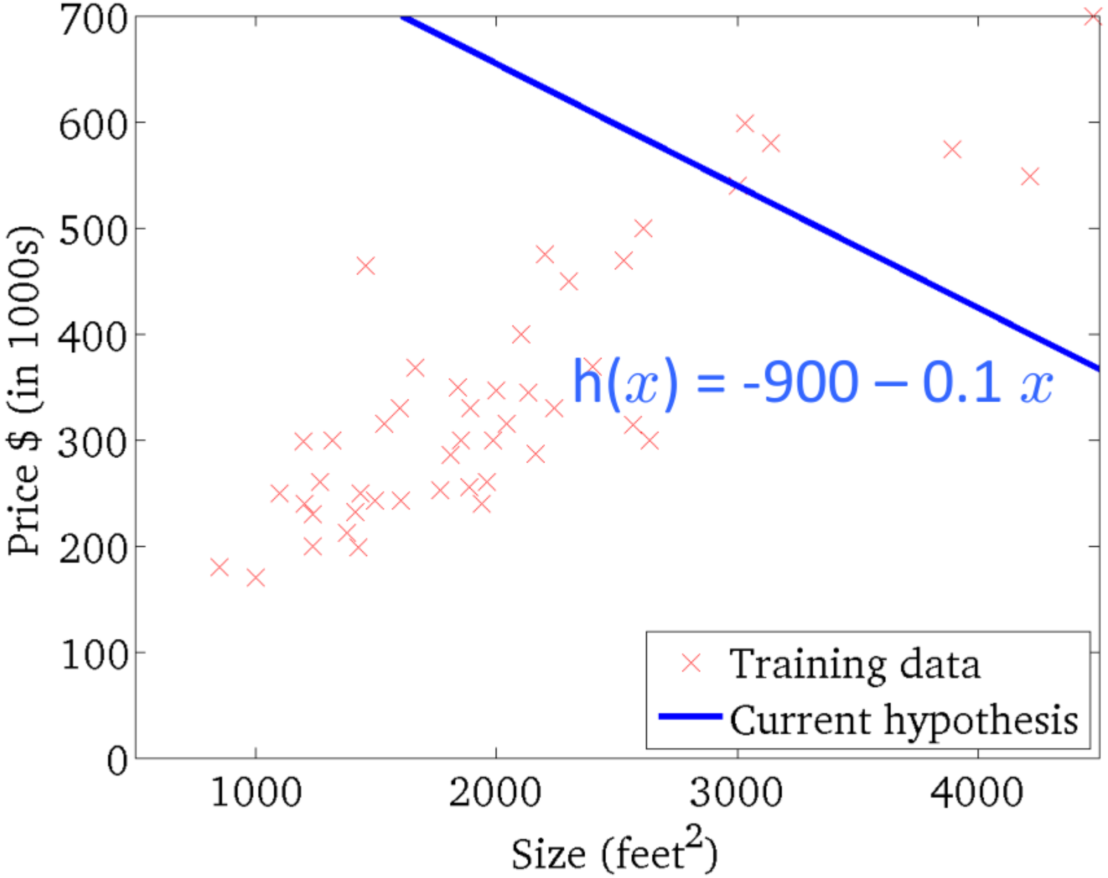
Theorem: MSE loss function is convex for linear regression.

- One minimum, global minimum

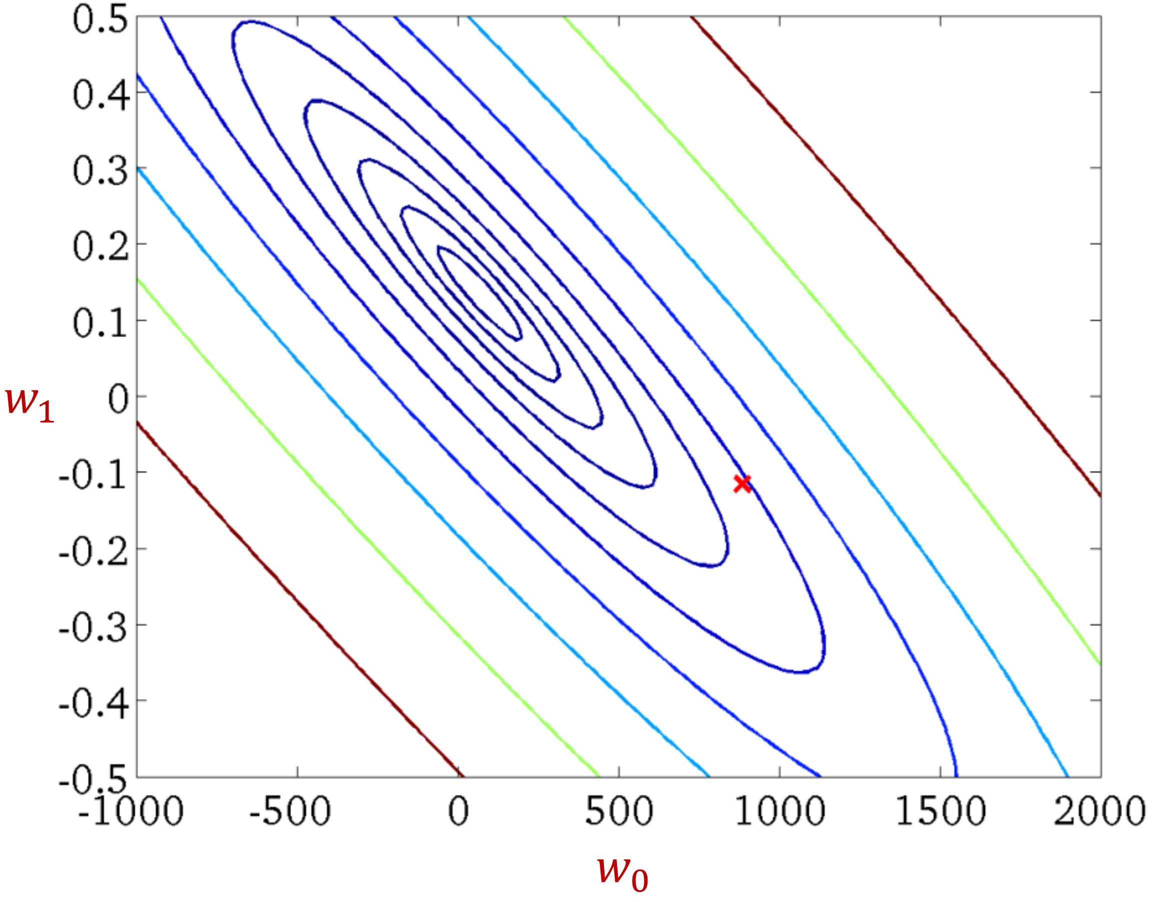
Image Credit: https://python.plainenglish.io/logistic-regression-in-machine-learning-from-scratch-872b1fedd05b

Linear Regression with Gradient Descent

$h_w(x)$



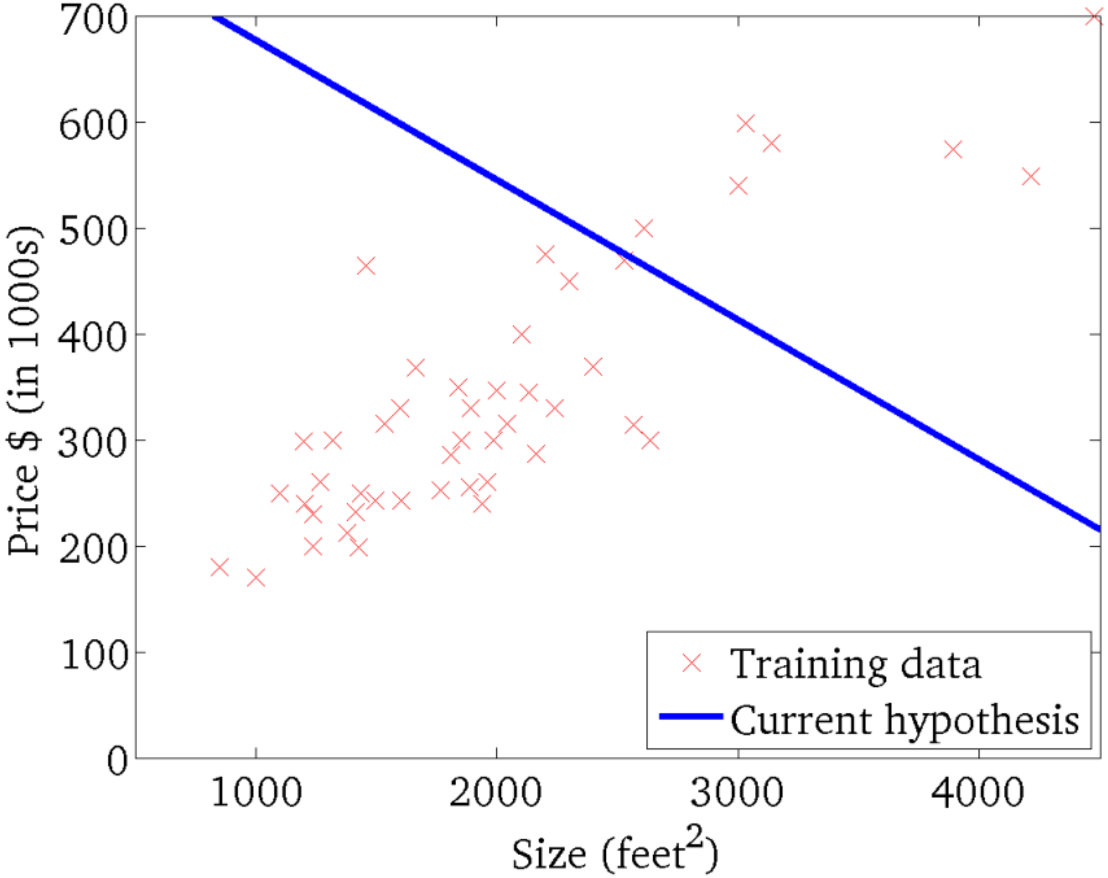
$J_{MSE}(w_0, w_1)$



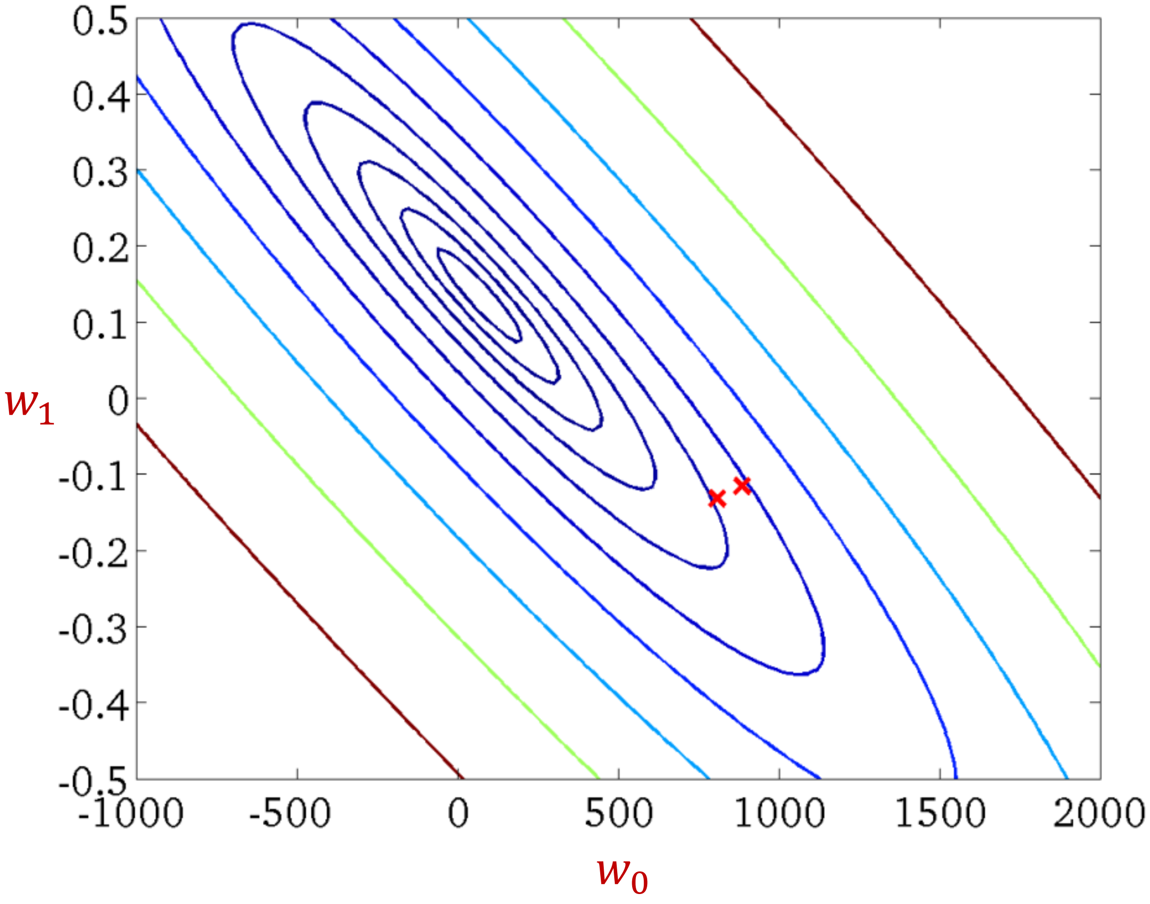
Credit: Andrew Ng

Linear Regression with Gradient Descent

$h_w(x)$



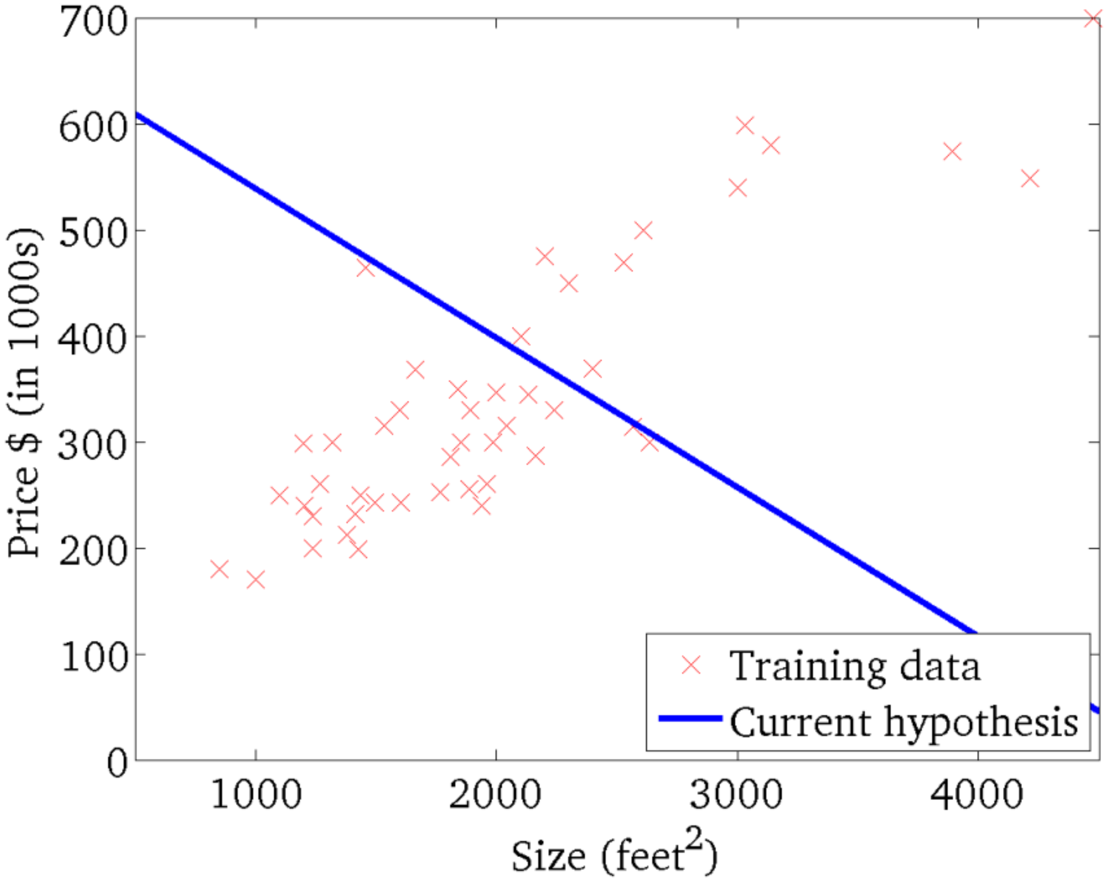
$J_{MSE}(w_0, w_1)$



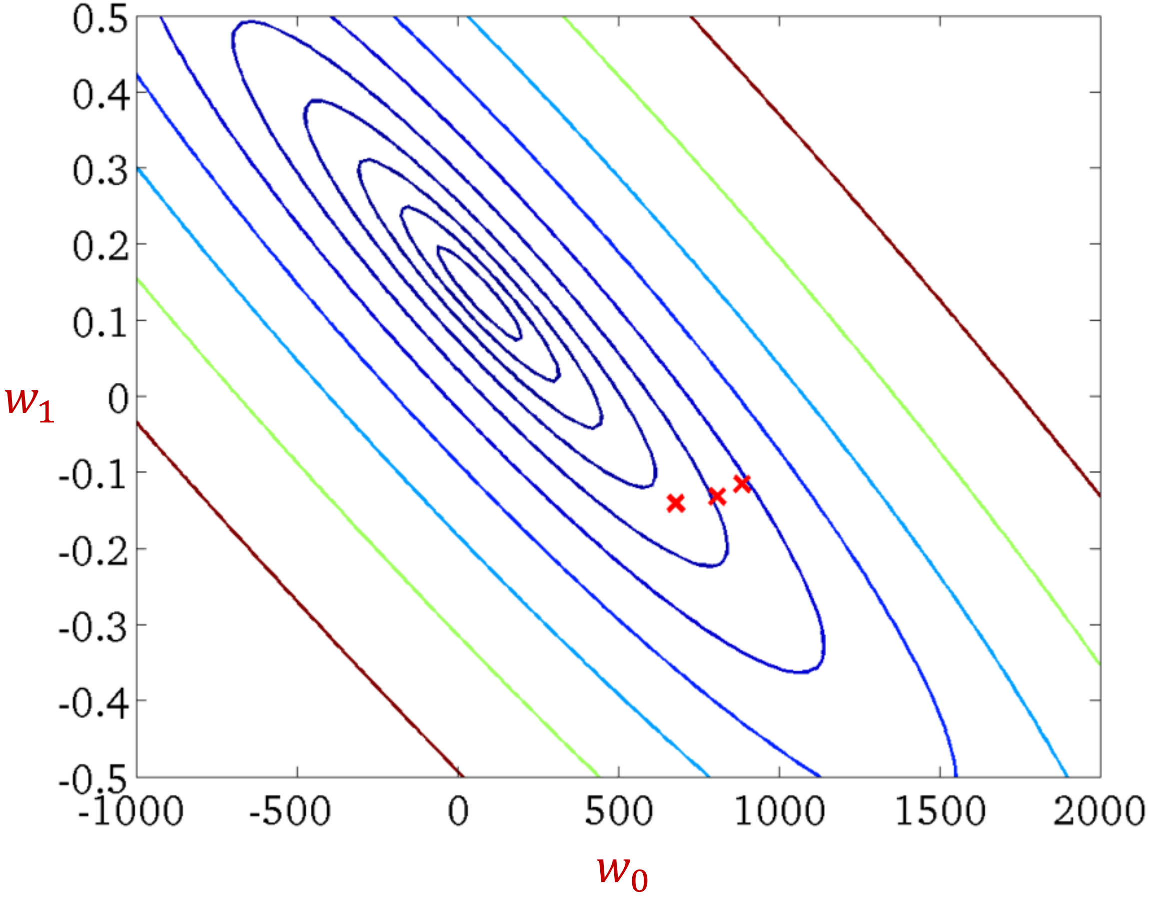
Credit: Andrew Ng

Linear Regression with Gradient Descent

$h_w(x)$



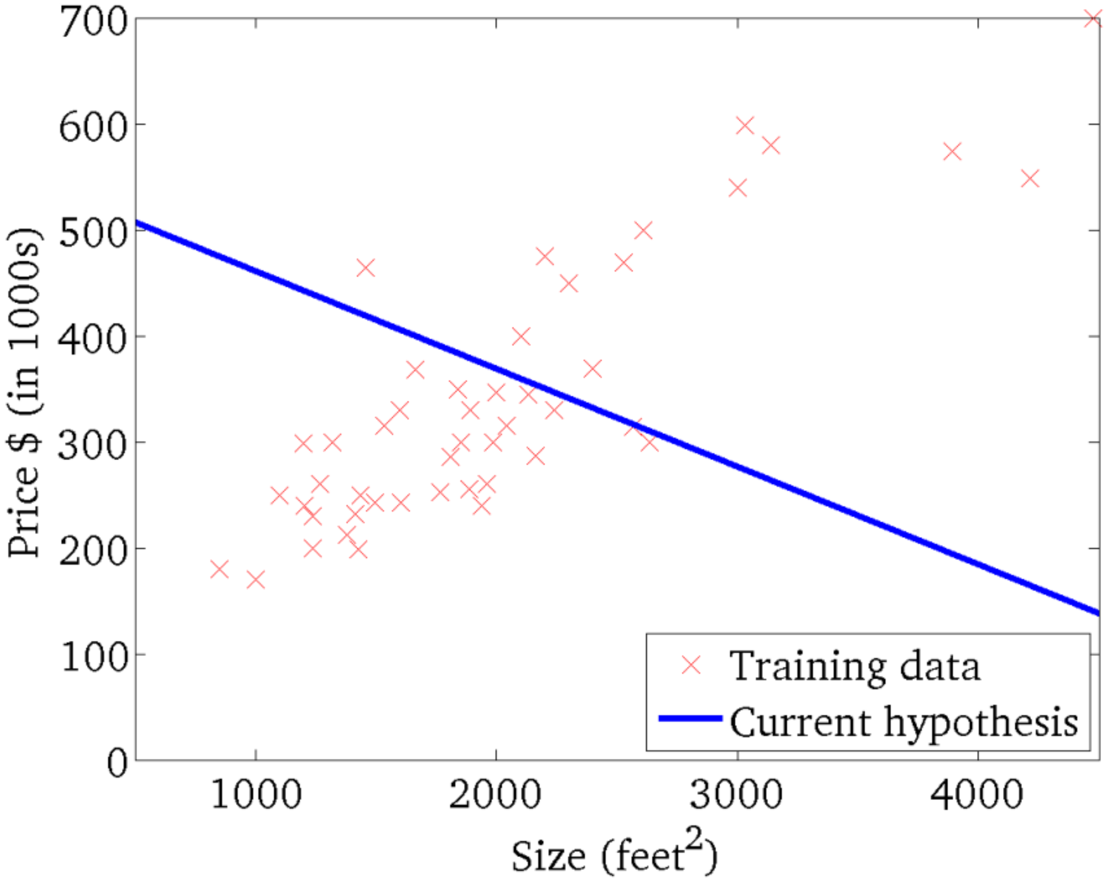
$J_{MSE}(w_0, w_1)$



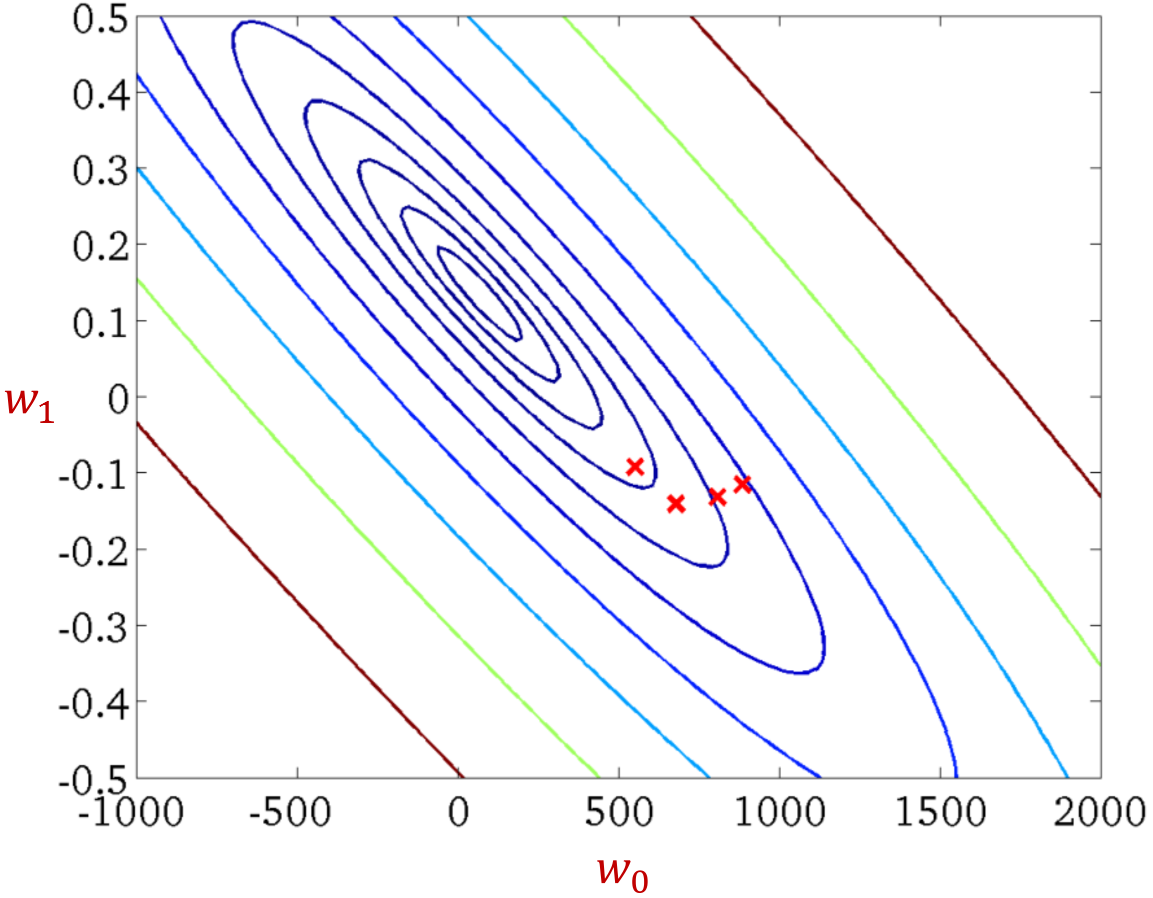
Credit: Andrew Ng

Linear Regression with Gradient Descent

$h_w(x)$



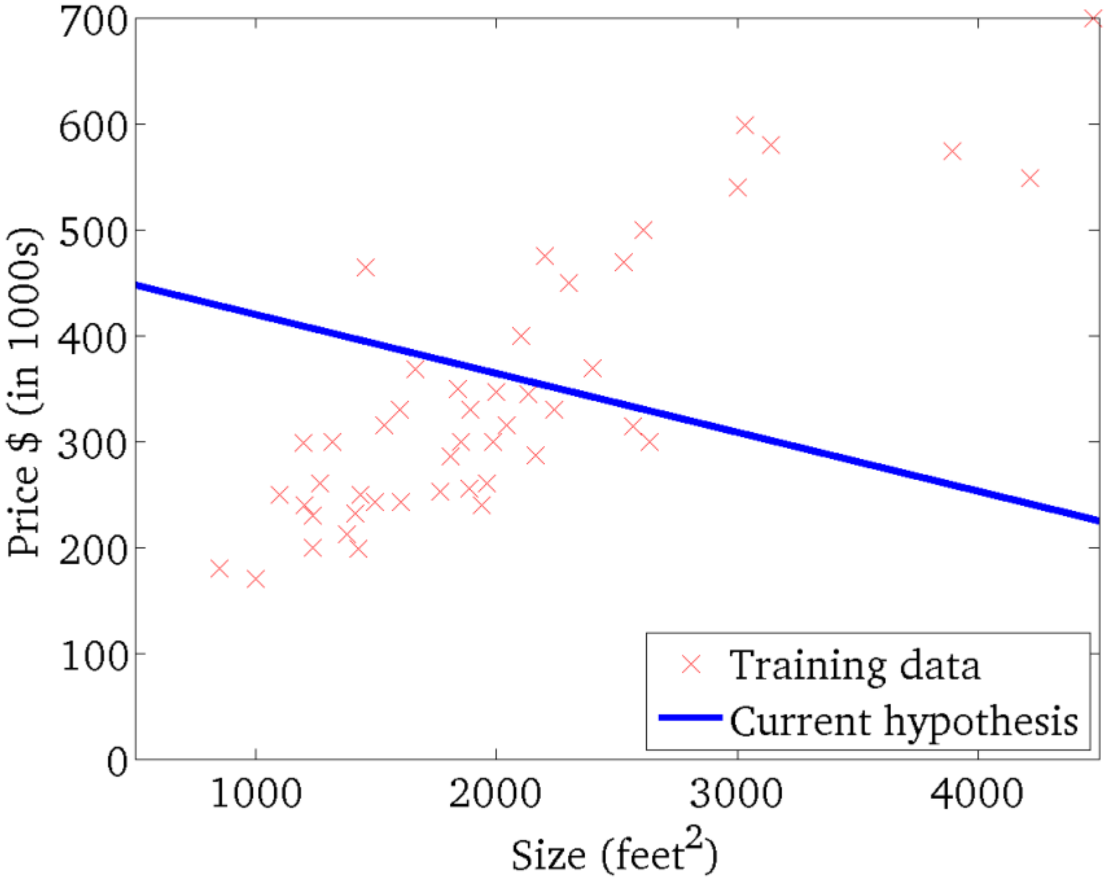
$J_{MSE}(w_0, w_1)$



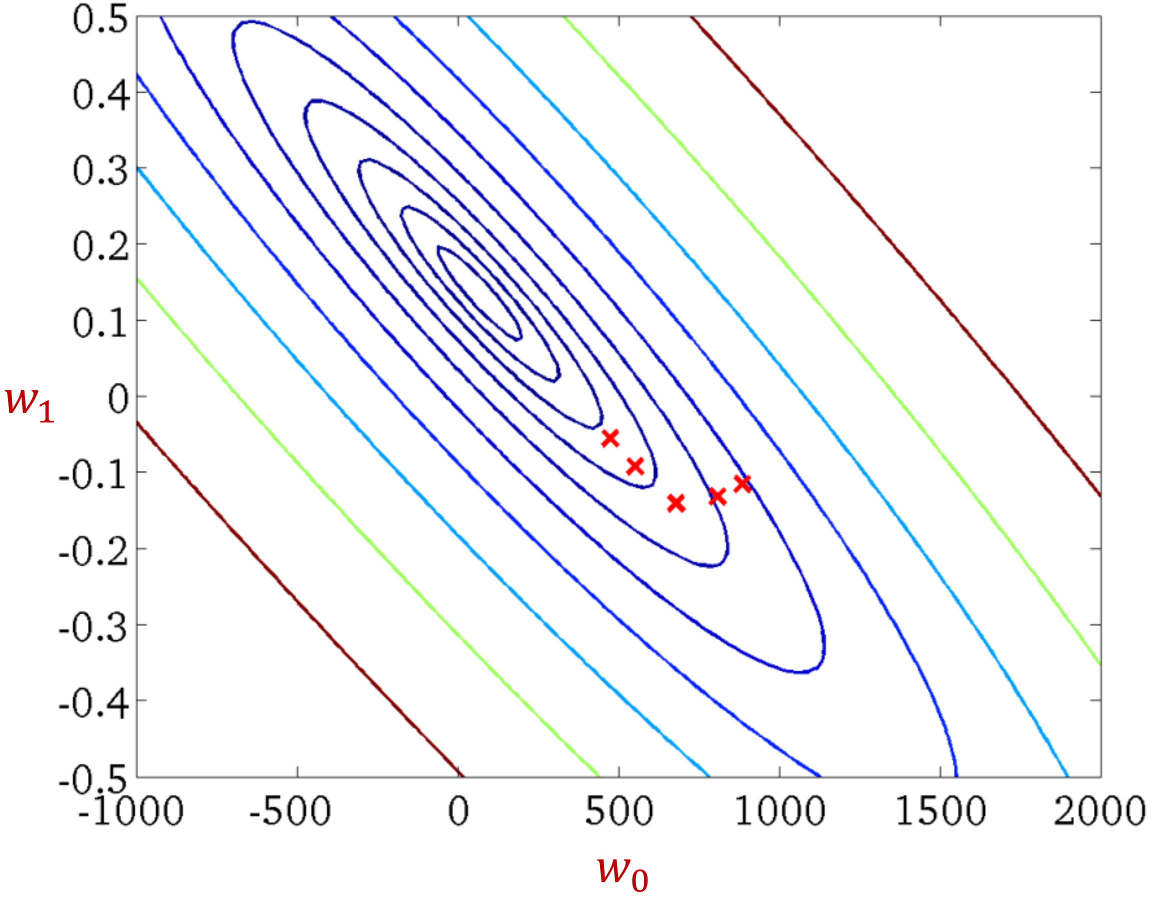
Credit: Andrew Ng

Linear Regression with Gradient Descent

$h_w(x)$



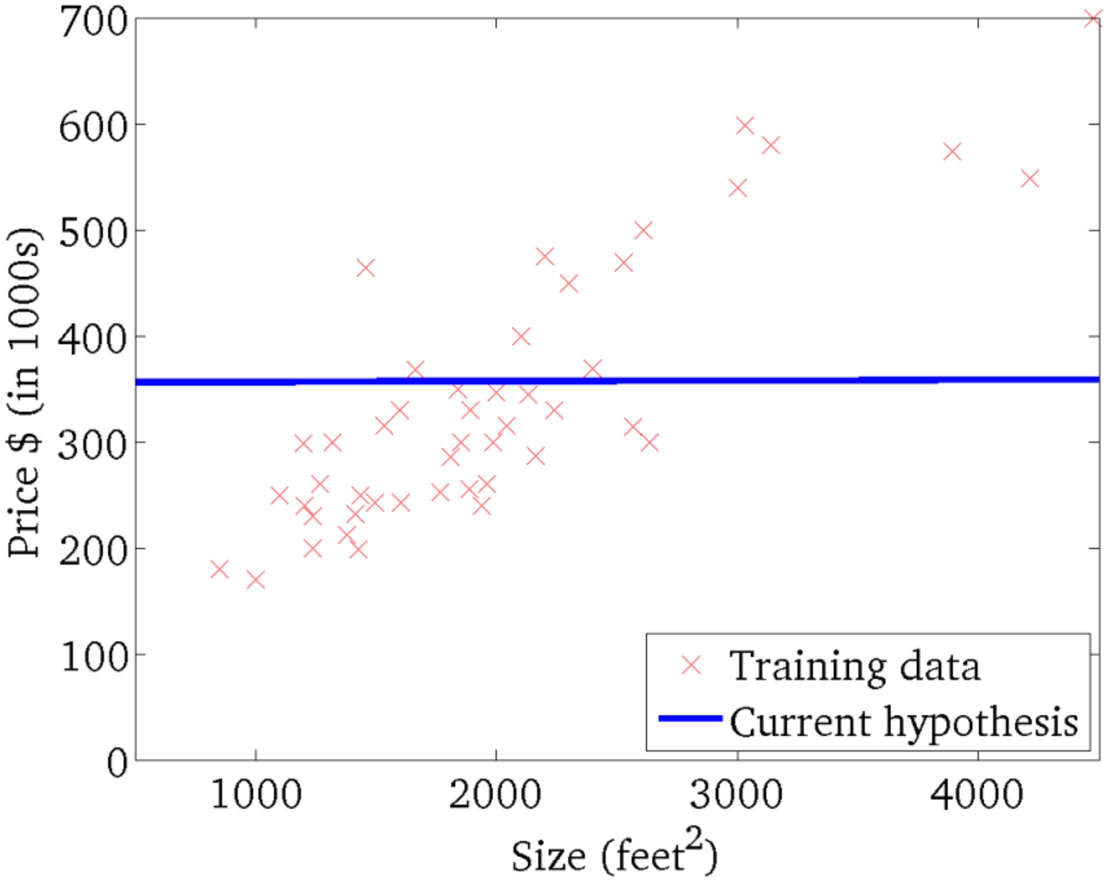
$J_{MSE}(w_0, w_1)$



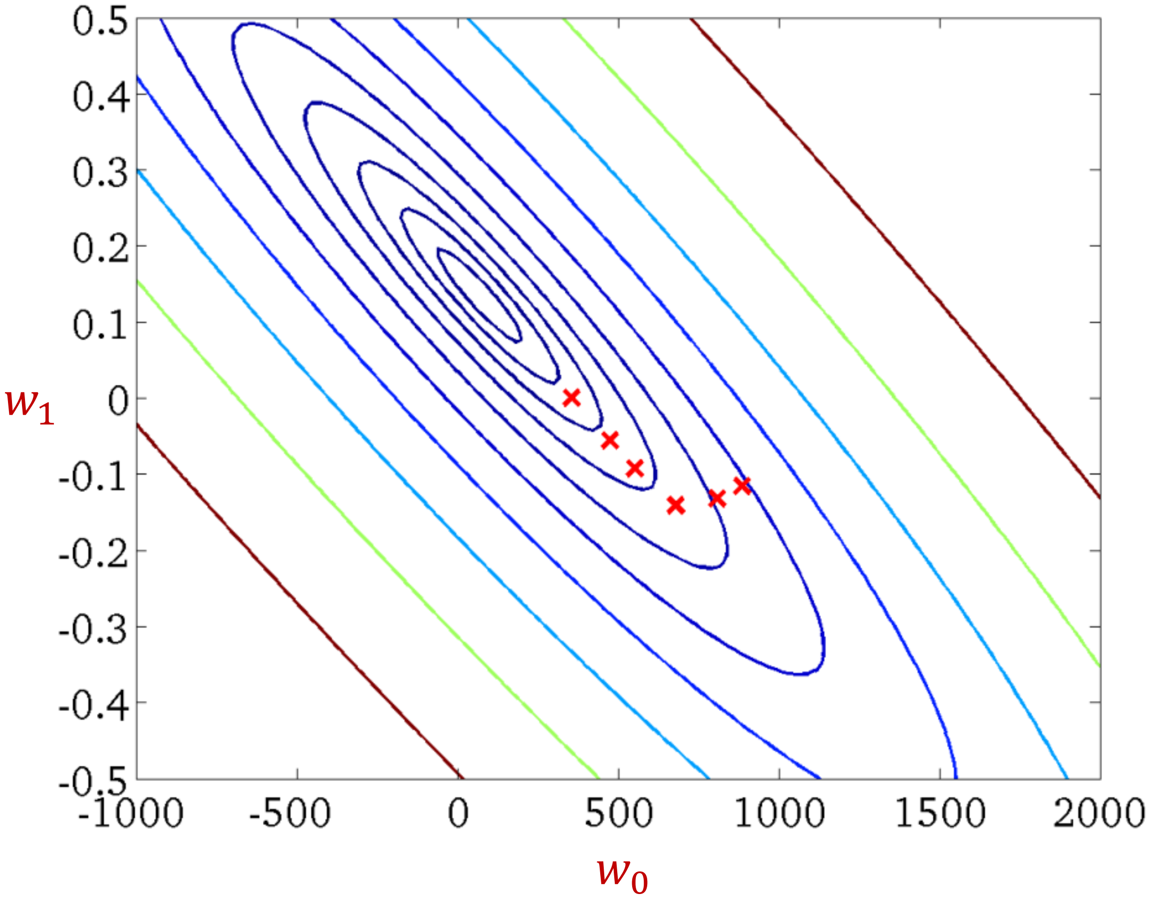
Credit: Andrew Ng

Linear Regression with Gradient Descent

$h_w(x)$

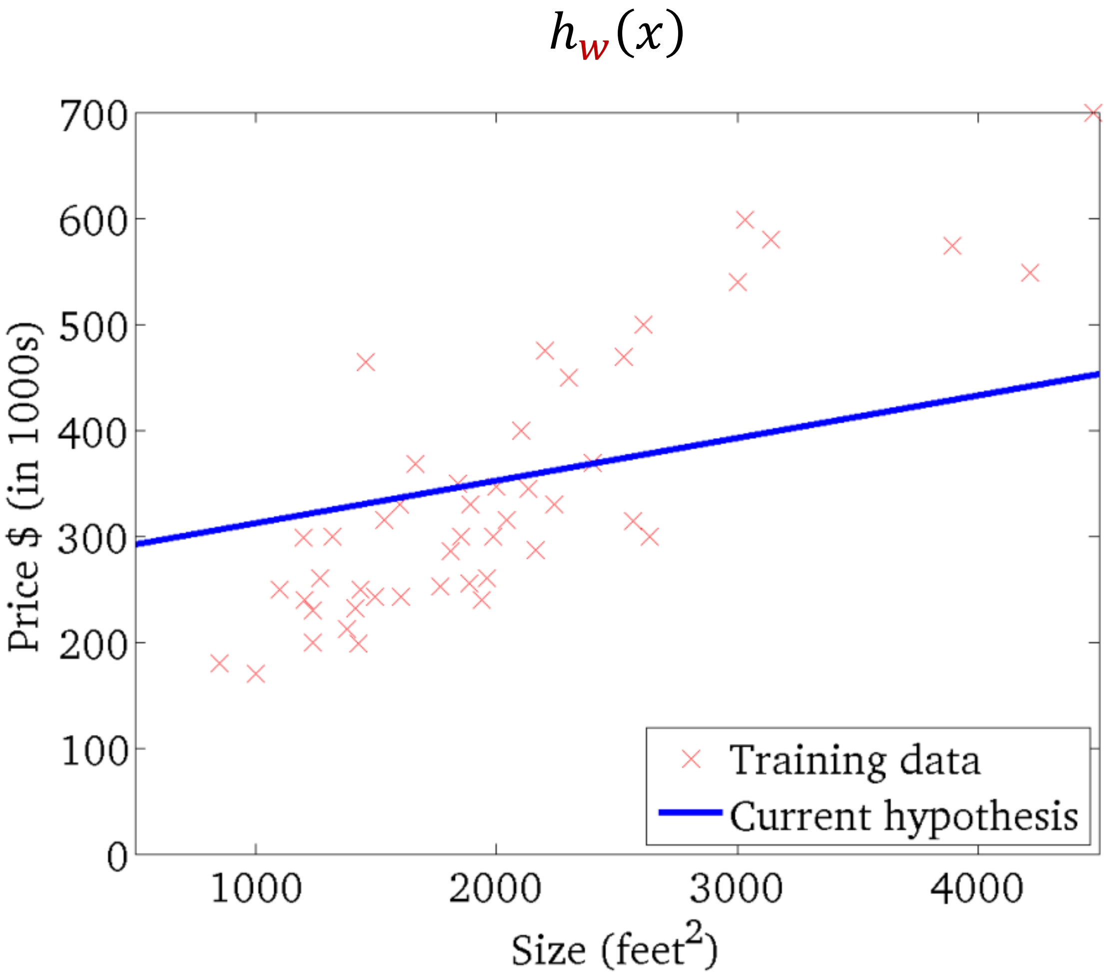


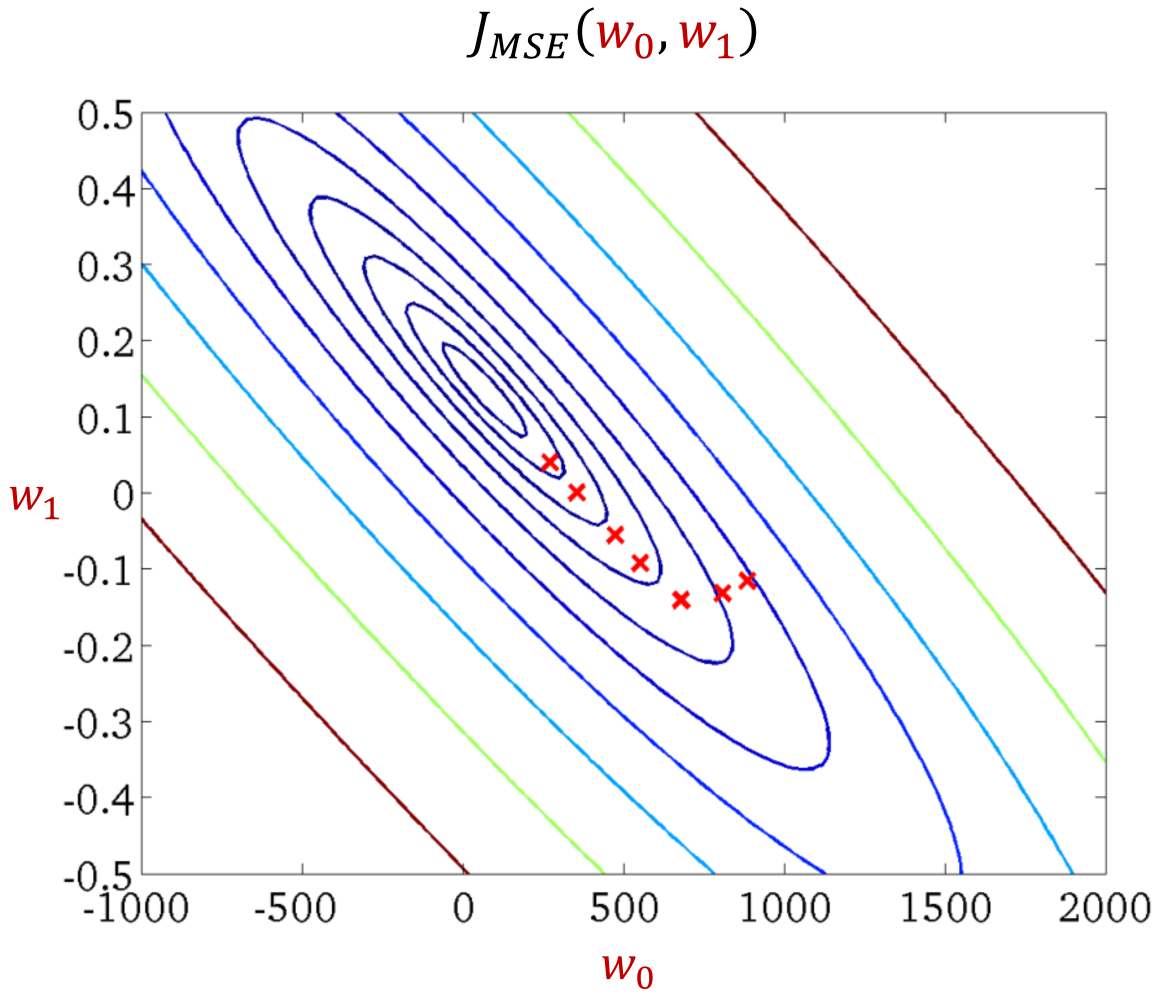
$J_{MSE}(w_0, w_1)$



Credit: Andrew Ng

Linear Regression with Gradient Descent

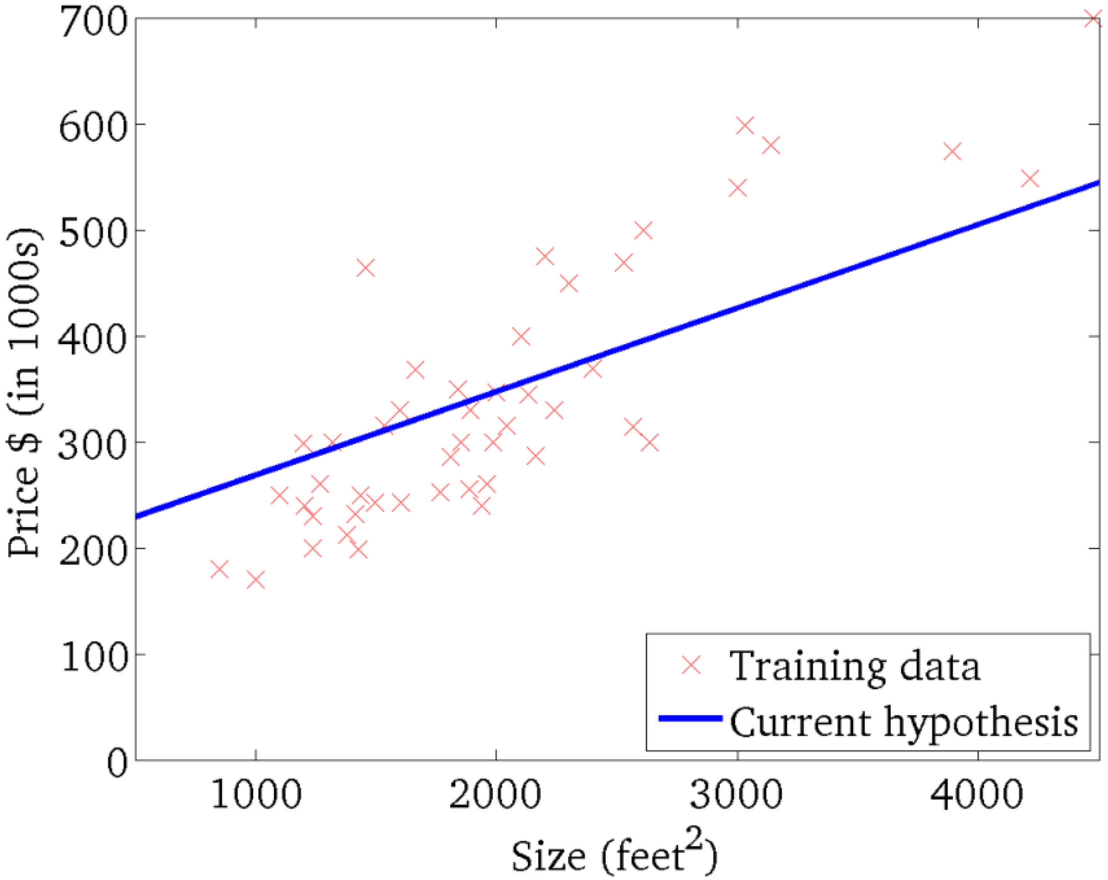




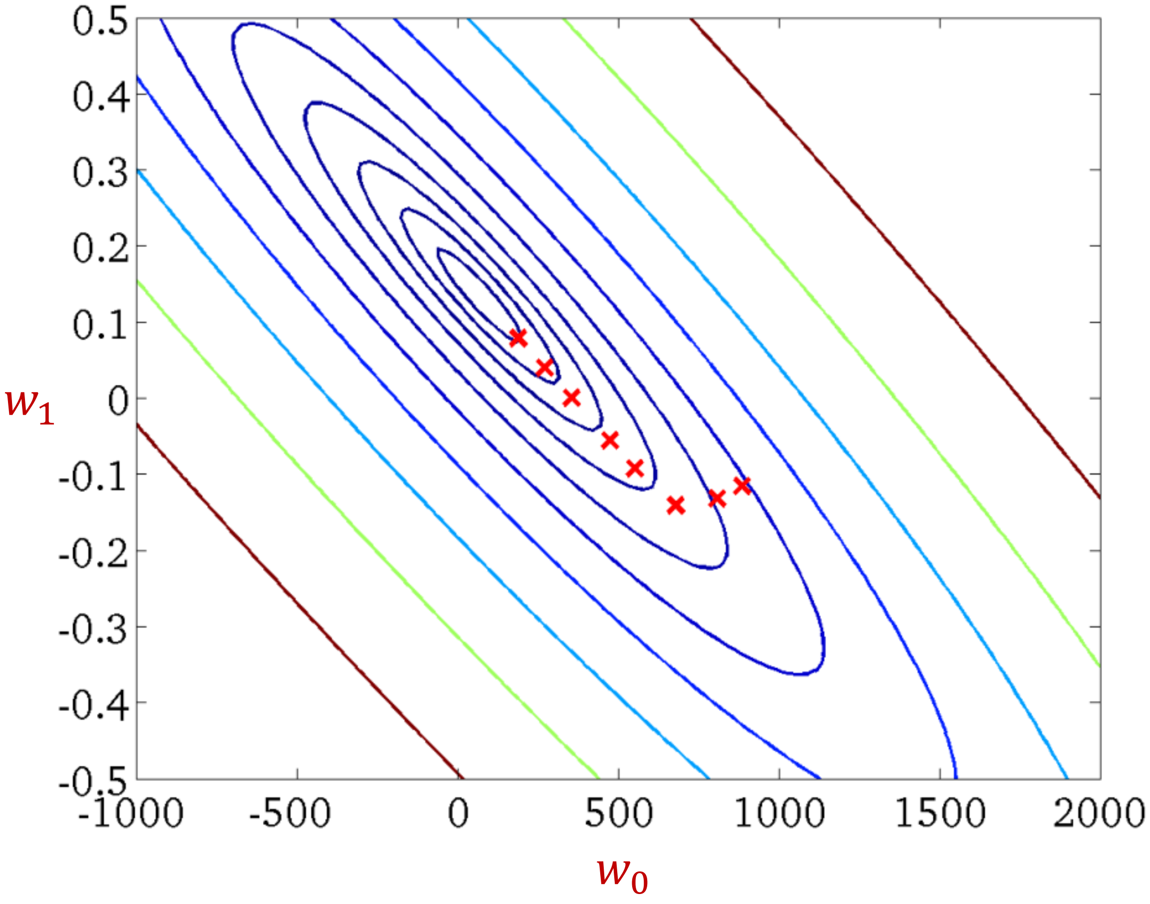
Credit: Andrew Ng

Linear Regression with Gradient Descent

$h_w(x)$



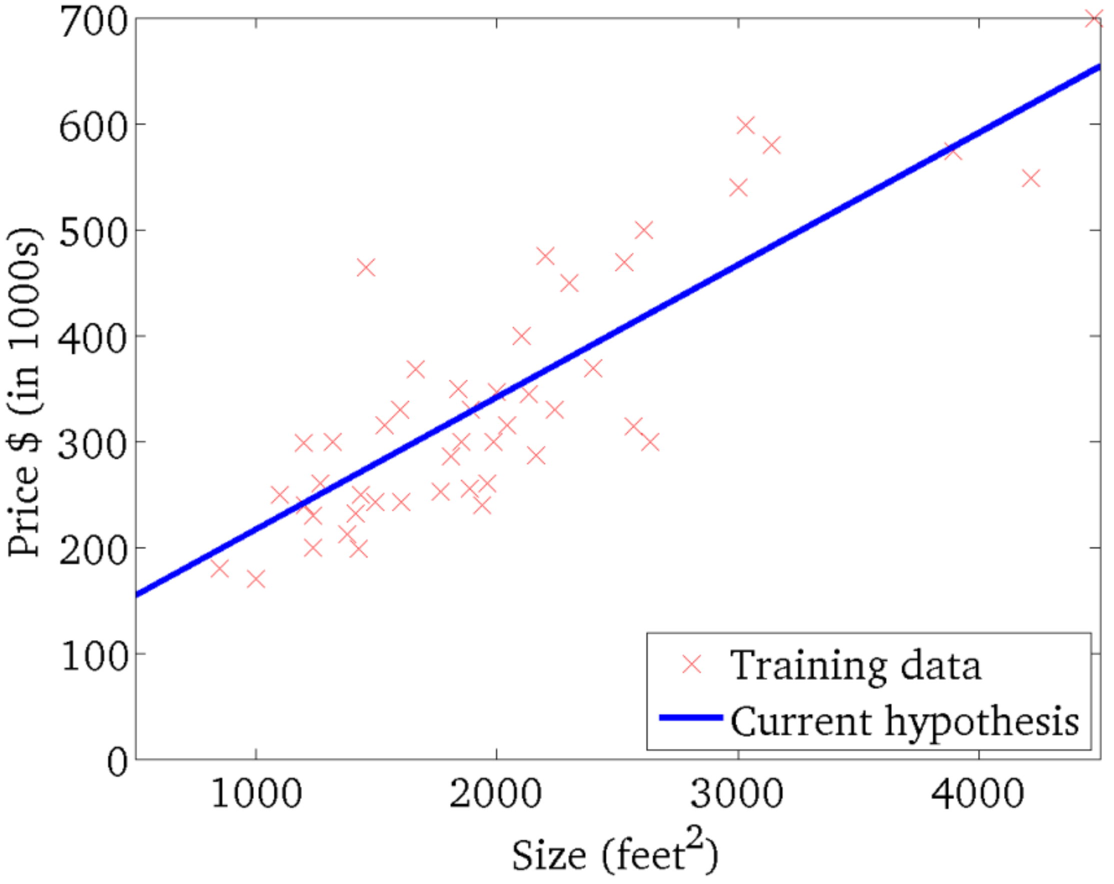
$J_{MSE}(w_0, w_1)$



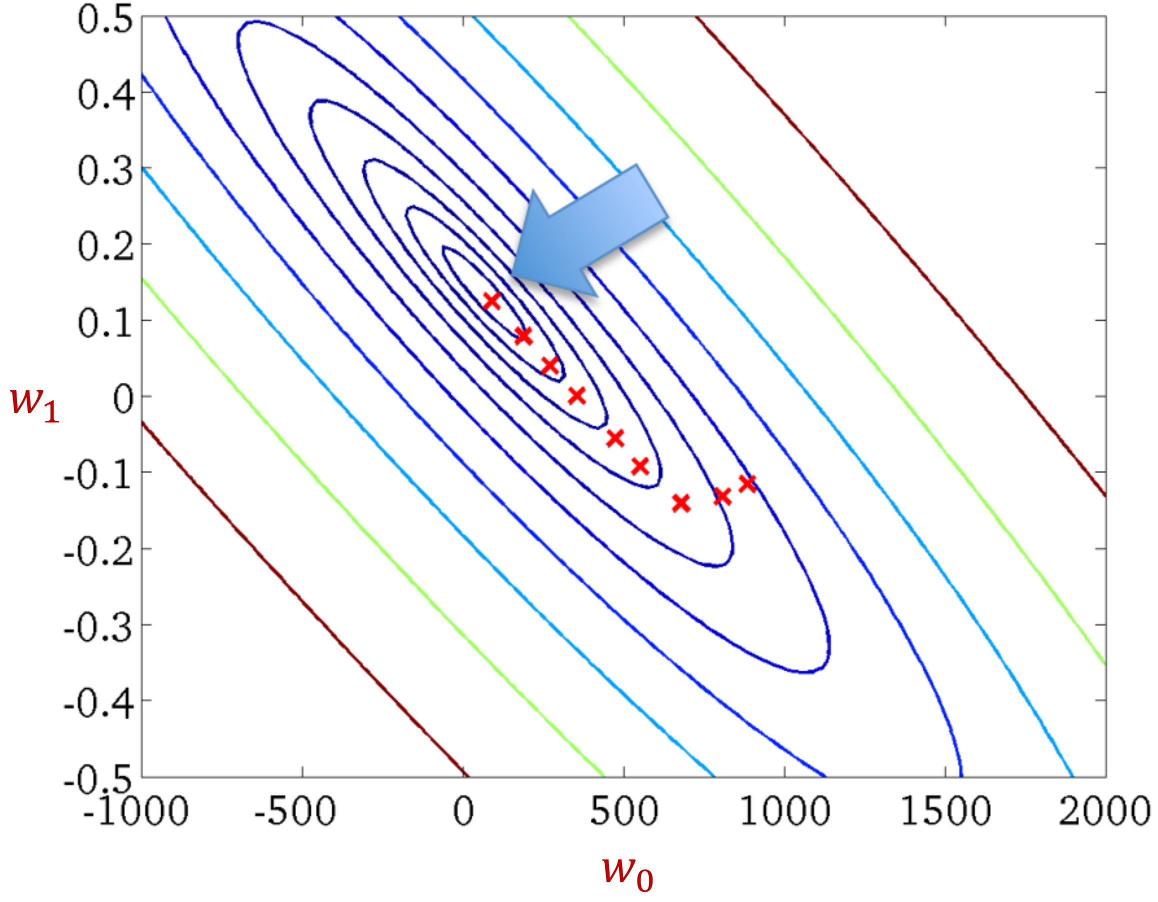
Credit: Andrew Ng

Linear Regression with Gradient Descent

$h_w(x)$

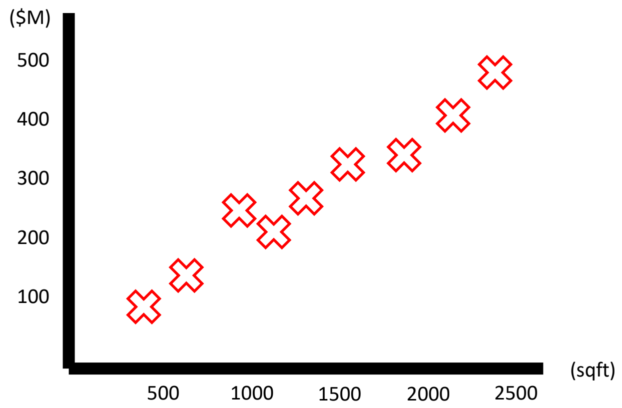


$J_{MSE}(w_0, w_1)$



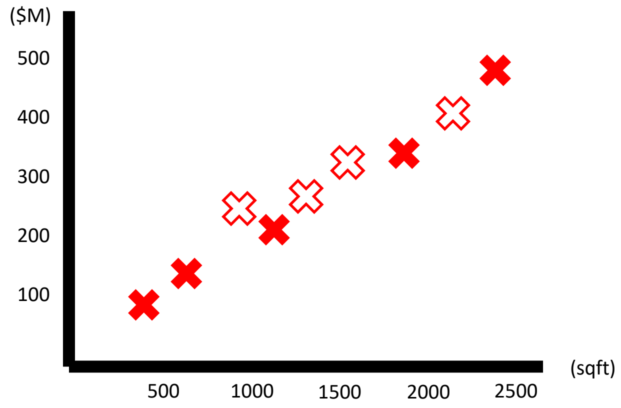
Credit: Andrew Ng

Variants of Gradient Descent



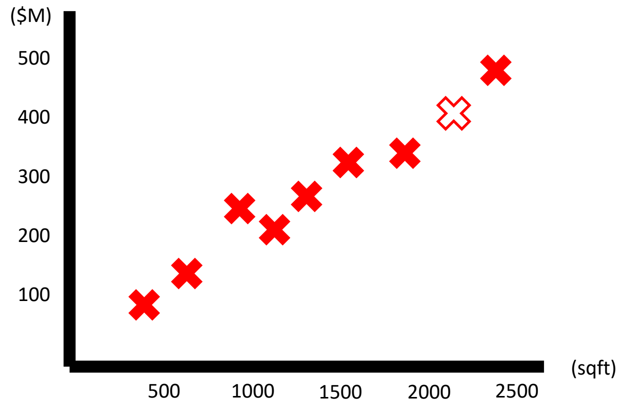
(Batch) Gradient Descent

- Consider all training examples



Mini-batch Gradient Descent

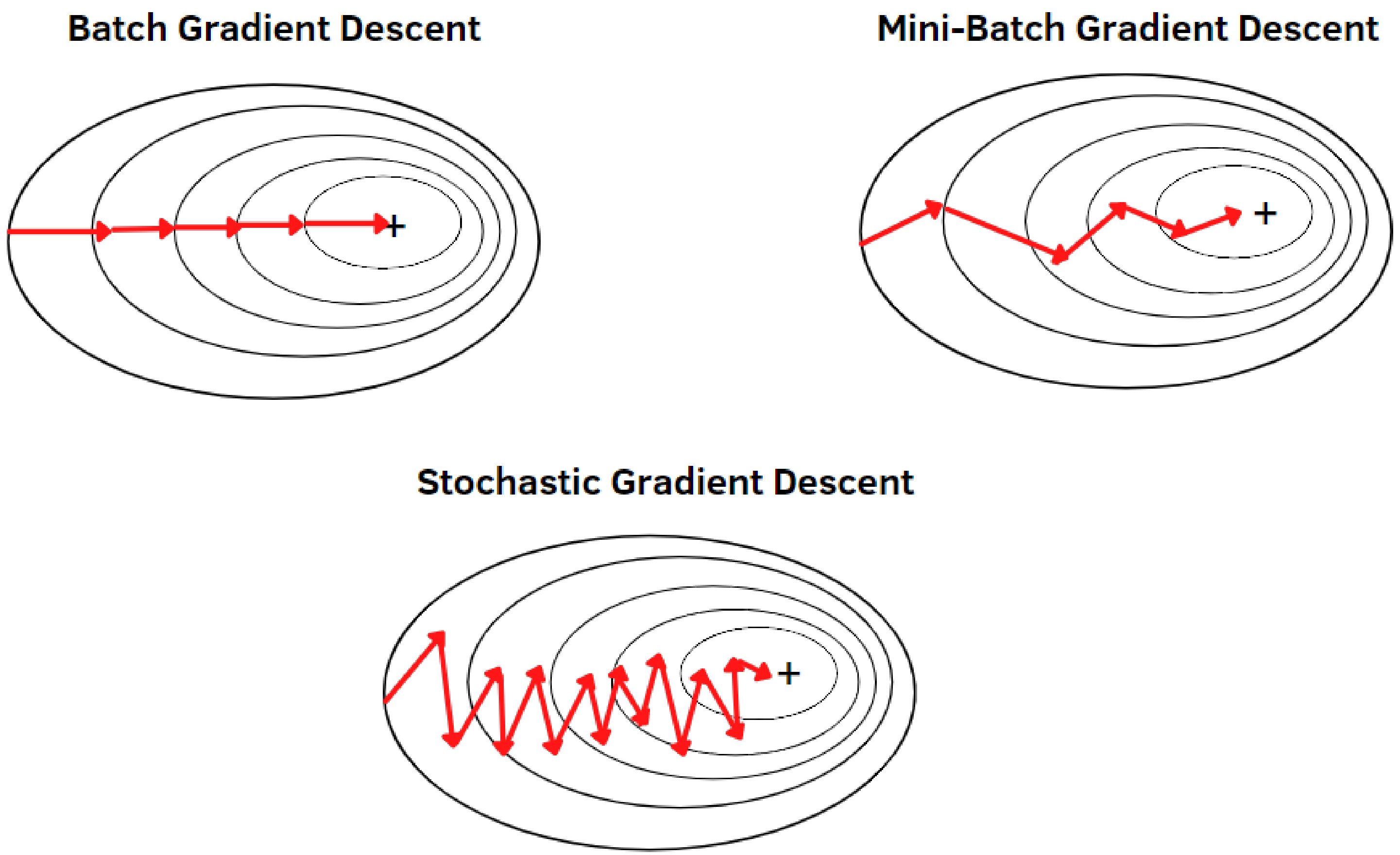
- Consider a subset of training examples at a time
- Cheaper (Faster) / iteration
- Randomness, may escape local minima



Stochastic Gradient Descent (SGD)

- Select one random data point at a time
- Cheapest (Fastest) / iteration
- More randomness, may escape local minima

Variants of Gradient Descent



Credit: analyticsvidhya.com

Outline

- Linear Regression
- Gradient Descent
  - Gradient Descent Algorithm
  - Linear Regression with Gradient Descent
  - Variants of Gradient Descent
- **Linear Regression: Challenges and Solutions**
  - Linear Regression with Many Attributes
  - Dealing with Features of Different Scales
  - Dealing with Non-Linear Relationship
- Normal Equation

Linear Regression with Many Attributes

Bias (x_0)	Year (x_1)	# bedrooms (x_2)	# bathrooms (x_3)	Size (m²) (x_4)	Price ($) ($y$)
1	2016	4	2	113	560,000
1	1998	3	2	102	739,000
1	1997	3	0	100	430,000
1	2014	3	2	84	698,000
1	2016	3	0	112	688,888
1	1979	2	2	68	390,000
1	1969	2	1	53	250,000
1	1986	3	2	122	788,000
1	1985	3	3	150	680,000
1	2009	3	2	90	828,000

HDB prices from SRX

Notation:

- n = number of features
- $x^{(i)}$ = input features of the i-th training example
- $x_j^{(i)}$ = value of feature j in i-th training example

Hypothesis:

$$h_w(x) = w_0 x_0 + w_1 x_1 + w_2 x_2 + w_3 x_3 + w_4 x_4$$

Hypothesis (for n features):

$$h_w(x) = \sum_{j=0}^{n} w_j x_j = w^T x$$

vector

Weight Update (for n features):

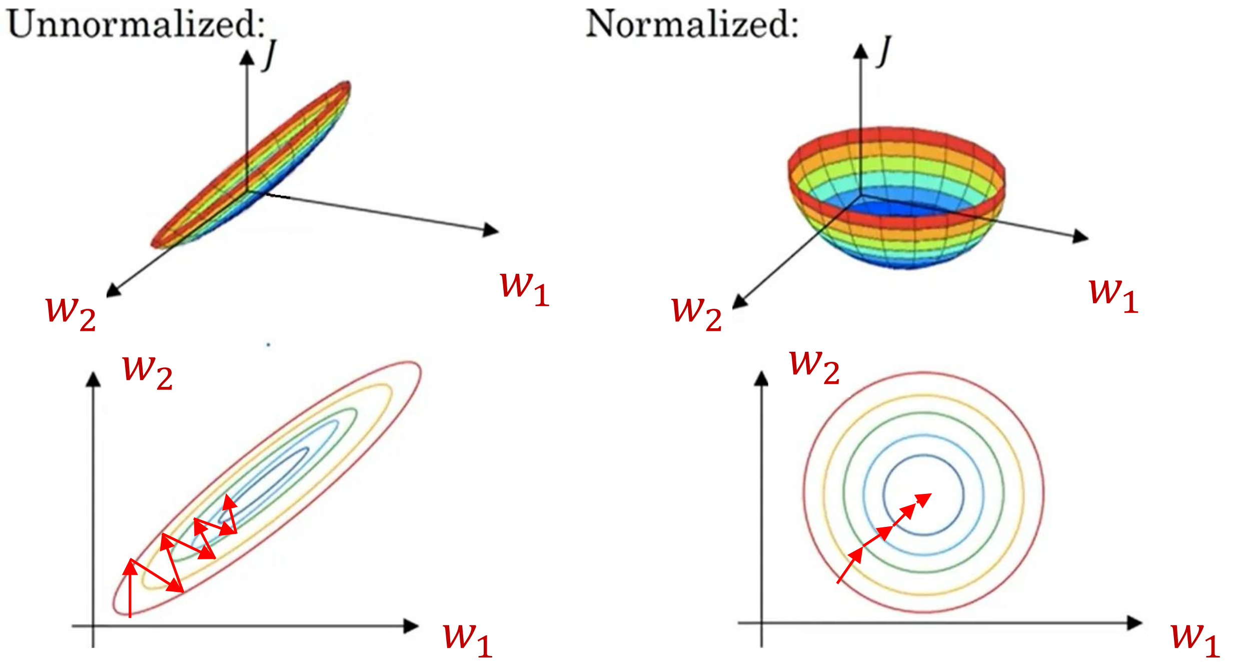
$$w_j \leftarrow w_j - \gamma \frac{\partial J_{MSE}(w_0, w_1, \ldots, w_n)}{\partial w_j}$$

$$w_j \leftarrow w_j - \gamma \frac{1}{m} \sum_{i=1}^{m} \left(h_w(x^{(i)}) - y^{(i)}\right). x_j^{(i)}$$

Dealing with Features of Different Scales

x_0	x_1	x_2	x_3	x_4	y
Bias	Year	# bedrooms	# bathrooms	Size (m^2)	Price ($)
1	2016	4	2	113	560,000
1	1998	3	2	102	739,000
1	1997	3	0	100	430,000
1	2014	3	2	84	698,000
1	2016	3	0	112	688,888
1	1979	2	2	68	390,000
1	1969	2	1	53	250,000
1	1986	3	2	122	788,000
1	1985	3	3	150	680,000
1	2009	3	2	90	828,000

HDB prices from SRX



Each attribute has different scale:

$$1969 \le x_1 \le 2016$$

$$2 \le x_2 \le 5$$

$$0 \le x_3 \le 2$$

$$84 \le x_4 \le 150$$

Other methods of standardization also exists:
Min-max scaling, robust scaling, etc

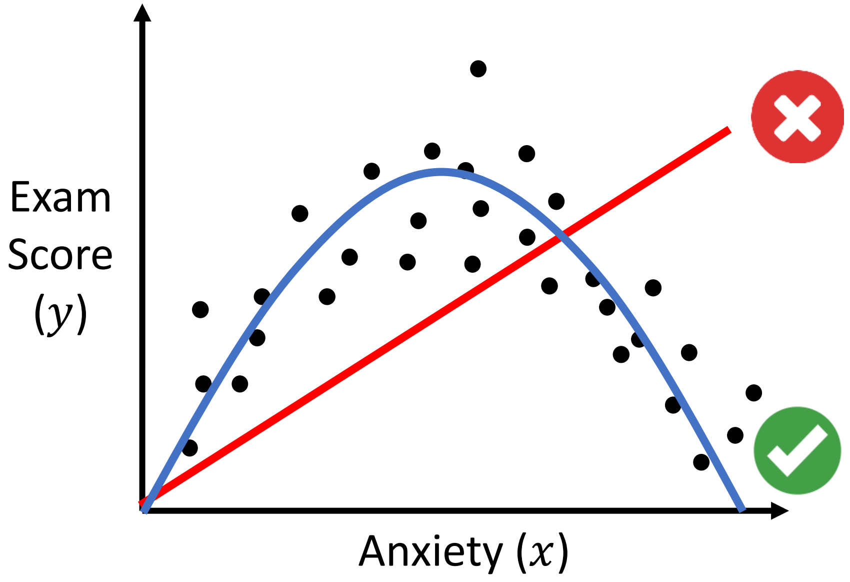
Mean normalization:

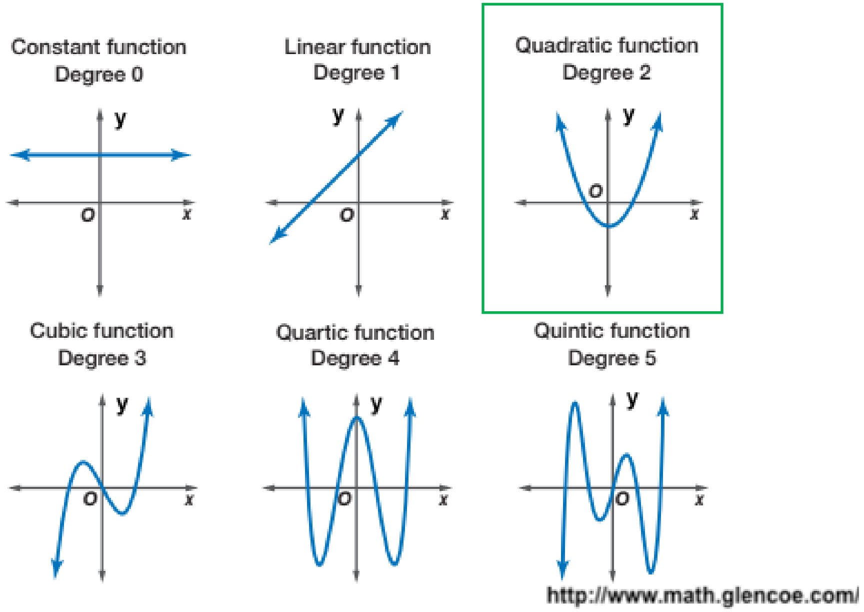
$$x_i \leftarrow \frac{x_i - \mu_i}{\sigma_i}$$

std dev

Image Credit: Andrew Ng

Dealing with Non-Linear Relationship





Which function?

$$h_w(x) = w_0 + w_1 x_1 + w_2 x_1^2$$

Polynomial Regression

Need to scale this!

Generally:

$$h_w(x) = w_0 + w_1 f_1 + w_2 f_2 + w_3 f_3 + \cdots + w_n f_n$$

Transformed features:

$e.g., f_1 = x_1, f_2 = x_1^2$

Outline

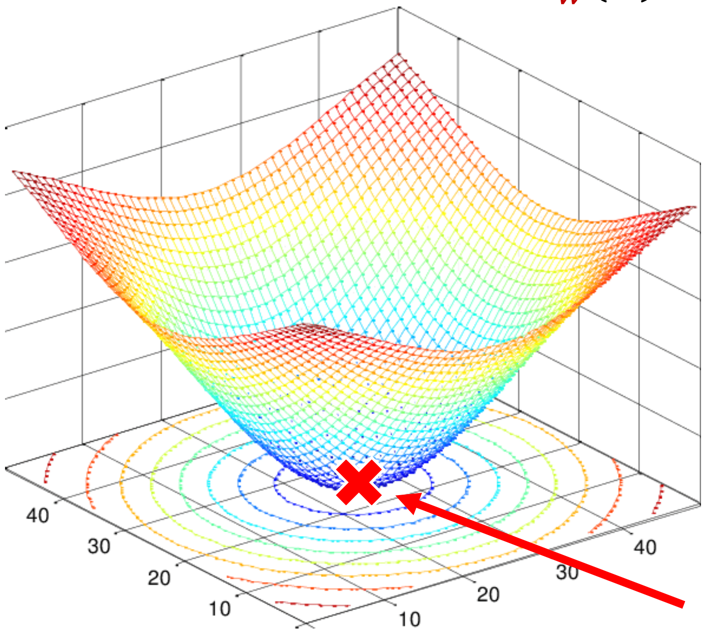
- Linear Regression
- Gradient Descent
  - Gradient Descent Algorithm
  - Linear Regression with Gradient Descent
  - Variants of Gradient Descent
- Linear Regression: Challenges and Solutions
  - Linear Regression with Many Attributes
  - Dealing with Features of Different Scales
  - Dealing with Non-Linear Relationship
- **Normal Equation**

Normal Equation

$$X = \begin{bmatrix} 1 & x_1^{(1)} & & x_n^{(1)} \\ 1 & x_1^{(2)} & \dots & x_n^{(2)} \\ 1 & \vdots & & \vdots \\ 1 & x_1^{(m)} & & x_n^{(m)} \end{bmatrix} \quad w = \begin{bmatrix} w_0 \\ w_1 \\ \dots \\ w_n \end{bmatrix} \quad Y = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \dots \\ y^{(m)} \end{bmatrix}$$

Bias

$$h_w(X) = Xw$$



Zero gradient!

Goal: find w that minimizes J_{MSE}:

Set $\dfrac{\partial J_{MSE}(w)}{\partial w} = 0$

A bunch of math…

$$2X^TXw - 2X^TY = 0$$

$$2X^TXw = 2X^TY$$

$$X^TXw = X^TY$$

$$\boxed{w = (X^TX)^{-1}X^TY}$$

Assume invertible

Gradient Descent vs Normal Equation

	Gradient Descent	**Normal Equation**
Need to choose γ	Yes	No
Iteration(s)	Many	None
Large number of samples n?	No problem	Slow, $(X^T X)^{-1} \rightarrow O(n^3)$
Feature scaling?	May be necessary	Not necessary
Constraints	-	$X^T X$ needs to be invertible

Summary

- Linear Regression: **fitting a line** to data
- Gradient Descent
  - Gradient Descent Algorithm: **follow –gradient** to reduce error
  - Linear Regression with Gradient Descent: **convex** optimization, **one minimum**
  - Variants of Gradient Descent: batch, mini-batch, stochastic
- Linear Regression: Challenges and Solutions
  - Linear Regression with Many Attributes: $h_w(x) = \sum_{j=0}^{n} w_j x_j = w^T x$
  - Dealing with Features of Different Scales: **normalize**!
  - Dealing with Non-Linear Relationship: **transform features**
- Normal Equation: **analytically** find the best parameters

Coming Up Next Week

- Logistic Regression
  - Gradient Descent
  - Multi-class classification
  - Non-linear decision boundary
- (More) Performance Measure
- (More) Model Evaluation

To Do

- **Lecture Training 5**
  - +100 Free EXP
  - +50 Early bird bonus
- **Problem Set 4**
  - Out today!