Social Media & Text Analysis

lecture 5 - Paraphrase Identification and Linear Regression

Follow @cocoweixu

CSE 5539-0010 Ohio State University
Instructor: Wei Xu
Website: socialmedia-class.org

In-class Presentation

- pick your topic and sign up
- a 10 minute presentation (20 points)
  - A Social Media Platform
  - Or a NLP Researcher

Reading #6 & Quiz #3

Identifying Products in Online Cybercrime Marketplaces: A Dataset for Fine-grained Domain Adaptation

Greg Durrett
UT Austin
gdurrett@cs.utexas.edu

Jonathan K. Kummerfeld
University of Michigan
jkummerf@umich.edu

Taylor Berg-Kirkpatrick
Carnegie Mellon University
tberg@cs.cmu.edu

Rebecca S. Portnoff
UC Berkeley
rsportnoff@cs.berkeley.edu

Sadia Afroz
ICSI, UC Berkeley
sadia@icsi.berkeley.edu

Damon McCoy
NYU
mccoy@nyu.edu

Kirill Levchenko
UC San Diego
klevchen@cs.ucsd.edu

Vern Paxson
ICSI, UC Berkeley
vern@berkeley.edu

Abstract

One weakness of machine-learned NLP models is that they typically perform poorly on out-of-domain data. In this work, we study the task of identifying products being bought and sold in online cybercrime forums, which exhibits particularly challenging cross-domain effects. We formulate a task that represents a hybrid of slot-filling information extraction and named entity recognition and annotate data from four different forums. Each of these forums constitutes its own "fine-grained domain" in that the forums

TITLE: [buy] Backconnect bot
BODY: Looking for a solid backconnect bot .
If you know of anyone who codes them please let me know

(a) File 0-initiator4856

TITLE: Exploit cleaning ?
BODY: Have some Exploits i need fud .

(b) File 0-initiator10815

Figure 1: Example posts and annotations from Darkode, with annotated product tokens underlined. The second example exhibits jargon (*fud* means "fully undetectable"), nouns that could be a product in other contexts (*Exploit*), and multiple lexically-distinct descriptions of a single service.

Mini Research Proposal

- propose/explore NLP problems in GitHub dataset

https://github.com

Mini Research Proposal

- propose/explore NLP problems in GitHub dataset
- GitHub:
  - a social network for programmers (sharing, collaboration, bug tracking, etc.)
  - hosting Git repositories (a version control system that tracks changes to files, usually source code)
  - containing potentially interesting text fields in natural language (comments, issues, etc.)

(Recap)

what is Paraphrase?

“sentences or phrases that convey approximately the same meaning using different words” — (Bhagat & Hovy, 2012)

(Recap)

what is Paraphrase?

"sentences or phrases that convey approximately the same meaning using different words" — (Bhagat & Hovy, 2012)

wealthy **word** *rich*

(Recap)

what is Paraphrase?

"sentences or phrases that convey approximately the same meaning using different words" — (Bhagat & Hovy, 2012)

wealthy	**word**	*rich*
the king's speech	**phrase**	*His Majesty's address*

(Recap)

what is Paraphrase?

"sentences or phrases that convey approximately the same meaning using different words" — (Bhagat & Hovy, 2012)

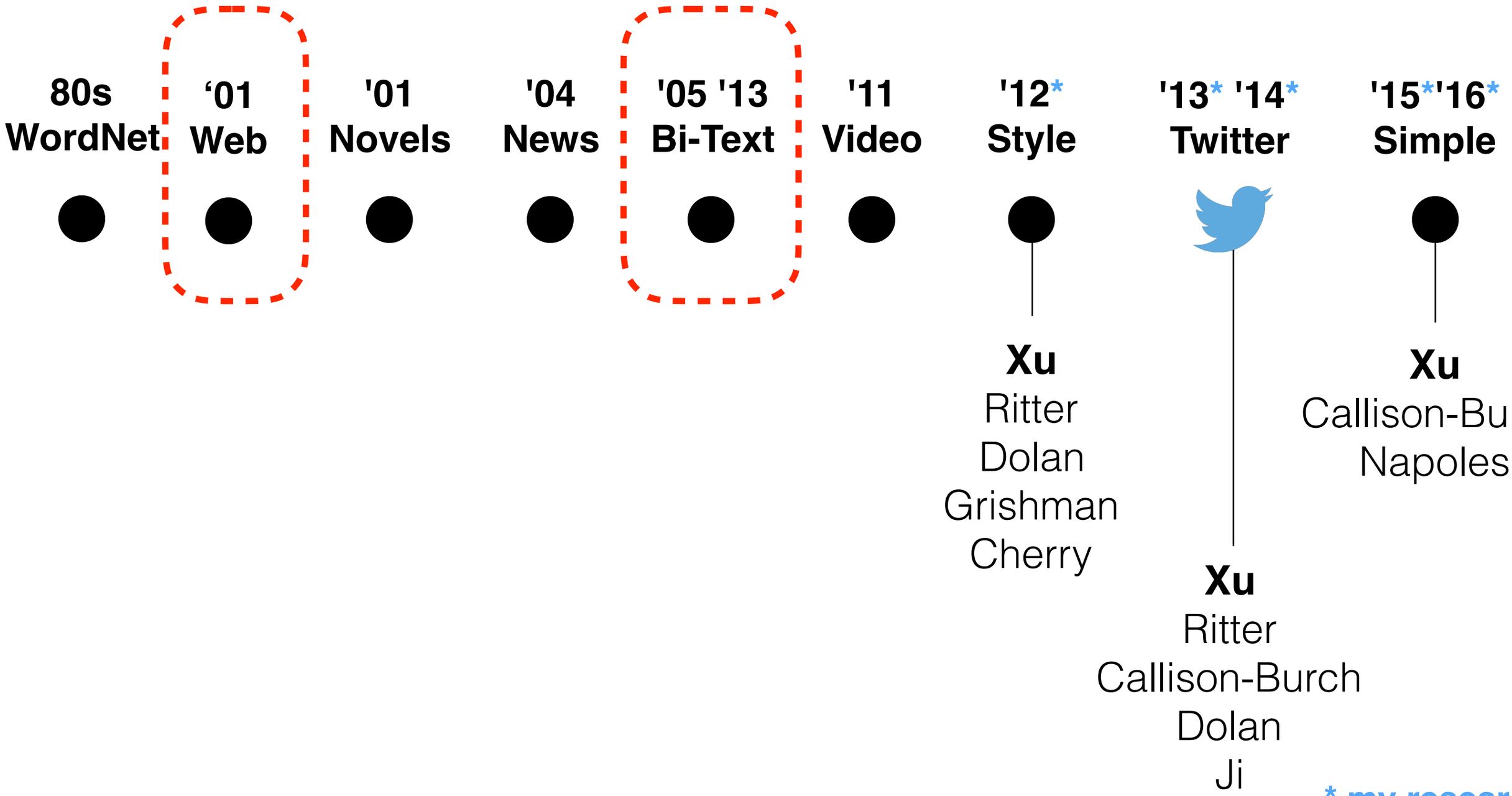
wealthy	**word**	*rich*
the king's speech	**phrase**	*His Majesty's address*
… the forced resignation of the CEO of Boeing, Harry Stonecipher, for …	**sentence**	*… after Boeing Co. Chief Executive Harry Stonecipher was ousted from …*

The Ideal

Translation: "You have a bruised rib."

(Recap)

Paraphrase Research



Wei Xu. “Data-driven Approaches for Paraphrasing Across Language Variations” PhD Thesis (2014)

Distributional Similarity

Lin and Panel (2001) operationalize the Distributional Hypothesis using **dependency relationships** to define **similar environments.**

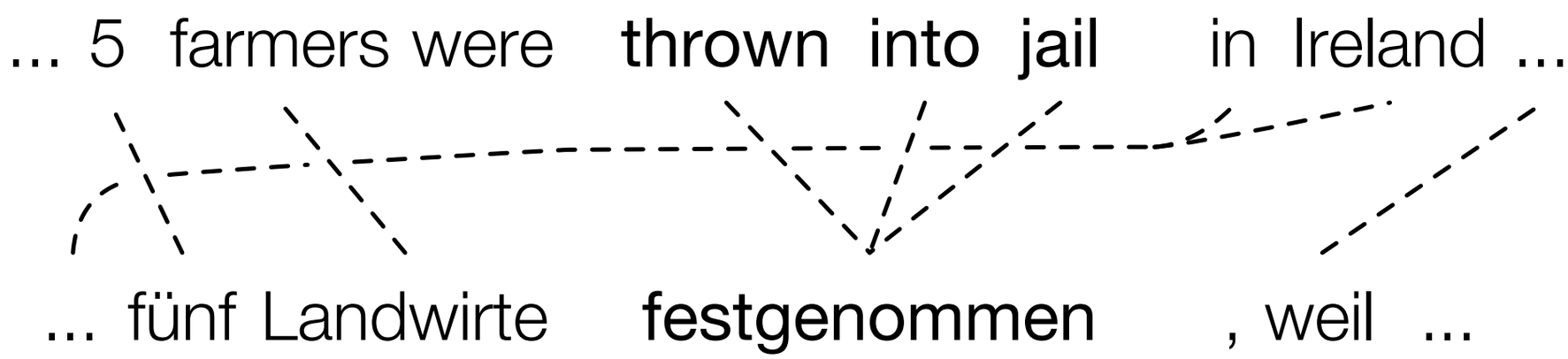
Duty and **responsibility** share a similar set of dependency contexts in large volumes of text:

modified by adjectives	objects of verbs
additional, administrative, assigned, assumed, collective, congressional, constitutional ...	assert, assign, assume, attend to, avoid, become, breach ...

Decking Lin and Patrick Pantel. “DIRT - Discovery of Inference Rules from Text” In KDD (2001)

Bilingual Pivoting

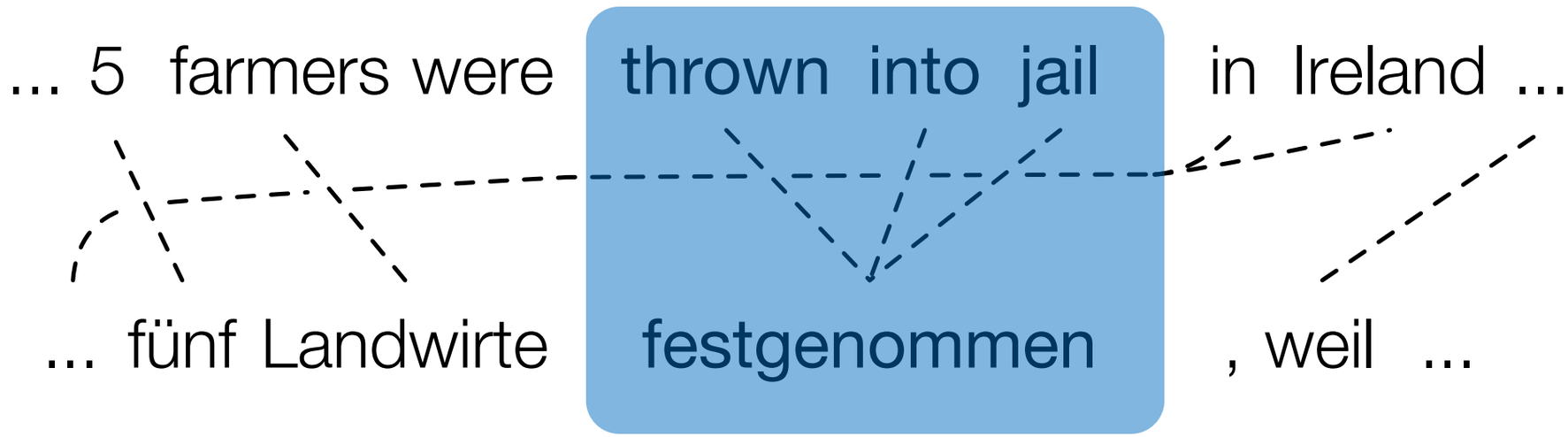
word alignment



Source: Chris Callison-Burch

Bilingual Pivoting

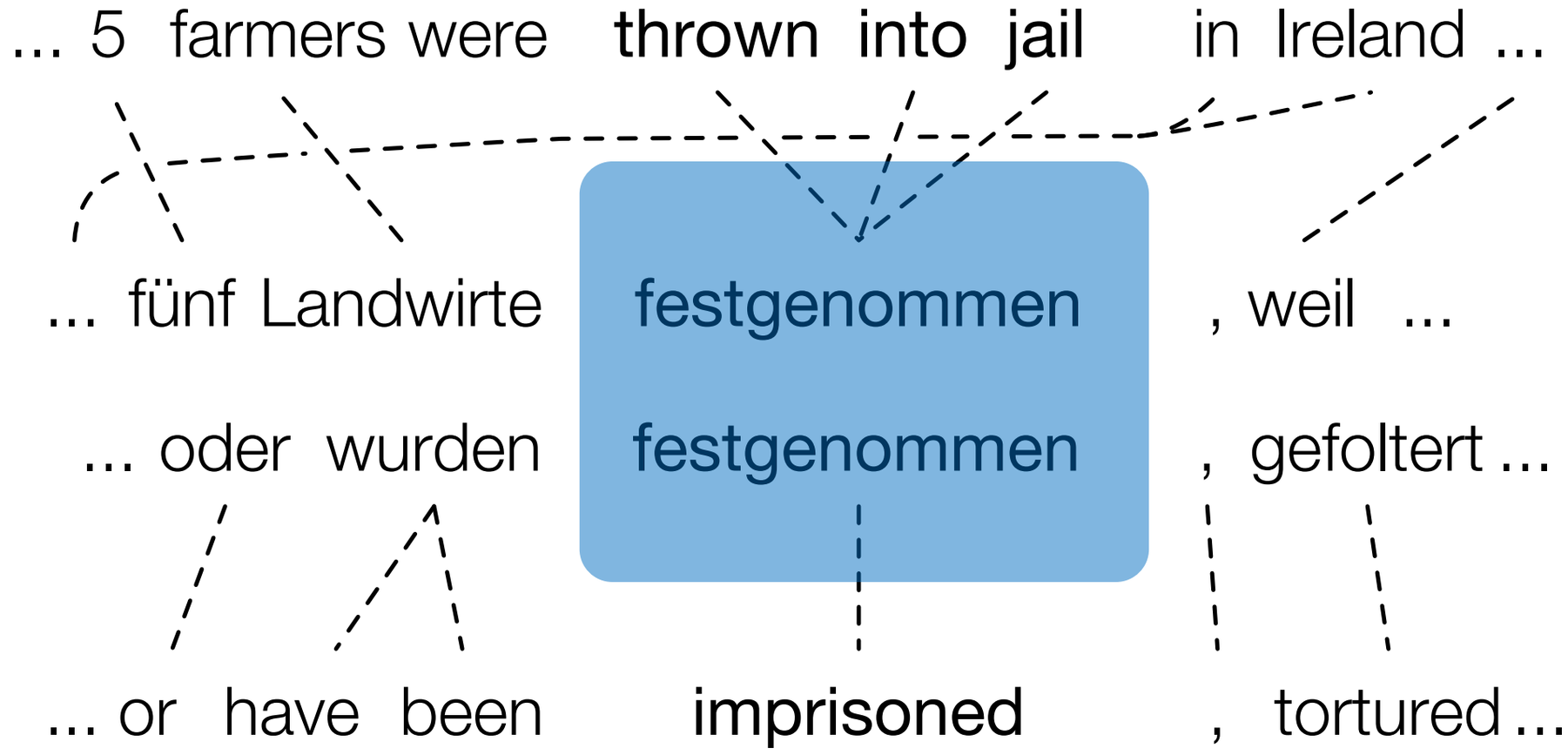
word alignment



Source: Chris Callison-Burch

Bilingual Pivoting

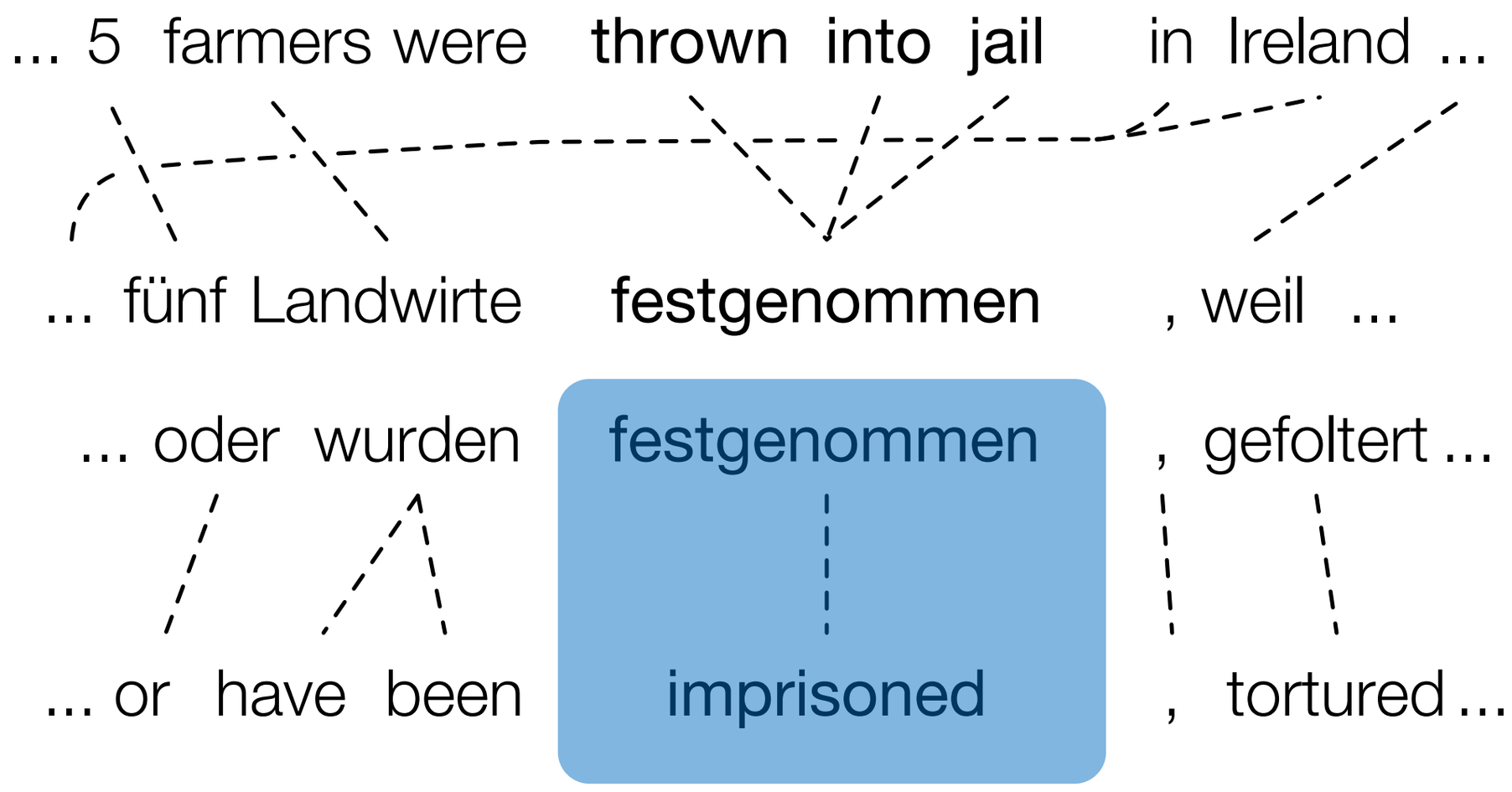
word alignment



Source: Chris Callison-Burch

Bilingual Pivoting

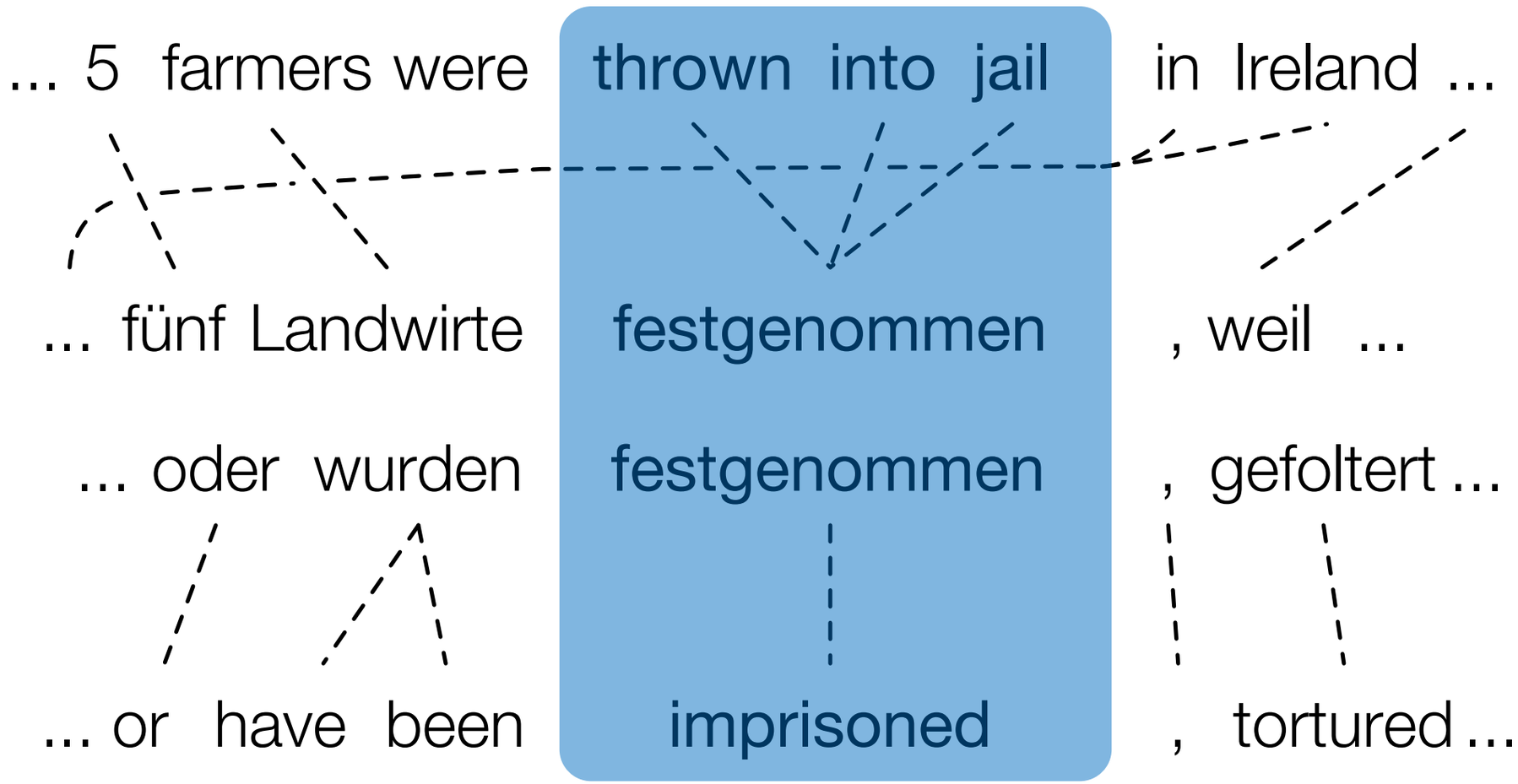
word alignment



Source: Chris Callison-Burch

Bilingual Pivoting

word alignment



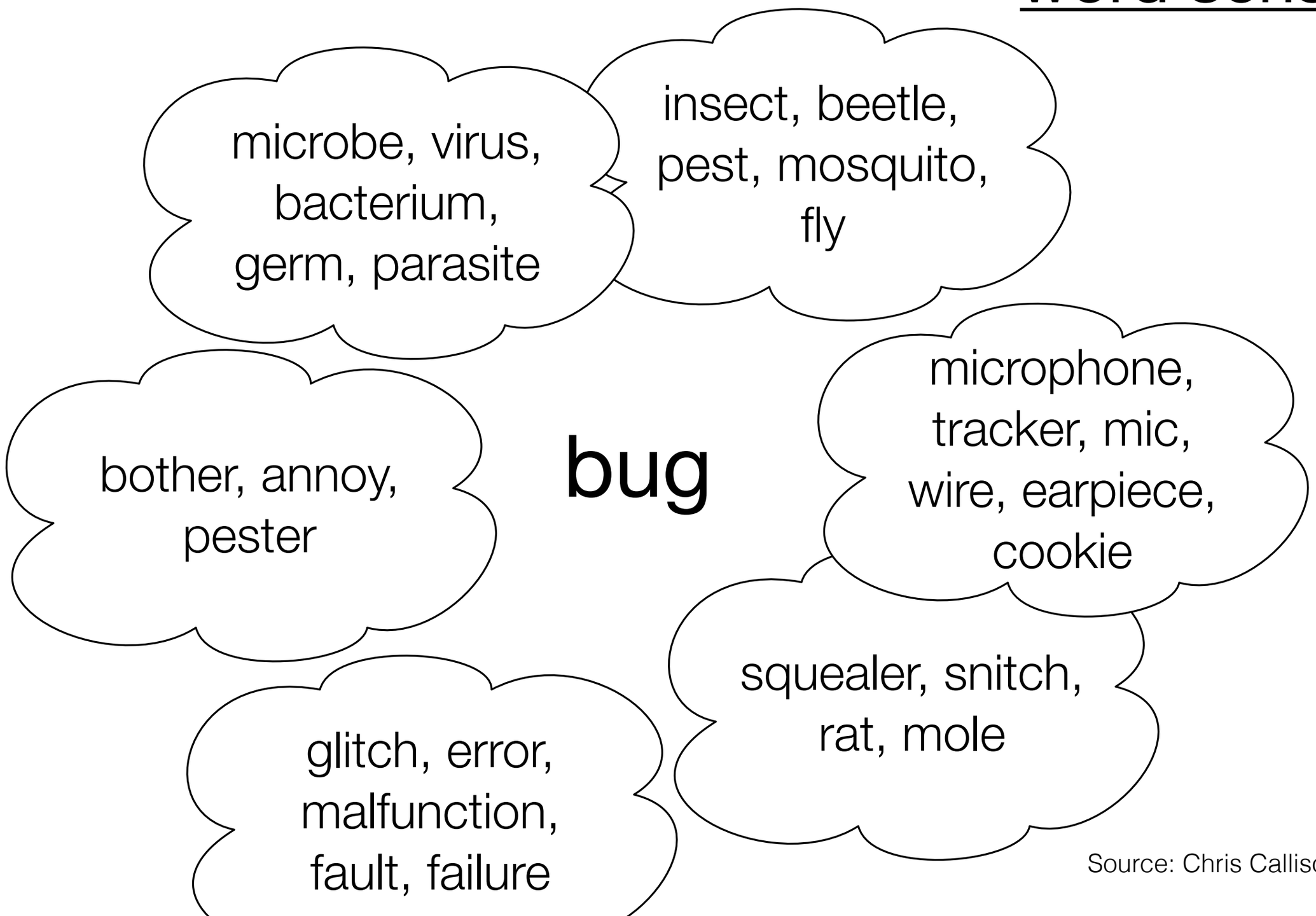
Source: Chris Callison-Burch

Quiz #2

Key Limitations of PPDB?

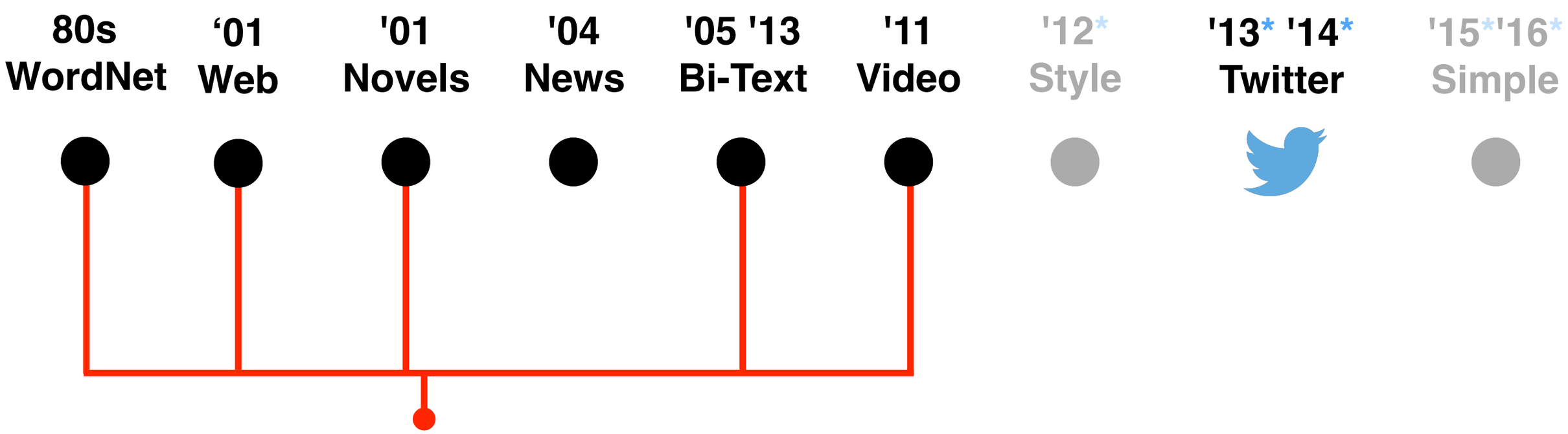
Quiz #2

word sense



Source: Chris Callison-Burch

Another Key Limitation



only paraphrases, no non-paraphrases

* my research

Wei Xu. “Data-driven Approaches for Paraphrasing Across Language Variations” PhD Thesis (2014)

Paraphrase Identification

obtain sentential paraphrases automatically

Mancini has been sacked by Manchester City

Mancini gets the boot from Man City

Yes!

WORLD OF JENKS IS ON AT 11

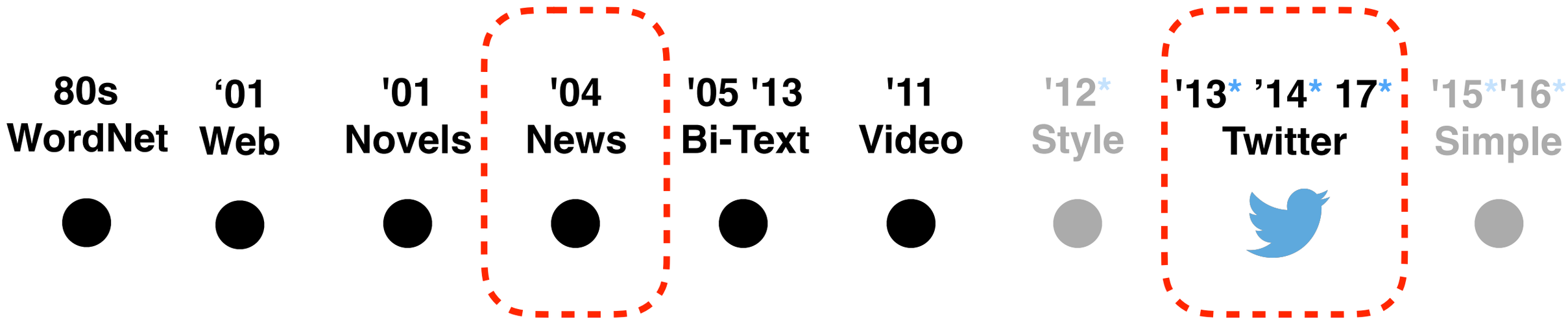
World of Jenks is my favorite show on tv

No!

(meaningful) non-paraphrases are needed to train classifiers!

Wei Xu, Alan Ritter, Chris Callison-Burch, Bill Dolan, Yangfeng Ji. "Extracting Lexically Divergent Paraphrases from Twitter" In TACL (2014)

Also Non-Paraphrases



(meaningful) non-paraphrases are needed to train classifiers!

*** my research**

Wei Xu. “Data-driven Approaches for Paraphrasing Across Language Variations” PhD Thesis (2014)

News Paraphrase Corpus

Microsoft Research Paraphrase Corpus

also contains some non-paraphrases

(Dolan, Quirk and Brockett, 2004; Dolan and Brockett, 2005; Brockett and Dolan, 2005)

Twitter Paraphrase Corpus

Rep. Stacey Newman @staceynewman · 5h
So sad to hear today of former WH Press Sec **James Brady**'s **passing**. @bradybuzz & family will carry on his legacy of #gunsense.

Jim Sciutto @jimsciutto · 4h
Breaking: Fmr. WH Press **Sec. James Brady** has died at 73, crusader for gun control after wounded in '81 Reagan assassination attempt

NBC News @NBCNews · 2h
James Brady, President Reagan's press secretary shot in 1981 assassination attempt, dead at 73 nbcnews.to/WX1Btq pic.twitter.com/1ZtuEakRd9

also contains a lot of non-paraphrases

Wei Xu, Alan Ritter, Chris Callison-Burch, Bill Dolan, Yangfeng Ji. “Extracting Lexically Divergent Paraphrases from Twitter” In TACL (2014)

Paraphrase Identification:

A Binary Classification Problem

- Input:
  - a sentence pair ***x***
  - a fixed set of binary classes ***Y = {0, 1}***
- Output:
  - a predicted class ***y*** ∈ ***Y*** (***y = 0*** or ***y = 1***)

Paraphrase Identification:

A Binary Classification Problem

- Input:
  - a sentence pair ***x***
  - a fixed set of binary classes ***Y = {0, 1}***

negative (non-paraphrases)

- Output:
  - a predicted class ***y*** ∈ ***Y*** (***y = 0*** or ***y = 1***)

Paraphrase Identification:

A Binary Classification Problem

- Input:
  - a sentence pair ***x***
  - a fixed set of binary classes ***Y = {0, 1}***

negative (non-paraphrases)

positive (paraphrases)

- Output:
  - a predicted class ***y*** $\in$ ***Y*** (***y = 0*** or ***y = 1***)

Paraphrase Identification:

A Binary Classification Problem

- Input:
  - a sentence pair ***x***
  - a fixed set of binary classes ***Y = {0, 1}***
- Output:
  - a predicted class ***y*** ∈ ***Y*** (***y = 0*** or ***y = 1***)

Classification Method:

Supervised Machine Learning

- Input:
  - a sentence pair ***x***
  - a fixed set of binary classes ***Y = {0, 1}***
  - a training set of ***m*** hand-labeled sentence pairs ***$(x^{(1)}, y^{(1)}), \ldots, (x^{(m)}, y^{(m)})$***

- Output:
  - a learned classifier ***$\gamma: x \rightarrow y \in Y$*** (***y = 0*** or ***y = 1***)

Classification Method:

Supervised Machine Learning

- Input:
  - a sentence pair ***x*** **(represented by features)**
  - a fixed set of binary classes ***Y = {0, 1}***
  - a training set of ***m*** hand-labeled sentence pairs ***$(x^{(1)}, y^{(1)}), \ldots, (x^{(m)}, y^{(m)})$***

- Output:
  - a learned classifier ***$\gamma: x \rightarrow y \in Y$*** (***y = 0*** or ***y = 1***)

(Recap) Classification Method:

Supervised Machine Learning

- **Naïve Bayes**
- Logistic Regression
- Support Vector Machines (SVM)
- …

(Recap)

Naïve Bayes

- ***Cons:***

features t_i are assumed independent given the class y

$$P(t_1, t_2, ..., t_n \mid y) = P(t_1 \mid y) \cdot P(t_2 \mid y) \cdot ... \cdot P(t_n \mid y)$$

- ***This will cause problems:***
  - *correlated features → double-counted evidence*
  - *while parameters are estimated independently*
  - *hurt classier's accuracy*

Classification Method:

Supervised Machine Learning

- Naïve Bayes
- **Logistic Regression**
- Support Vector Machines (SVM)
- …

Logistic Regression

- One of the most useful **supervised machine learning algorithm** for classification!

- Generally high performance for a lot of problems.

- Much more robust than Naïve Bayes (better performance on various datasets).

Before Logistic Regression

Let's start with something simpler!

Paraphrase Identification:

Simplified Features

- We use only one feature:

  - number of words that two sentence shared in common

A very related problem of Paraphrase Identification:

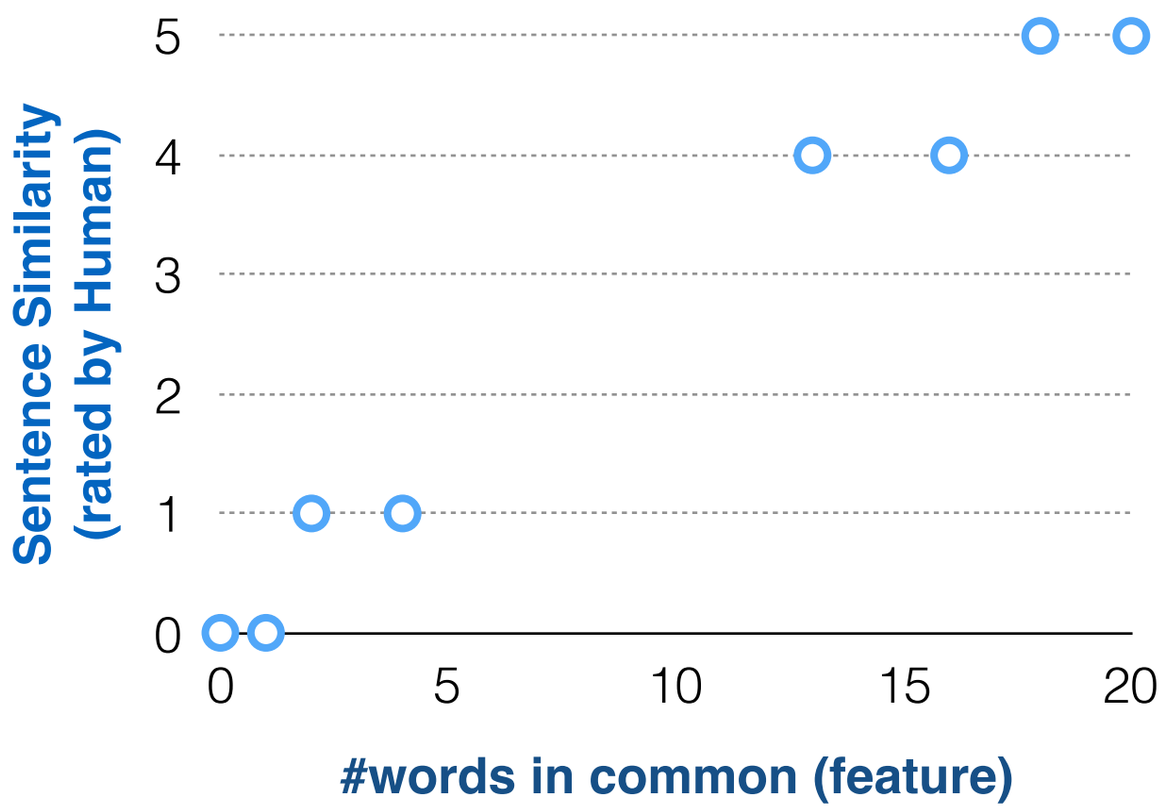
Semantic Textual Similarity

- How similar (close in meaning) two sentences are?

5:	completely equivalent in meaning
4:	mostly equivalent, but some unimportant details differ
3:	roughly equivalent, some important information differs/missing
2:	not equivalent, but share some details
1:	not equivalent, but are on the same topic
0:	completely dissimilar

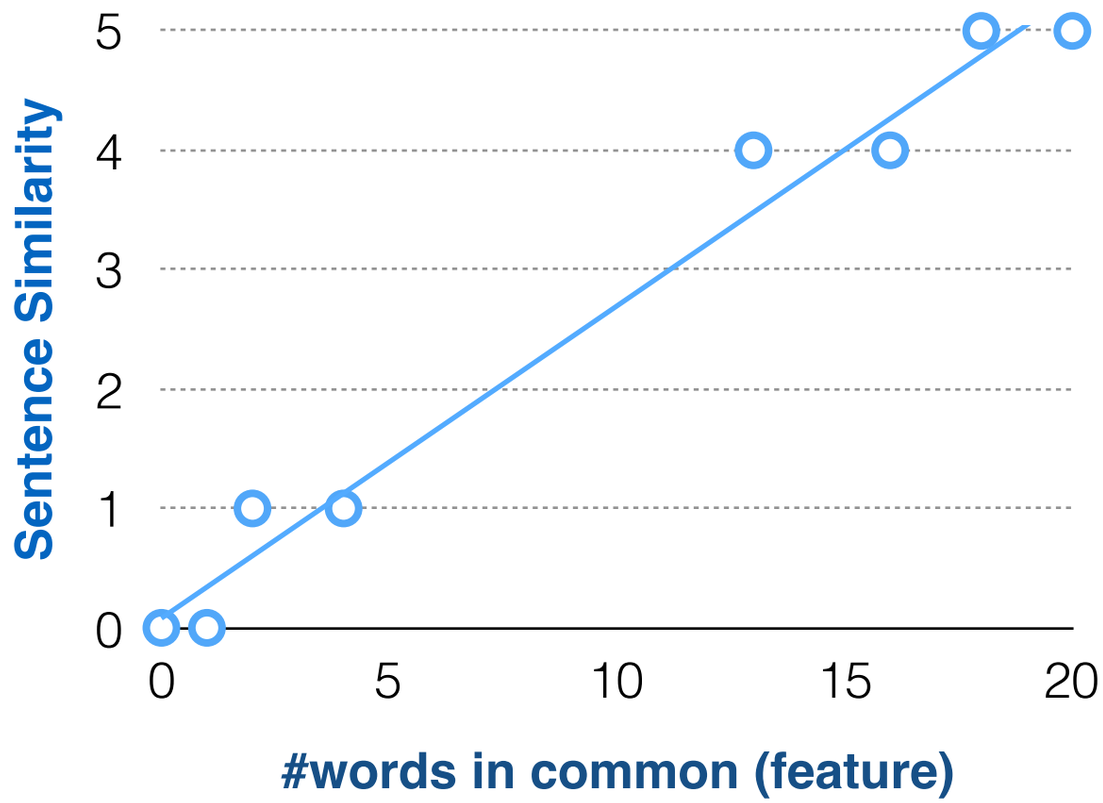
A Simpler Model:

Linear Regression



A Simpler Model:

Linear Regression

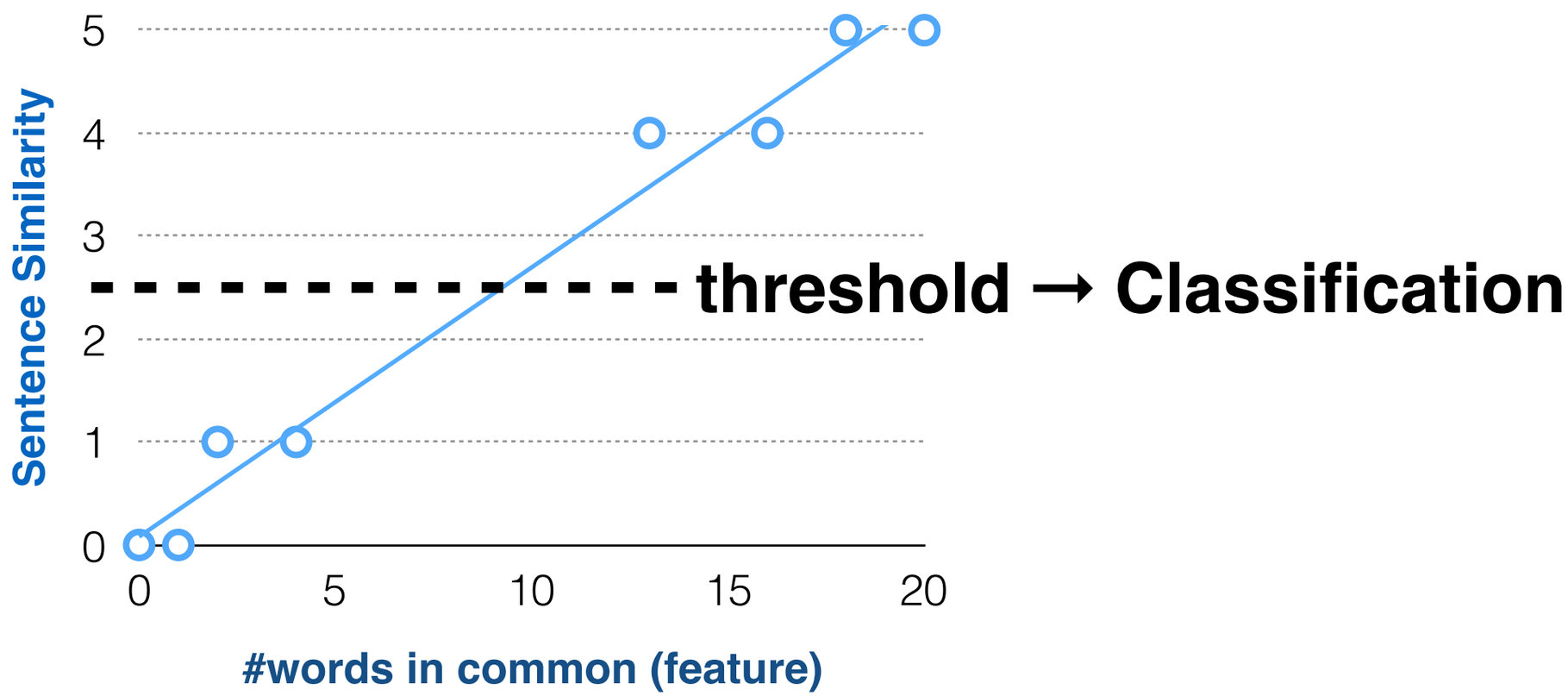


- also supervised learning (learn from annotated data)

- but for **Regression**: predict **real-valued** output
  (Classification: predict discrete-valued output)

A Simpler Model:

Linear Regression

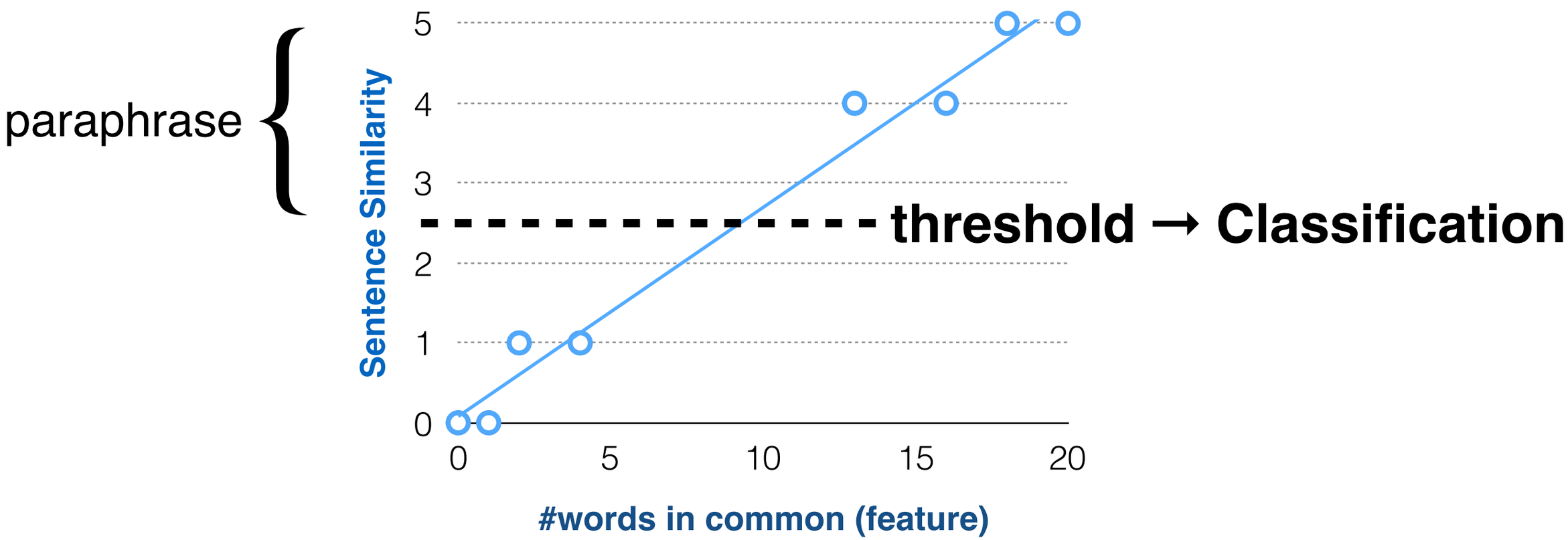


- also supervised learning (learn from labeled data)

- but for **Regression**: predict **real-valued** output
  (Classification: predict discrete-valued output)

A Simpler Model:

Linear Regression

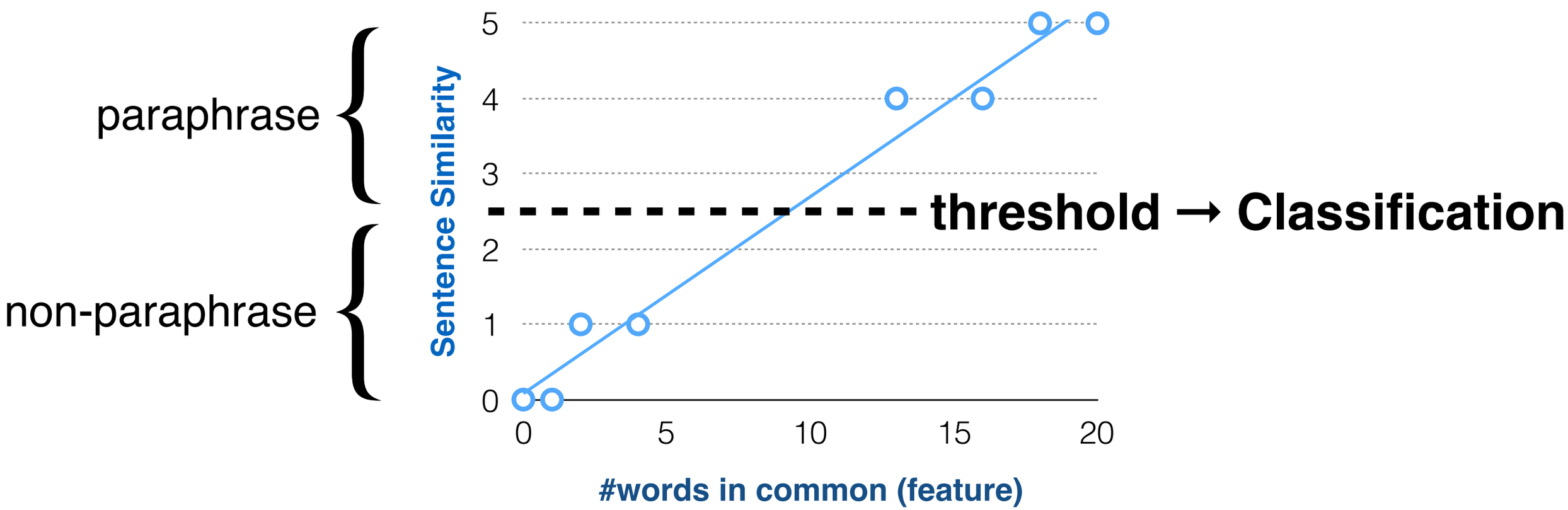


- also supervised learning (learn from labeled data)

- but for **Regression**: predict **real-valued** output
  (Classification: predict discrete-valued output)

A Simpler Model:

Linear Regression



- also supervised learning (learn from labeled data)

- but for **Regression**: predict **real-valued** output
  (Classification: predict discrete-valued output)

Training Set

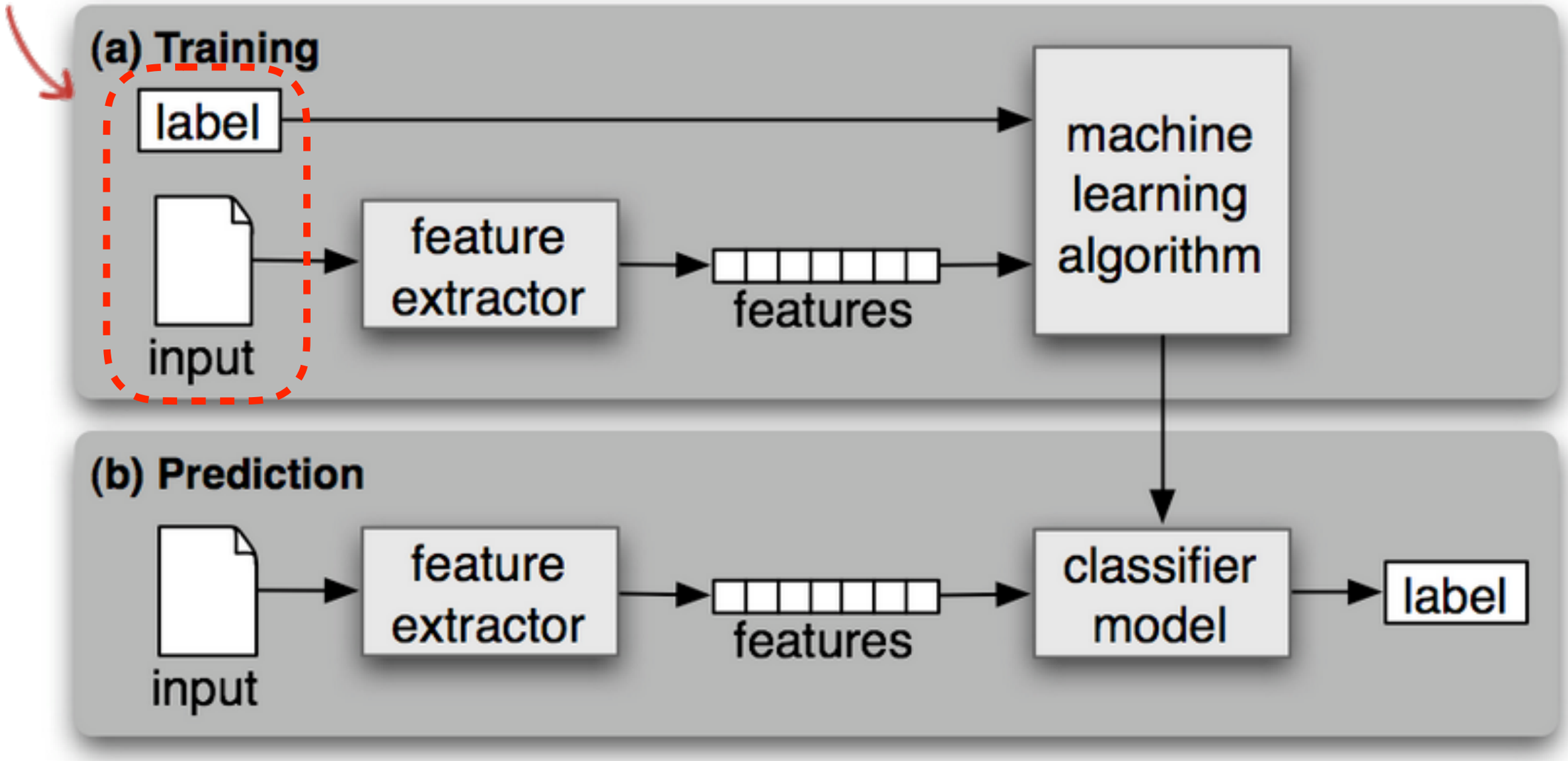
#words in common (*x*)	Sentence Similarity (*y*)
1	0
4	1
13	4
18	5
…	…

- ***m*** hand-labeled sentence pairs ***$(x^{(1)}, y^{(1)}), \ldots, (x^{(m)}, y^{(m)})$***
- ***x***'s: "input" variable / features
- ***y***'s: "output"/"target" variable

(Recap)

Supervised Machine Learning

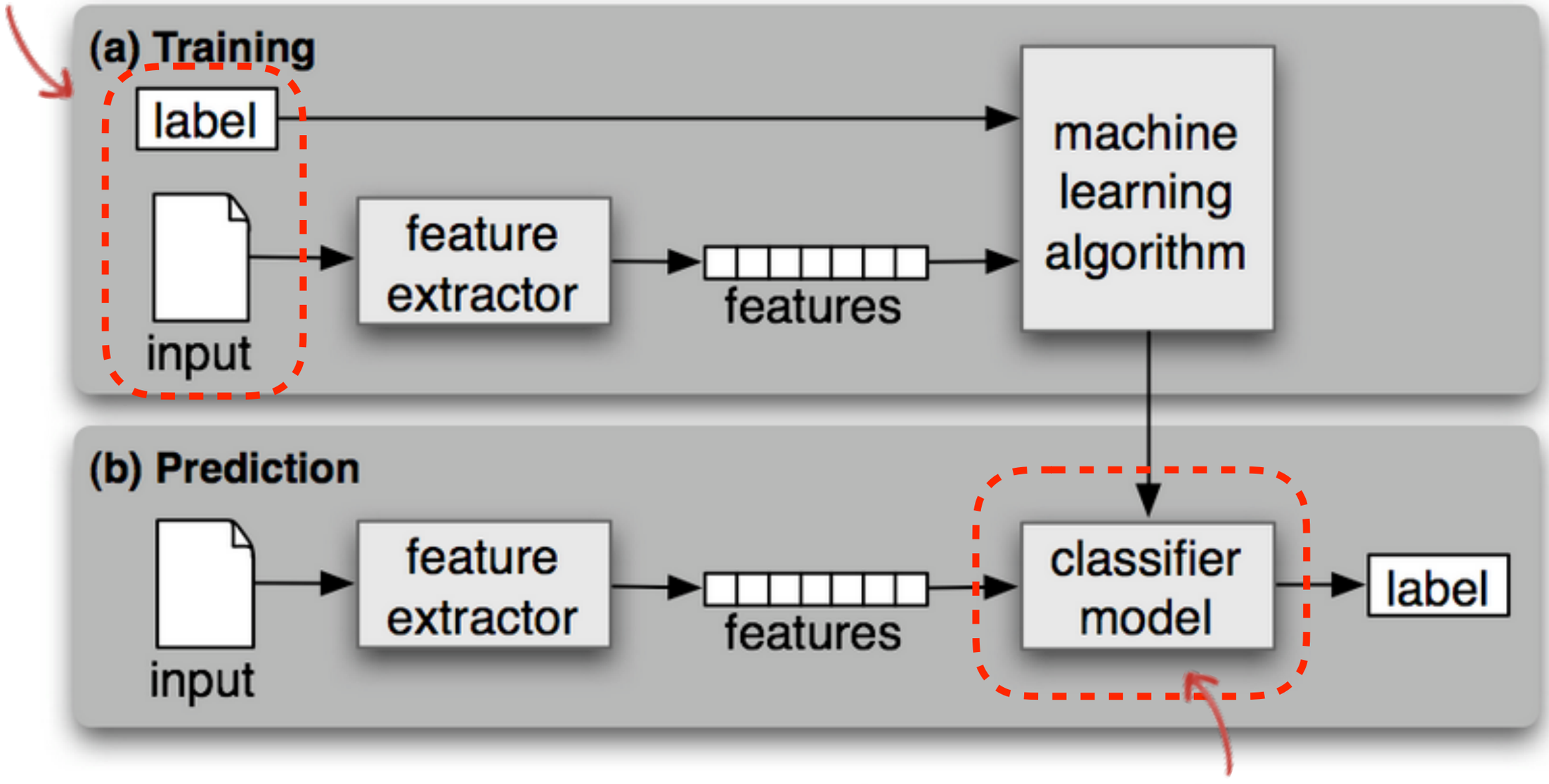
training set



Source: NLTK Book

Supervised Machine Learning

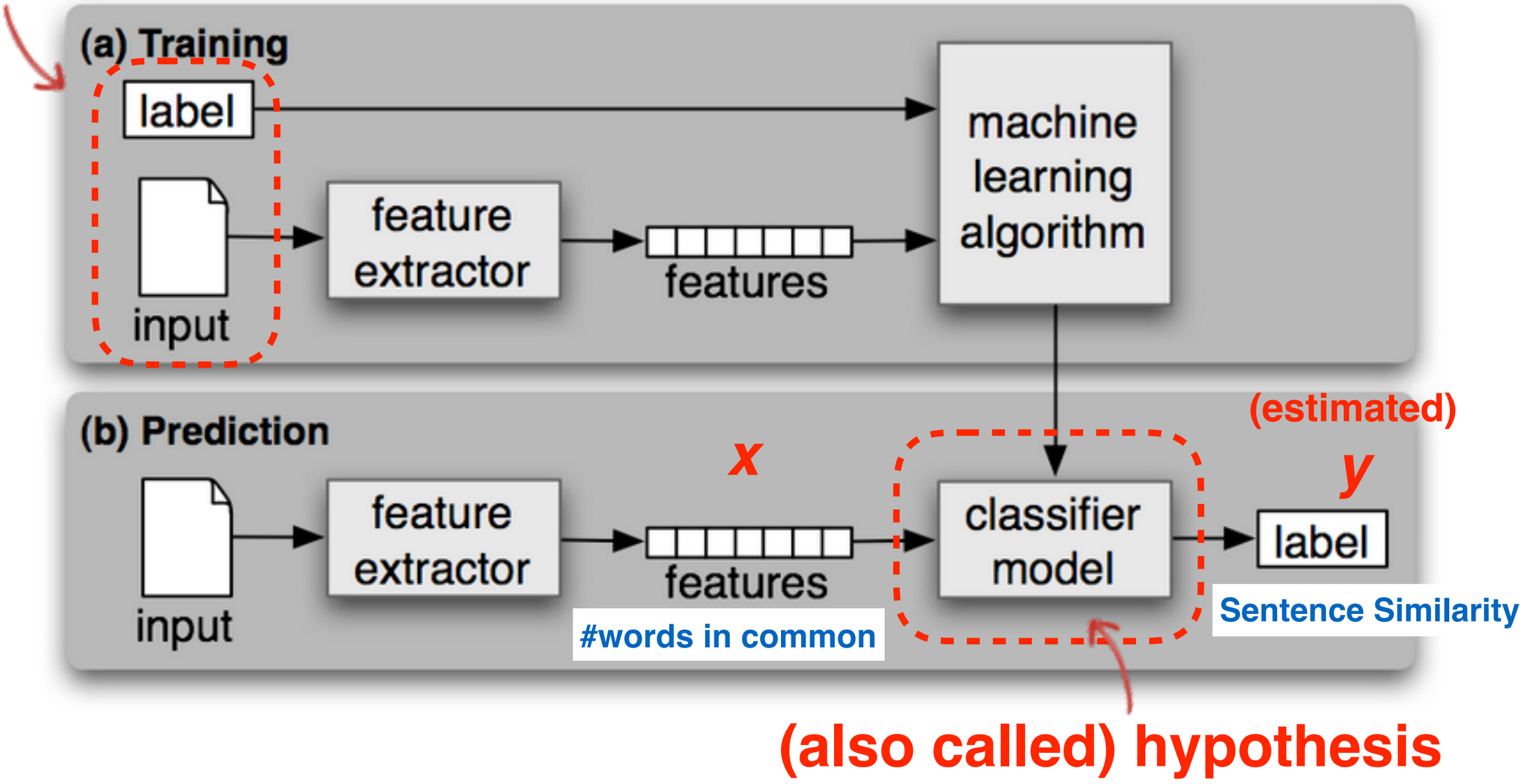
training set



(also called) hypothesis

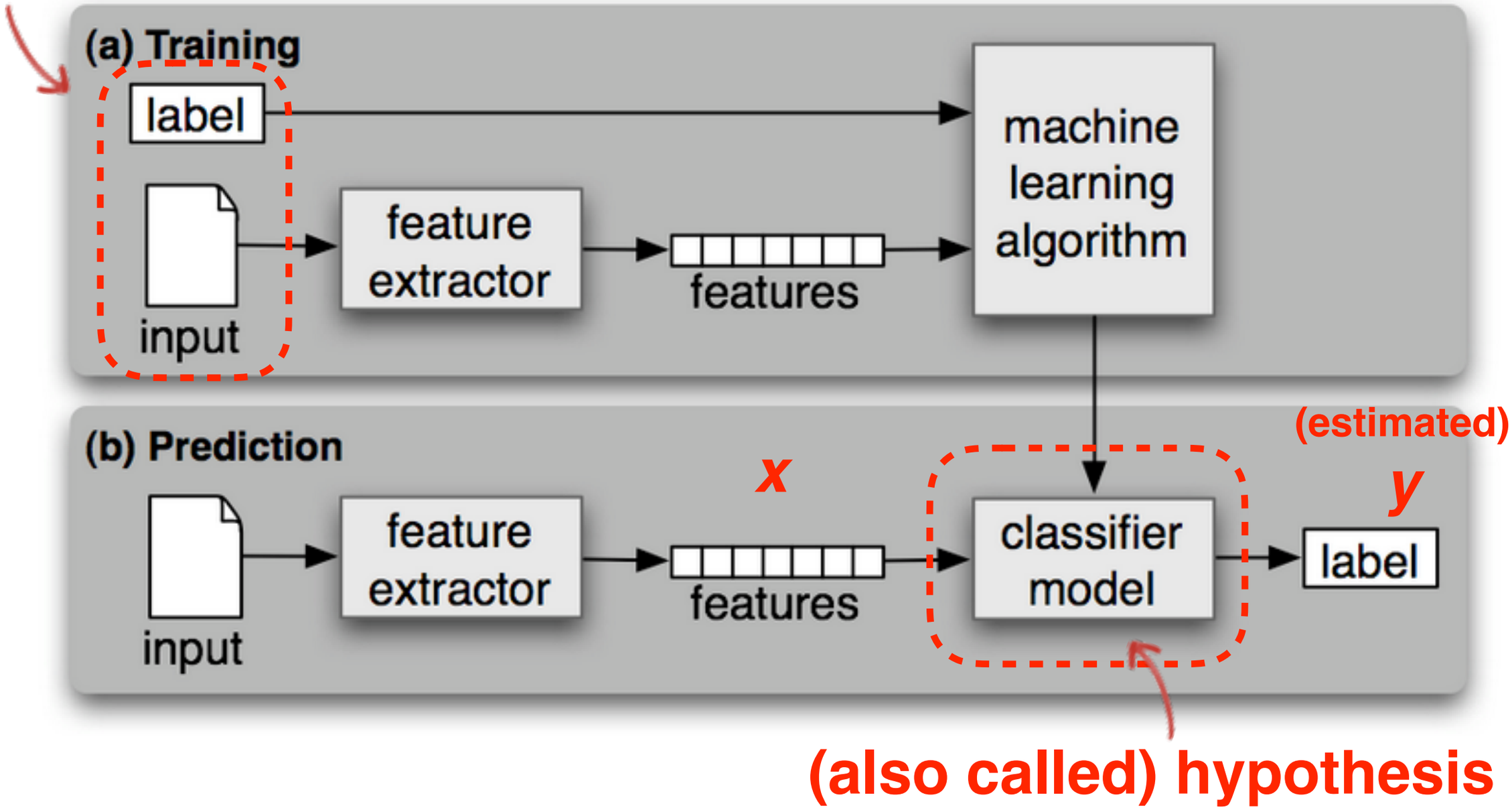
Supervised Machine Learning

training set



Supervised Machine Learning

training set



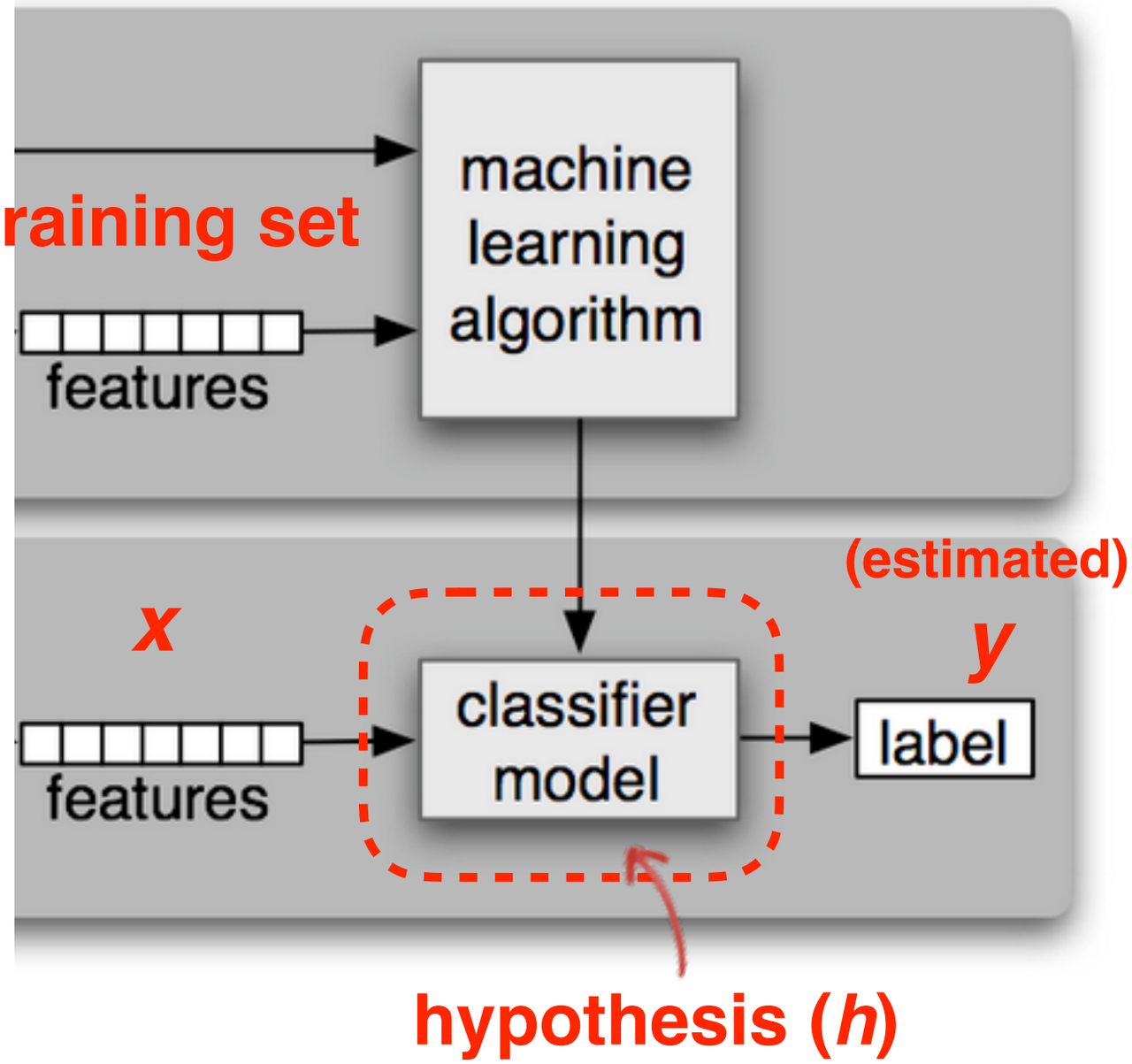
Linear Regression:

Model Representation

- How to represent ***h*** ?

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

Linear Regression
w/ one variable



Linear Regression w/ one variable:

Model Representation

#words in common (*x*)	Sentence Similarity (*y*)
1	0
4	1
13	4
18	5
…	…

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

- ***m*** hand-labeled sentence pairs ***$(x^{(1)}, y^{(1)}), \ldots, (x^{(m)}, y^{(m)})$***
  θ's: parameters

Source: many following slides are adapted from Andrew Ng

Linear Regression w/ one variable:

Model Representation

#words in common (*x*)	Sentence Similarity (*y*)
1	0
4	1
13	4
18	5
…	…

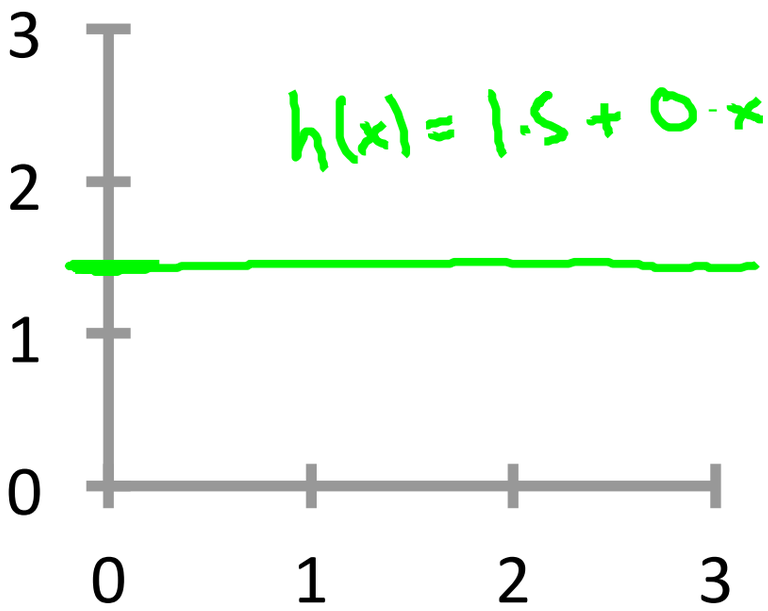
$$h_\theta(x) = \theta_0 + \theta_1 x$$

- ***m*** hand-labeled sentence pairs ***$(x^{(1)}, y^{(1)}), \ldots, (x^{(m)}, y^{(m)})$***
- ***θ***'s: parameters

Linear Regression w/ one variable::

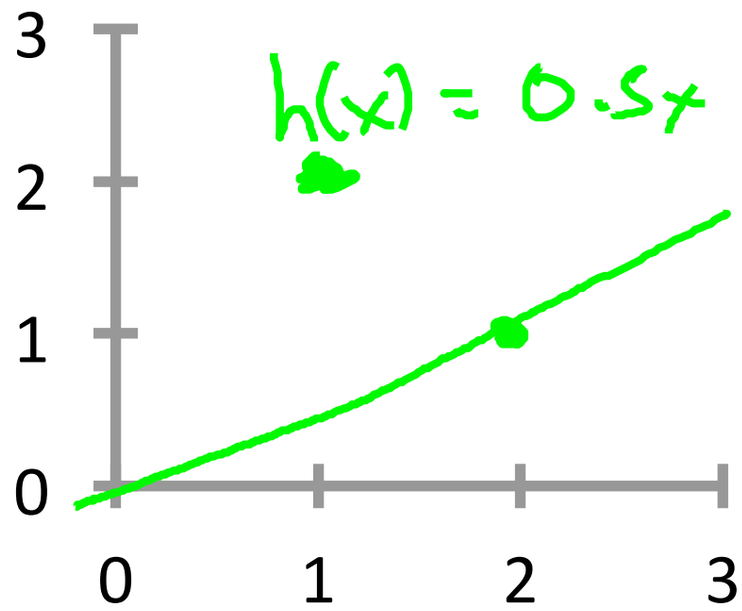
Model Representation

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



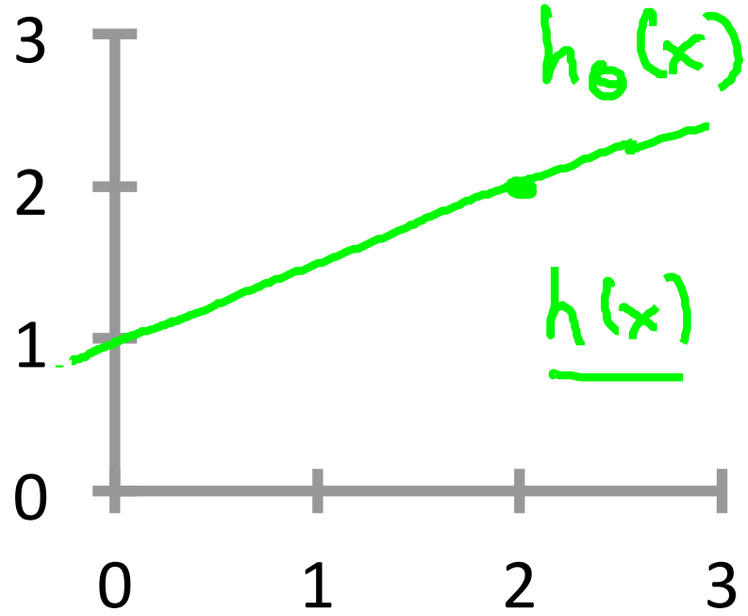
$\theta_0 = 1.5$

$\theta_1 = 0$



$\theta_0 = 0$

$\theta_1 = 0.5$

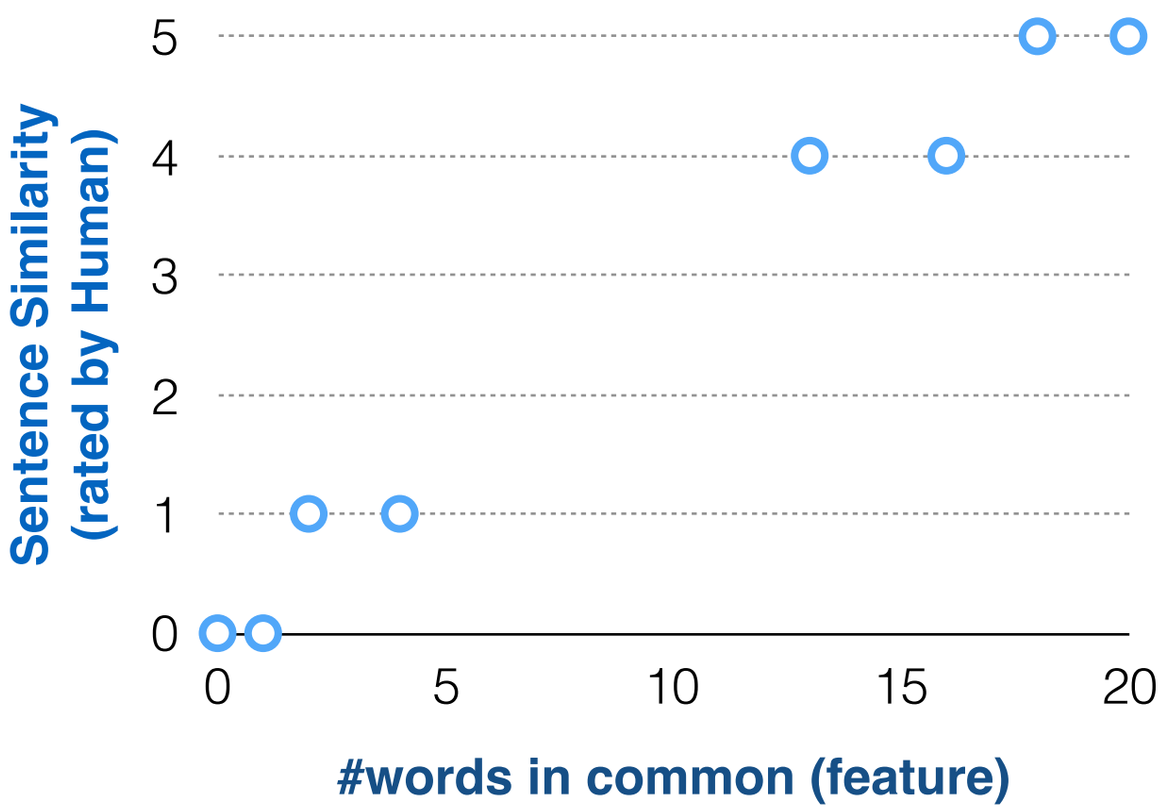


$\theta_0 = 1$

$\theta_1 = 0.5$

Linear Regression w/ one variable:

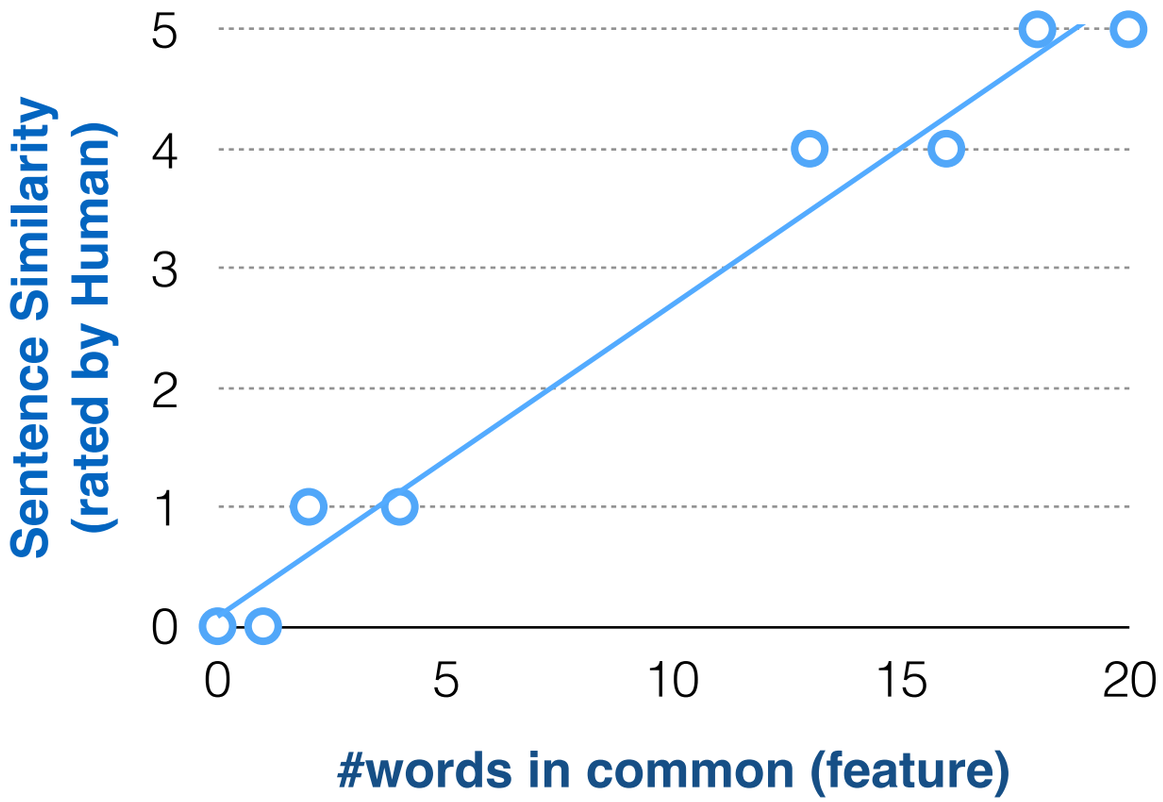
Cost Function



- **Idea:** choose θ_0, θ_1 so that $h_\theta(x)$ is close to y for training examples (x, y)

Linear Regression w/ one variable:

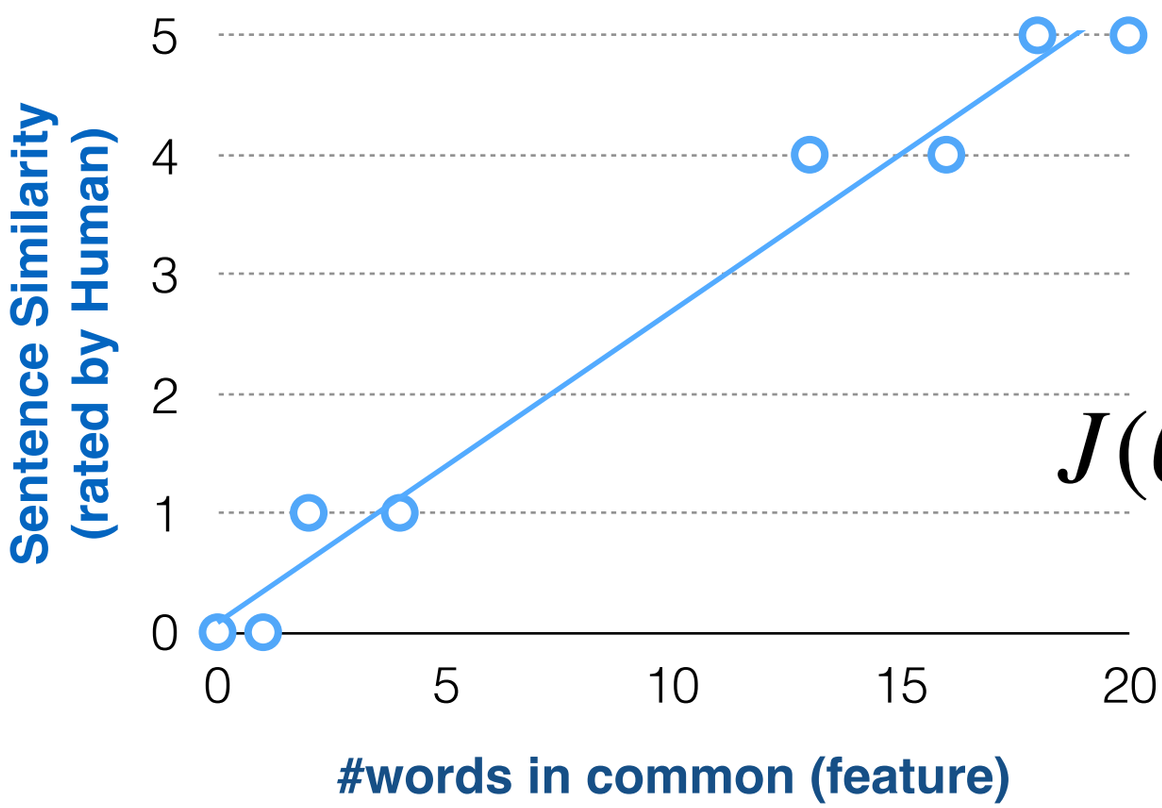
Cost Function



- **Idea:** choose θ_0, θ_1 so that $h_\theta(x)$ is close to y for training examples (x, y)

Linear Regression w/ one variable:

Cost Function



squared error function:

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

- **Idea:** choose θ_0, θ_1 so that $h_\theta(x)$ is close to y for training examples (x, y)

$$\underset{\theta_0,\,\theta_1}{\text{minimize}}\, J(\theta_0,\theta_1)$$

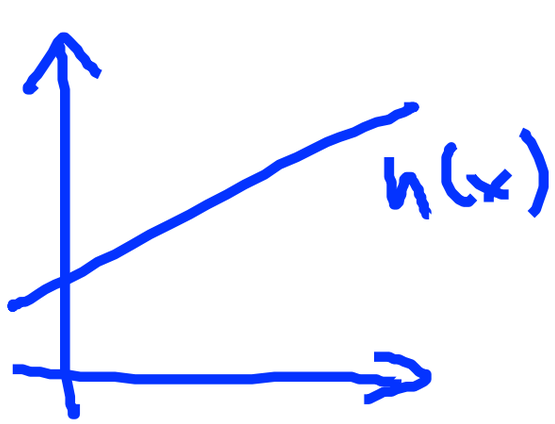
Linear Regression

- **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:** $\underset{\theta_0, \theta_1}{\text{minimize}}\, J(\theta_0, \theta_1)$

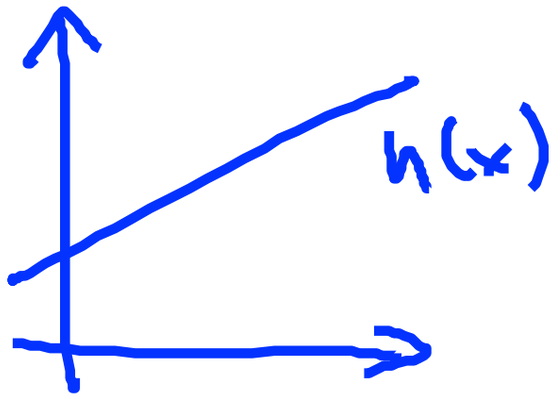
Linear Regression

- **Hypothesis**:

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

- **Parameters:**

θ_0, θ_1



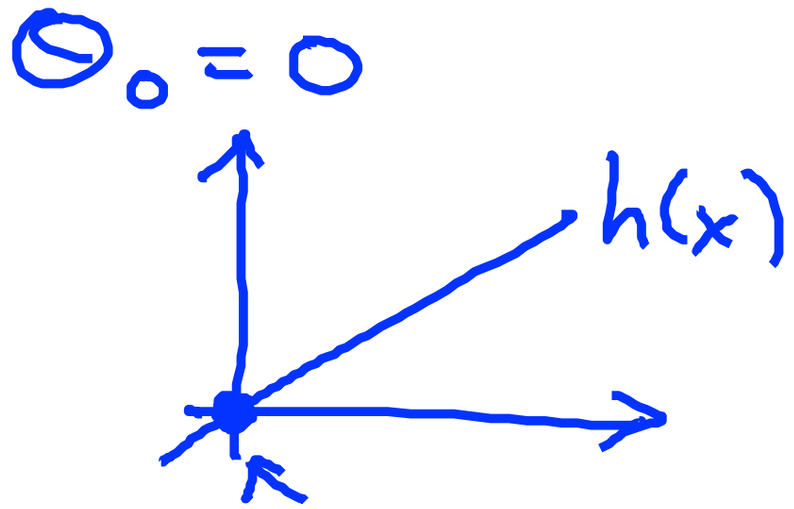
- **Cost Function:**

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

- **Goal:** $\underset{\theta_0, \theta_1}{\text{minimize}}\, J(\theta_0, \theta_1)$

Simplified

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



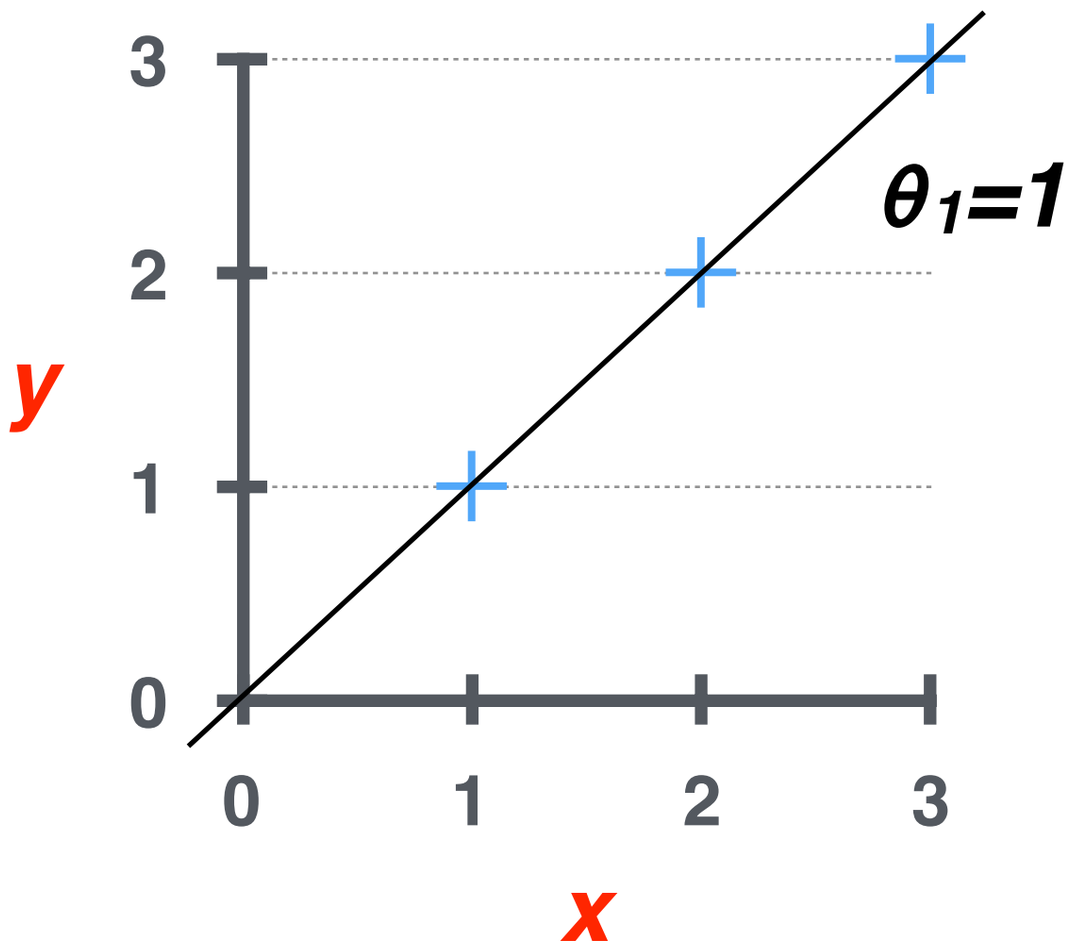
$$J(\theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$$

$$\underset{\theta_1}{\text{minimize}}\, J(\theta_1)$$

Hypothesis

$$h_\theta(x)$$

(for fixed θ_1, this is a function of x)



Cost Function

$$J(\theta_1)$$

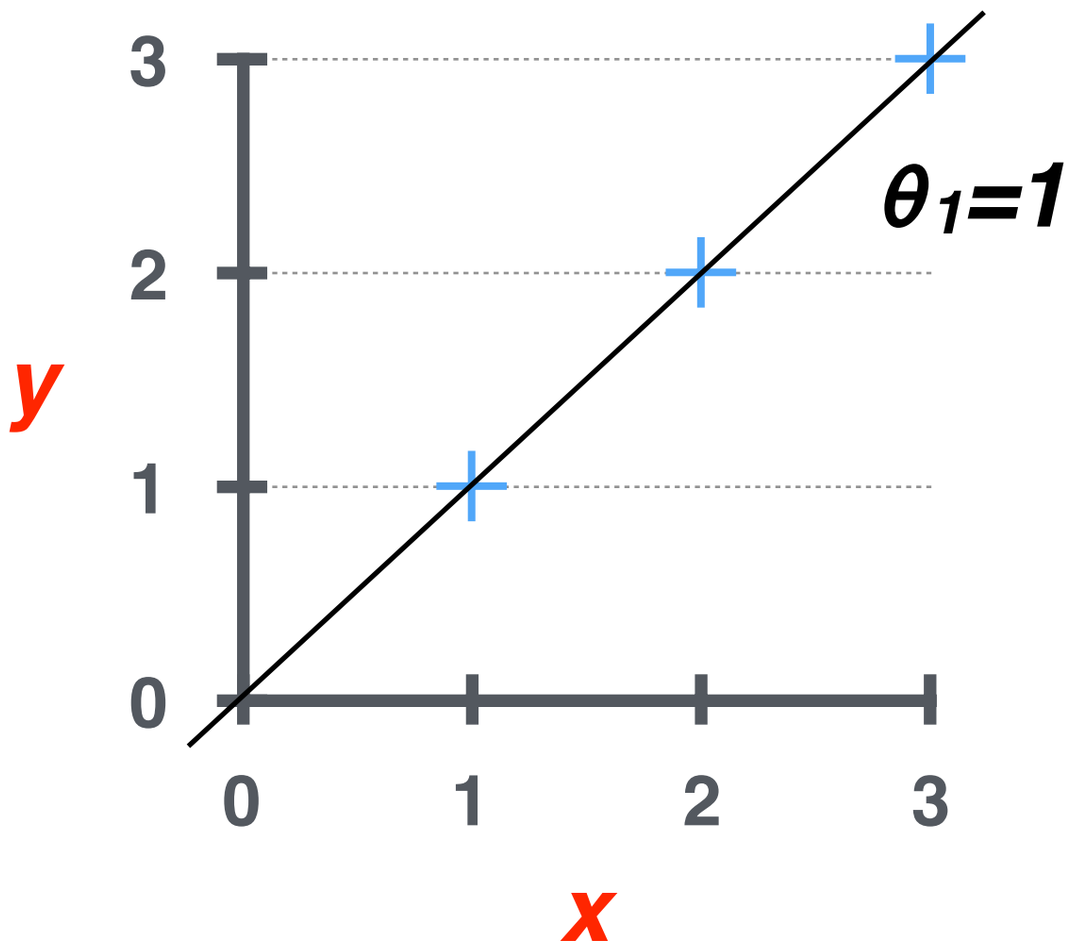
(function of the parameter θ_1)

Q: $J(1) = ?$

Hypothesis

$$h_\theta(x)$$

(for fixed θ_1, this is a function of x)



Cost Function

$$J(\theta_1)$$

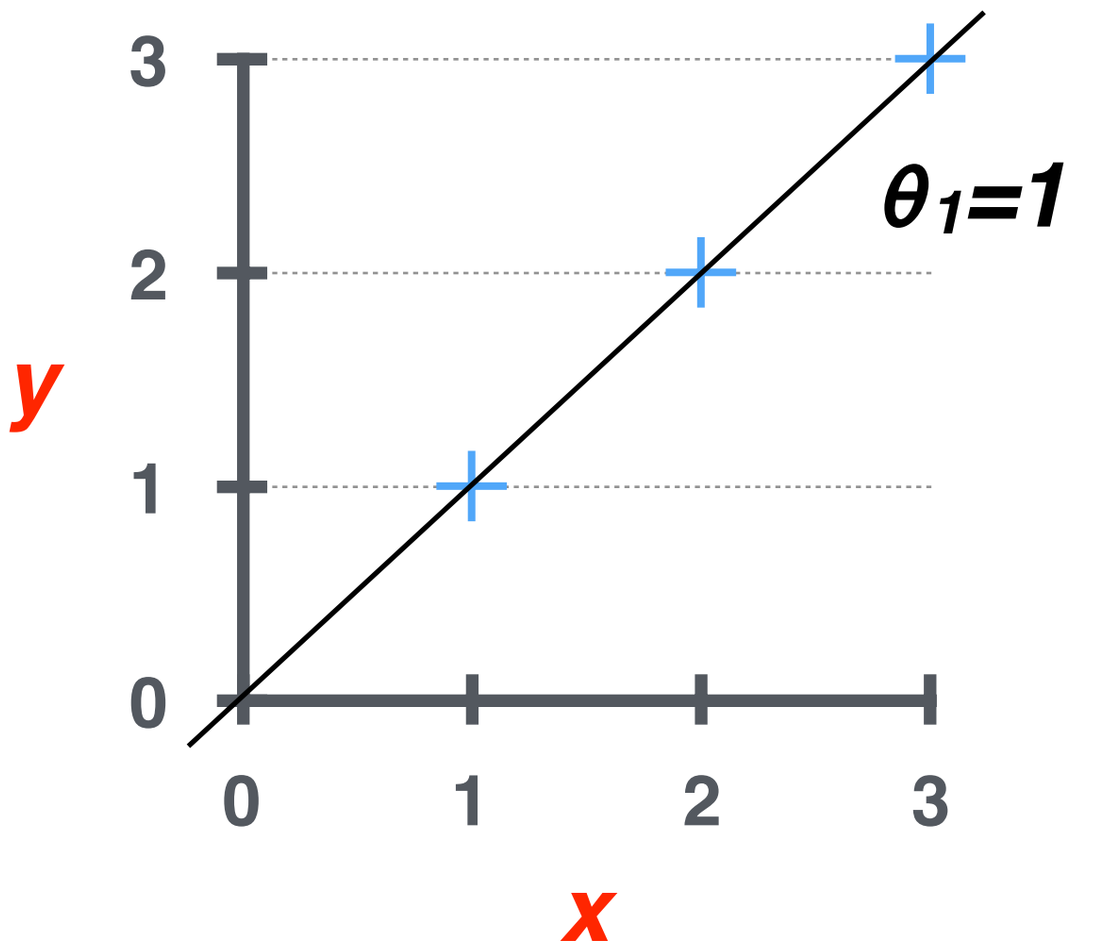
(function of the parameter θ_1)

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

Hypothesis

$$h_\theta(x)$$

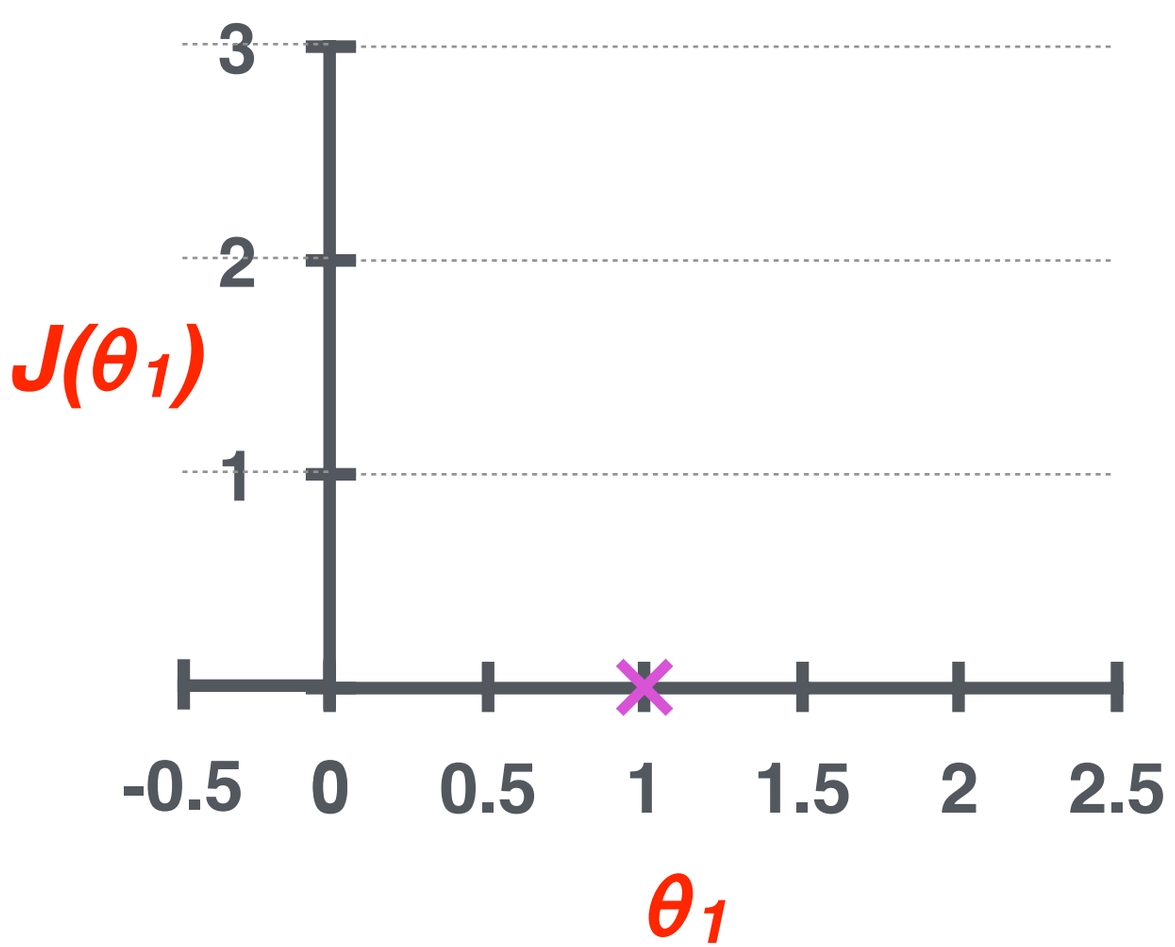
(for fixed θ_1, this is a function of x)



Cost Function

$$J(\theta_1)$$

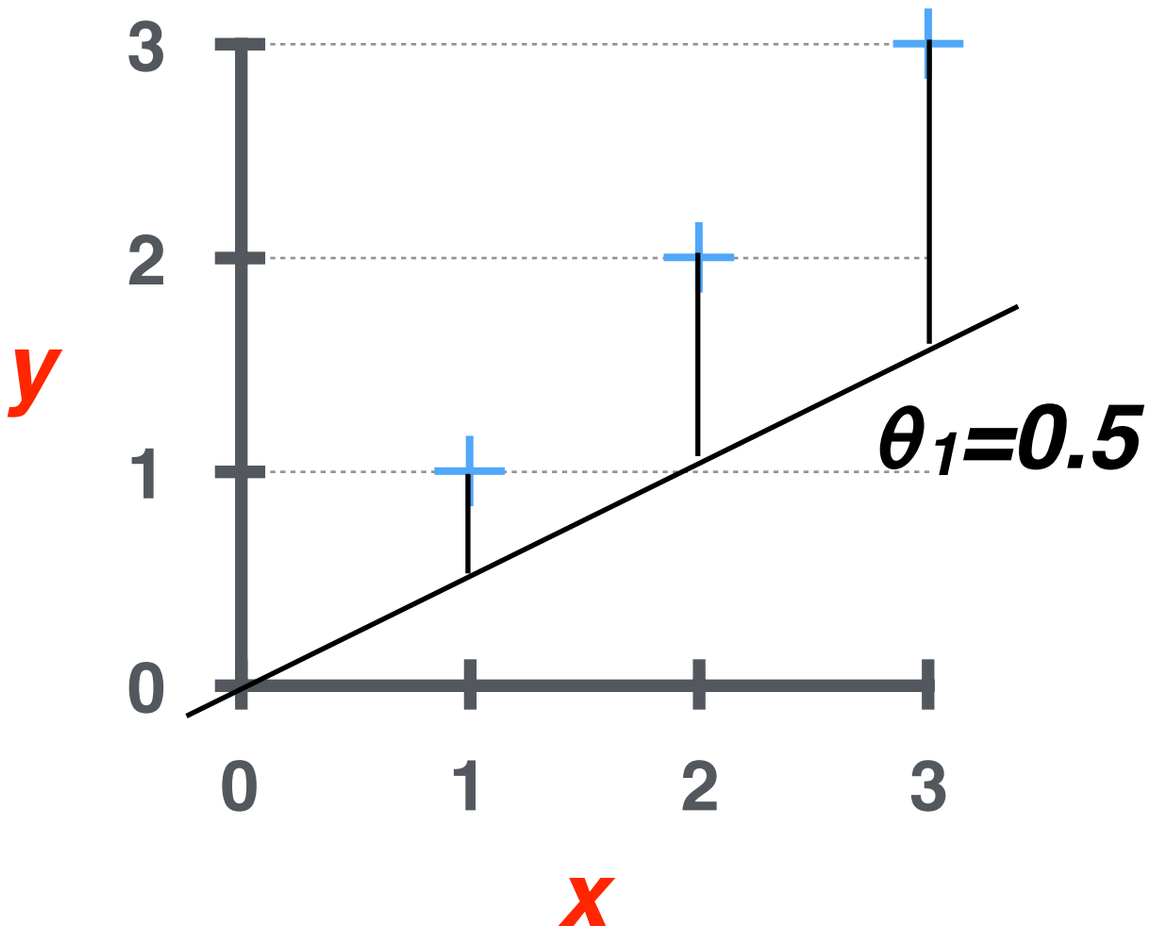
(function of the parameter θ_1)



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

$h_\theta(x)$

(for fixed θ_1, this is a function of x)



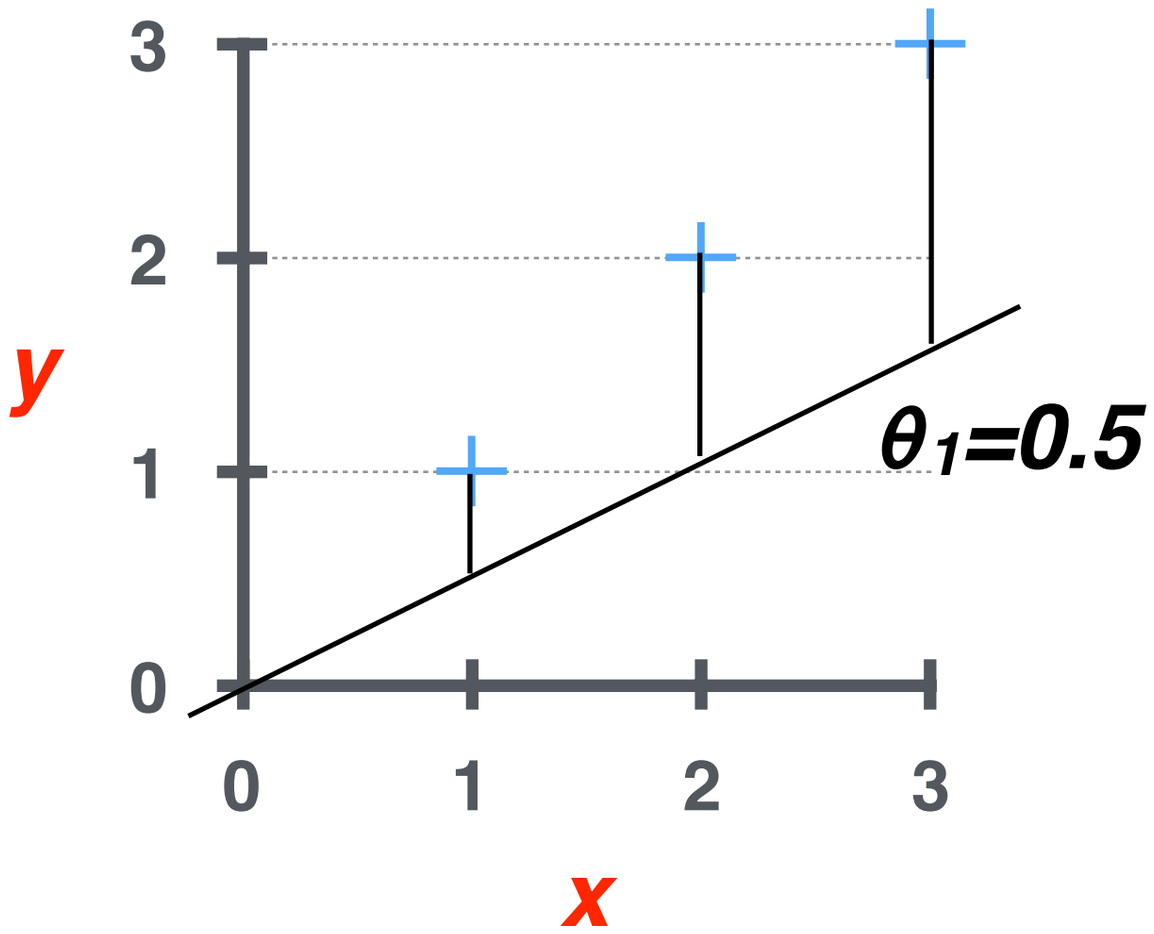
$J(\theta_1)$

(function of the parameter θ_1)

Q: $J(0.5) = ?$

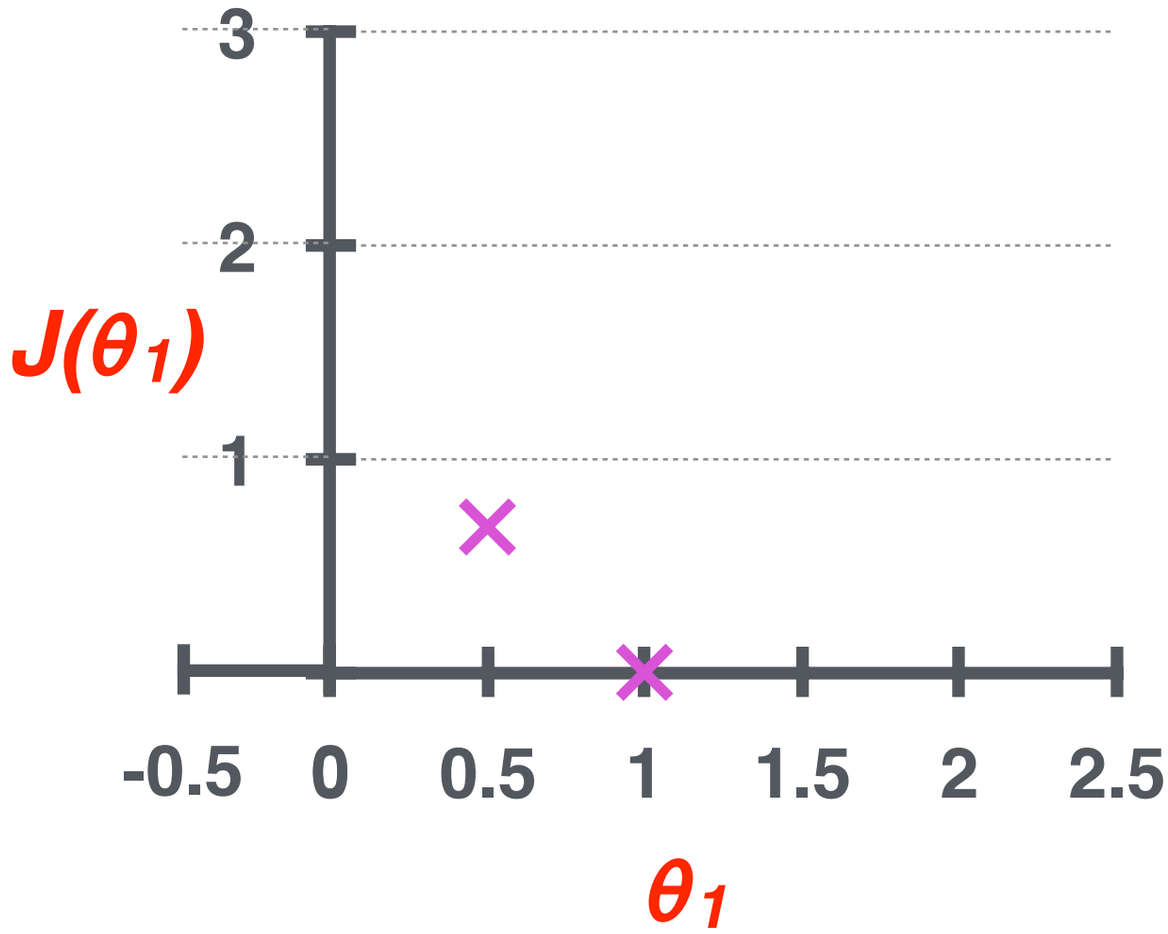
$h_\theta(x)$

(for fixed θ_1, this is a function of x)



$J(\theta_1)$

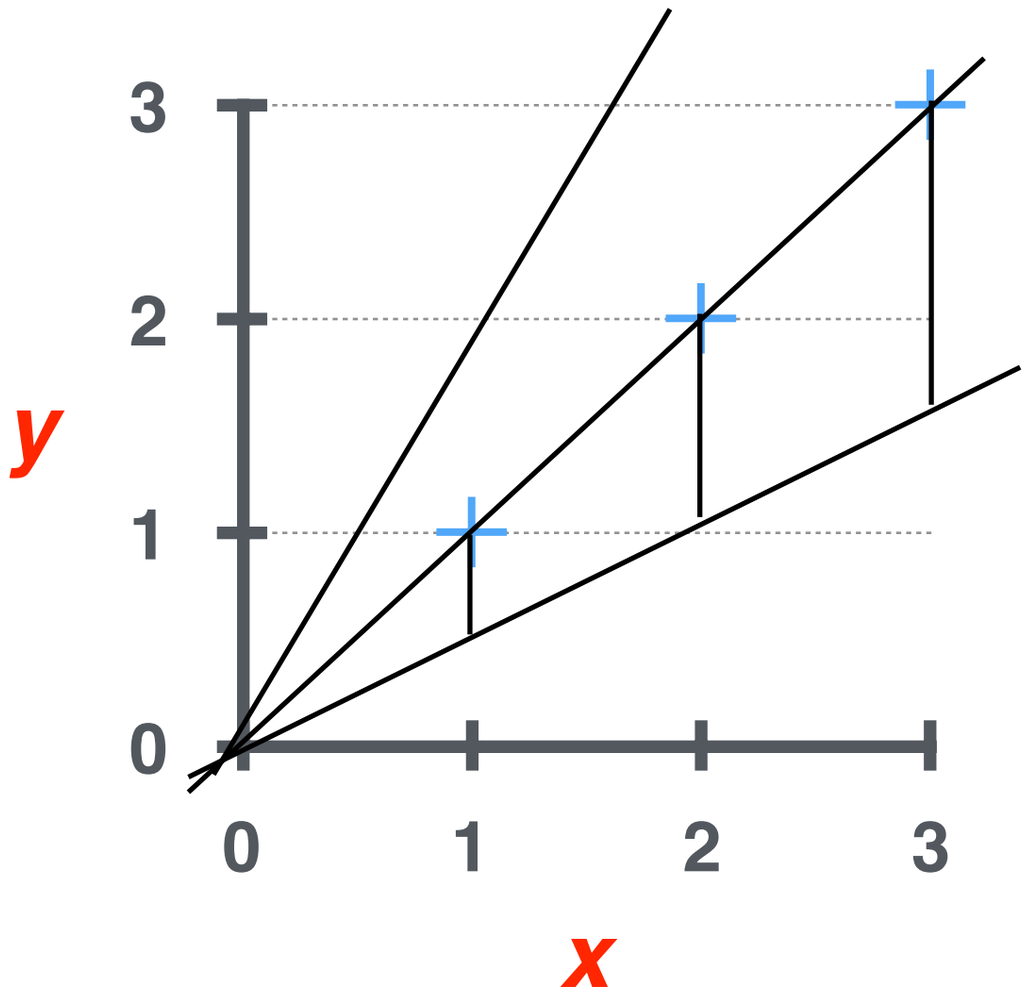
(function of the parameter θ_1)



$$J(0.5) = \frac{1}{2 \times 3}[(0.5-1)^2 + (1-2)^2 + (1.5-3)^2] = 0.68$$

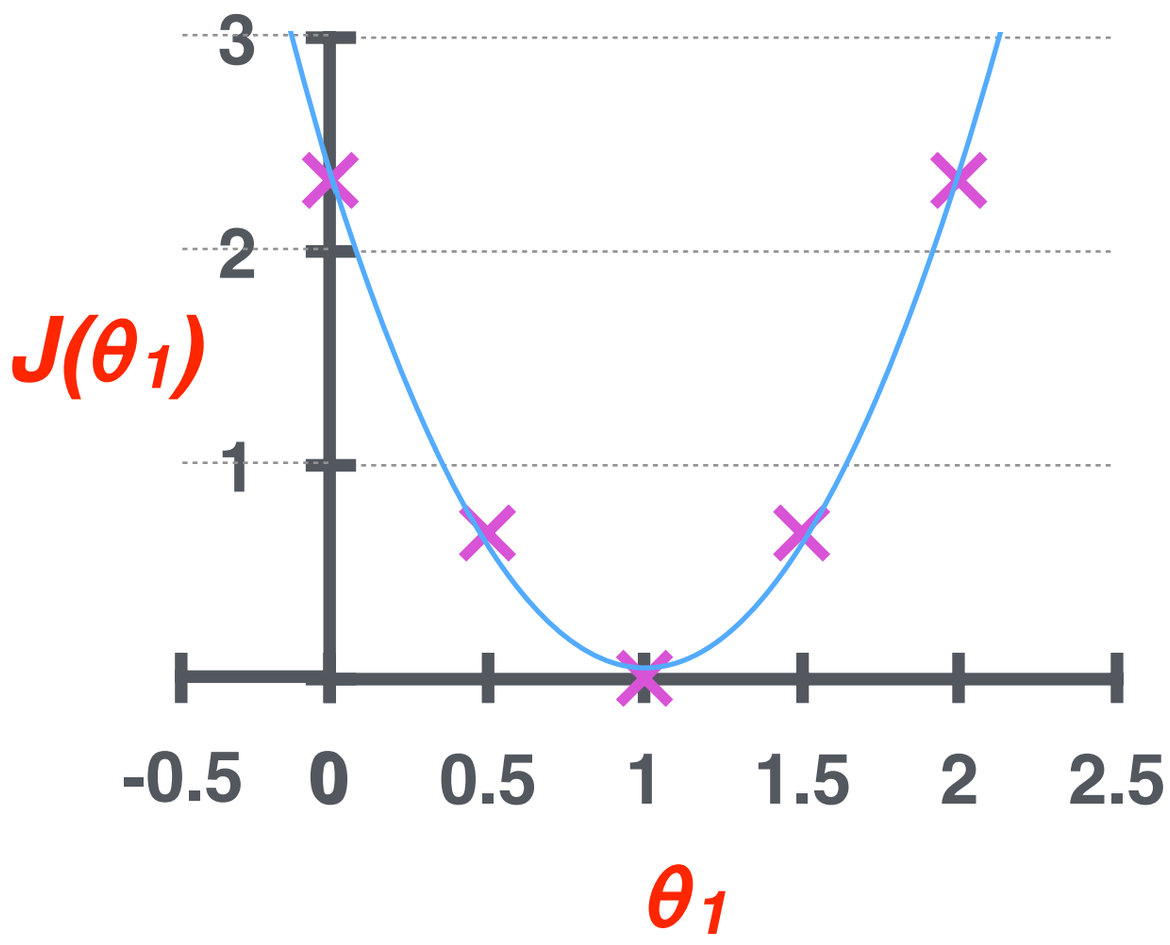
$h_\theta(x)$

(for fixed θ_1, this is a function of x)



$J(\theta_1)$

(function of the parameter θ_1)



$$\underset{\theta_1}{\text{minimize}}\, J(\theta_1)$$

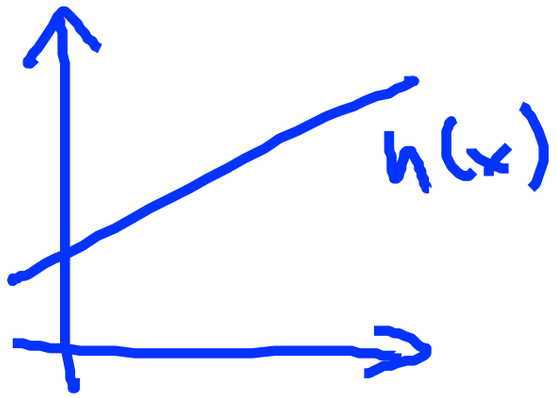
Linear Regression

- **Hypothesis**:

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

- **Parameters:**

θ_0, θ_1



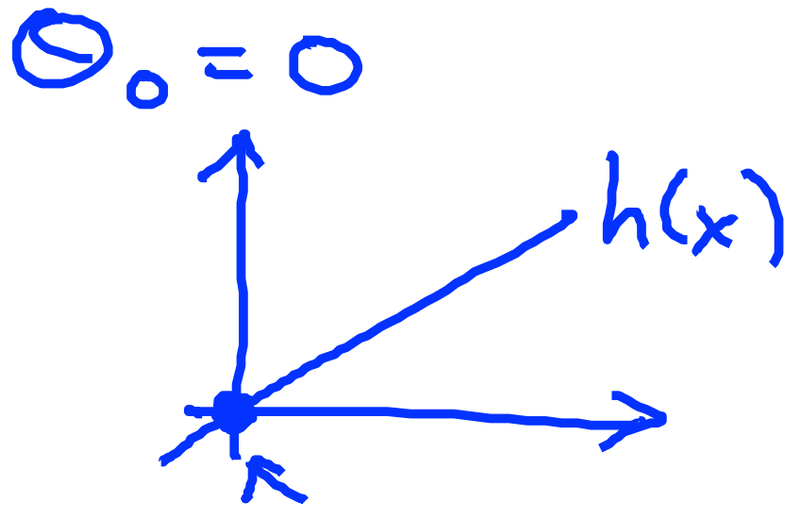
- **Cost Function:**

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

- **Goal:** $\underset{\theta_0, \theta_1}{\text{minimize}}\, J(\theta_0, \theta_1)$

Simplified

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



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

$$\underset{\theta_1}{\text{minimize}}\, J(\theta_1)$$

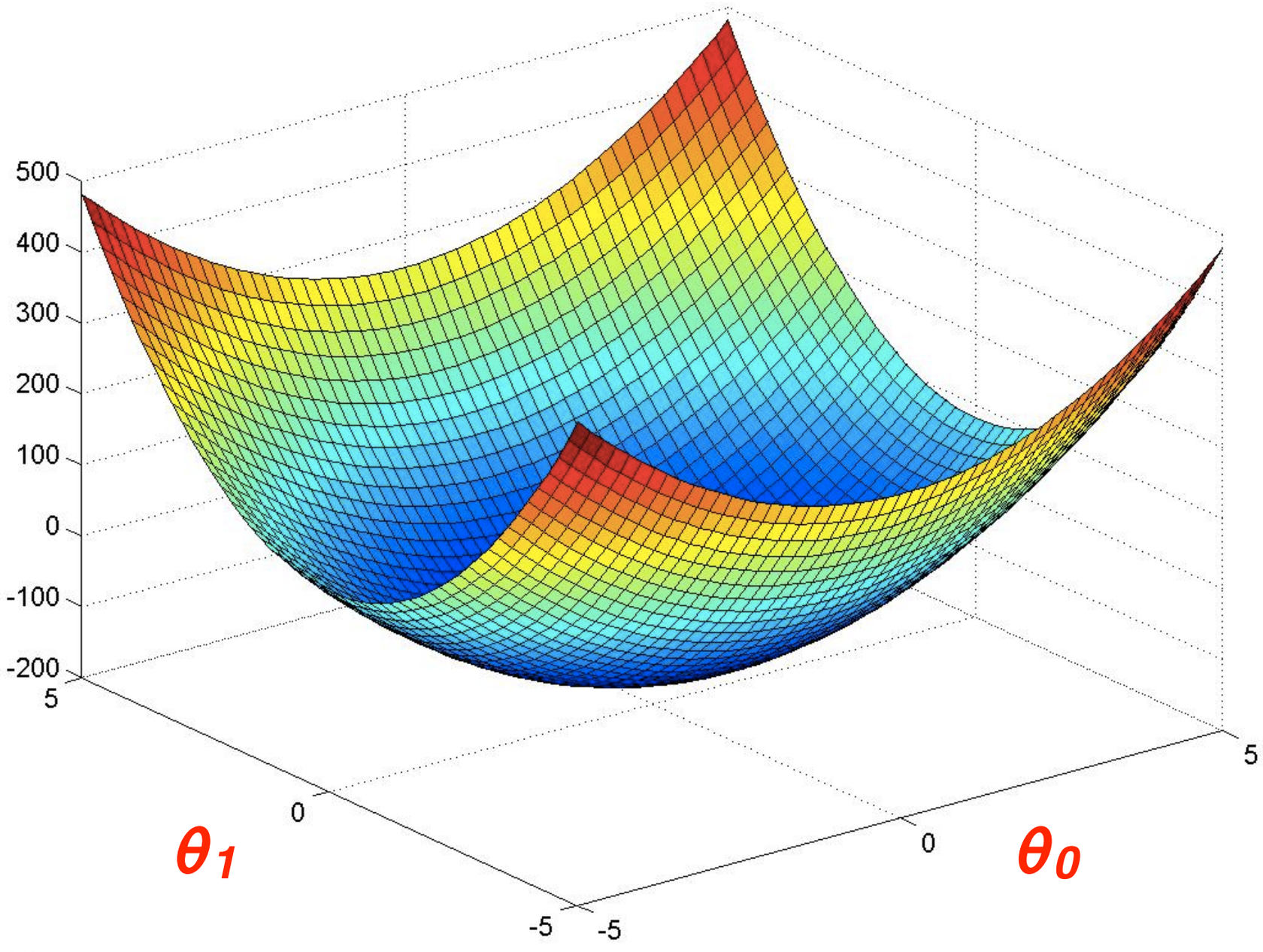
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)

$J(\theta_0, \theta_1)$

(function of the parameter θ_0, θ_1)

$J(\theta_1, \theta_2)$

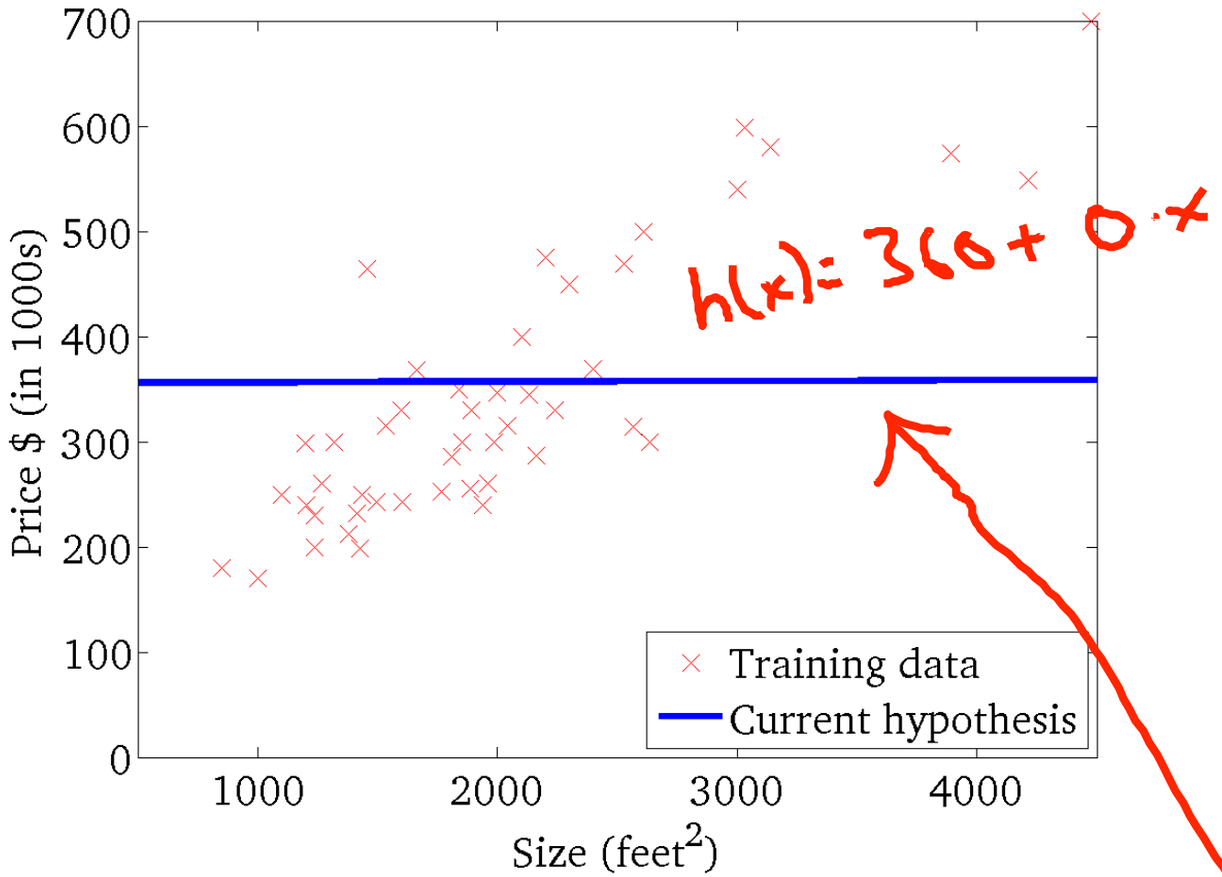


$h_\theta(x)$

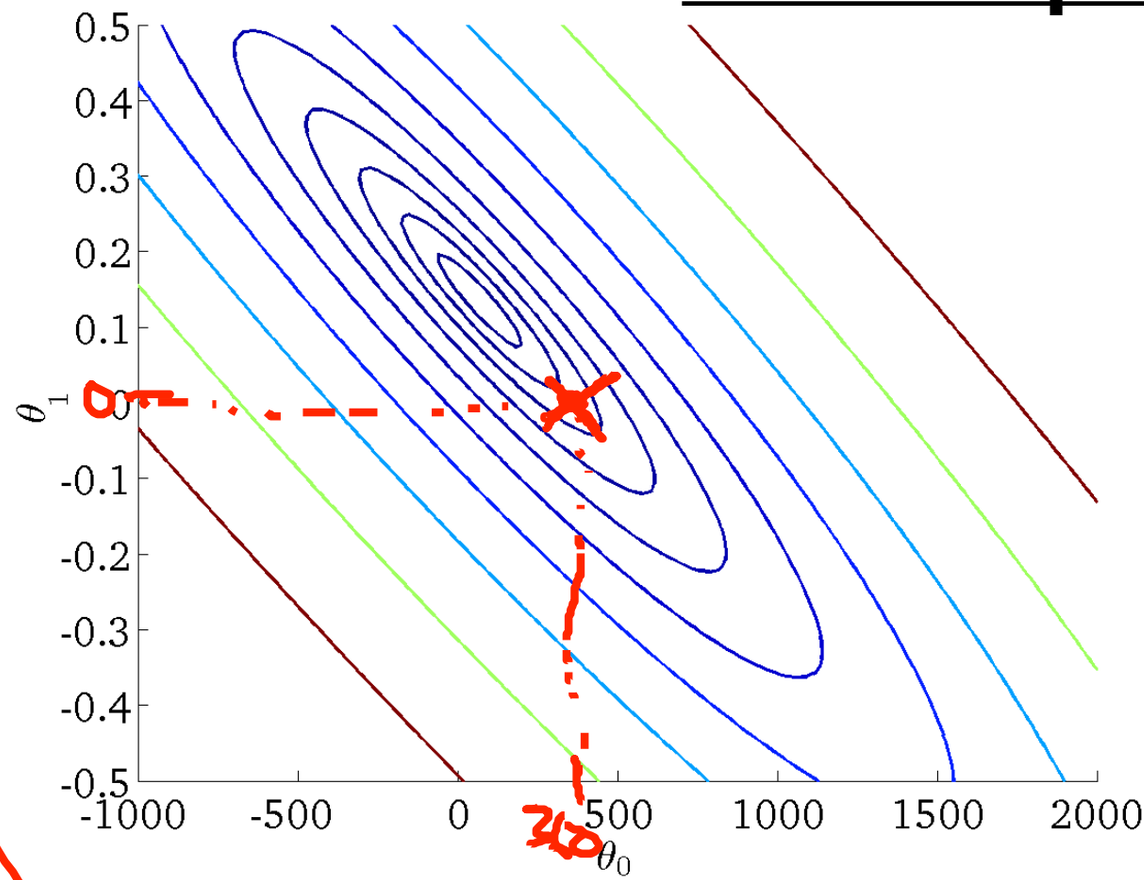
(for fixed θ_0, θ_1, this is a function of x)

$J(\theta_0, \theta_1)$

(function of the parameter θ_0, θ_1)



contour plot



$\theta_0 = 360$

$\theta_1 = 0$

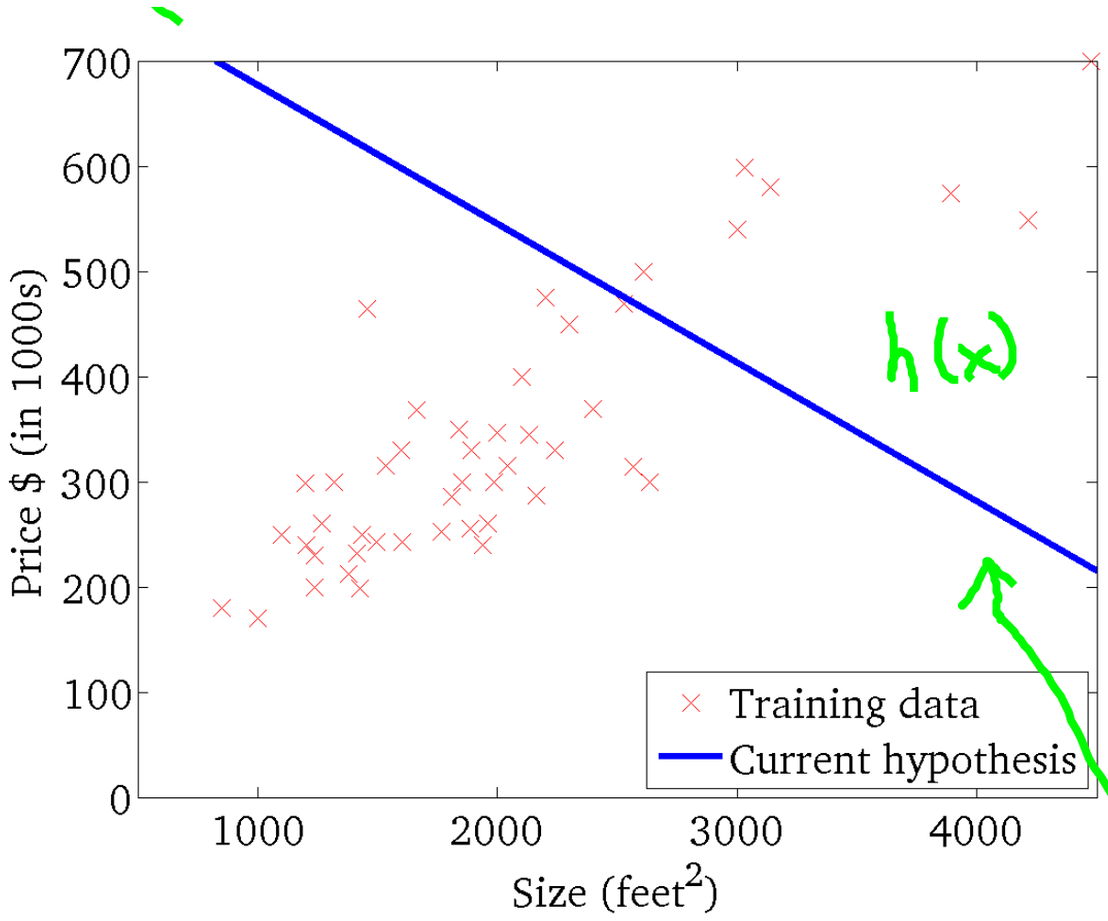
Andrew Ng

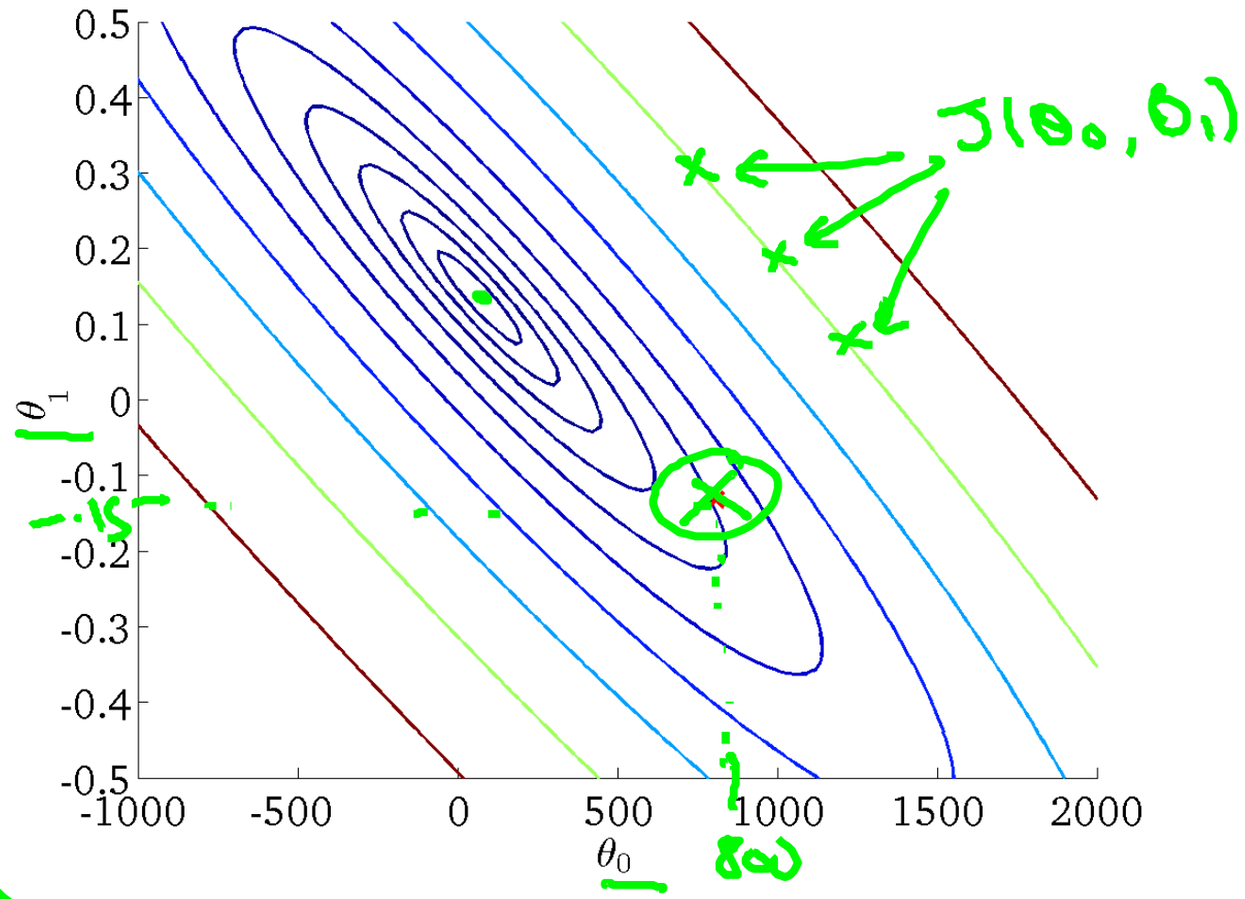
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)

$J(\theta_0, \theta_1)$

(function of the parameter θ_0, θ_1)





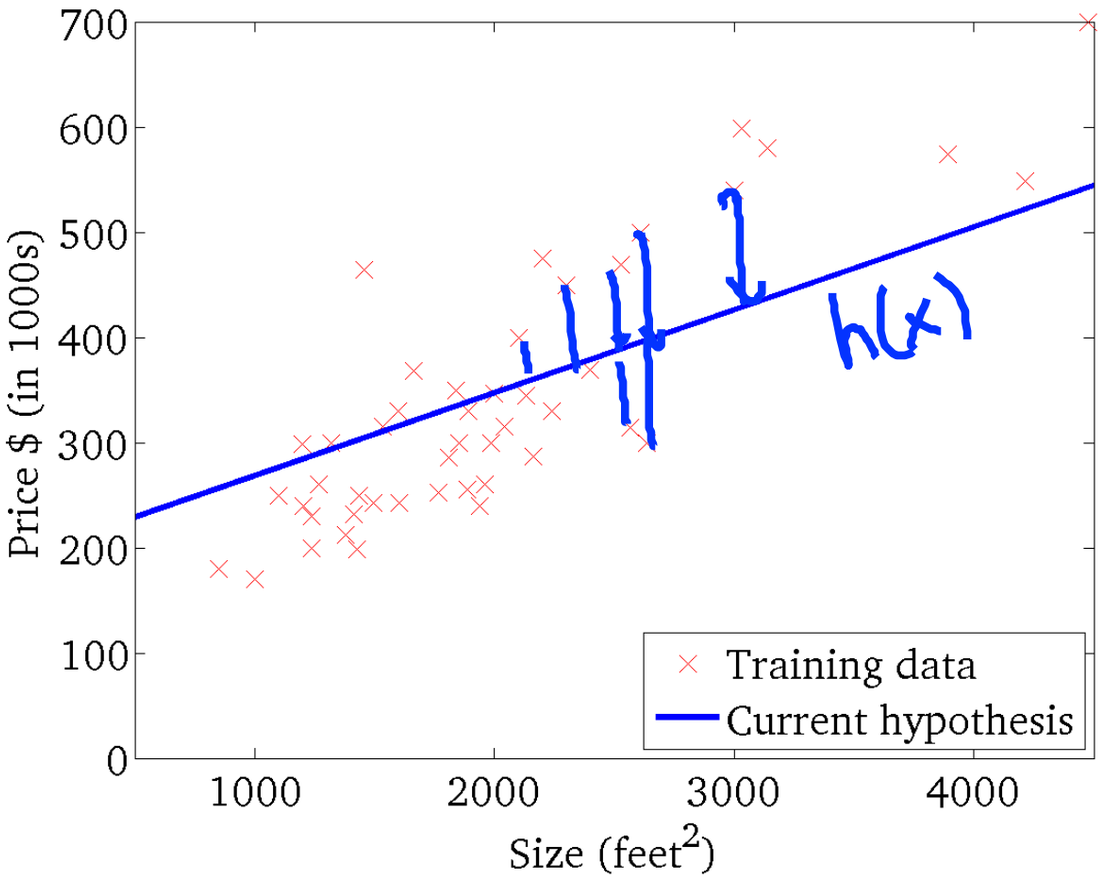
Andrew Ng

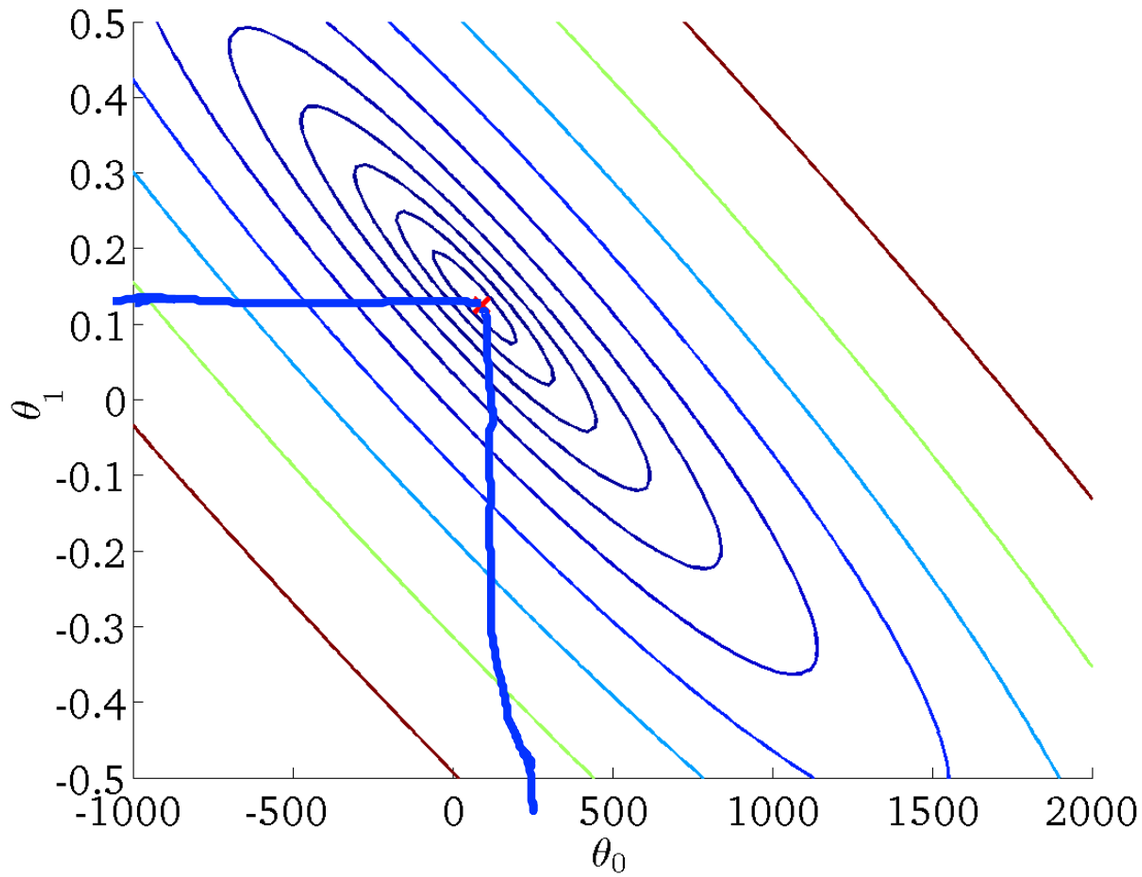
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)

$J(\theta_0, \theta_1)$

(function of the parameter θ_0, θ_1)





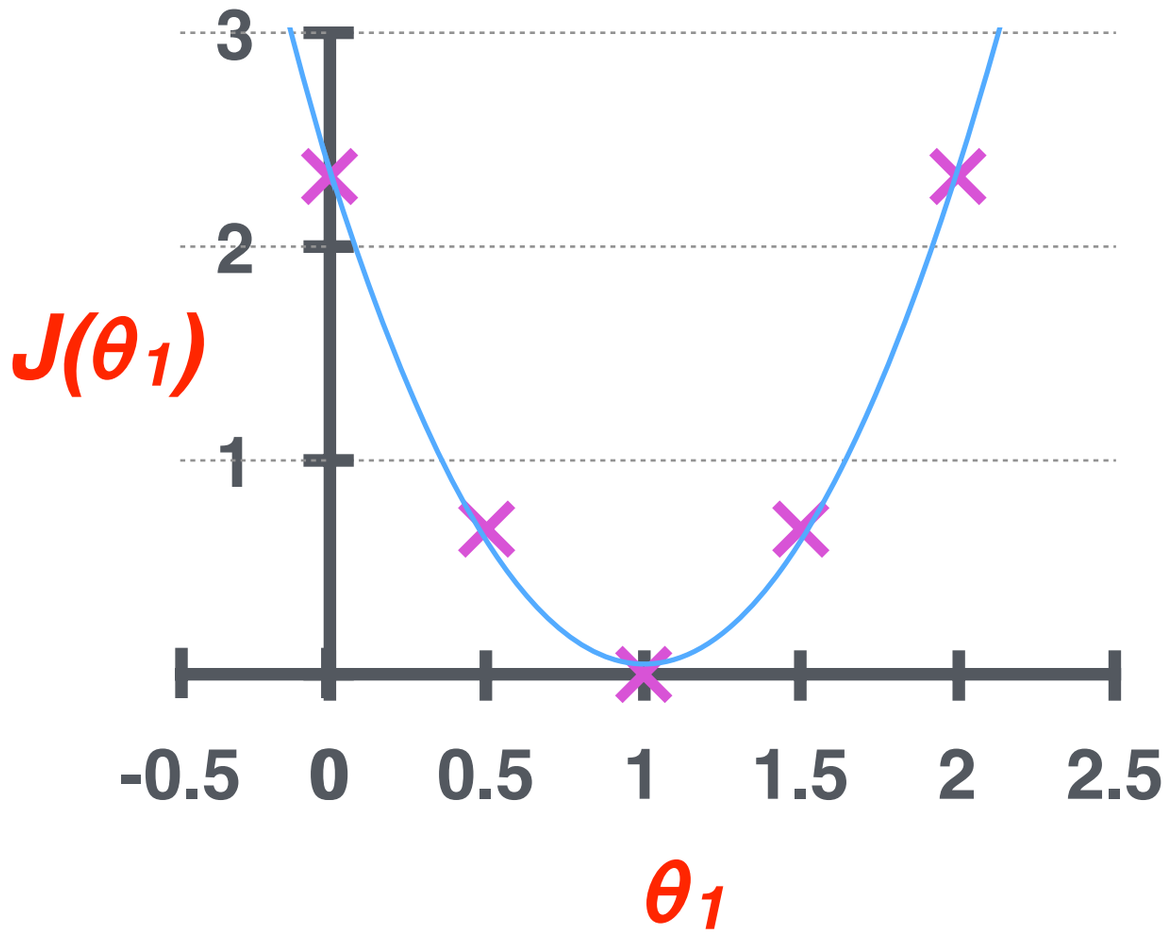
Andrew Ng

Parameter Learning

- 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
  - Keep changing θ_0, θ_1 to reduce $J(\theta_1, \theta_2)$ until we hopefully end up at a minimum

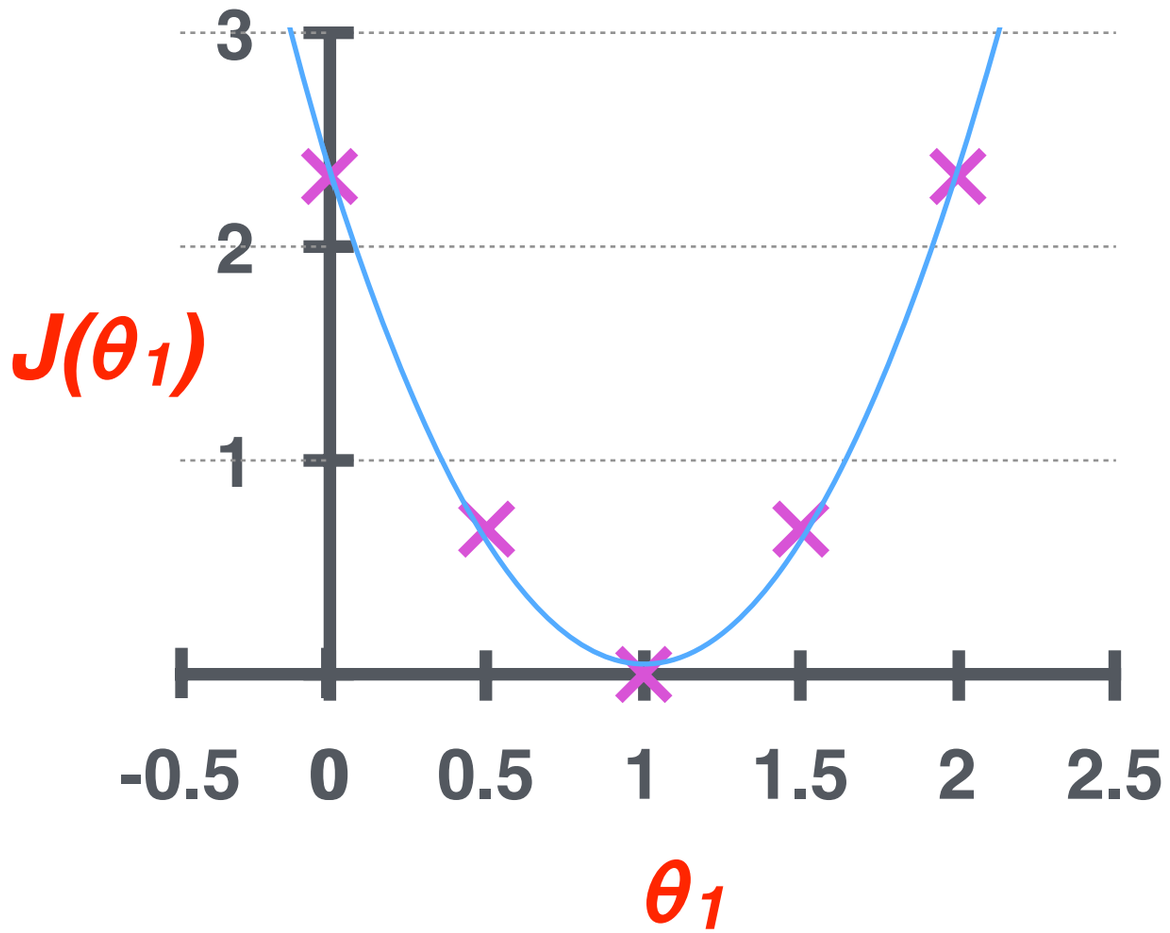
Gradient Descent



Simplified

$$\underset{\theta_1}{\text{minimize}}\, J(\theta_1)$$

Gradient Descent



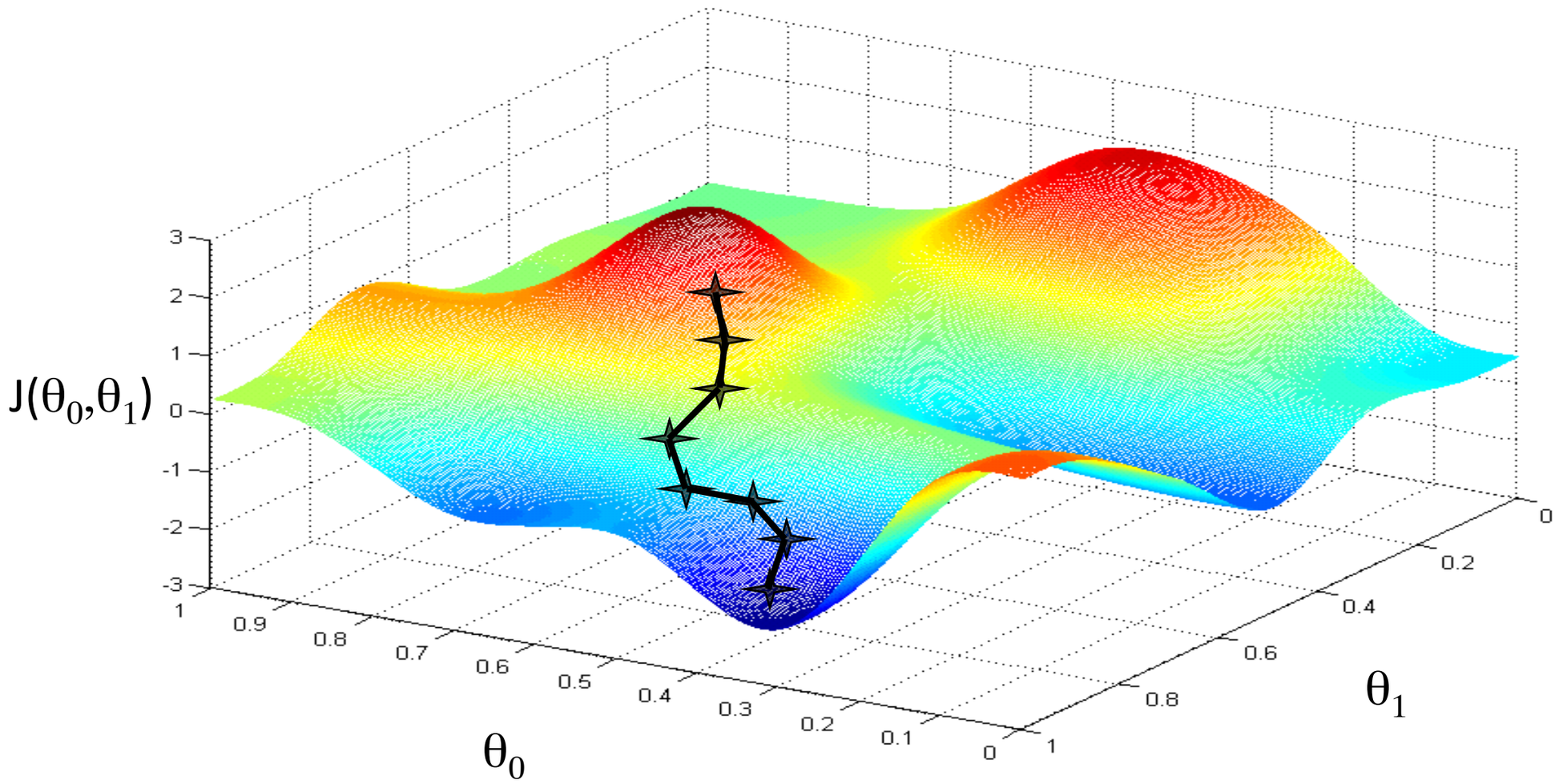
Simplified

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

learning rate

$$\underset{\theta_1}{\text{minimize}}\, J(\theta_1)$$

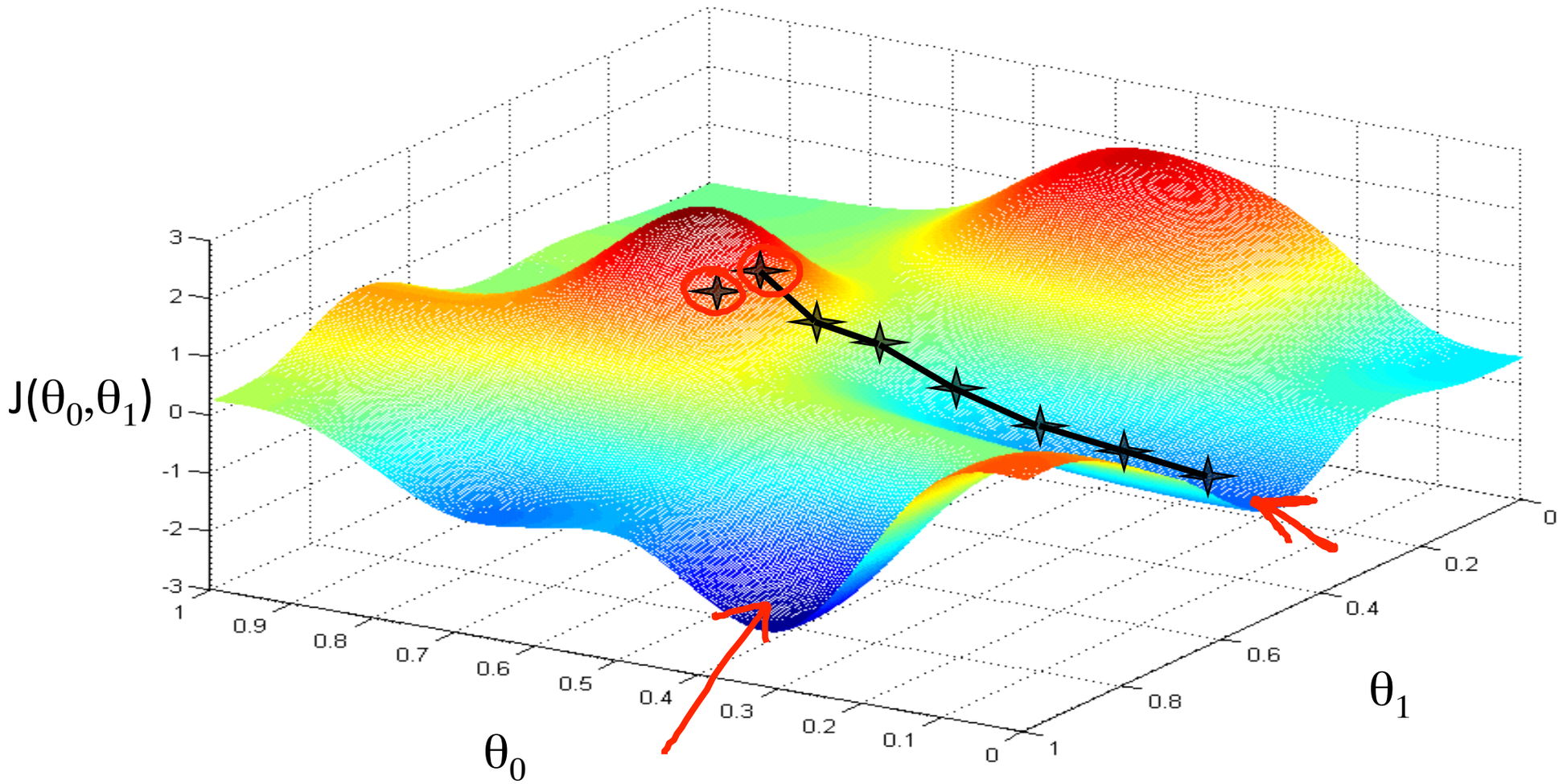
Gradient Descent



$$\underset{\theta_0,\theta_1}{\text{minimize}}\, J(\theta_0,\theta_1)$$

Andrew Ng

Gradient Descent



$$\underset{\theta_0,\theta_1}{\text{minimize}}\, J(\theta_0,\theta_1)$$

Andrew Ng

Gradient Descent

repeat until convergence {

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

simultaneous update for j=0 and j=1

}

learning rate

Linear Regression w/ one variable:

Gradient Descent

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

Cost Function

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

$$\frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1) = ?$$

$$\frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1) = ?$$

Linear Regression w/ one variable:

Gradient Descent

repeat until convergence {

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})$$

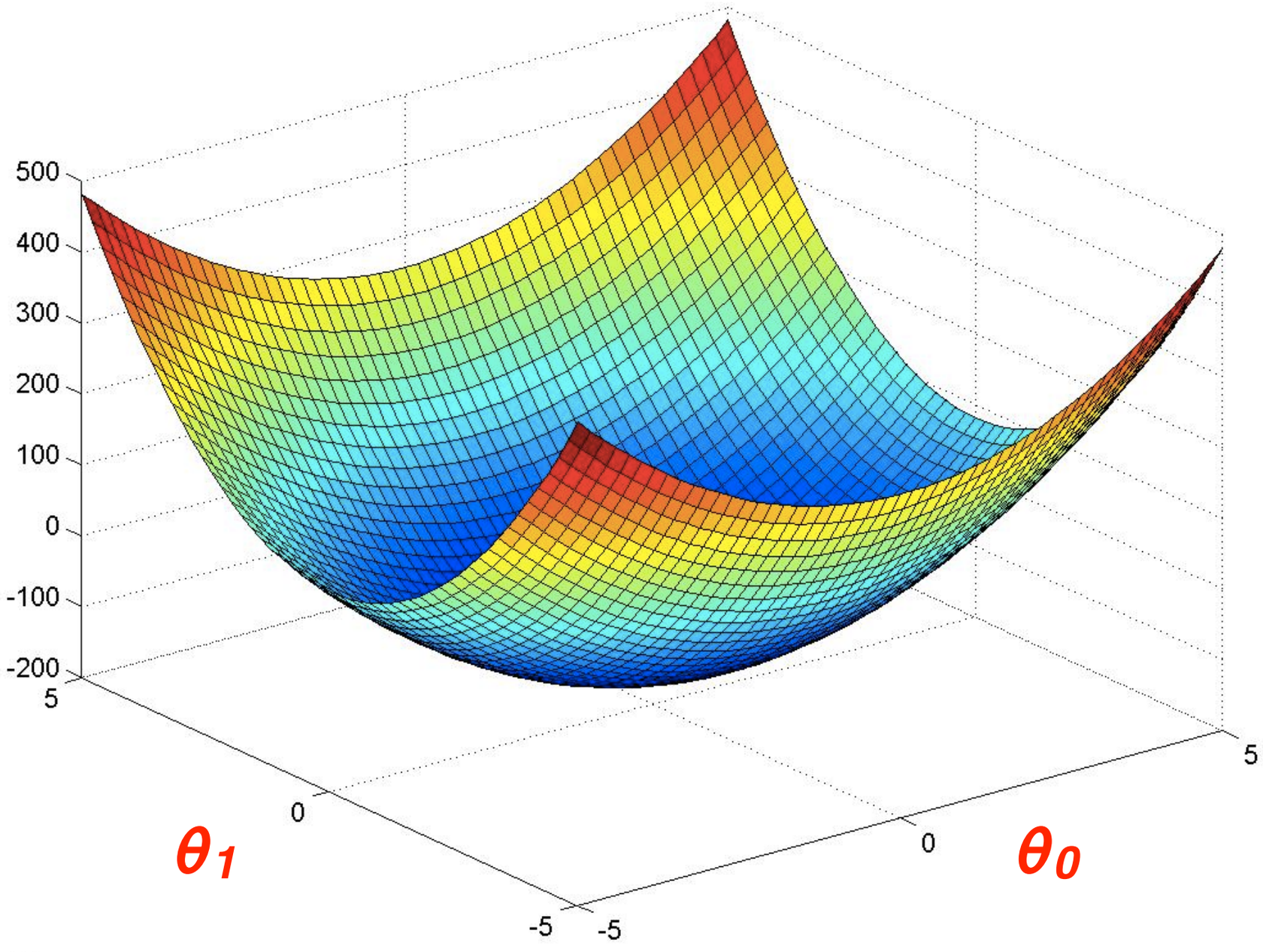
$$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) \cdot x^{(i)}$$

}

simultaneous update θ_0, θ_1

Linear Regression

cost function is convex



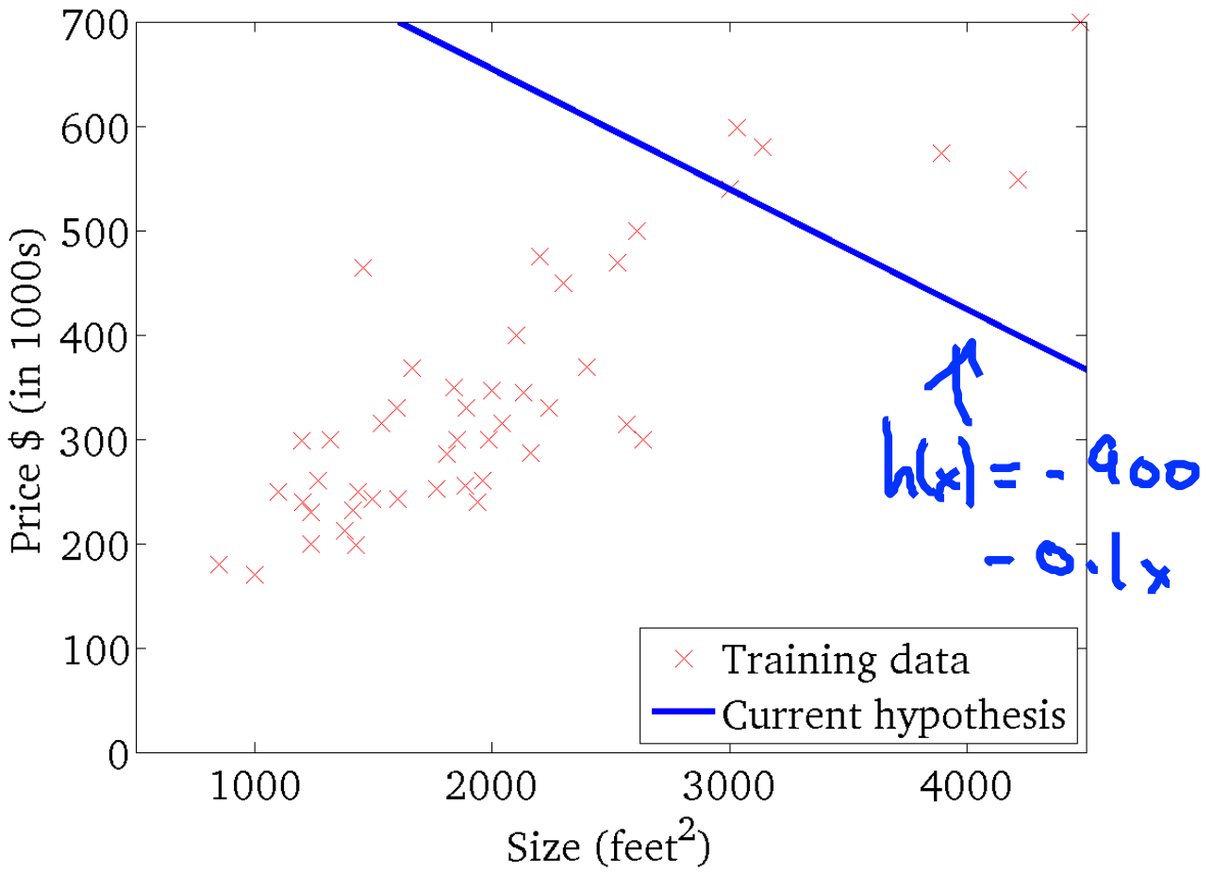
$J(\theta_1, \theta_2)$

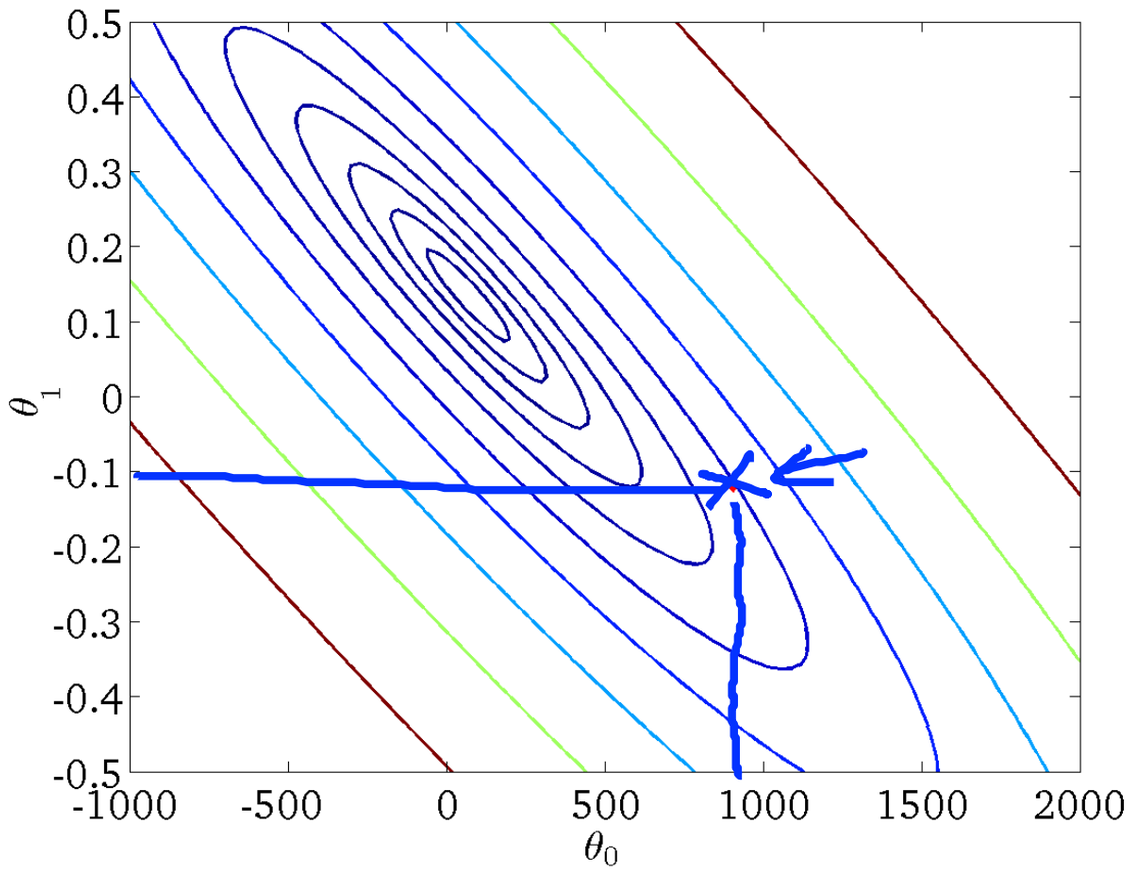
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)

$J(\theta_0, \theta_1)$

(function of the parameter θ_0, θ_1)



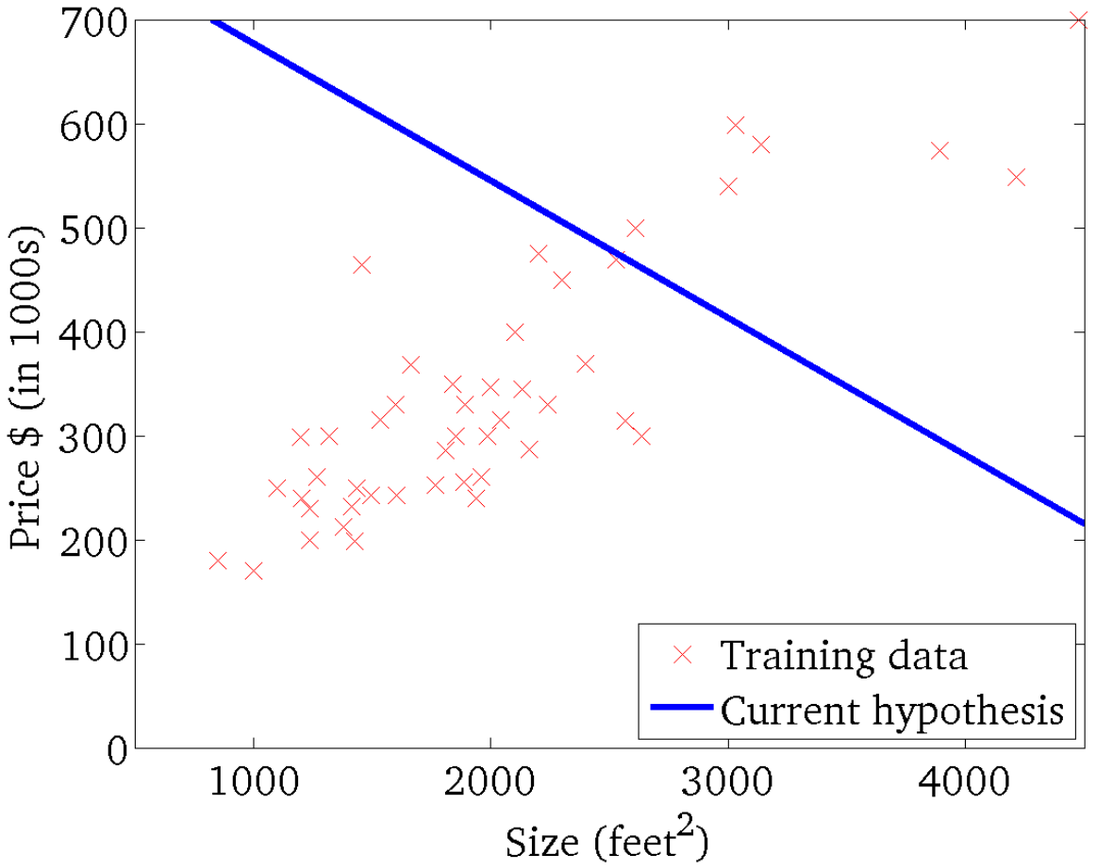


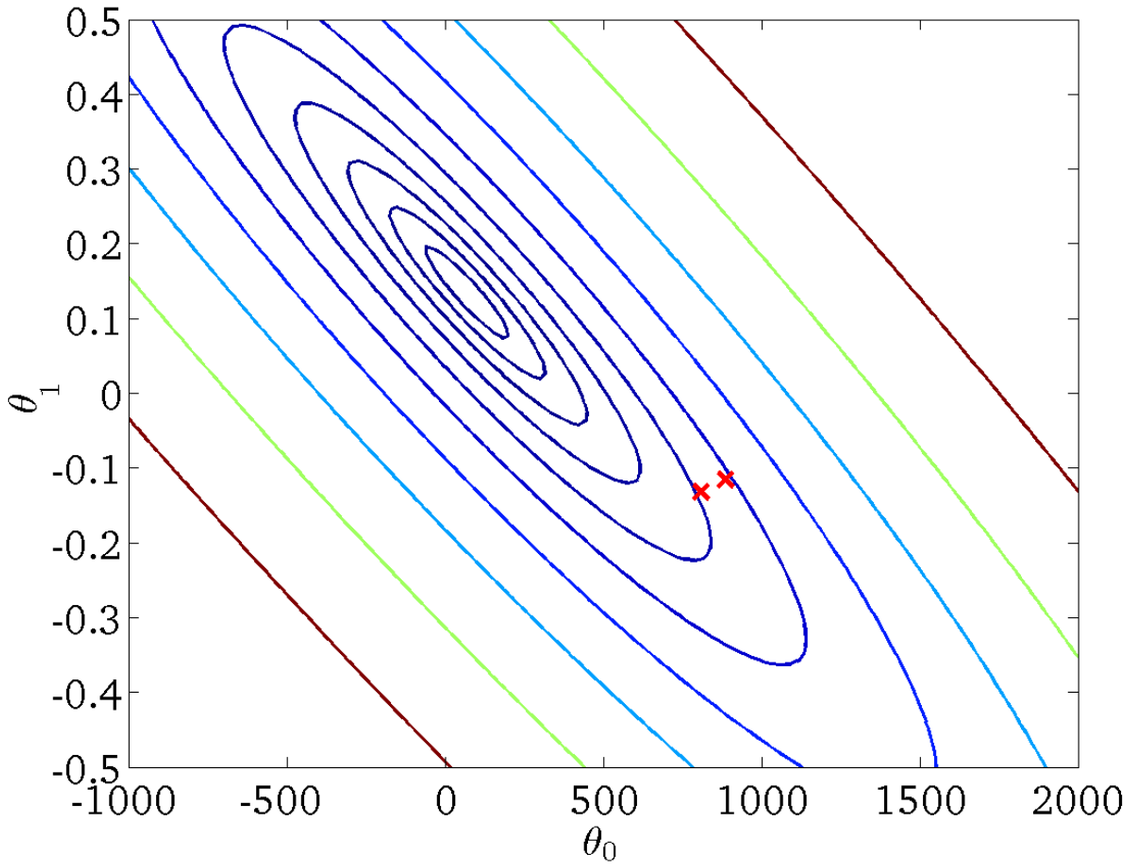
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)

$J(\theta_0, \theta_1)$

(function of the parameter θ_0, θ_1)



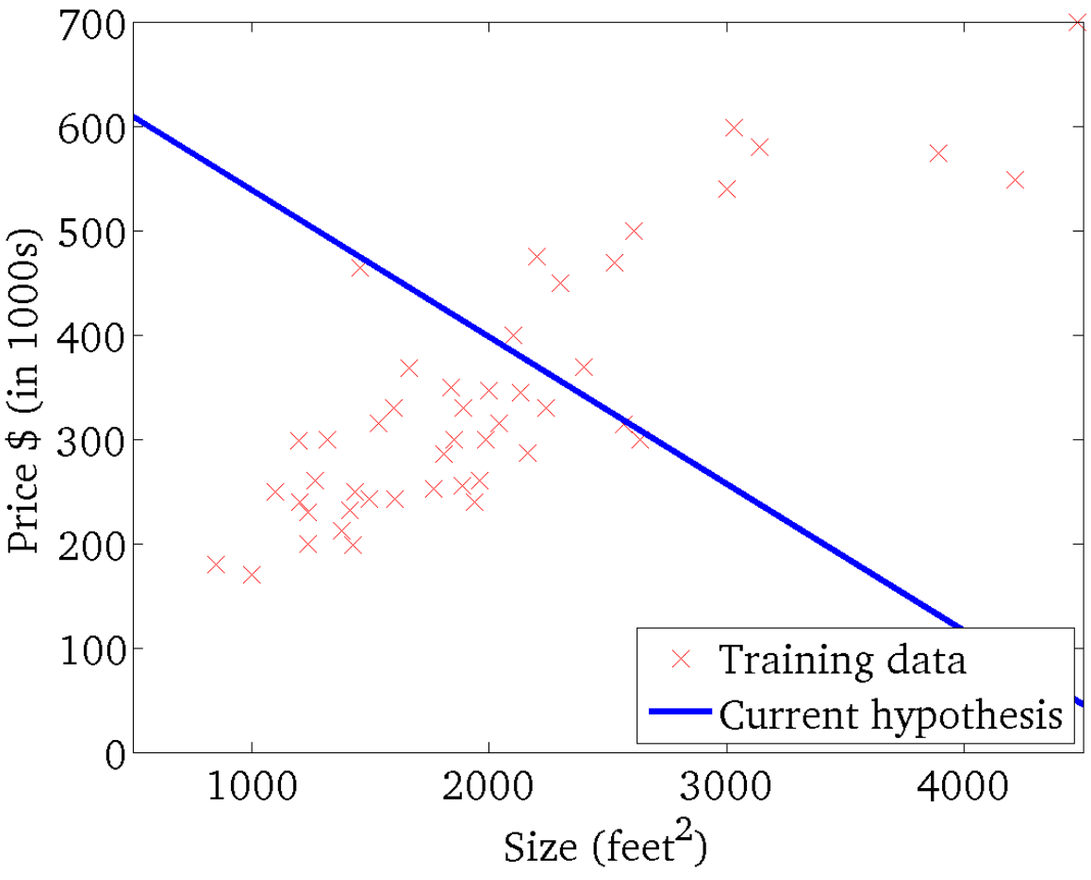


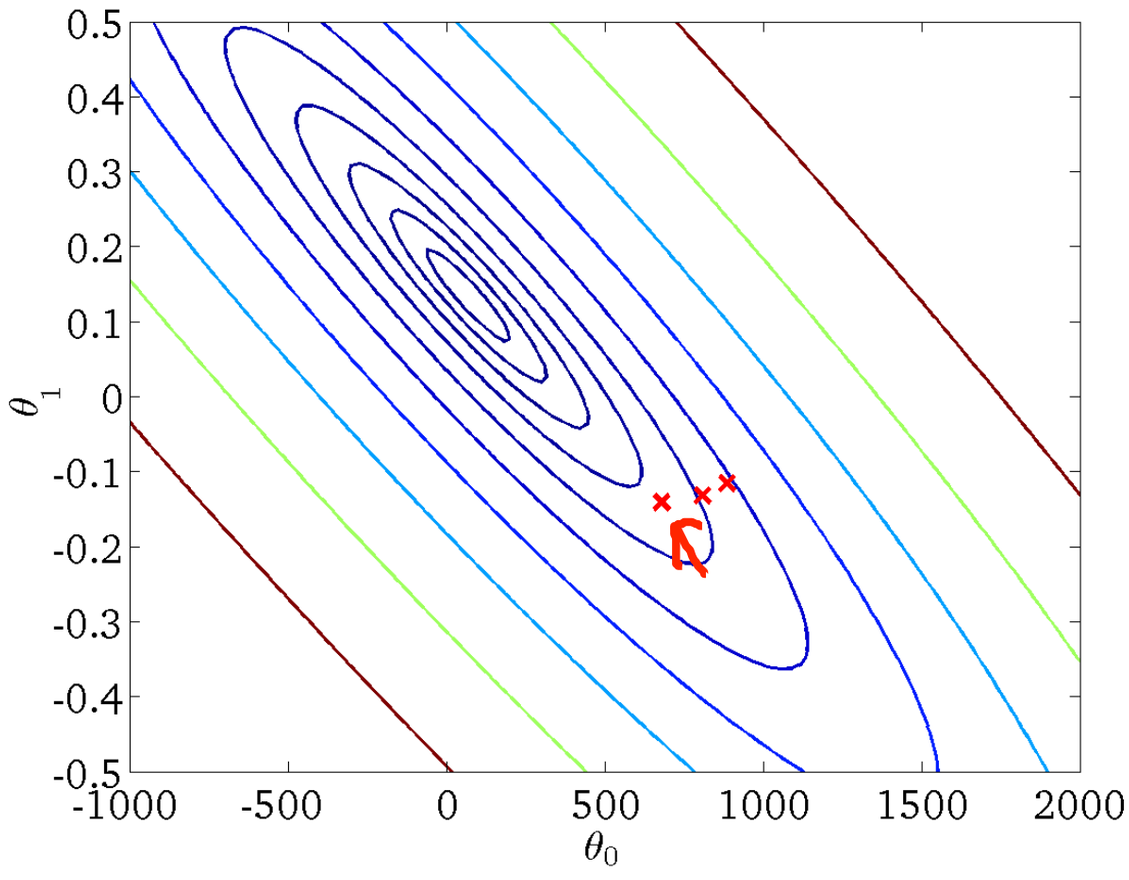
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)

$J(\theta_0, \theta_1)$

(function of the parameter θ_0, θ_1)



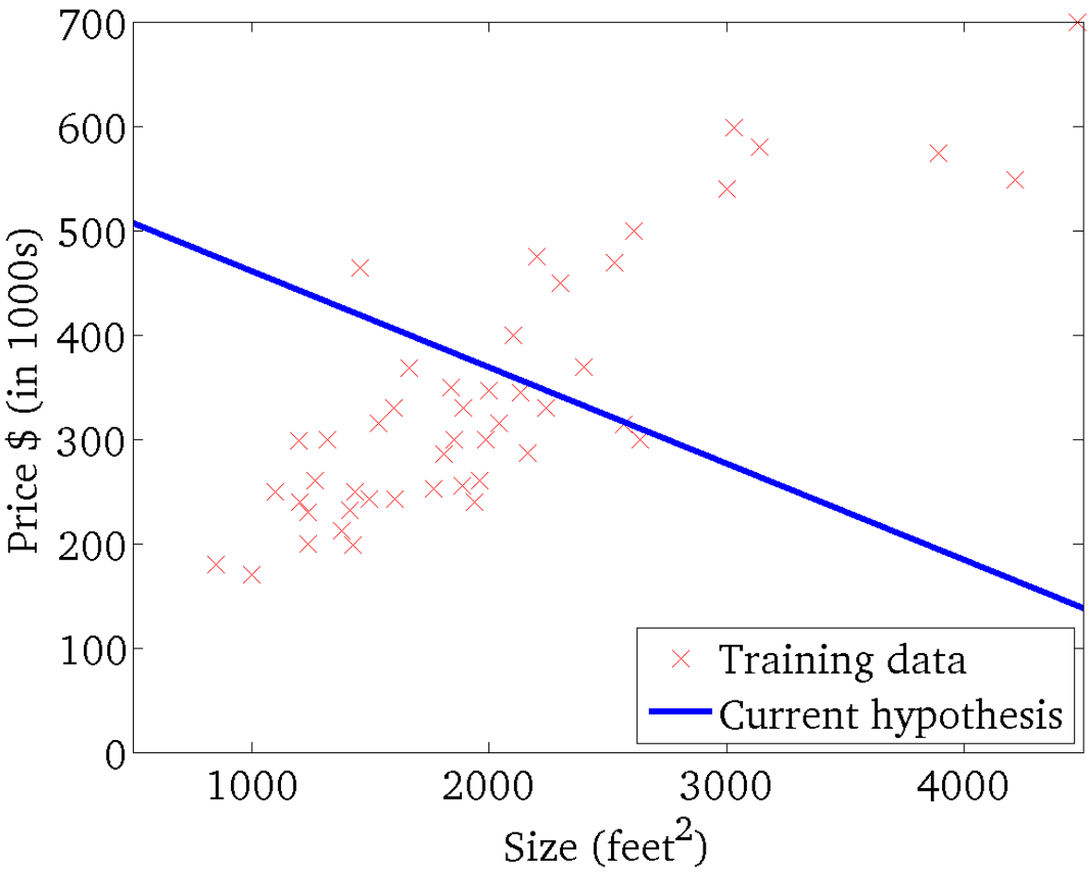


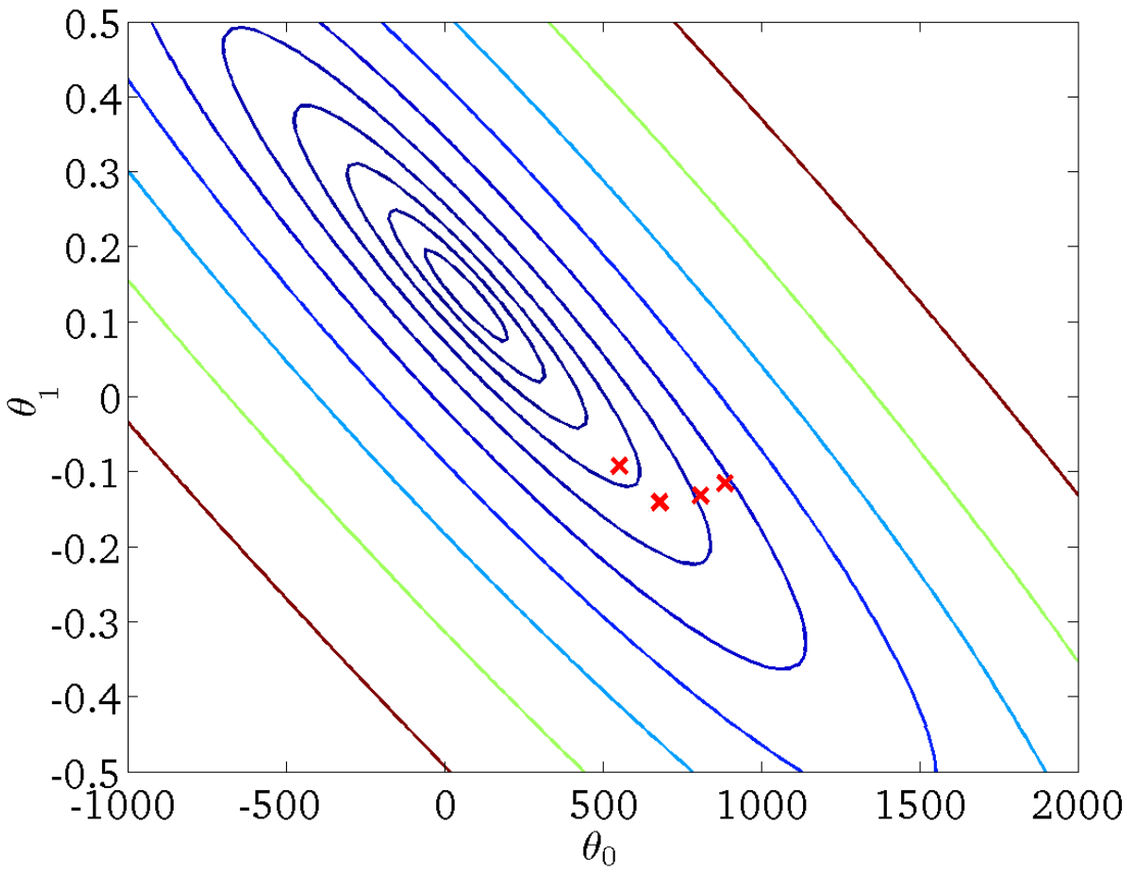
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)

$J(\theta_0, \theta_1)$

(function of the parameter θ_0, θ_1)



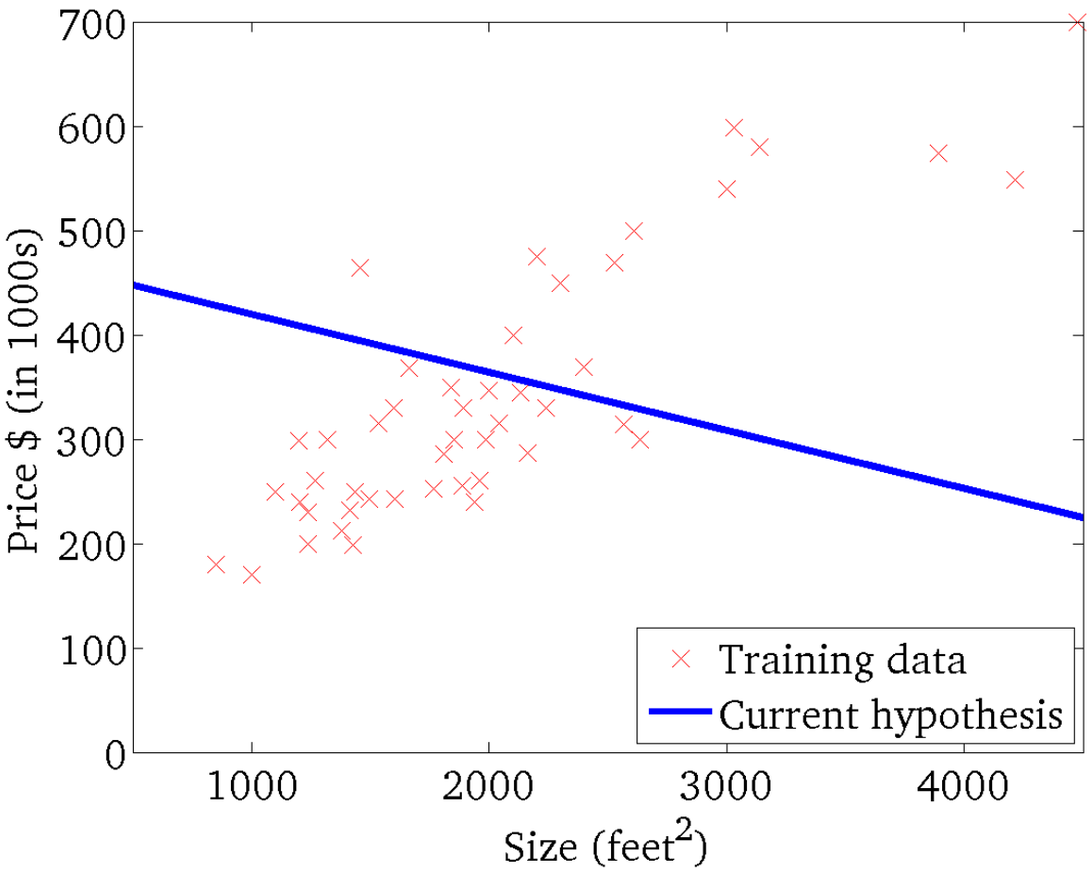


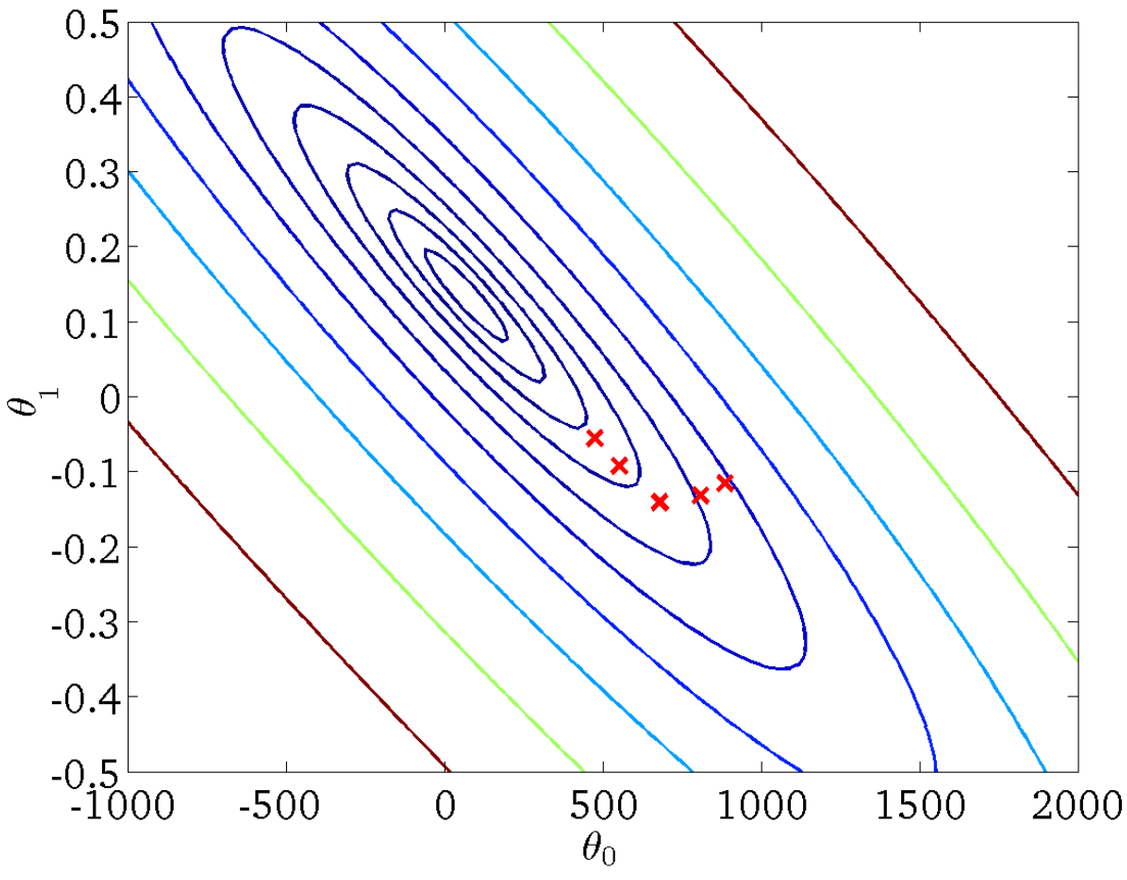
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)

$J(\theta_0, \theta_1)$

(function of the parameter θ_0, θ_1)



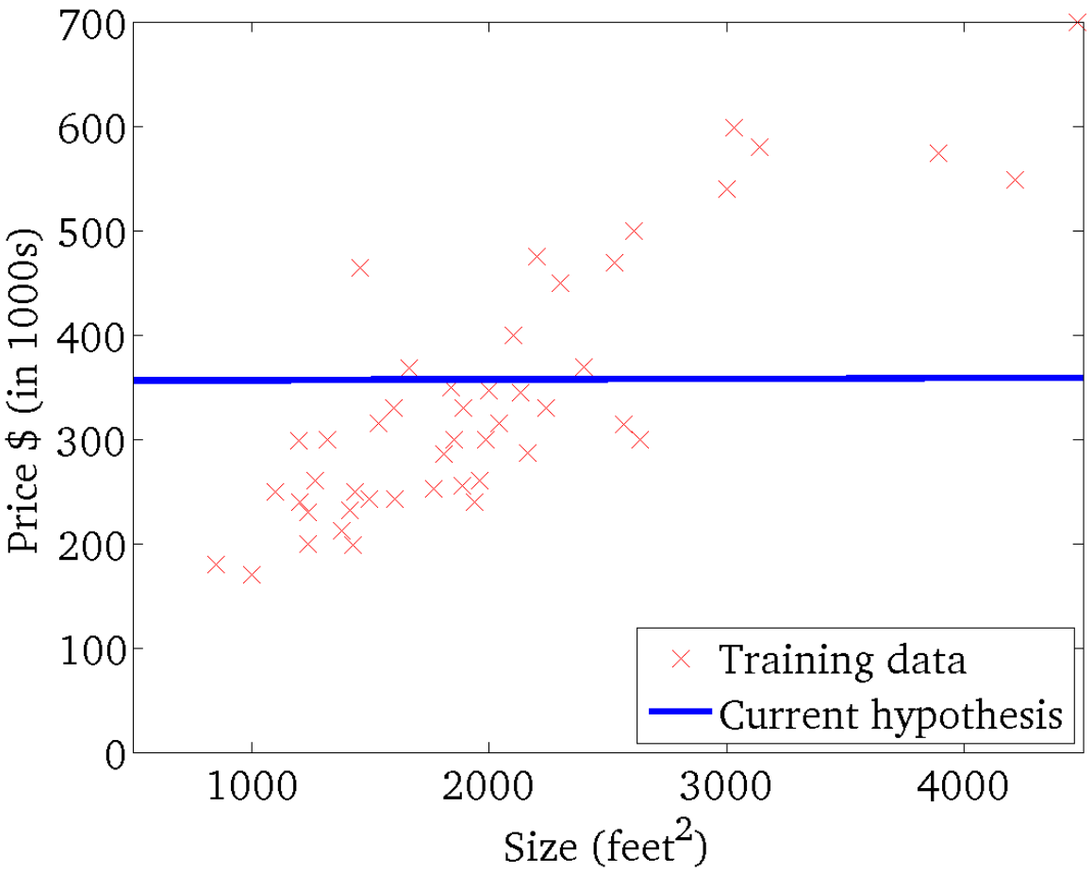


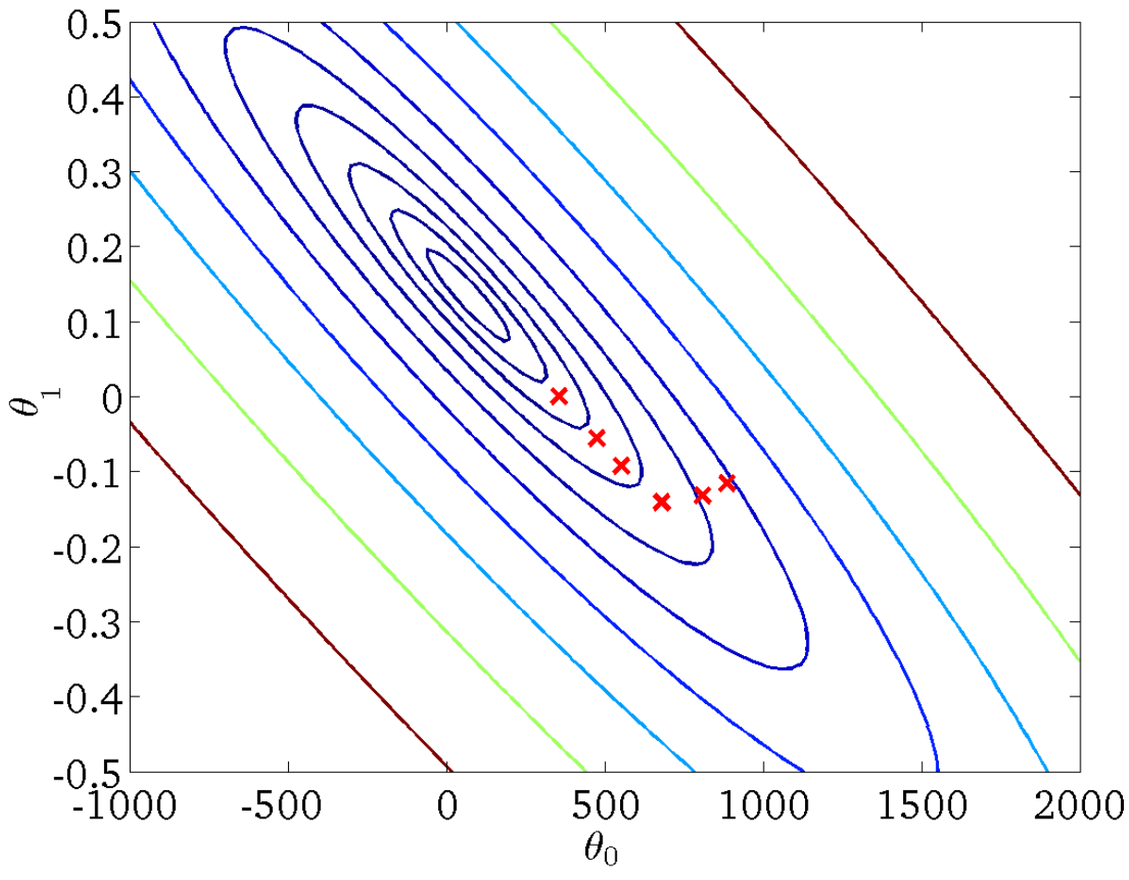
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)

$J(\theta_0, \theta_1)$

(function of the parameter θ_0, θ_1)



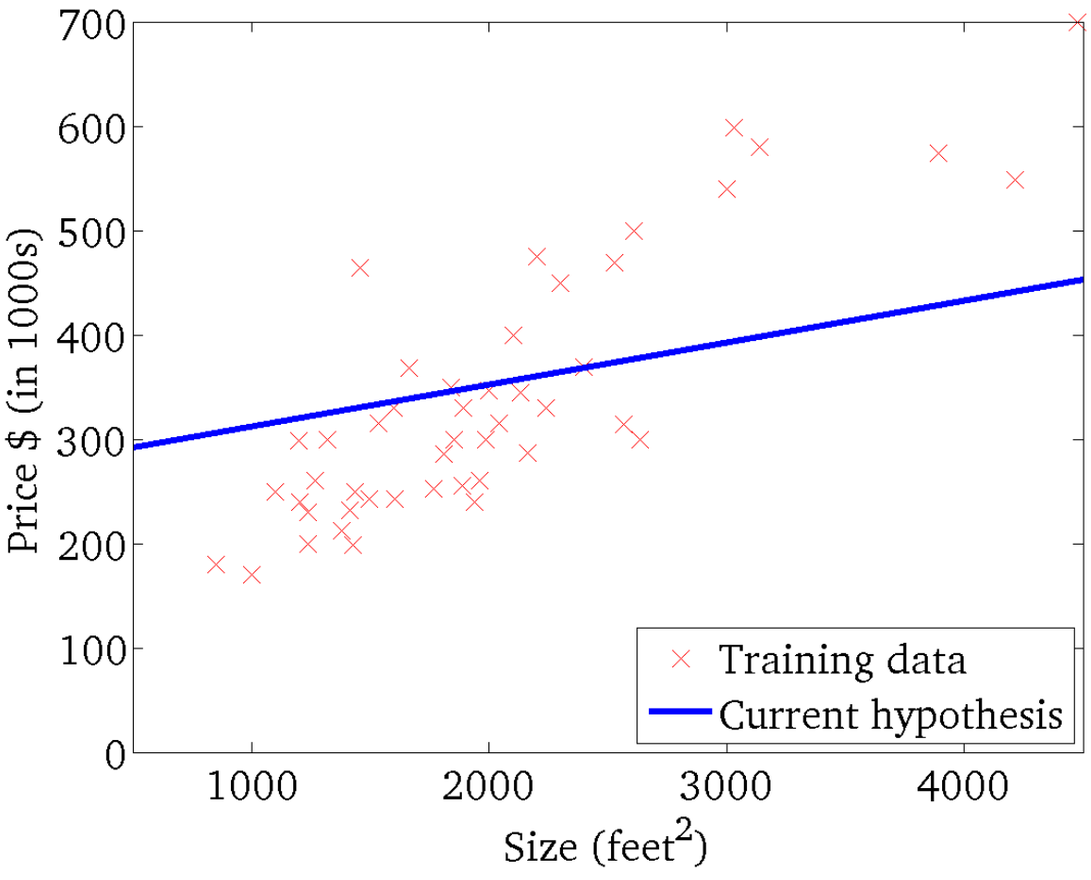


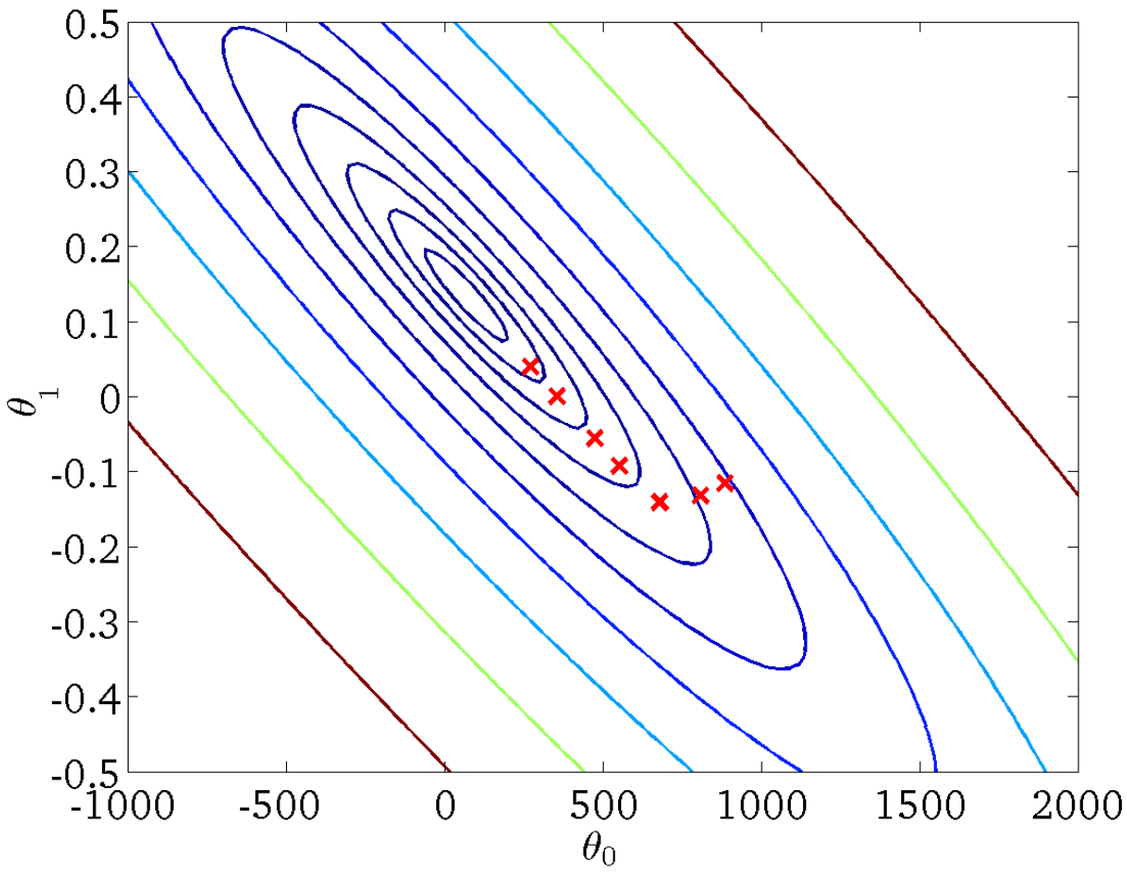
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)

$J(\theta_0, \theta_1)$

(function of the parameter θ_0, θ_1)



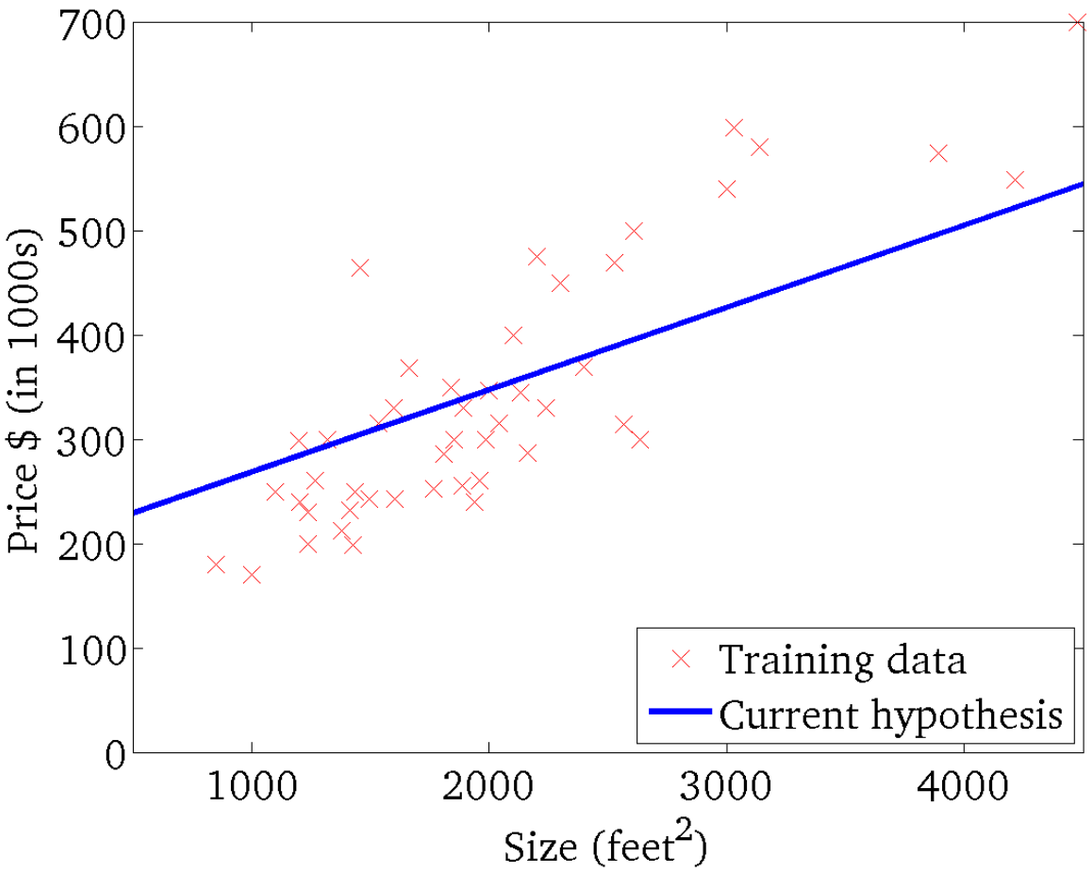


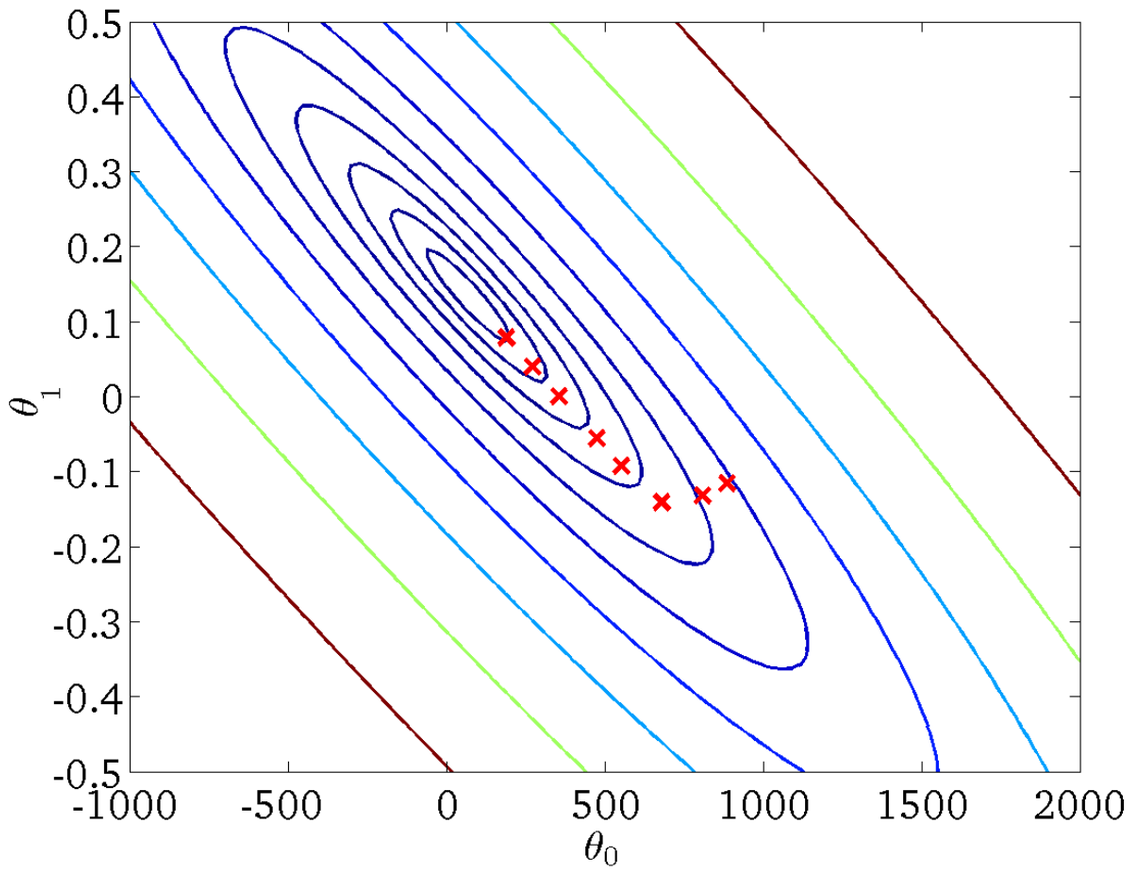
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)

$J(\theta_0, \theta_1)$

(function of the parameter θ_0, θ_1)



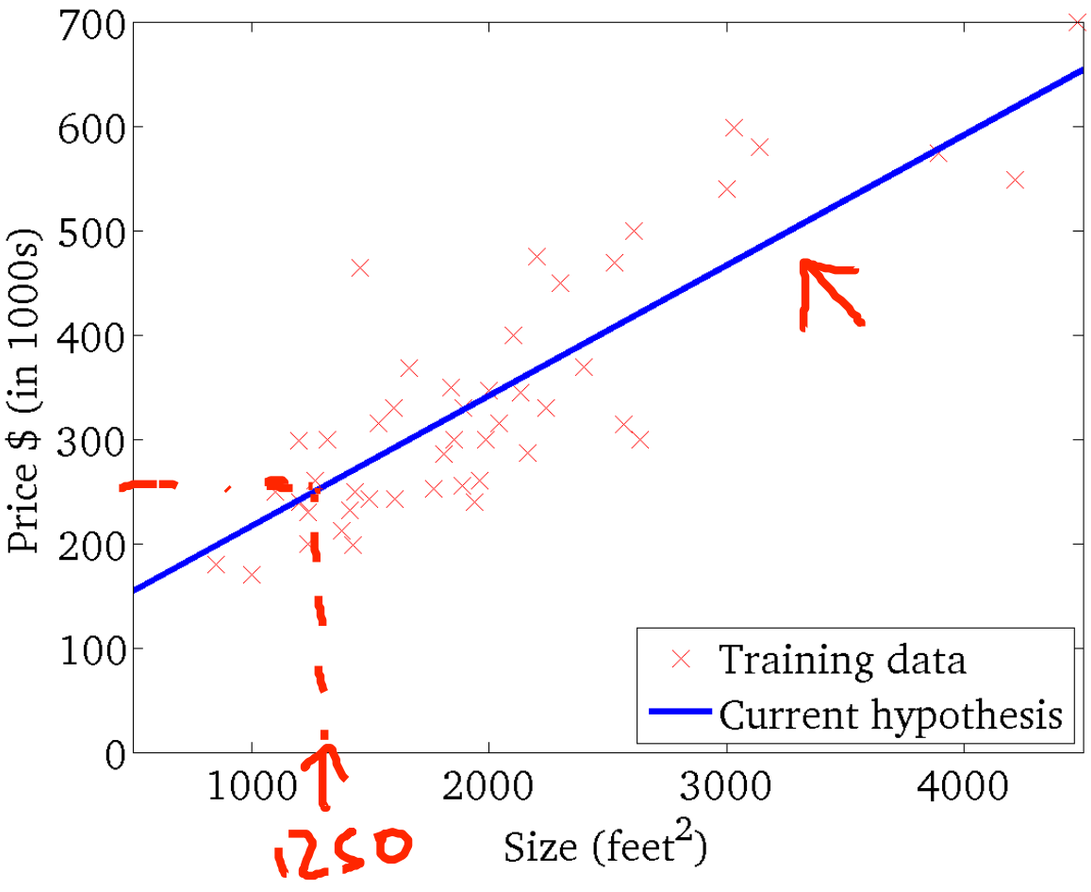


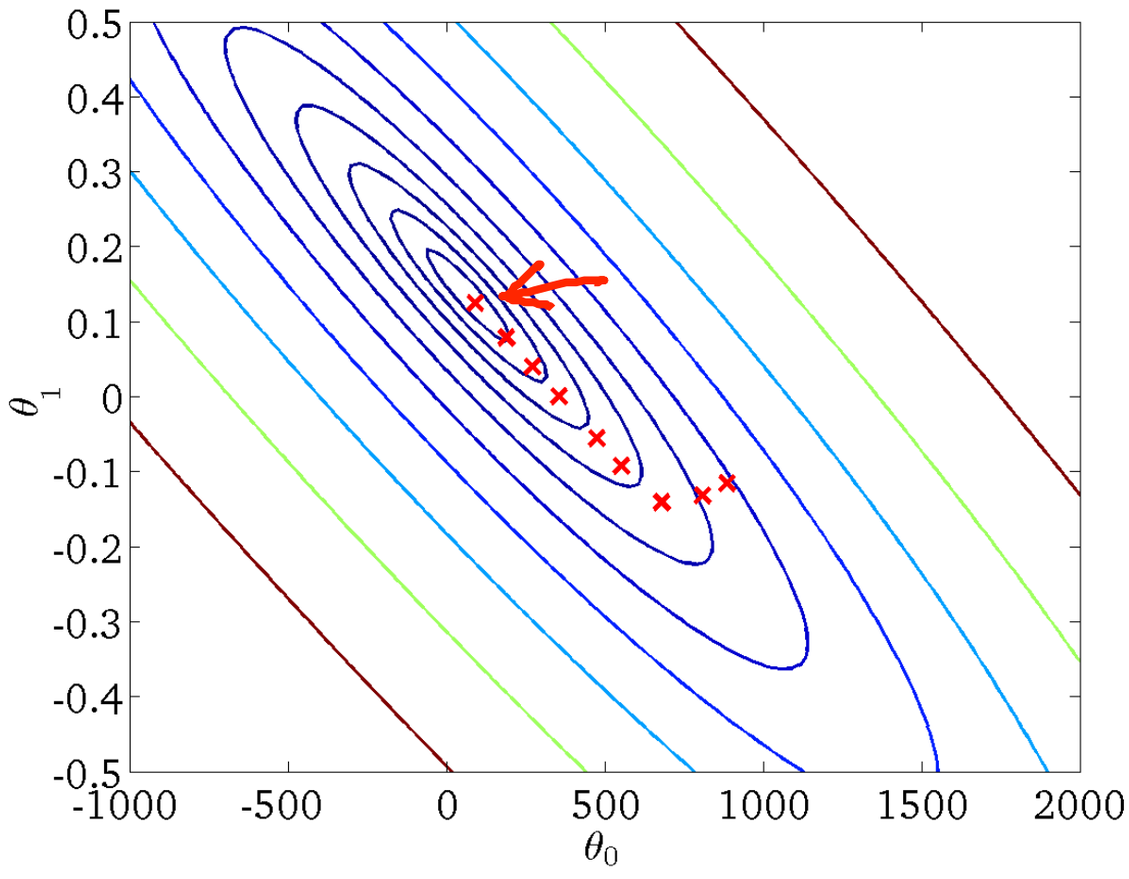
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)

$J(\theta_0, \theta_1)$

(function of the parameter θ_0, θ_1)





Batch Gradient Descent

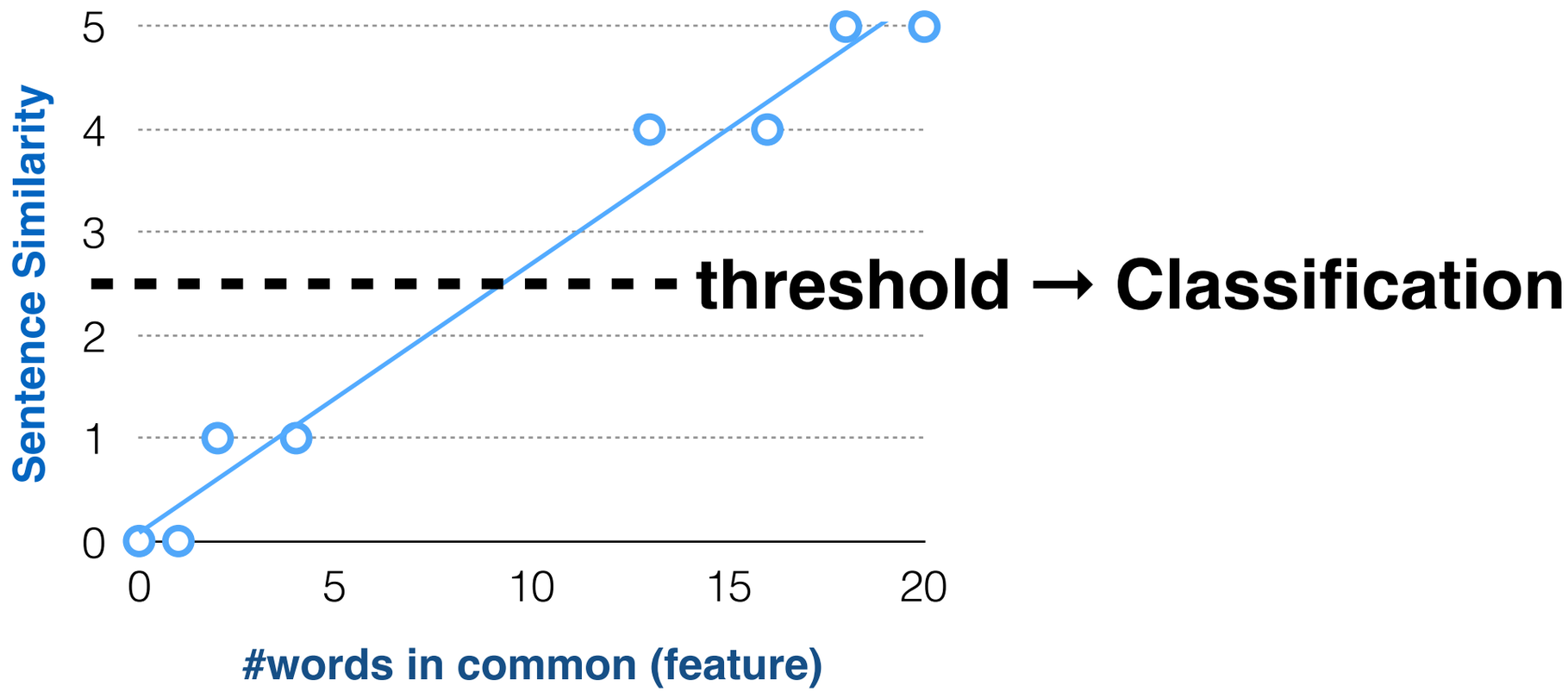
- Each step of gradient descent uses all the training examples

Cost Function

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

(Recap)

Linear Regression

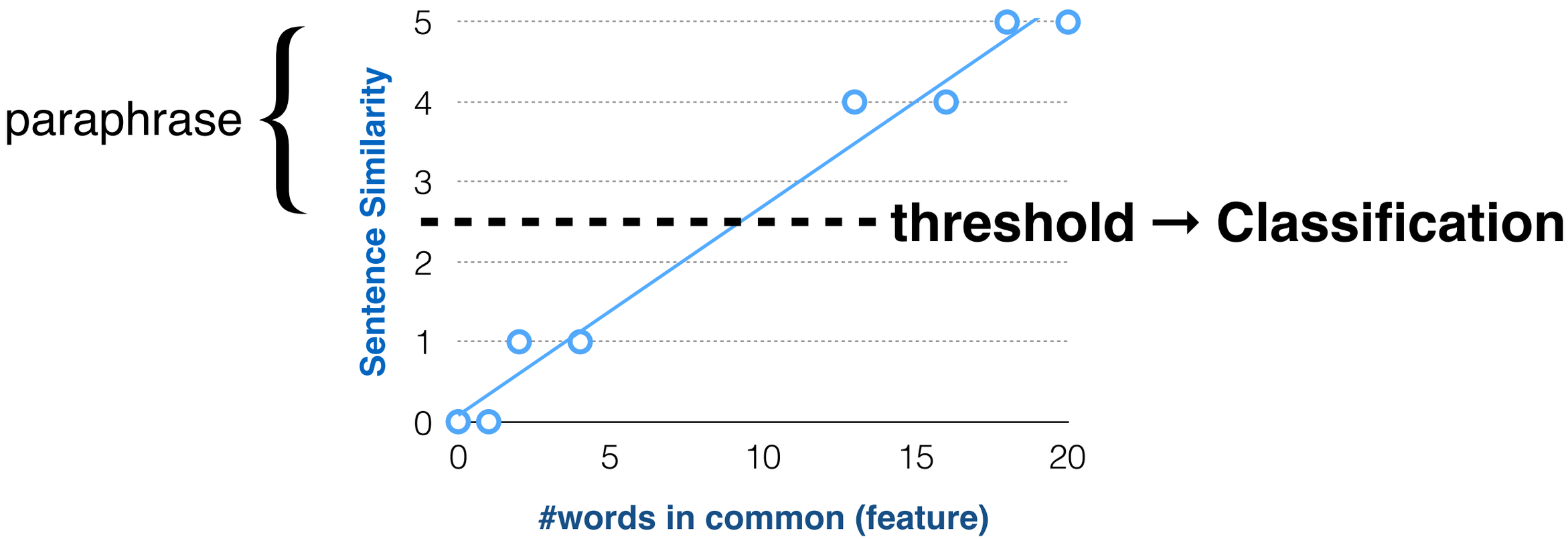


- also supervised learning (learn from annotated data)

- but for **Regression**: predict **real-valued** output
  (Classification: predict discrete-valued output)

(Recap)

Linear Regression

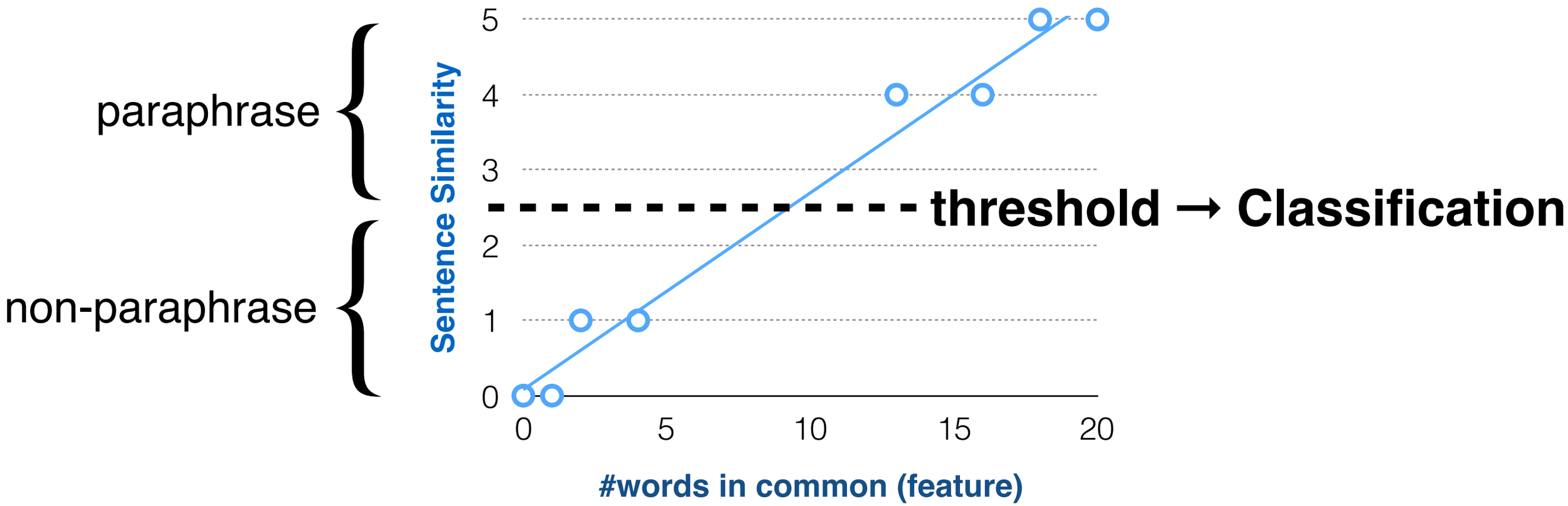


- also supervised learning (learn from annotated data)

- but for **Regression**: predict **real-valued** output
  (Classification: predict discrete-valued output)

(Recap)

Linear Regression



- also supervised learning (learn from annotated data)

- but for **Regression**: predict **real-valued** output
  (Classification: predict discrete-valued output)

(Recap)

Linear Regression

- **Hypothesis**:

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

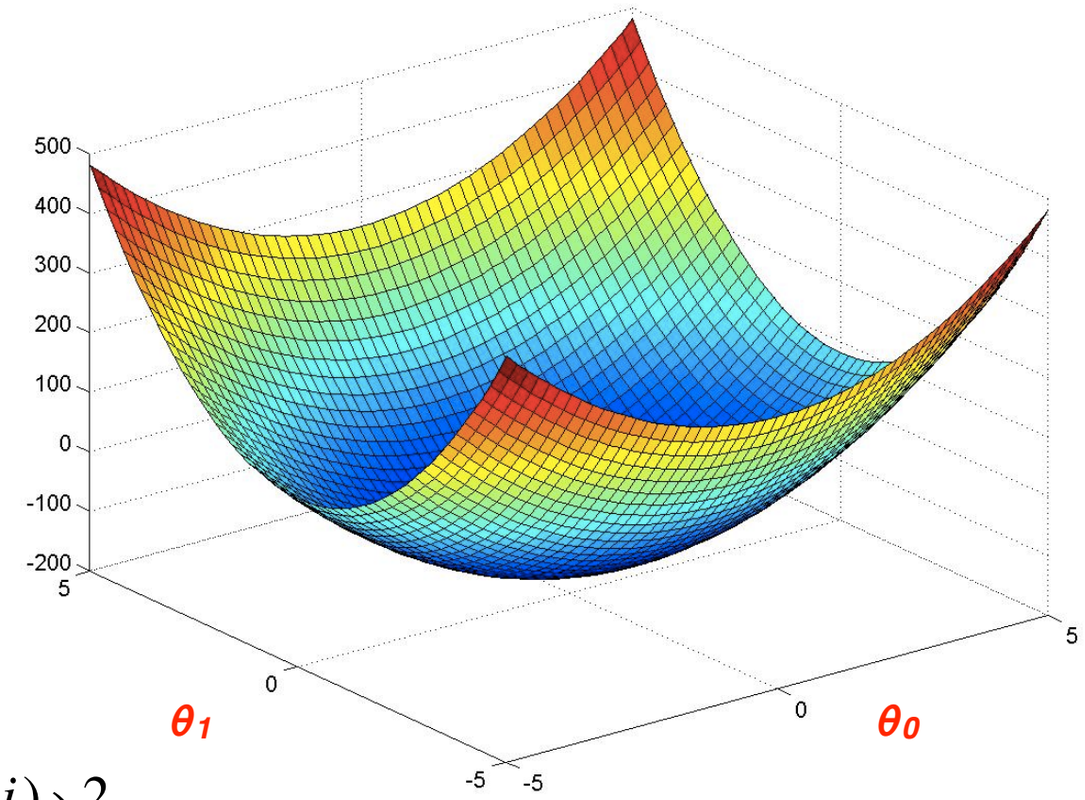
- **Parameters:**

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:** $\underset{\theta_0, \theta_1}{\text{minimize}}\, J(\theta_0, \theta_1)$



(Recap)

Gradient Descent

repeat until convergence {

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

(simultaneous update for j=0 and j=1)

}

learning rate

Next Class:

- Logistic Regression

socialmedia-class.org