Imperial College
London

Machine Learning for Imaging

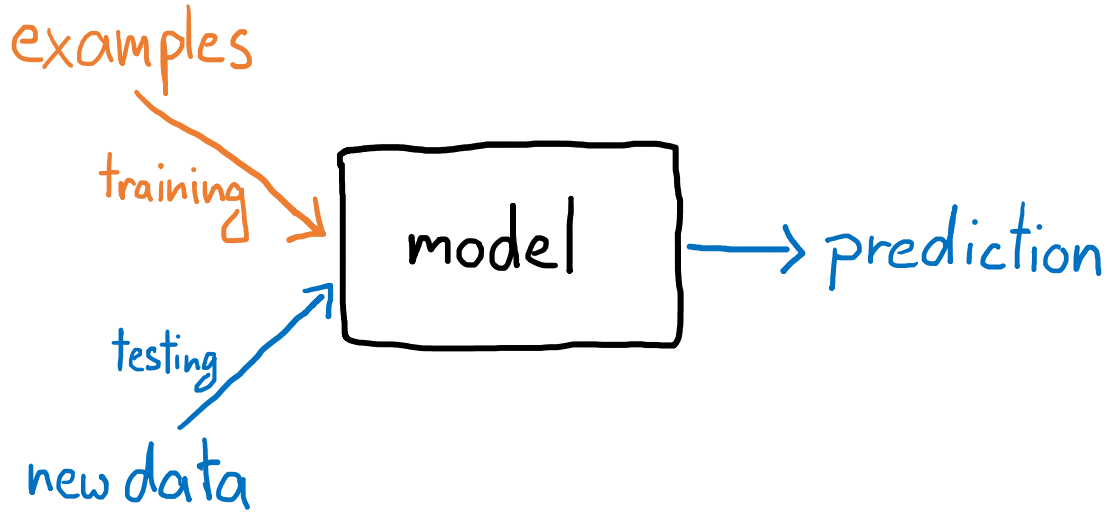
Introduction to Machine Learning

Ben Glocker
b.glocker@imperial.ac.uk

What is Machine Learning?

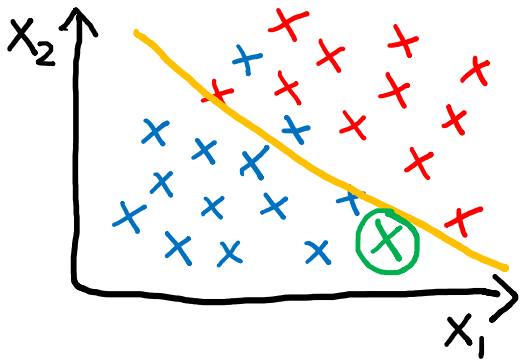
“Field of study that gives computers the ability to learn without being explicitly programmed.” (Arthur Samuel, 1959)

“Machine learning algorithms build a mathematical model of sample data in order to make predictions or decisions without being explicitly programmed to perform the task.” (Wikipedia)

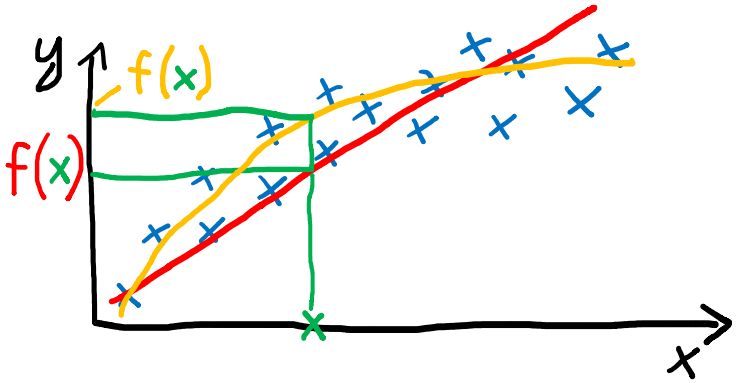


Supervised Learning

Classification
predicts a qualitative output



Regression
predicts a quantitative output



Supervised Learning

Given a vector of input features $\mathbf{x}$,

we want to learn a predictor $h_\Theta(\mathbf{x})$,

that outputs an accurate prediction y

$$h_\Theta(\mathbf{x}) = y$$

hypothesis

parameters

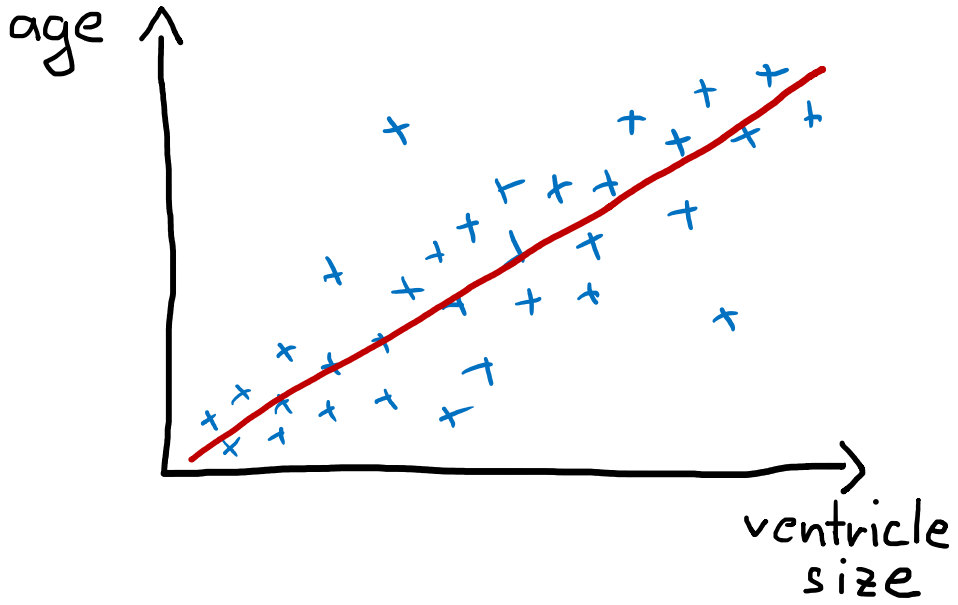
Using a training set $T = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^{m}$ with m examples,

the predictor is trained (somehow) such that

$$\forall i \left[h_\Theta(\mathbf{x}^{(i)}) \approx y^{(i)} \right]$$

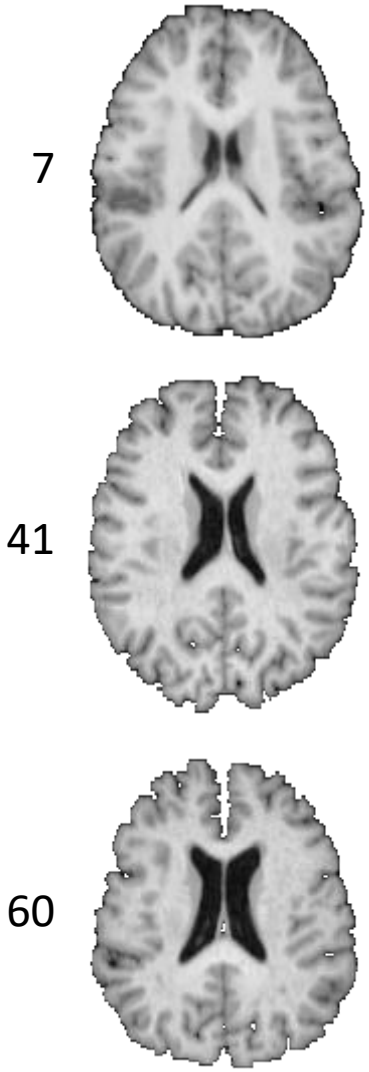
Regression

Linear Regression



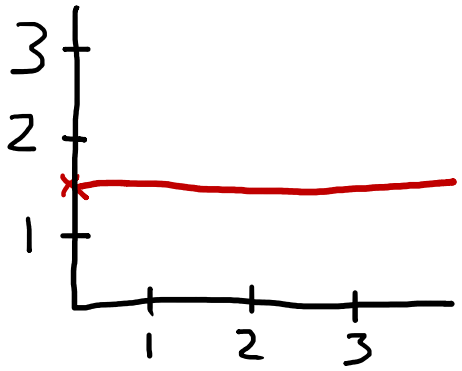
$$h_\Theta(x) = \Theta_0 + \Theta_1 x$$

bias



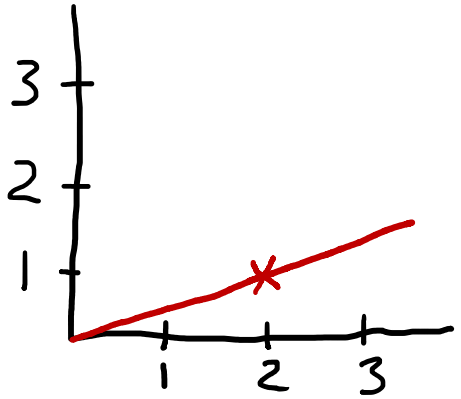
Linear Regression

$$h_\Theta(x) = \Theta_0 + \Theta_1 x$$



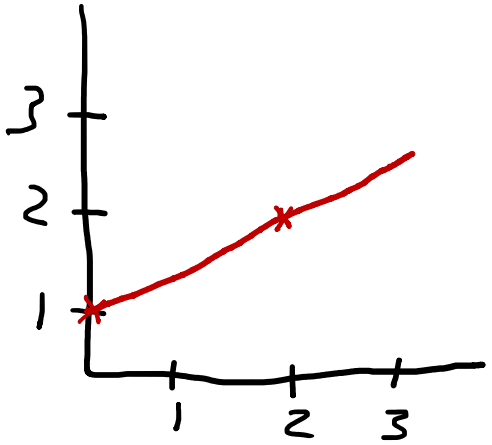
$\Theta_0 = 1.5$

$\Theta_1 = 0$



$\Theta_0 = 0$

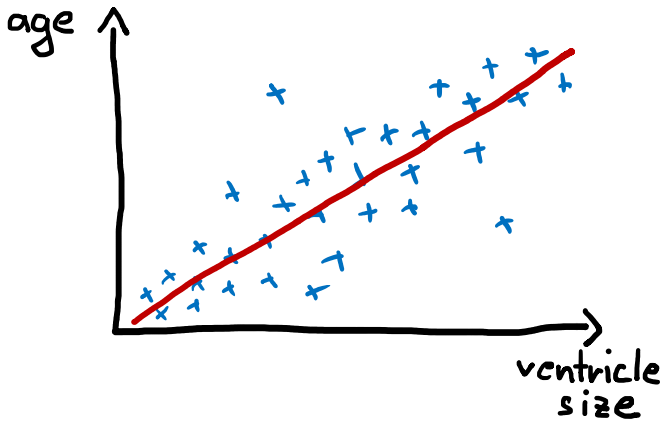
$\Theta_1 = 0.5$



$\Theta_0 = 1$

$\Theta_1 = 0.5$

Linear Regression



$$h_\Theta(x) = \Theta_0 + \Theta_1 x$$

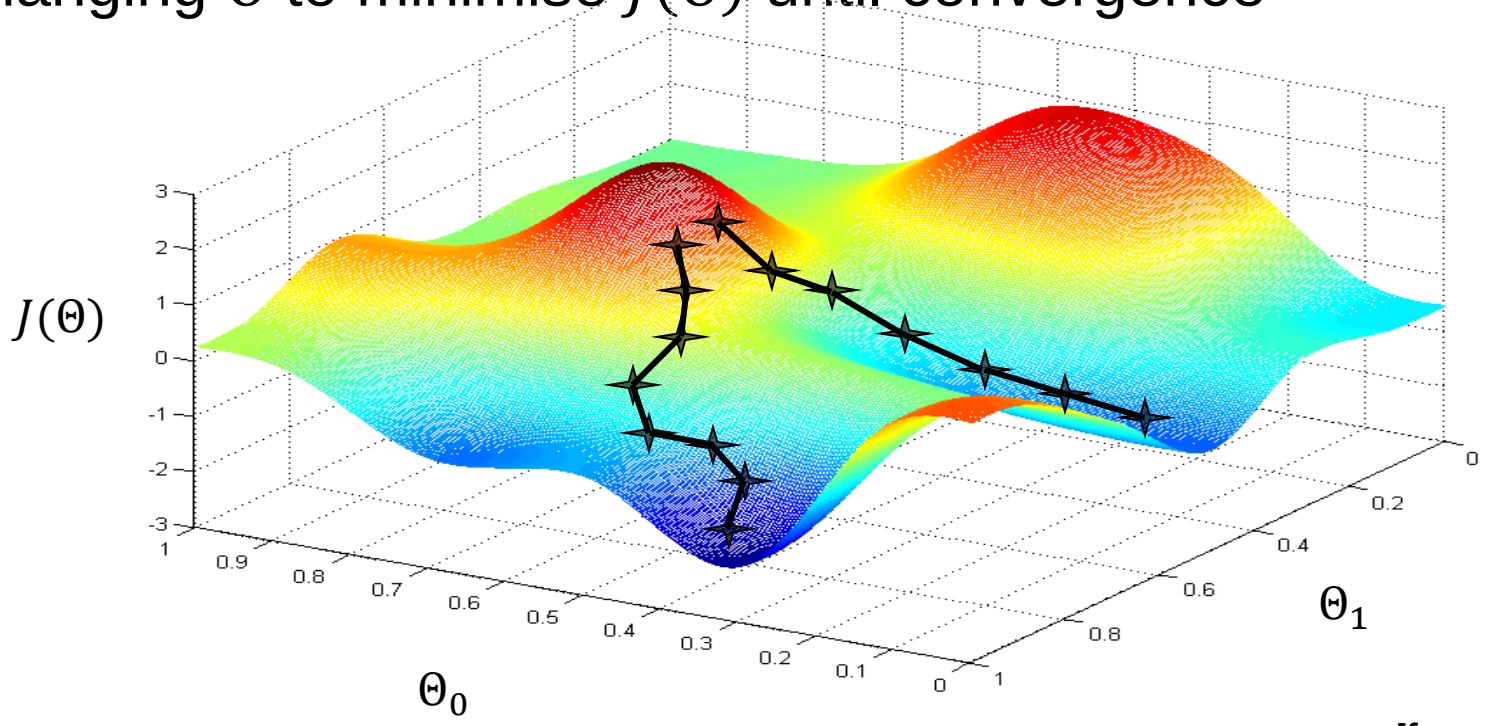
$$\min_\Theta \frac{1}{m}\sum_{i=1}^{m}\left(h_\Theta\left(x^{(i)}\right) - y^{(i)}\right)^2$$

cost function $J(\theta)$

Gradient Descent

Given a hypothesis h_Θ and a cost function $J(\Theta)$

- Take an initial guess for the parameters Θ
- Keep changing Θ to minimise $J(\Theta)$ until convergence



[from Andrew Ng's slides]

Gradient Descent

Given a hypothesis h_Θ and a cost function $J(\Theta)$

repeat until convergence

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

learning rate

partial derivative

parameter update

Gradient Descent for Linear Regression

Rewriting the cost function

$$h_\Theta(x) = \Theta_0 + \Theta_1 x$$

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

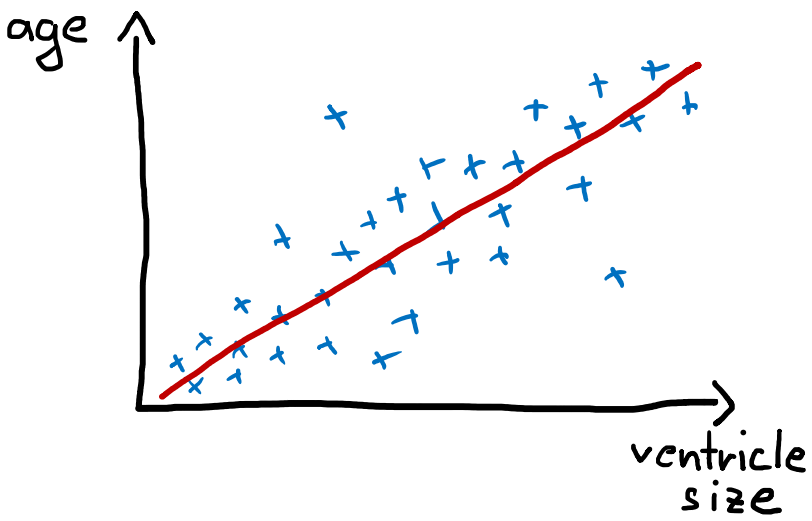
$$= \frac{1}{2m}\sum_{i=1}^{m}\left(\Theta_0 + \Theta_1 x^{(i)} - y^{(i)}\right)^2$$

$$\frac{\partial}{\partial \Theta_j} J(\Theta) = ?$$

$$\frac{\partial}{\partial \Theta_0} J(\Theta) = \frac{1}{m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)$$

$$\frac{\partial}{\partial \Theta_1} J(\Theta) = \frac{1}{m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right) \cdot x^{(i)}$$

Gradient Descent for Linear Regression



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

repeat until convergence

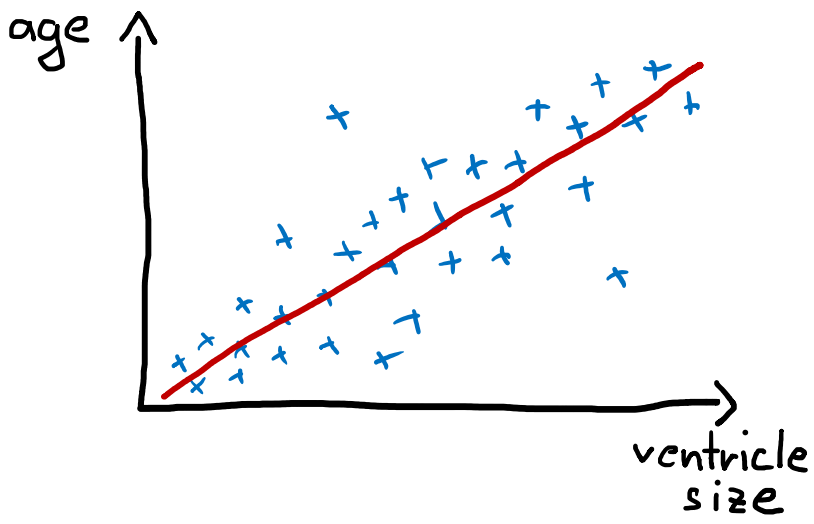
$$\Theta_0 := \Theta_0 - \alpha\frac{1}{m}\sum_{i=1}^{m}(h_\Theta(x^{(i)}) - y^{(i)})$$

$$\Theta_1 := \Theta_1 - \alpha\frac{1}{m}\sum_{i=1}^{m}(h_\Theta(x^{(i)}) - y^{(i)}) \cdot x^{(i)}$$

General form

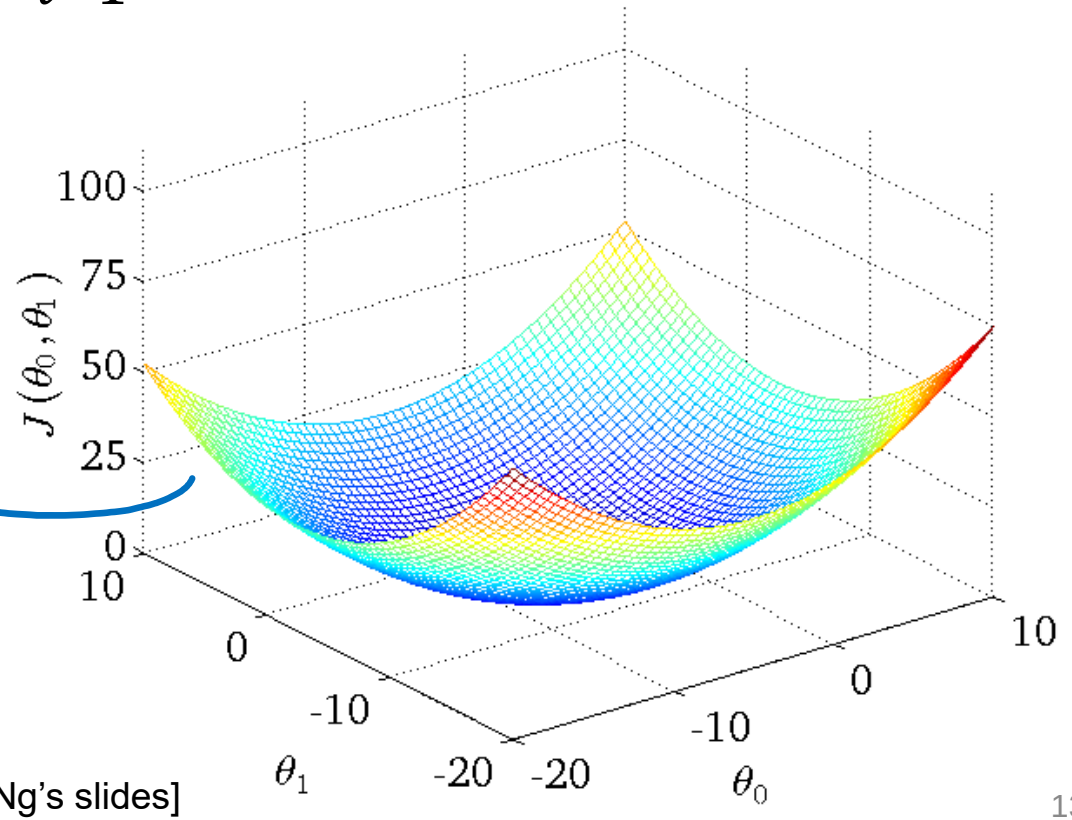
$$\Theta_j := \Theta_j - \alpha\frac{1}{m}\sum_{i=1}^{m}(h_\Theta(\mathbf{x}^{(i)}) - y^{(i)}) \cdot x_j^{(i)}$$

Gradient Descent for Linear Regression



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

convex
⇕
no local minima

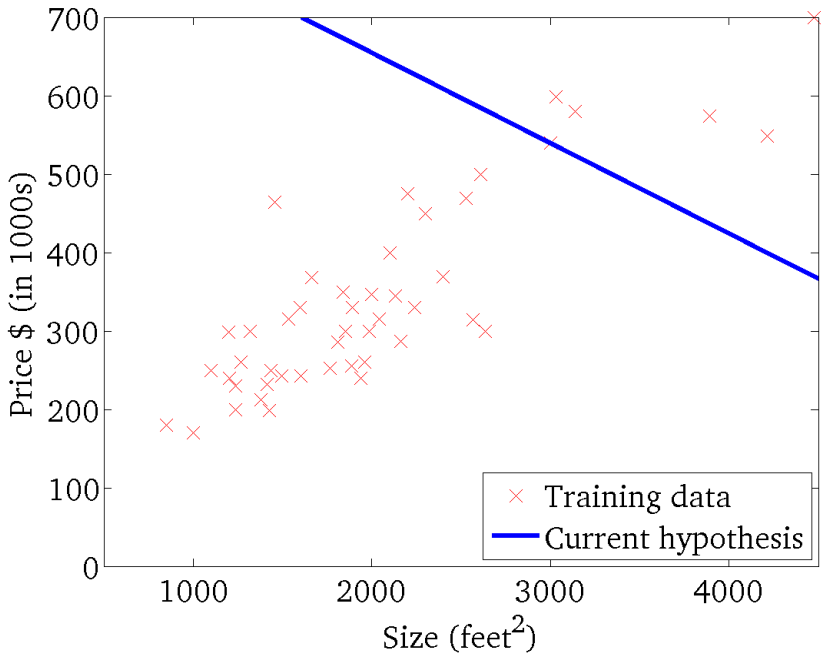


[from Andrew Ng's slides]

Gradient Descent for Linear Regression

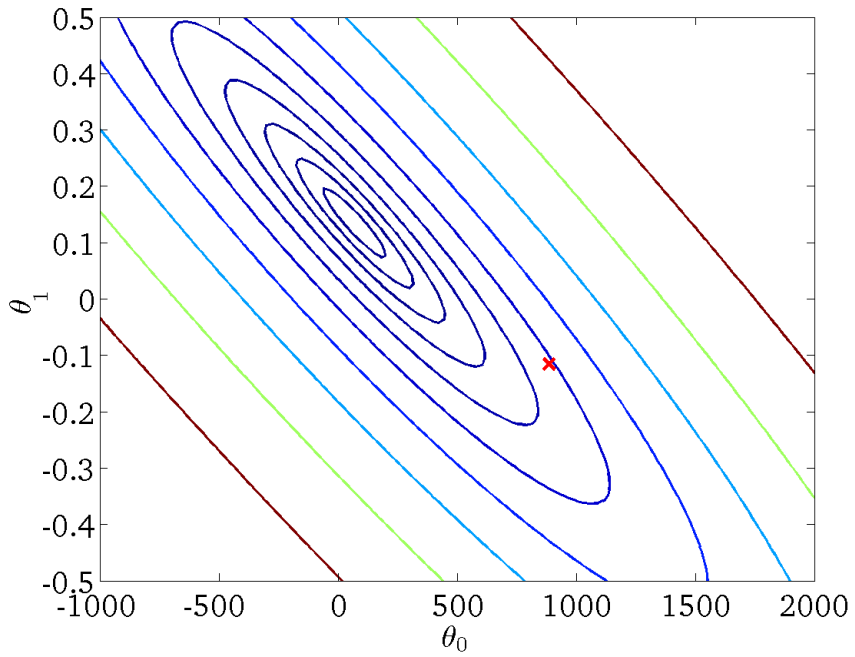
$h_\theta(x)$

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



$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)

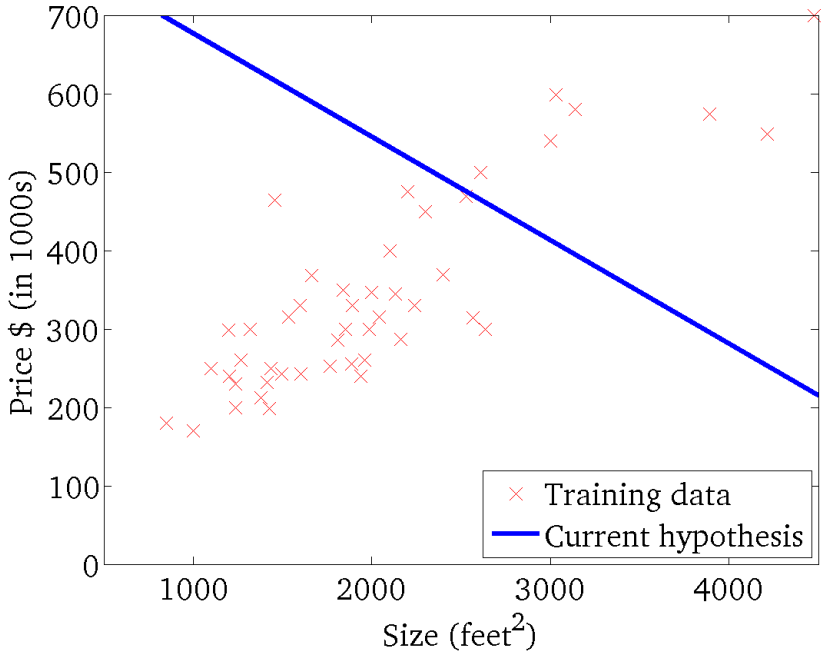


[from Andrew Ng's slides]

Gradient Descent for Linear Regression

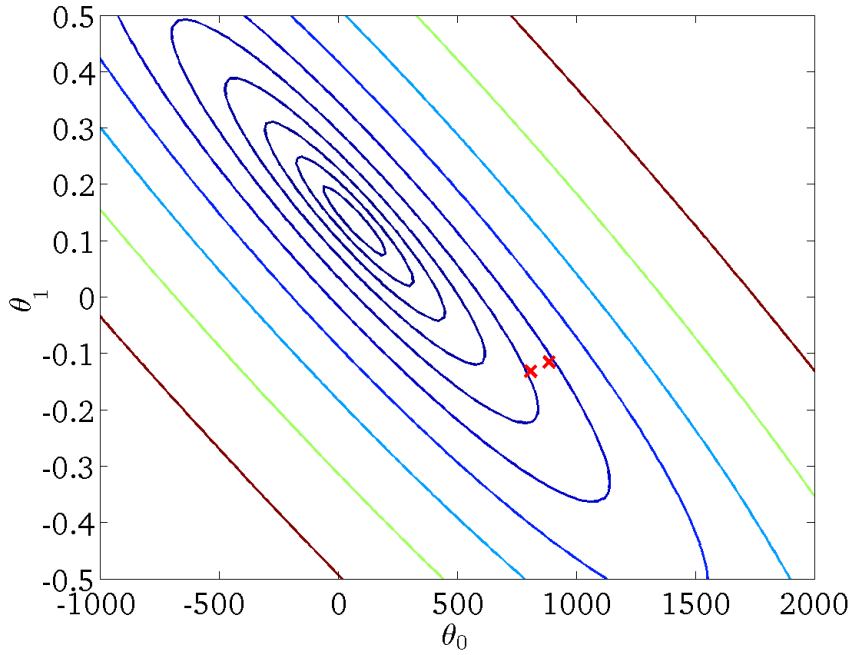
$h_\theta(x)$

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



$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)

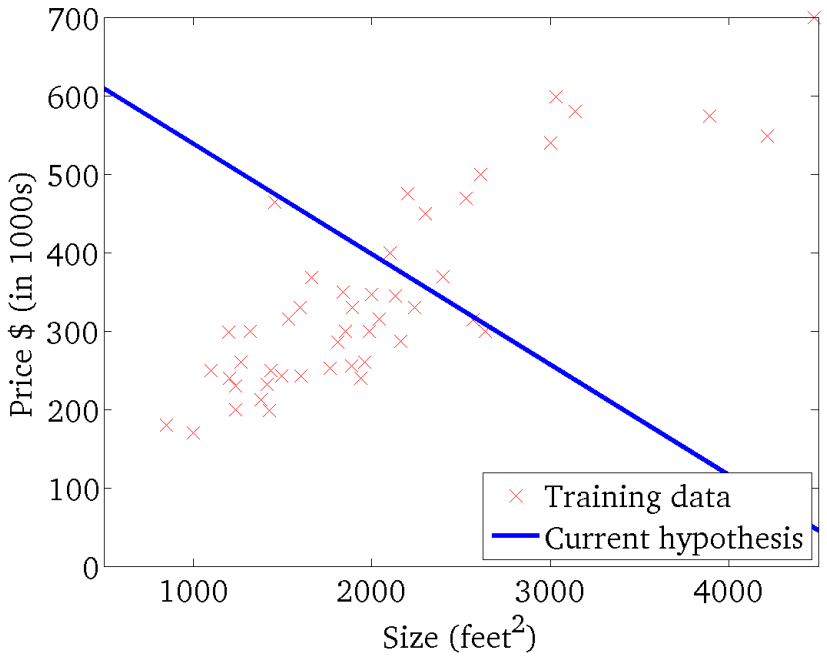


[from Andrew Ng's slides]

Gradient Descent for Linear Regression

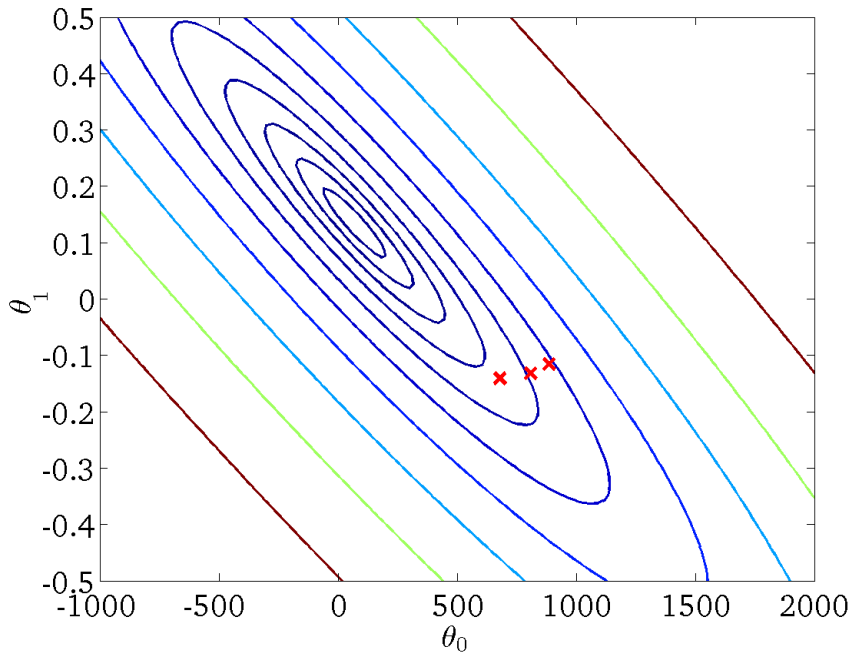
$h_\theta(x)$

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



$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)

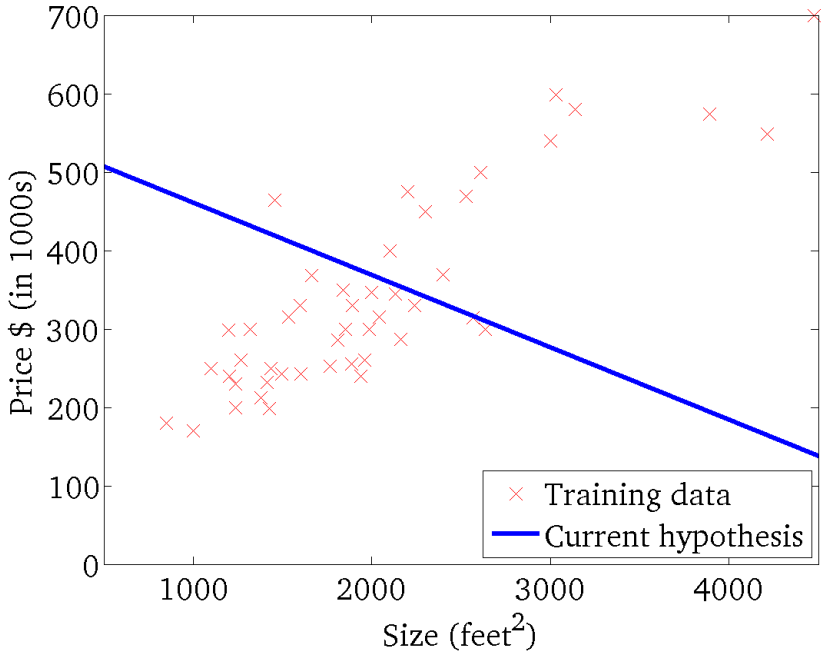


[from Andrew Ng's slides]

Gradient Descent for Linear Regression

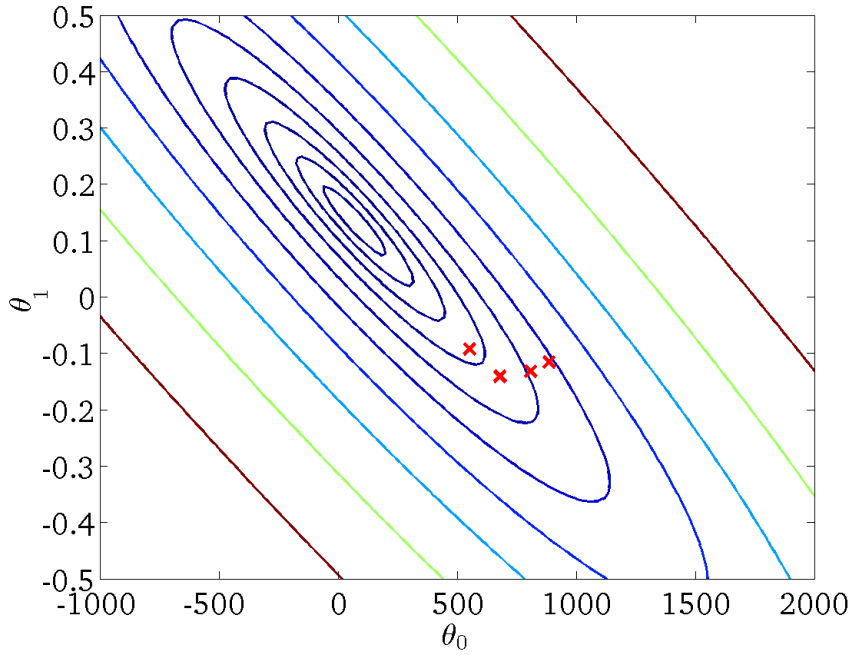
$h_\theta(x)$

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



$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)

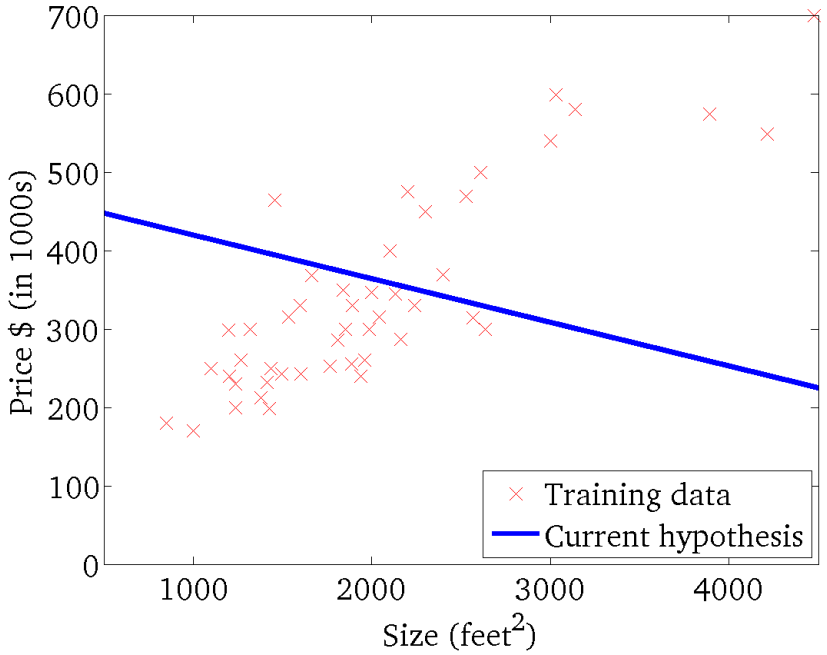


[from Andrew Ng's slides]

Gradient Descent for Linear Regression

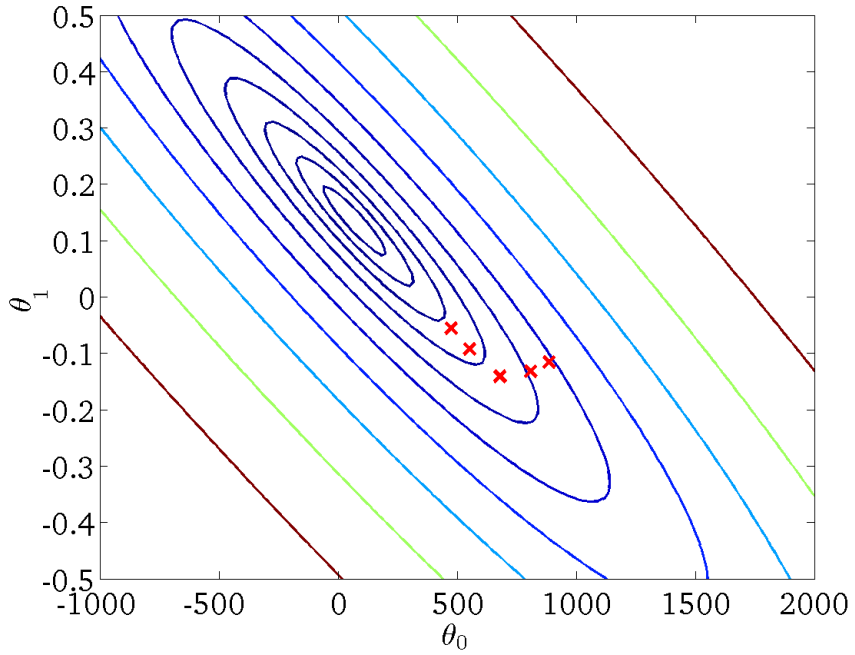
$h_\theta(x)$

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



$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)

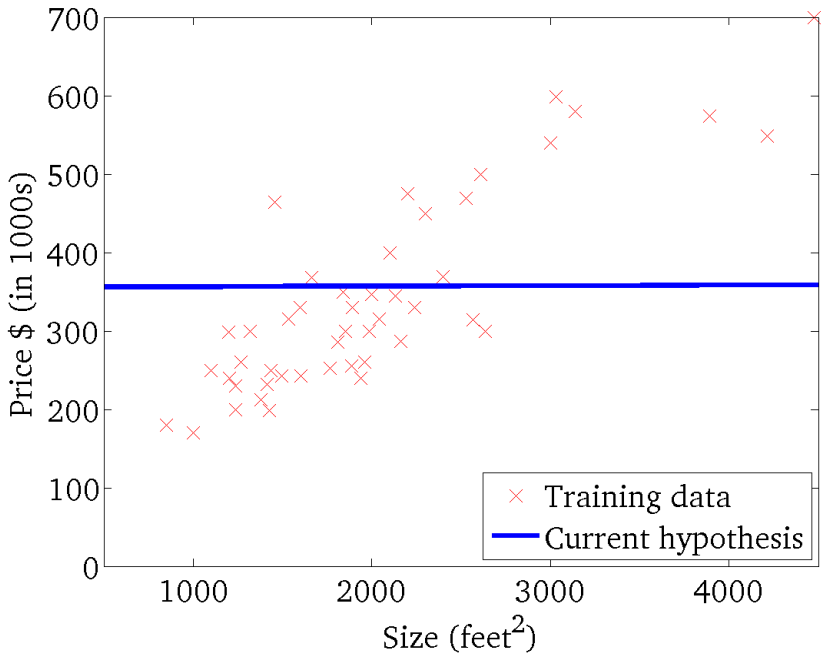


[from Andrew Ng's slides]

Gradient Descent for Linear Regression

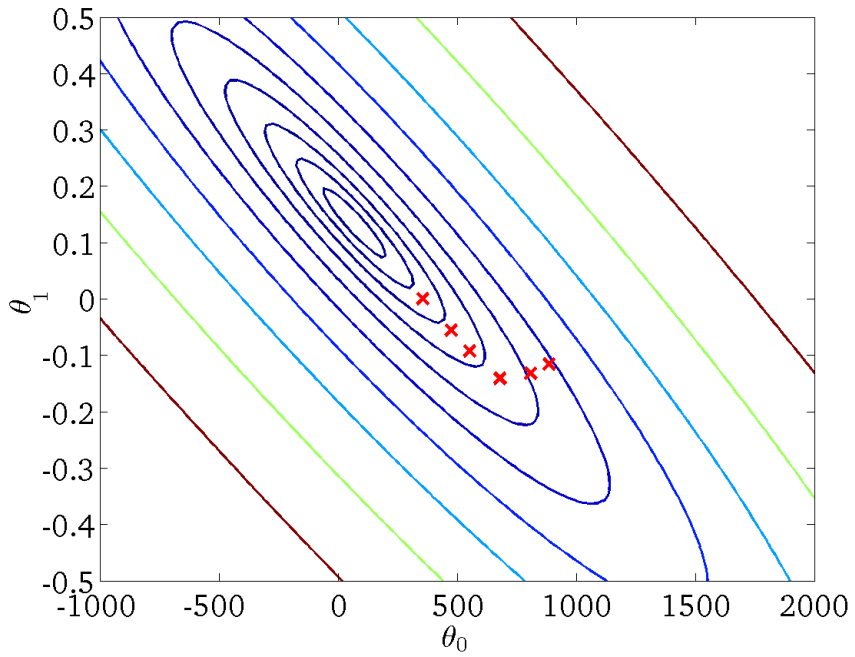
$h_\theta(x)$

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



$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)

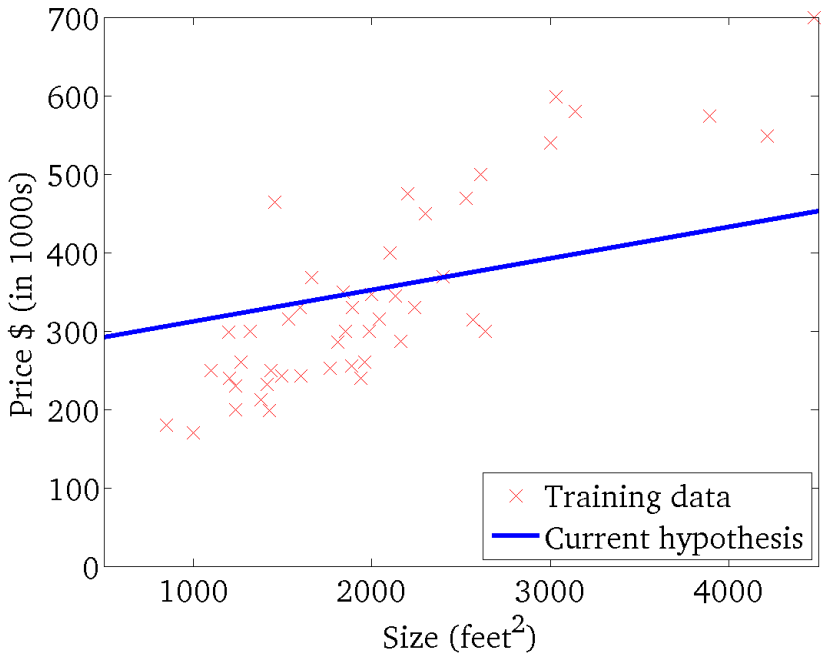


[from Andrew Ng's slides]

Gradient Descent for Linear Regression

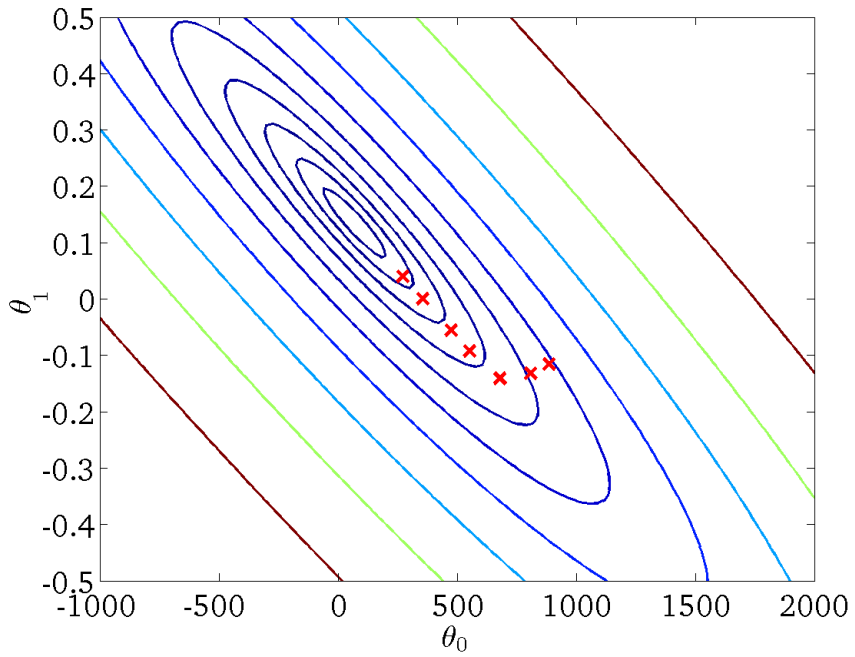
$h_\theta(x)$

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



$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)

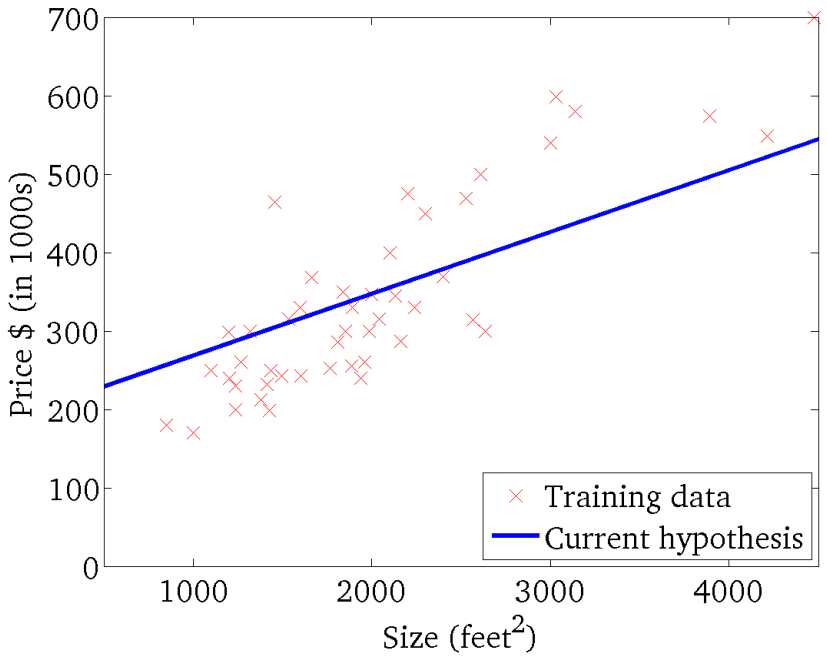


[from Andrew Ng's slides]

Gradient Descent for Linear Regression

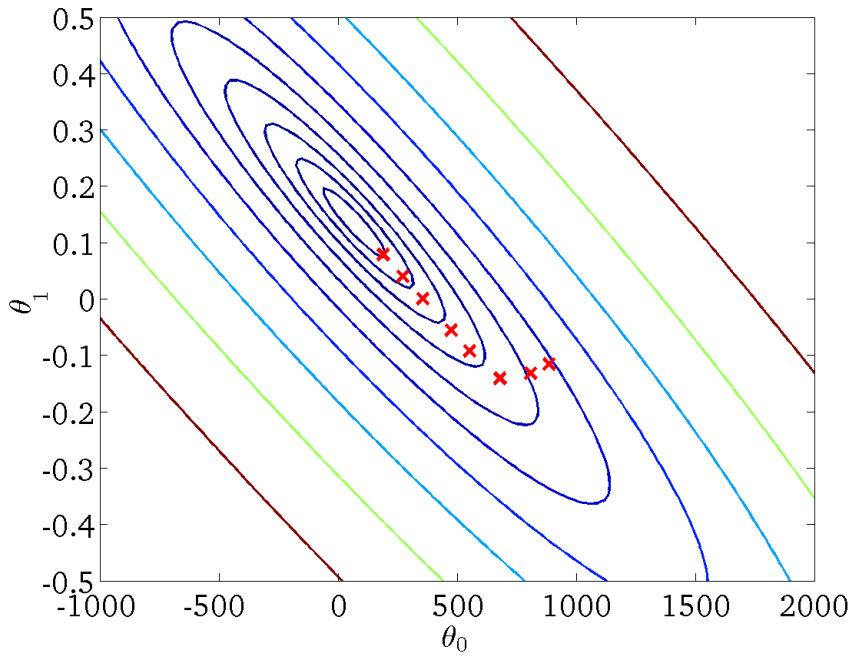
$h_\theta(x)$

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



$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)

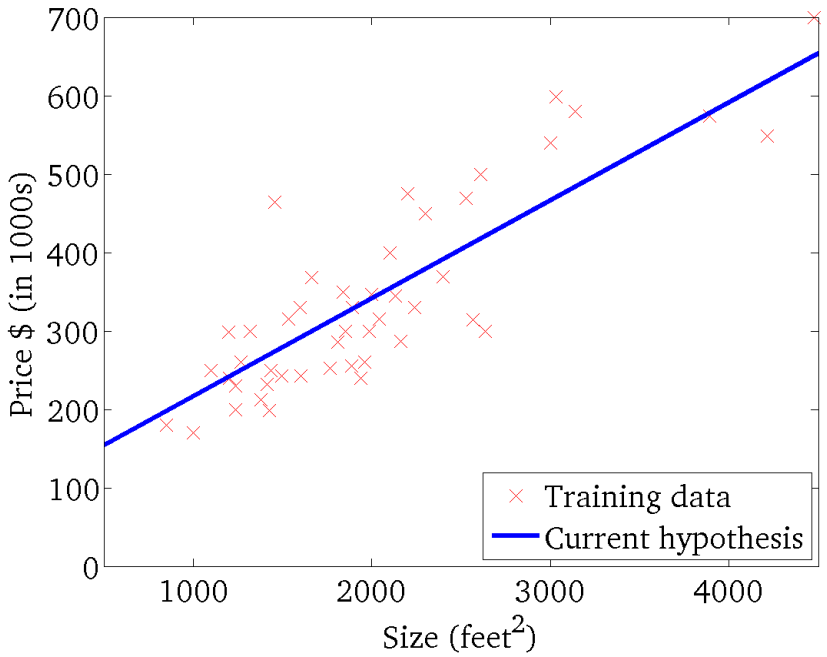


[from Andrew Ng's slides]

Gradient Descent for Linear Regression

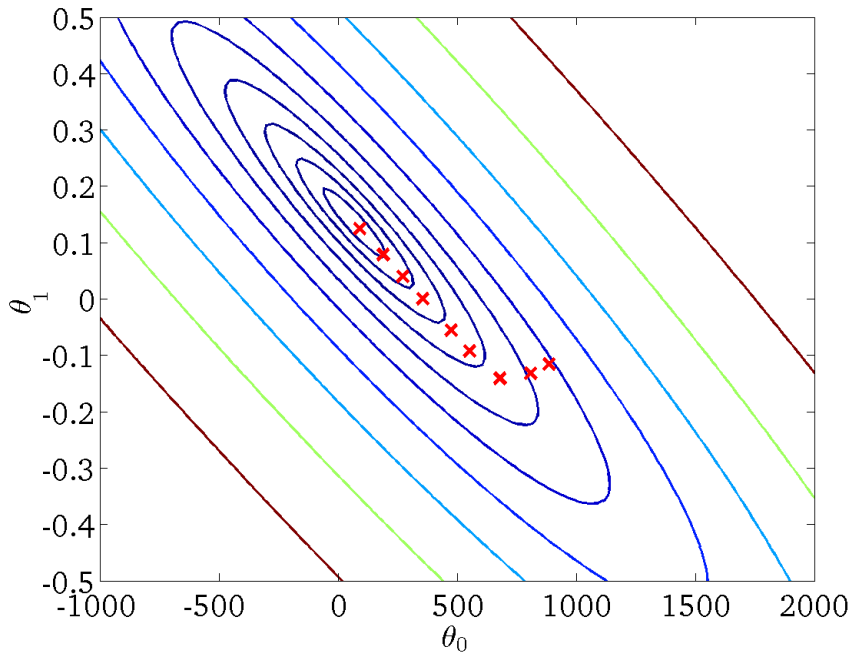
$h_\theta(x)$

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



$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)



[from Andrew Ng's slides]

Linear Regression for Multiple Variables

Given input vector $\mathbf{x}$ and output y

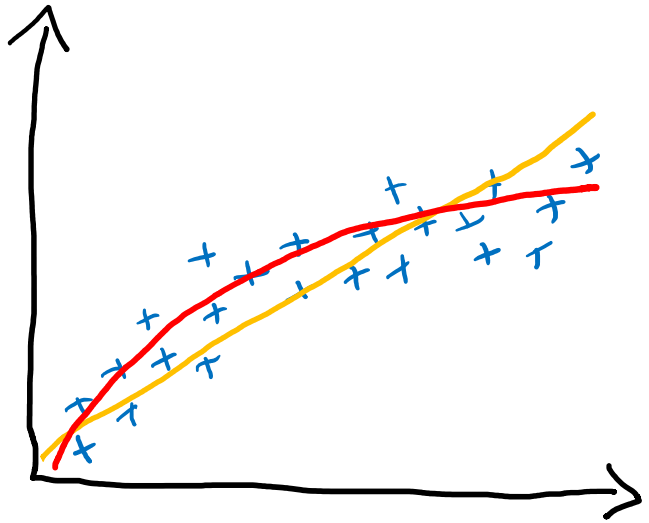
$$h_\Theta(\mathbf{x}) = \Theta^\top \mathbf{x}$$

$$= \Theta_0 x_0 + \Theta_1 x_1 + \Theta_2 x_2 + \cdots + \Theta_n x_n$$

$x_0 = 1$

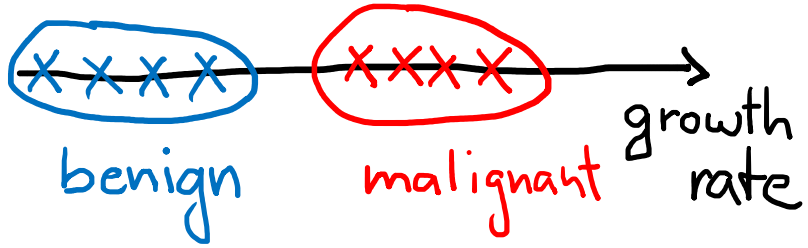
Polynomial Regression

$$h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2$$

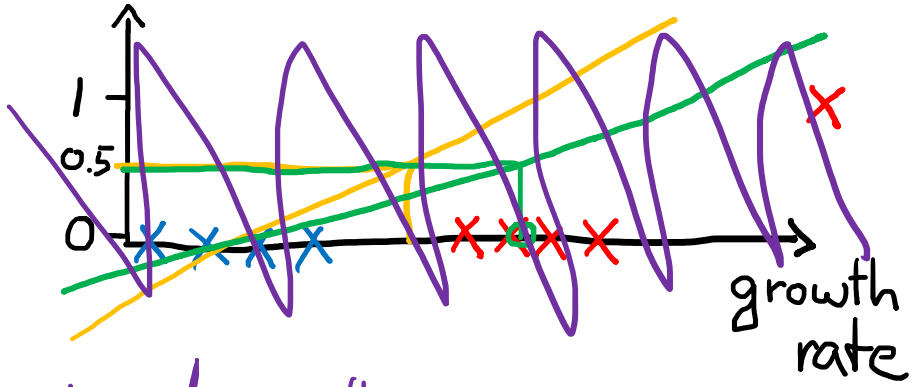


Classification

Logistic Regression



$h_\Theta(\mathbf{x}) = \Theta^\top \mathbf{x}$



$$h_\Theta(\mathbf{x}) = g(\Theta^\top \mathbf{x})$$

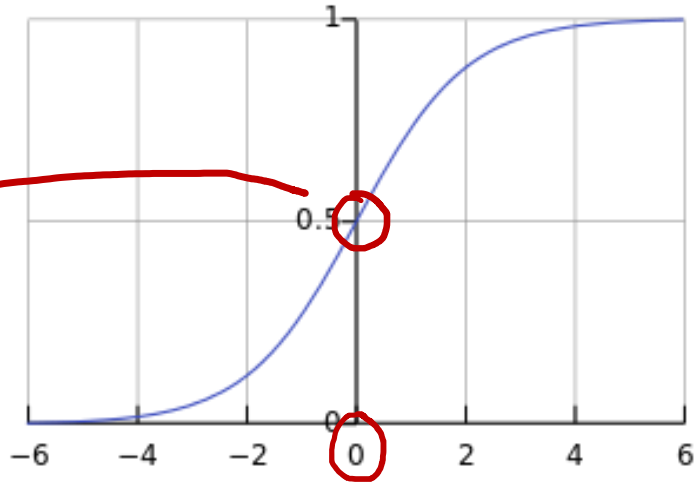
$$g(z) = \frac{1}{1+e^{-z}}$$

sigmoid function
logistic function

Logistic Regression

Hypothesis

$$h_\Theta(\mathbf{x}) = \frac{1}{1 + e^{-\Theta^T \mathbf{x}}} \in [0,1]$$



Prediction

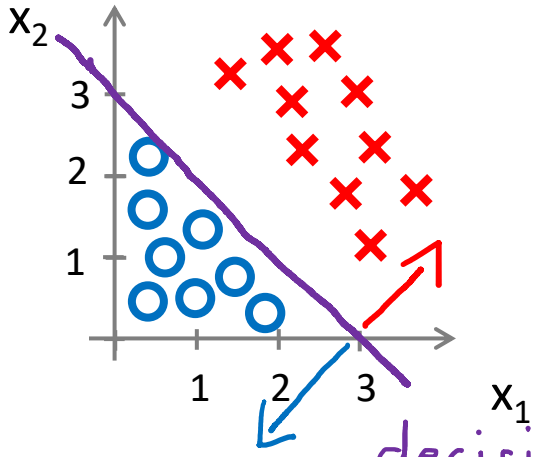
$$h_\Theta(\mathbf{x}) \geq 0.5 \rightarrow y = 1 \quad \text{if } \theta^T x \geq 0$$

$$h_\Theta(\mathbf{x}) < 0.5 \rightarrow y = 0$$

$$g(z) = \frac{1}{1 + e^{-z}}$$

$g(z) \geq 0.5$ if $z \geq 0$

Decision Boundary



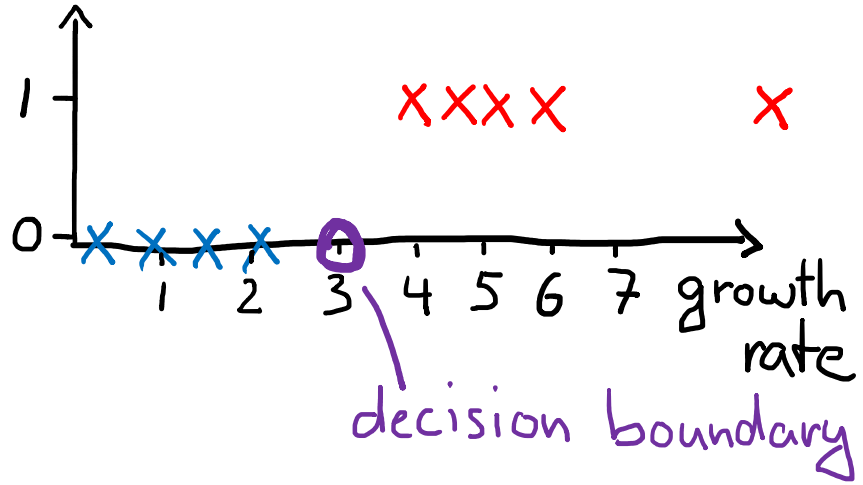
[from Andrew Ng's slides]

$$h_\Theta(\mathbf{x}) = g(\Theta_0 + \Theta_1 x_1 + \Theta_2 x_2)$$

$$\Theta = \begin{bmatrix} -3 \\ 1 \\ 1 \end{bmatrix} \rightarrow -3 + x_1 + x_2 \geq 0$$

$$x_1 + x_2 \geq 3$$

decision boundary $x_1 + x_2 = 3$



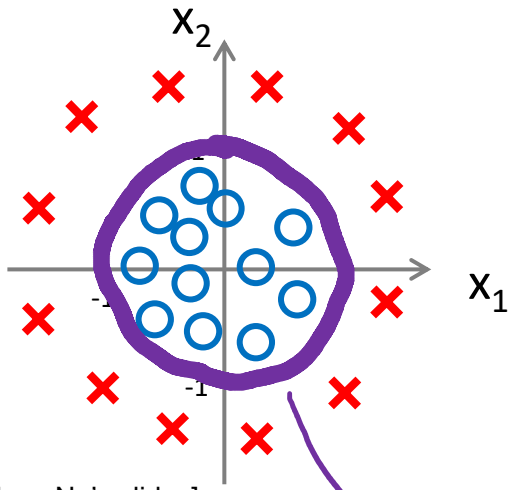
$$h_\Theta(\mathbf{x}) = g(\Theta_0 + \Theta_1 x_1)$$

$$\Theta = \begin{bmatrix} -3 \\ 1 \end{bmatrix} \rightarrow -3 + x_1 \geq 0$$

$$x_1 \geq 3$$

$x_1 = 3$

Non-linear Decision Boundaries



[from Andrew Ng's slides]

$$h_\Theta(\mathbf{x}) = g(\Theta_0 + \Theta_1 x_1 + \Theta_2 x_2 + \Theta_3 x_1^2 + \Theta_4 x_2^2)$$

$\Theta_0 = -1$, $\Theta_1 = 0$, $\Theta_2 = 0$, $\Theta_3 = 1$, $\Theta_4 = 1$

$-1 + x_1^2 + x_2^2 \geq 0$

$x_1^2 + x_2^2 \geq 1$

$x_1^2 + x_2^2 = 1$

decision boundary

Cost Function for Logistic Regression

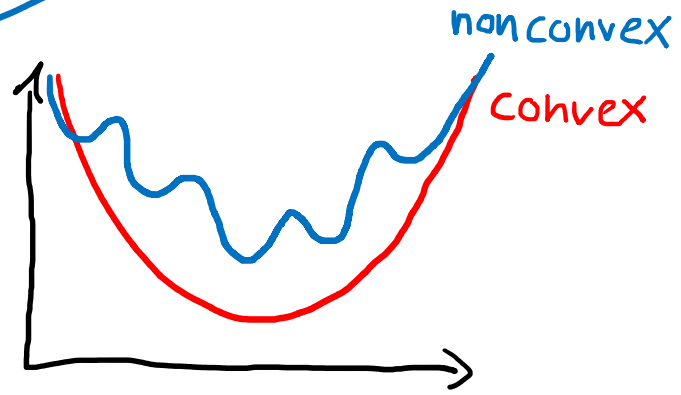
Remember: Linear Regression

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

Logistic Regression

$$h_\Theta(\mathbf{x}) = \frac{1}{1 + e^{-\Theta^\top \mathbf{x}}}$$

non linear



Rewrite Cost Function

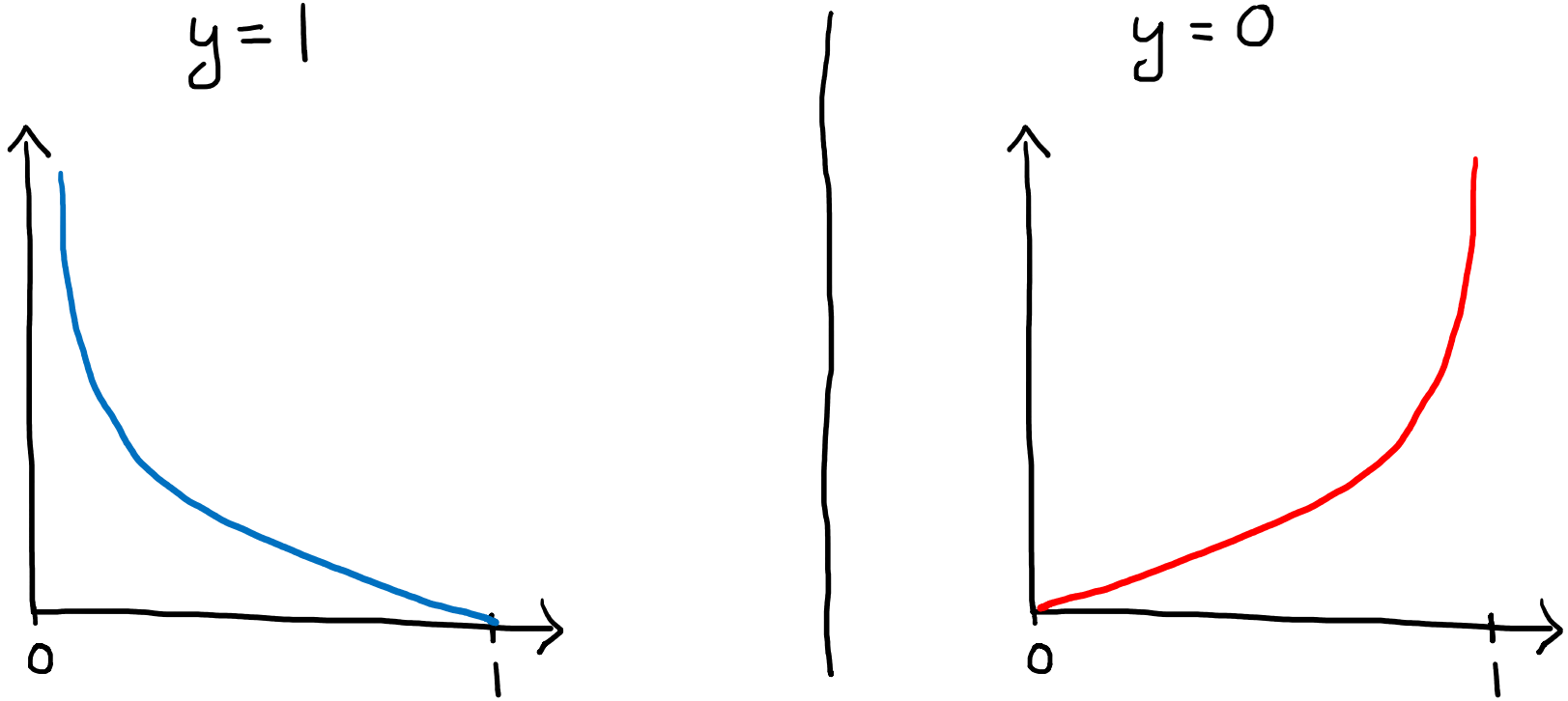
$$J(\Theta) = \frac{1}{2m}\sum_{i=1}^{m}\left(h_\Theta\left(\mathbf{x}^{(i)}\right) - y^{(i)}\right)^2$$

$\mathrm{cost}\left(h_\theta(x^{(i)}), y^{(i)}\right)$

Cost Function for Logistic Regression

$$J(\Theta) = \frac{1}{m}\sum_{i=1}^{m} cost\left(h_\Theta\left(\mathbf{x}^{(i)}\right), y^{(i)}\right)$$

$$cost(h_\Theta(\mathbf{x}), y) = \begin{cases} -\log(h_\Theta(\mathbf{x})) & \text{if } y = 1 \\ -\log(1 - h_\Theta(\mathbf{x})) & \text{if } y = 0 \end{cases}$$



Cost Function for Logistic Regression

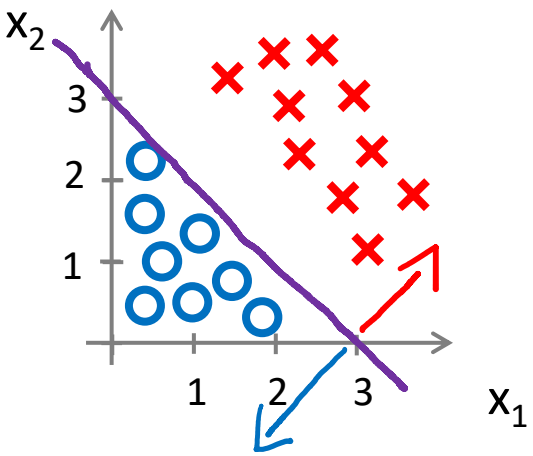
$$J(\Theta) = \frac{1}{m}\sum_{i=1}^{m} cost\left(h_\Theta\left(\mathbf{x}^{(i)}\right), y^{(i)}\right)$$

$$cost(h_\Theta(\mathbf{x}), y) = \begin{cases} -\log(h_\Theta(\mathbf{x})) & \text{if } y = 1 \\ -\log(1 - h_\Theta(\mathbf{x})) & \text{if } y = 0 \end{cases}$$

Simplify

$$cost(h_\Theta(\mathbf{x}), y) = -y\log\left(h_\Theta(\mathbf{x})\right) - (1 - y)\log\left(1 - h_\Theta(\mathbf{x})\right)$$

Gradient Descent for Logistic Regression



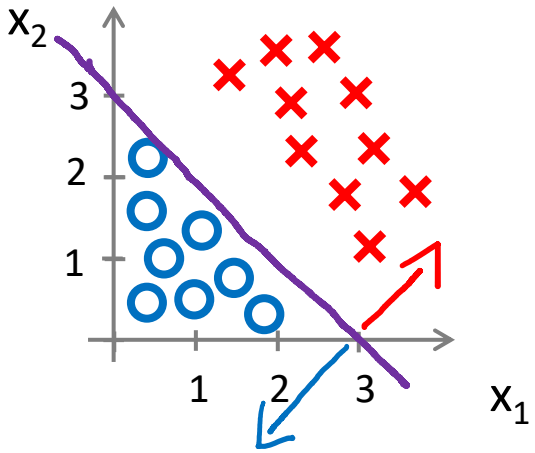
$$J(\Theta) = \frac{1}{m}\sum_{i=1}^{m} -y^{(i)} \log\left(h_\Theta\left(\mathbf{x}^{(i)}\right)\right) - (1 - y^{(i)}) \log\left(1 - h_\Theta\left(\mathbf{x}^{(i)}\right)\right)$$

$$\frac{\partial}{\partial \Theta_j} J(\Theta) = \frac{1}{m}\sum_{i=1}^{m} \left(h_\Theta\left(\mathbf{x}^{(i)}\right) - y^{(i)}\right) \cdot x_j^{(i)}$$

repeat until convergence

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

Gradient Descent for Logistic Regression



$$J(\Theta) = \frac{1}{m}\sum_{i=1}^{m} -y^{(i)} \log\left(h_\Theta\left(\mathbf{x}^{(i)}\right)\right) - \left(1 - y^{(i)}\right) \log\left(1 - h_\Theta\left(\mathbf{x}^{(i)}\right)\right)$$

$$\frac{\partial}{\partial \Theta_j} J(\Theta) = \frac{1}{m}\sum_{i=1}^{m} \left(h_\Theta\left(\mathbf{x}^{(i)}\right) - y^{(i)}\right) \cdot x_j^{(i)}$$

repeat until convergence

$$\Theta_j := \Theta_j - \alpha \frac{1}{m}\sum_{i=1}^{m} \left(h_\Theta\left(\mathbf{x}^{(i)}\right) - y^{(i)}\right) \cdot x_j^{(i)}$$

Identical to linear regression

only difference

Logarithm

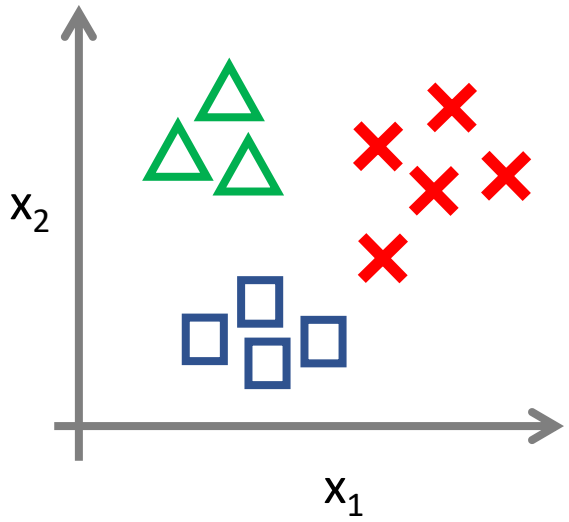
$$\frac{d \log x}{dx} = \frac{1}{x}$$

Sigmoid

$$\frac{dg(x)}{dx} = g(x)(1 - g(x))$$

Multiclass Classification

One-vs-all: learn k-binary classifiers

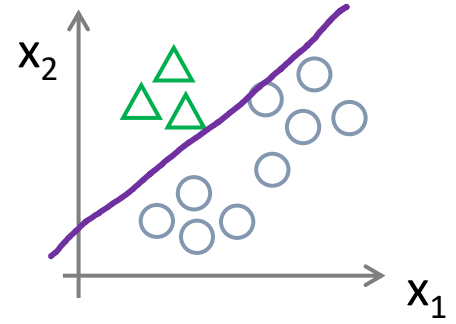


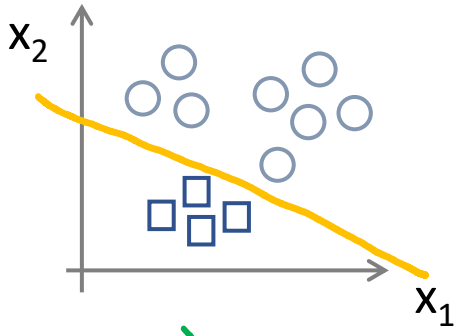
Class 1: △

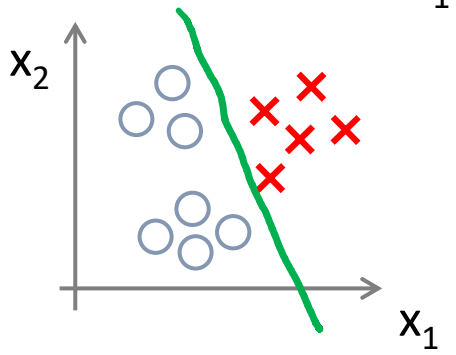
Class 2: □

Class 3: ×

[from Andrew Ng's slides]





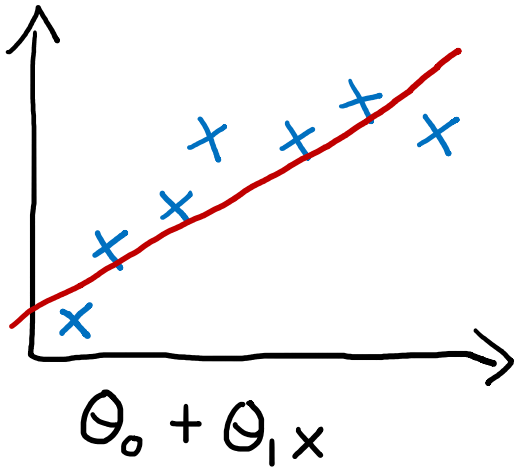


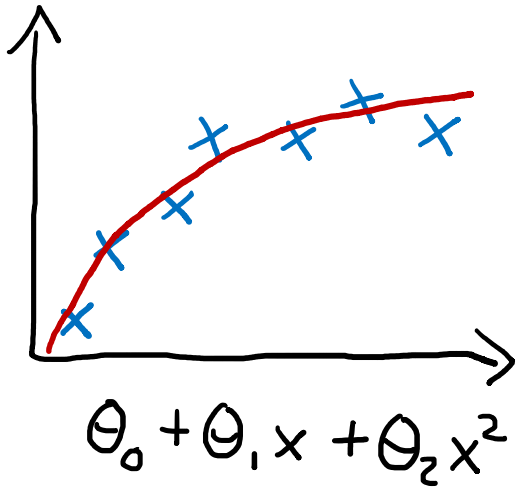
$$\max_i h_\Theta^{(i)}(\mathbf{x})$$

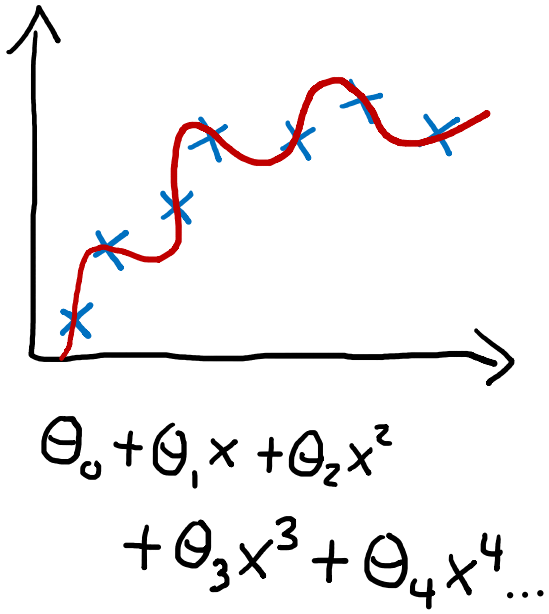
Over/Underfitting

Bias vs Variance

Linear regression







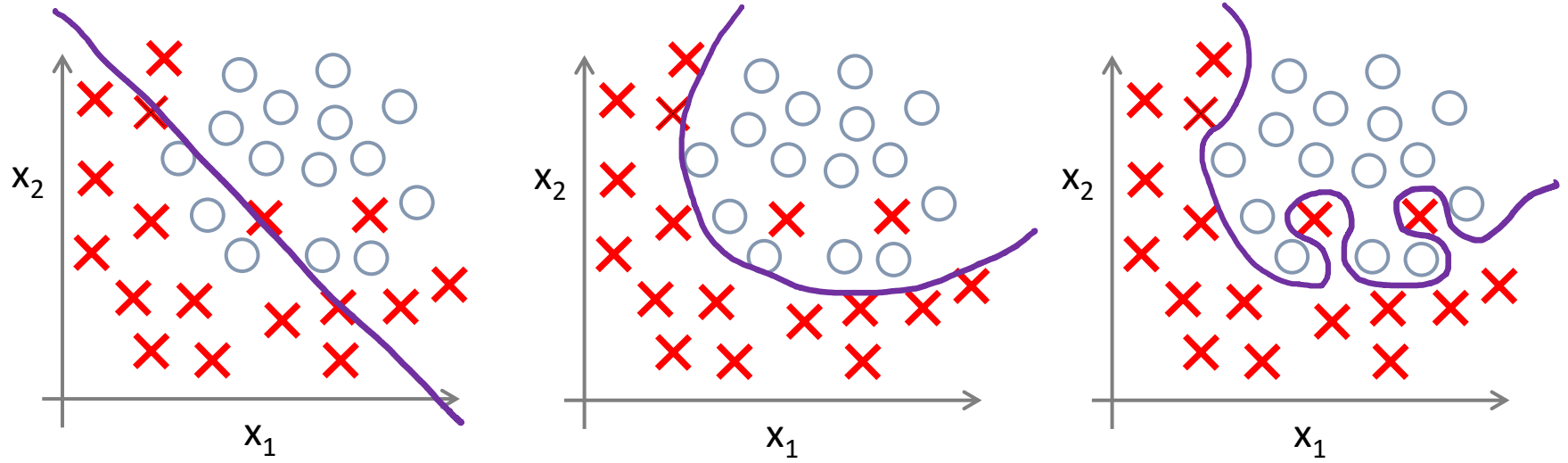
underfitting
"high bias"

just right

overfitting
"high variance"

Bias vs Variance

Logistic regression



[from Andrew Ng's slides]

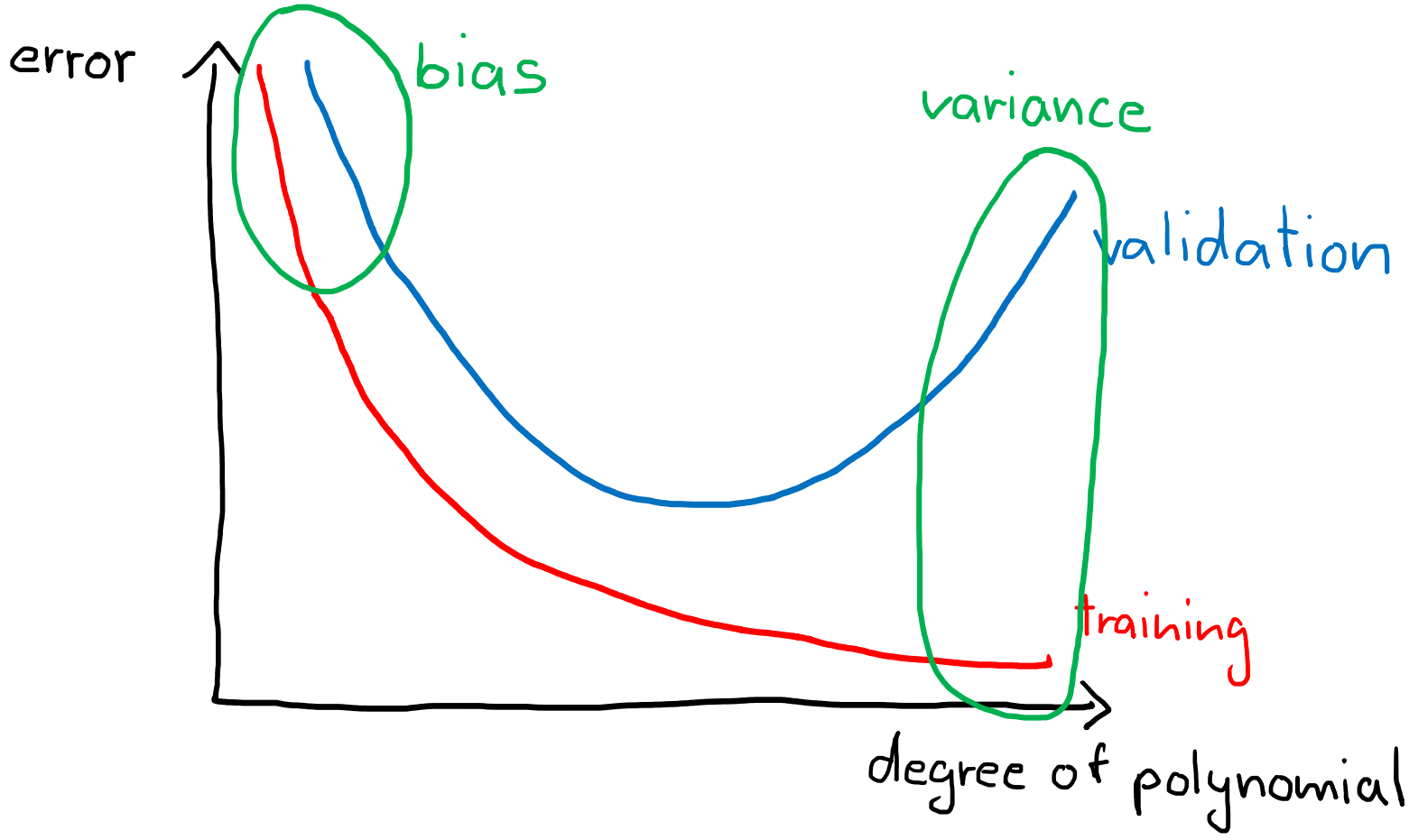
underfitting
"high bias"

just right

overfitting
"high variance"

Diagnosing Bias/Variance

Plotting training and validation prediction error



Regularisation

Add a penalty term on the coefficients (e.g., sum of squares)

Linear regression

$$J(\Theta) = \frac{1}{2m}\left[\sum_{i=1}^{m}\left(h_\Theta\left(\mathbf{x}^{(i)}\right) - y^{(i)}\right)^2 + \lambda\sum_{j=1}^{n}\Theta_j^2\right]$$

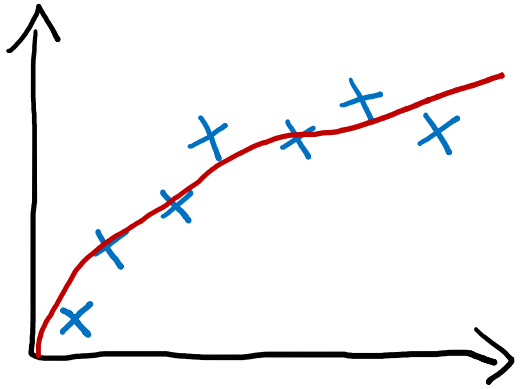
Logistic regression

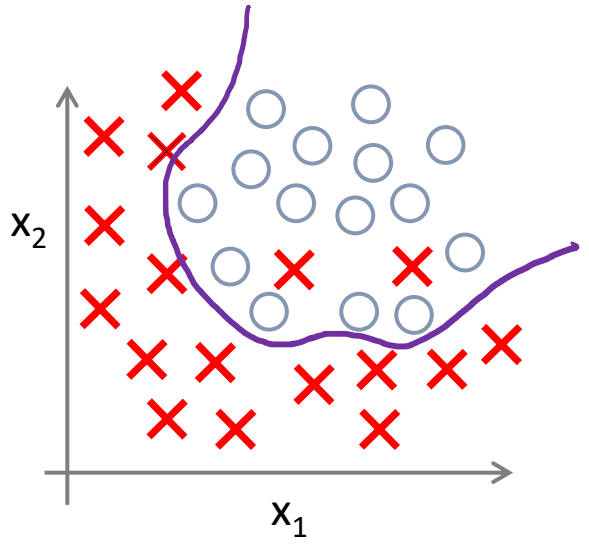
$$J(\Theta) = \frac{1}{m}\left[\sum_{i=1}^{m}\left(-y^{(i)}\log\left(h_\Theta\left(\mathbf{x}^{(i)}\right)\right) - (1-y^{(i)})\log\left(1-h_\Theta\left(\mathbf{x}^{(i)}\right)\right)\right) + \frac{\lambda}{2}\sum_{j=1}^{n}\Theta_j^2\right]$$

Gradient Descent with Regularisation

repeat until convergence

$$\Theta_j := \Theta_j - \alpha \left[\frac{1}{m} \sum_{i=1}^{m} \left(\left(h_\Theta\left(\mathbf{x}^{(i)}\right) - y^{(i)} \right) \cdot x_j^{(i)} \right) - \frac{\lambda}{m} \Theta_j \right]$$





$\Theta_0 + \Theta_1 x + \Theta_2 x^2$

$+ \Theta_3 x^3 + \Theta_4 x^4 \ldots$

Types of Regularisation

L1 regularisation (lasso* penalty)

- favours few non-zero coefficients

$$\left[\sum_{i=1}^{m}\left(h_\Theta\left(\mathbf{x}^{(i)}\right)-y^{(i)}\right)^2+\lambda\sum_{j=1}^{n}|\Theta_j|\right]$$

L2 regularisation (ridge penalty)

- favours small coefficients

$$\left[\sum_{i=1}^{m}\left(h_\Theta\left(\mathbf{x}^{(i)}\right)-y^{(i)}\right)^2+\lambda\sum_{j=1}^{n}\Theta_j^2\right]$$

Mixed L1/L2 regularisation (elastic net)

$$\left[\sum_{i=1}^{m}\left(h_\Theta\left(\mathbf{x}^{(i)}\right)-y^{(i)}\right)^2+\lambda_1\sum_{j=1}^{n}|\Theta_j|+\lambda_2\sum_{j=1}^{n}\Theta_j^2\right]$$

*least absolute shrinkage and selection operator

Debugging ML Algorithms

Debugging Machine Learning Algorithms

What to do if your ML method makes large errors?

- Get more training data
  - potentially very time-consuming/costly
- Change the feature set
  - remove features
  - add features
    (e.g. new features, combination of features)
- Change regularization (e.g. λ)
  - increase
  - decrease
- Change the model's flexibility (e.g. degree of polynomial)
  - increase
  - decrease

Source: https://xkcd.com/1838/

Use a systematic approach to debug your algorithms!

Machine Learning Diagnostic

- Get insights about what is/isn't working with a learning algorithm
- Diagnostics can take significant amount of time, but it's much better than guessing what the problems are

Possible diagnostics

- Evaluate prediction performance
  - Cross-validation, 70%/30%
  - Split data into train/test set
  - Define a performance measure
- Diagnose bias vs. variance (underfitting vs. overfitting)

Model Selection

- Fit hyper-parameters to test set, but then it’s not possible to report realistic test errors
- Instead, split data into training/validation/test set (60%/20%/20%)

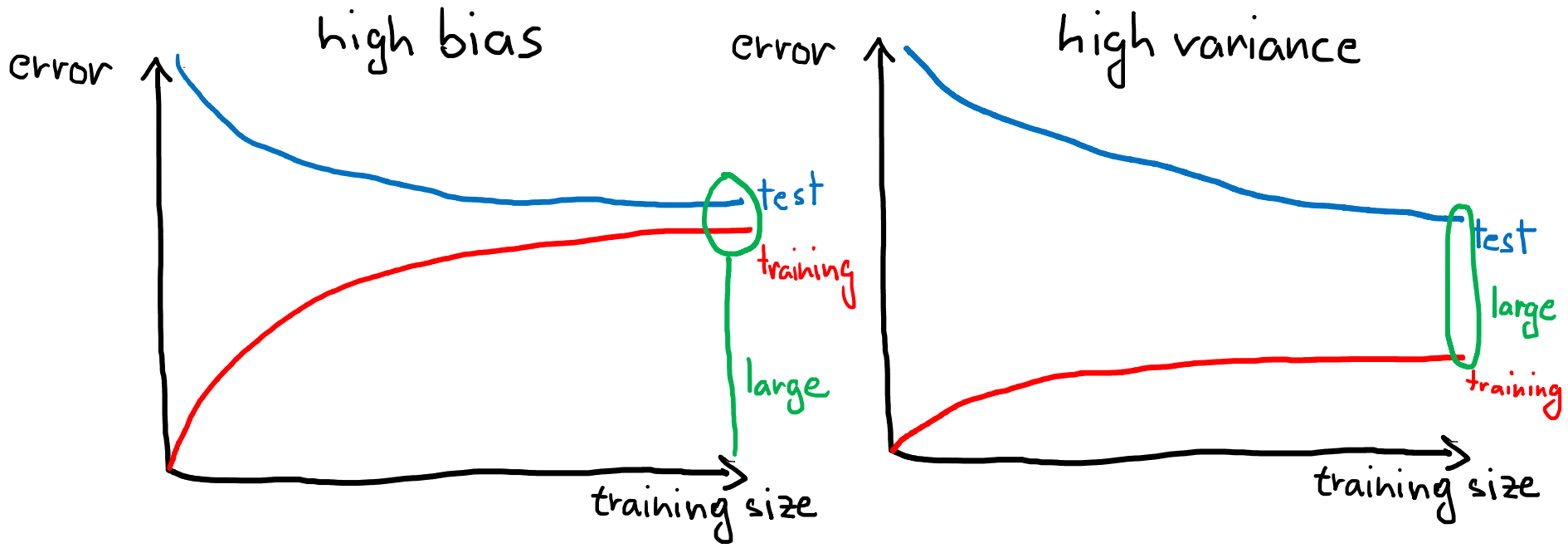
Examples

- degree of polynomial
- regularization parameter
- number of layers (neural networks)

Learning Curves

Plot errors vs. training size

- High bias: high training, high test error (more training data won't help)
- High variance: low training, high test error (more training data is likely to help)



Debugging Machine Learning Algorithms

What to do if your ML method makes large errors?

- Get more training data ✓
  - potentially very time-consuming/costly
- Change the feature set
  - remove features ✓
  - add features ✓
    (e.g. new features, combination of features)
- Change regularization (e.g. λ)
  - increase ✓
  - decrease ✓
- Change the model's flexibility (e.g. degree of polynomial)
  - increase ✓
  - decrease ✓

✓ fixes high bias

✓ fixes high variance