ENGN6528

Lecture 9B: Logistic Regression

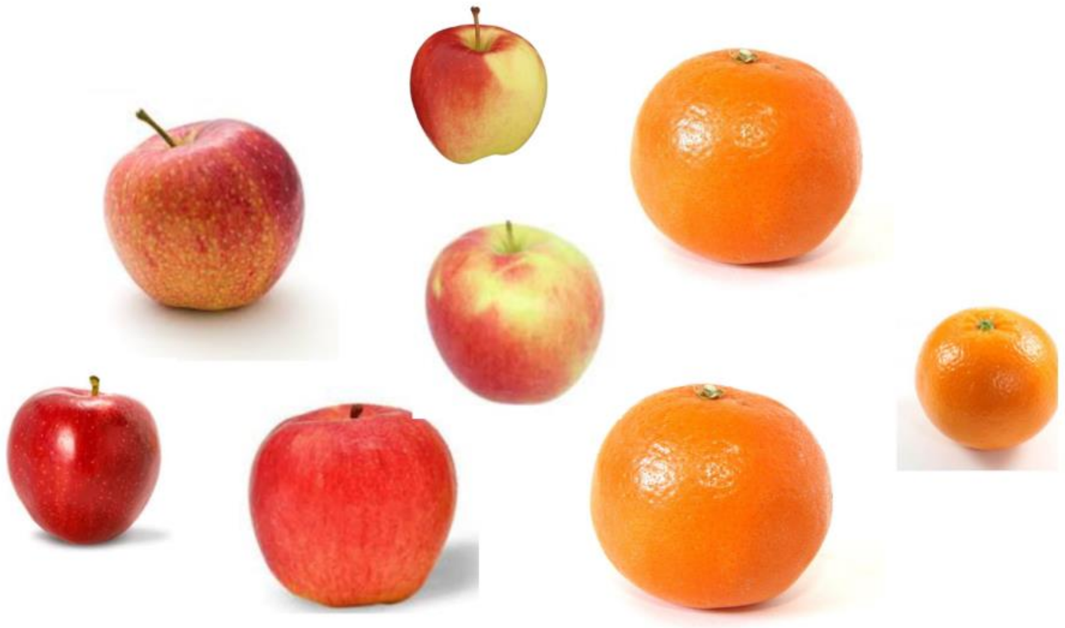
Thalaiyasingam Ajanthan

Slides from Andrew Ng, Miaomiao Liu and others.

Lecture schedule

Week	**Tuesday (2-hr)**	**Lecturer**	**Wednesday (1-hr)**	**Lecturer**
Week-9	3D vision	Richard	Logistic regression	Ajanthan
Week-10	Deep learning-A	Ajanthan	Deep learning-B	TBD
Week-11	Guest lecture: generative models	Fatemeh	Tutorial: deep learning	Lin
Week-12	Deep learning-C	Ajanthan	Review	Ajanthan

Example: Apple and Orange Classification



Visual Recognition Basics

Image formation
(+database+labels)

Capture and annotation.

Features
(saliency+description)

Feature Engineering

Classifier

(decision making)

Decision function

Visual Recognition

Binary classification

- Collect a set of images (apples and oranges) and label them (0: apple, 1: orange)
- Divide the dataset into training set, validation set and test set.
- Feature engineering (eg, bag of words)
- Classifier: logistic regression

$$\begin{aligned} &\text{if} \quad h_\theta(x) \geq 0.5, \quad \text{predict} \quad y = 1 \\ &\text{if} \quad h_\theta(x) < 0.5, \quad \text{predict} \quad y = 0 \end{aligned}$$

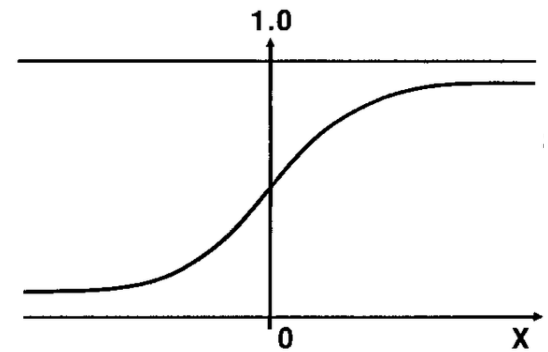
Logistic regression

$$\text{Want} \quad 0 \leq h_\theta(x) \leq 1$$

$$h_\theta(x) = g(\theta^T x) \quad \text{where} \quad g(z) = \frac{1}{1+e^{-z}}$$

g : sigmoid/logistic function

θ : parameters of the classifier



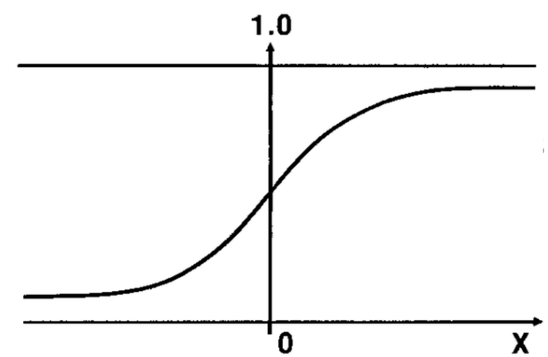
Logistic regression

Want $\quad 0 \leq h_\theta(x) \leq 1$

$$h_\theta(x) = g(\theta^T x) \quad \text{where} \quad g(z) = \frac{1}{1+e^{-z}}$$

g : sigmoid/logistic function

θ : parameters of the classifier



Objective: Learn the parameters θ using the training set.

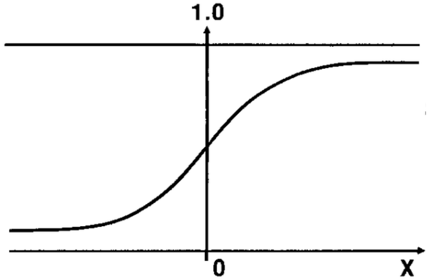
Interpretation: $h_\theta(x) = P(y=1|x;\theta)$

Decision Boundary

- What kind of a decision boundary can be learned by this classifier?

$$h_\theta(x) = g(\theta^T x) \geq 0.5 \quad \text{predict} \quad y = 1 \quad \text{otherwise} \quad y = 0$$

Decision Boundary



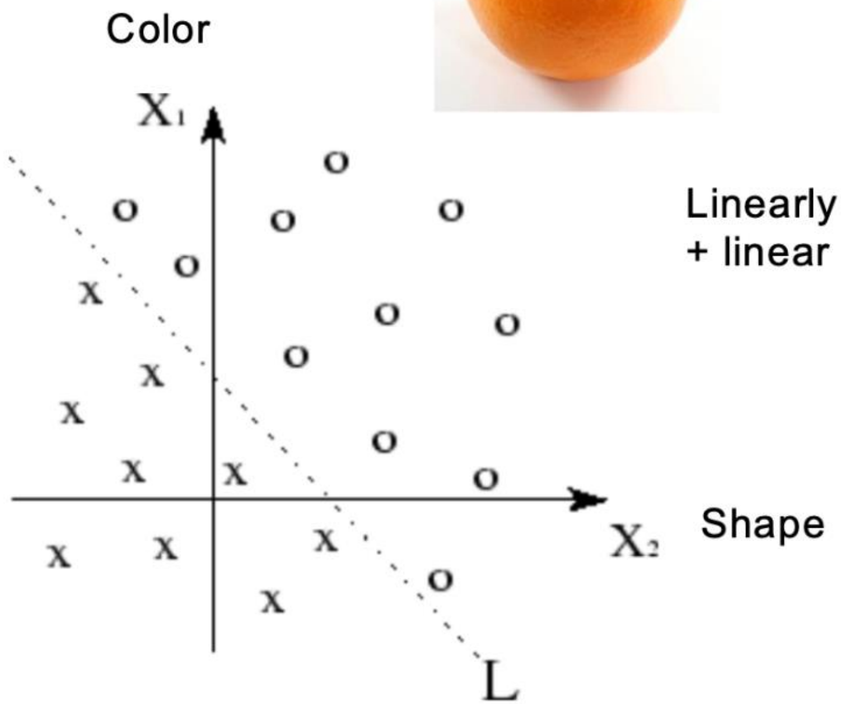
- What kind of a decision boundary can be learned by this classifier?

$$h_\theta(x) = g(\theta^T x) \geq 0.5 \quad \text{predict} \quad y = 1 \quad \text{otherwise} \quad y = 0$$

$$h_\theta(x) \geq 0.5 \quad \text{if} \quad \theta^T x \geq 0 \quad \text{otherwise} \quad \theta^T x < 0$$

- Learns a linear classifier

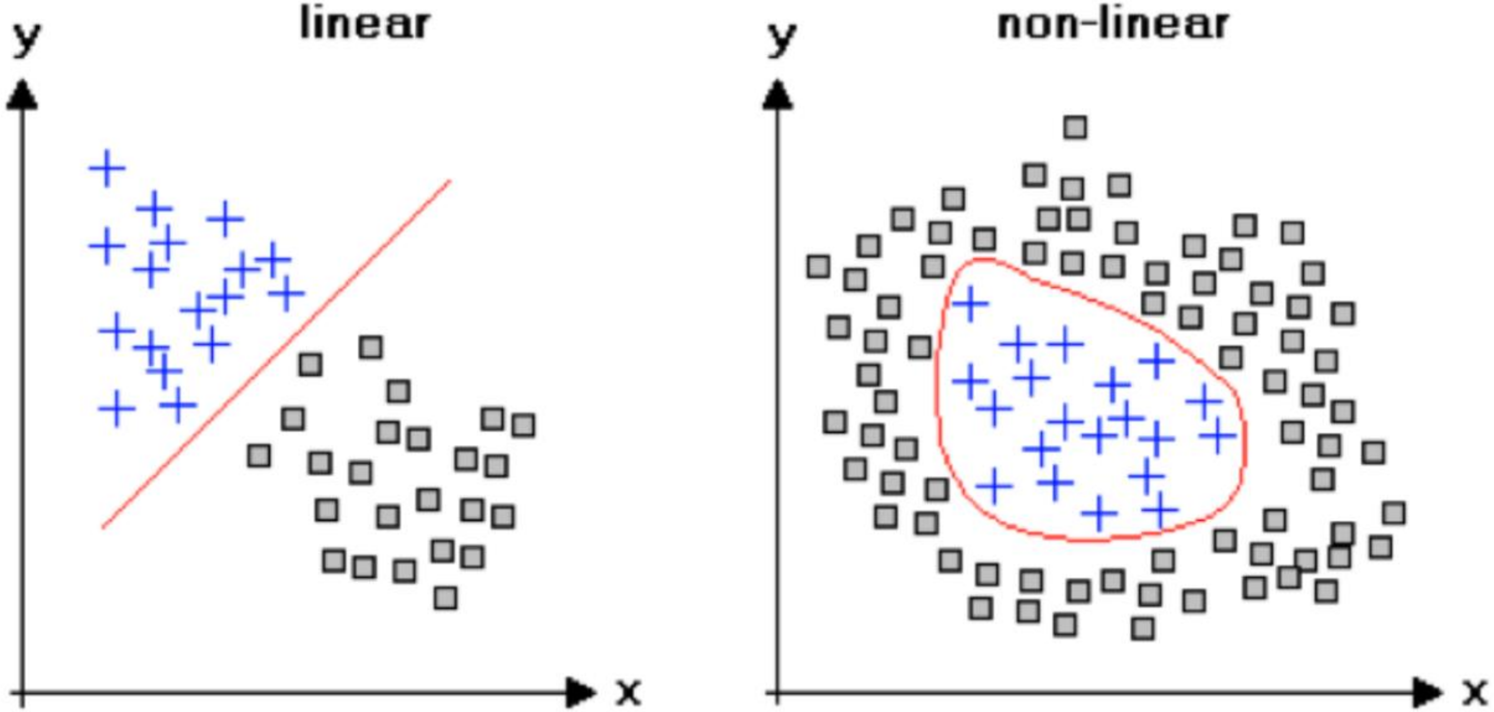
Decision Boundary



Linearly separable data
+ linear classifer = good.

Kawaguchi

Non-linearly separable case



$$h_\theta(x) = g(f(\theta, x)) \quad \text{where } f \text{ is a nonlinear function } x$$

Logistic regression

Training set: $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \cdots, (x^{(m)}, y^{(m)})\}$

m examples

$$x \in \underbrace{\begin{bmatrix} x_0 \\ x_1 \\ \dots \\ x_n \end{bmatrix}}_{} \quad \mathbb{R}^{n+1} \qquad x_0 = 1, y \in \{0, 1\}$$

$$h_\theta(x) = \frac{1}{1 + e^{-\theta^T x}}$$

How to choose parameters θ ?

Cost function

- For a given datapoint,

$$\mathrm{Cost}(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & \text{if } y = 1 \\ -\log(1 - h_\theta(x)) & \text{if } y = 0 \end{cases}$$

$$\text{if } y = 1 \text{ and } h_\theta(x) = 1 \Rightarrow \mathrm{Cost} = 0$$

$$\text{if } y = 1 \text{ and } h_\theta(x) \to 0 \Rightarrow \mathrm{Cost} \to \infty$$

Cost function

$$\mathrm{Cost}(h_\theta(x), y) = -y \log(h_\theta(x)) - (1-y)\log(1-h_\theta(x))$$

$$L(\theta; \mathcal{D}) = \frac{1}{m}\sum_{i=1}^{m} \mathrm{Cost}(h_\theta(x^{(i)}), y^{(i)})$$

Referred to as binary (sigmoid) cross entropy

Training: $\min_\theta L(\theta; \mathcal{D})$

Prediction: $h_\theta(x) = \dfrac{1}{1+e^{-\theta^T x}} = P(y=1|x;\theta)$

Training

$$L(\theta; \mathcal{D}) = -\frac{1}{m}\sum_{i=1}^{m}\Big(y^{(i)}\log h_\theta(x^{(i)}) + (1 - y^{(i)})\log(1 - h_\theta(x^{(i)}))\Big)$$

- Optimize the loss using gradient descent

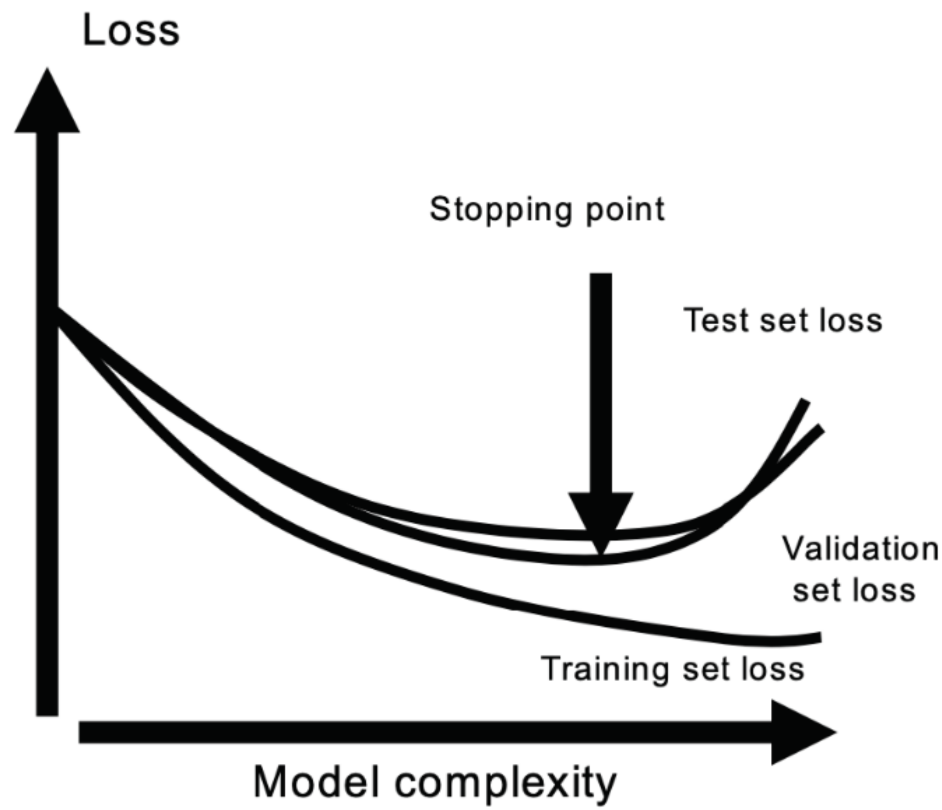
$$\theta^{t+1} = \theta^t - \alpha\nabla_\theta L(\theta^t; \mathcal{D})$$

- Stochastic (mini-batch) gradient descent

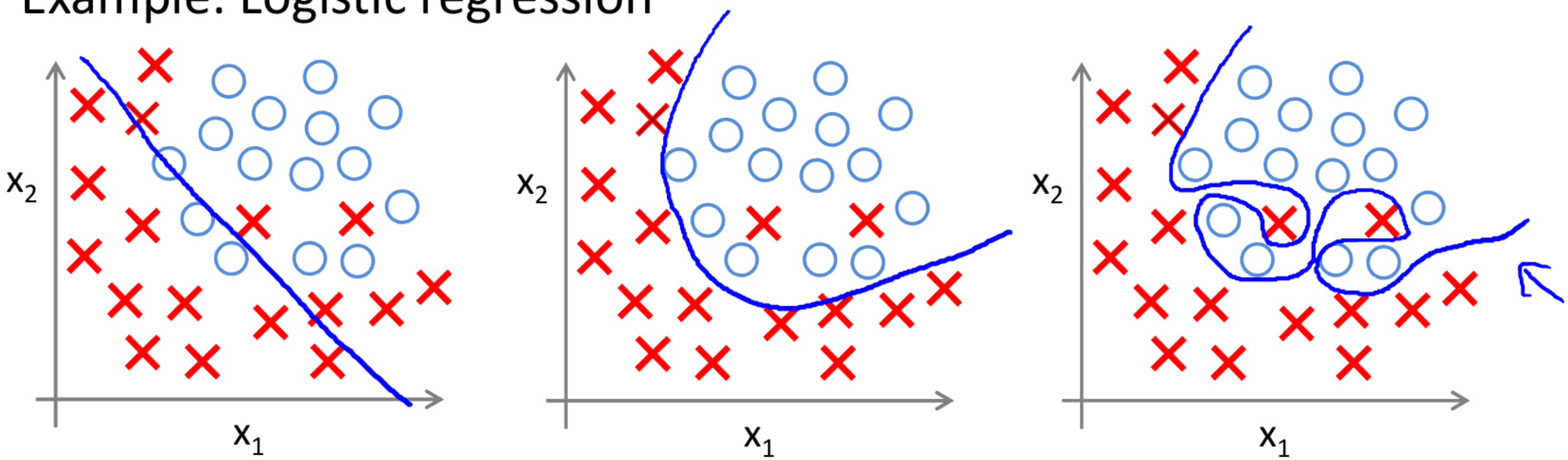
$$\theta^{t+1} = \theta^t - \alpha\nabla_\theta L(\theta^t; \mathcal{B}^t) \quad \text{where} \quad \mathcal{B}^t \subset \mathcal{D}$$

Overfitting

- Split the dataset
  - Training set
  - Validation set
  - Test set
- Use training set to **optimize** model parameters
- Use validation test to **choose** the best model
- Use test set only to **measure** the expected loss



Example: Logistic regression



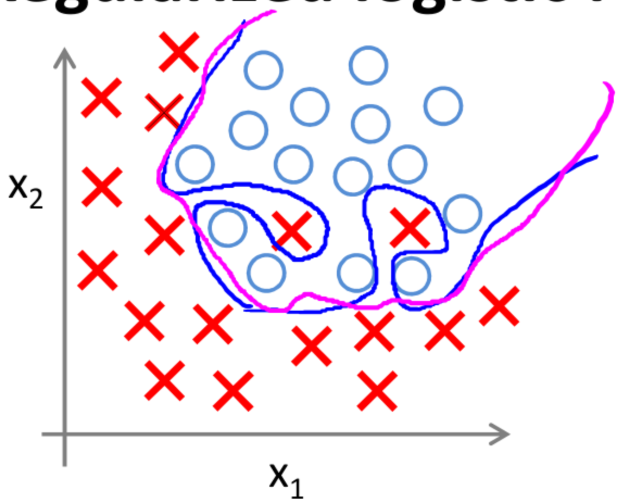
$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$

(g = sigmoid function)

$$g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2 + \theta_5 x_1 x_2)$$

$$g(\theta_0 + \theta_1 x_1 + \theta_2 x_1^2 + \theta_3 x_1^2 x_2 + \theta_4 x_1^2 x_2^2 + \theta_5 x_1^2 x_2^3 + \theta_6 x_1^3 x_2 + \ldots)$$

Regularized logistic regression.



$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_1^2 + \theta_3 x_1^2 x_2 + \theta_4 x_1^2 x_2^2 + \theta_5 x_1^2 x_2^3 + \dots)$$

Cost function:

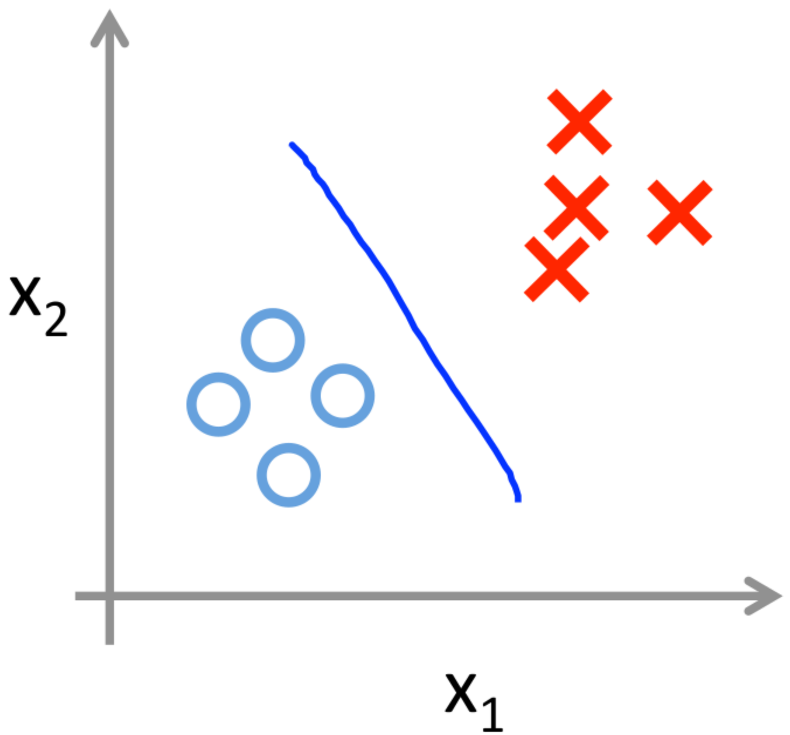
$$J(\theta) = -\left[\frac{1}{m}\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1-y^{(i)}) \log(1 - h_\theta(x^{(i)}))\right]$$

$$+ \frac{\lambda}{2m} \sum_{j=1}^{n} \theta_j^2$$

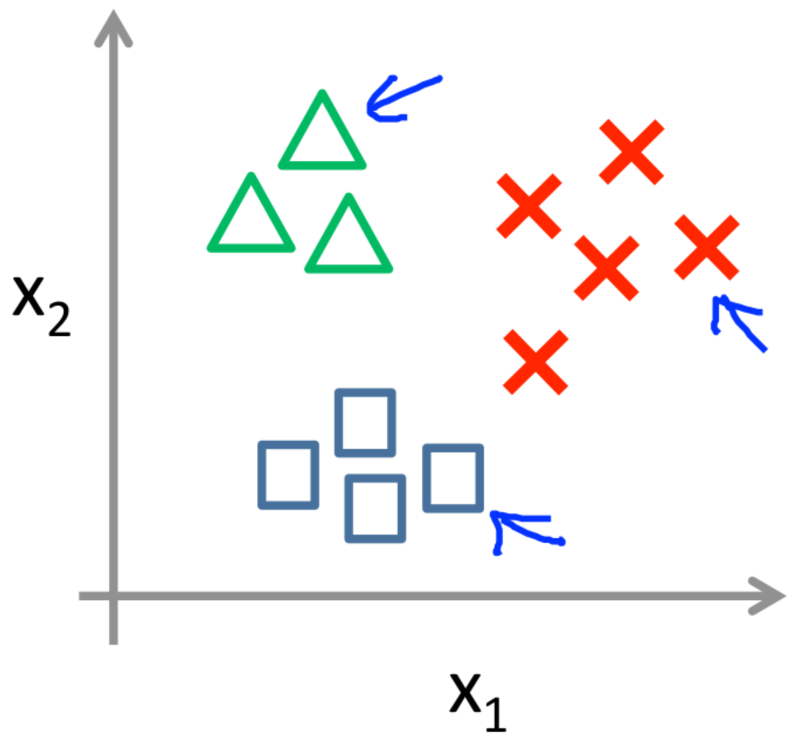
$\boxed{\theta_1, \theta_2, \dots, \theta_n}$

Multi-class classification

Binary classification:

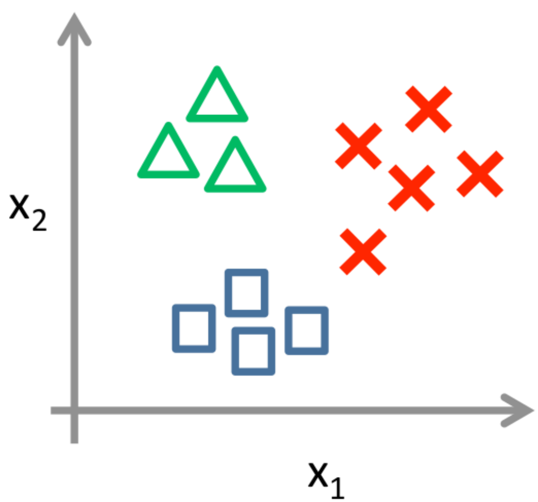


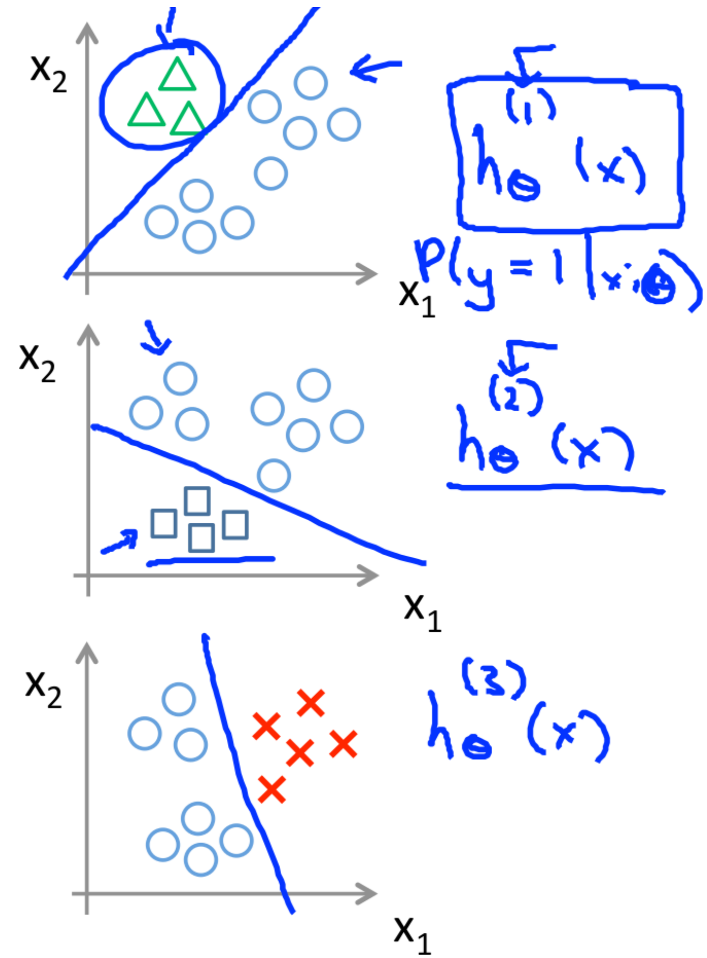
Multi-class classification:



Multi-class classification

One-vs-all (one-vs-rest):





Class 1: △

Class 2: □

Class 3: ×

$$\boxed{h_\theta^{(i)}(x) = P(y = i|x;\theta) \qquad (i = 1, 2, 3)}$$

Multi-class classification

One-vs-all

Train a logistic regression classifier $h_\theta^{(i)}(x)$ for each class i to predict the probability that $y = i$.

On a new input x, to make a prediction, pick the class i that maximizes

$$\max_i h_\theta^{(i)}(x)$$

Multi-class classification

Training set: $(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})$

$y^{(i)}$ one of $\begin{bmatrix}1\\0\\0\\0\end{bmatrix}$, $\begin{bmatrix}0\\1\\0\\0\end{bmatrix}$, $\begin{bmatrix}0\\0\\1\\0\end{bmatrix}$, $\begin{bmatrix}0\\0\\0\\1\end{bmatrix}$ $(x^{(i)}, y^{(i)})$

pedestrian car motorcycle truck

Loss: softmax cross entropy $\text{Cost}(h_\Theta(x), y) = -\sum_{k=1}^{K} y_k \log h_\Theta^k(x)$

$$h_\Theta^k(x) = \text{softmax}(\Theta_k^T x)$$

$$\text{softmax}(z_j) = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}$$

Recap: Logistic regression

- A binary classifier (linear/nonlinear) based on the sigmoid function
- Uses hand-designed features
- Optimized via (stochastic) gradient descent
- Can be extended to multi-class in a one-vs-all approach using the softmax function
- Use regularization to reduce overfitting

Deep learning for classification (next week)

- Classifier: logistic regression
- ~~Uses hand designed features~~ Learn features from data
- Optimized via (stochastic) gradient descent
- Can be extended to multi-class in a one-vs-all approach using the softmax function
- Use regularization, data augmentation, etc. to reduce overfitting