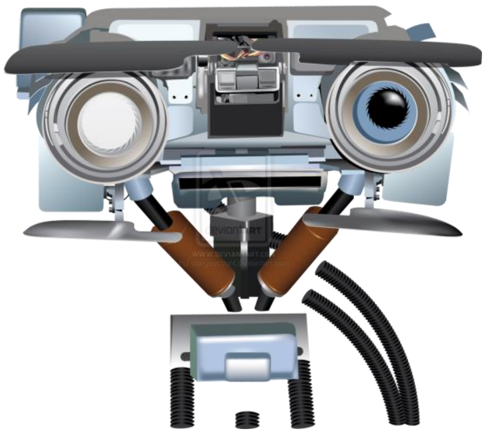
Today

- Learning from Examples Cont’d

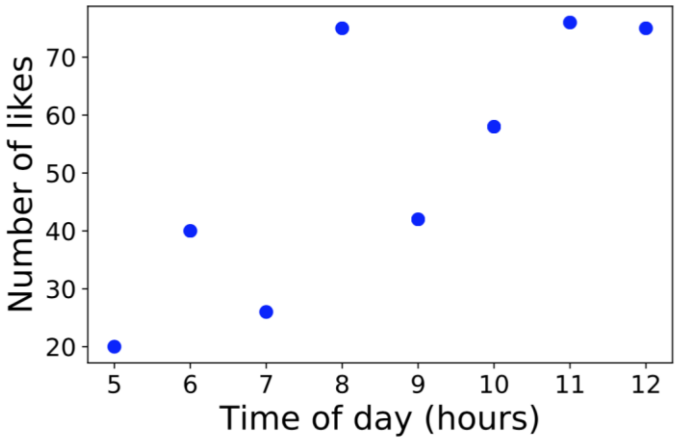
- Announcement:
  Pre-lecture Material posted for Thu Sept 17

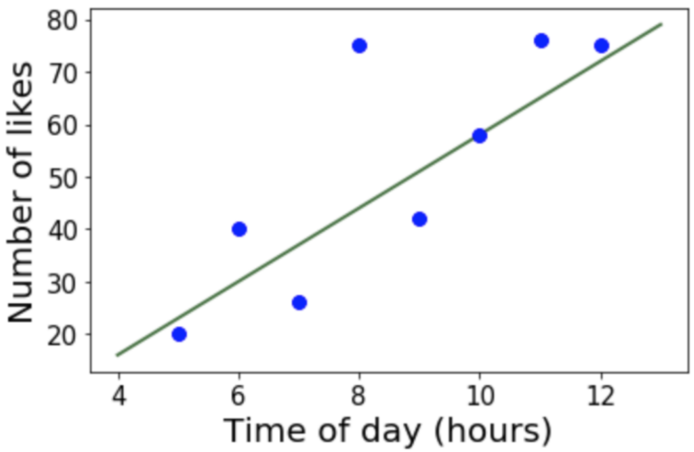


Supervised Learning: Linear Regression

Example of Supervised Learning: Linear Regression

- When the label is a real number
- Training a **model** to find a relationship between input and output values
- Learning a **line of best fit**



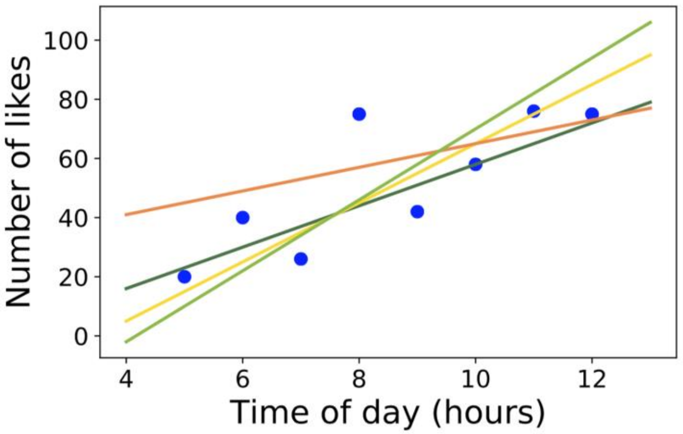


Linear Regression: Model Parameters

- Learning a best fit line means learning parameters (or weights) for our model.

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

θ_i's: Parameters



Linear Regression: Cost Function

- How do we know that the model parameters result in a "best fit" line? If parameter values minimize our cost function.

Cost Function:

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

SSD = sum of squared differences, also
SSE = sum of squared errors

Multivariate Linear Regression

Hypothesis:

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$

For convenience of notation, define $x_0 = 1$.

θ_i 's: Parameters

Cost Function:

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

Goal: $\underset{\theta_0, \theta_1, \ldots, \theta_n}{\text{minimize}} \; J(\theta_0, \theta_1, \ldots, \theta_n)$ **How??**

Two potential solutions

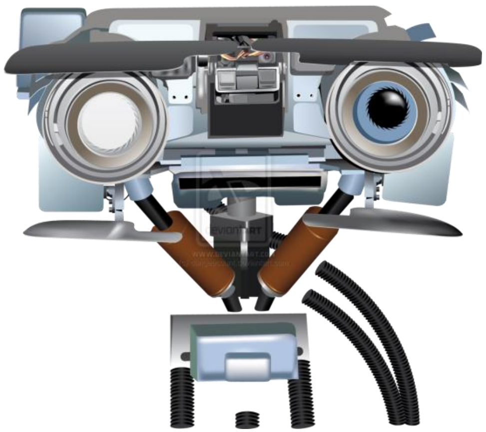
$$\min_{\theta} J(\theta; x^{(1)}, y^{(1)}, \ldots, x^{(m)}, y^{(m)})$$

Direct minimization

- Take derivative, set to zero
- Not possible for most "interesting" cost functions

Gradient descent (or other iterative algorithm)

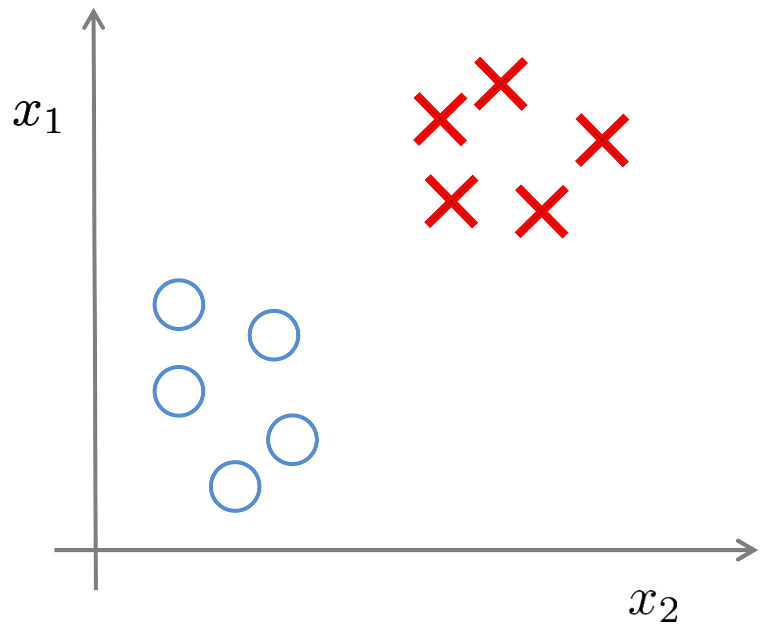
- Start with a guess for θ
- Change θ to decrease $J(\theta)$
- Until reach minimum



Supervised Learning: Classification

Tumor: Malignant / Benign?
Email: Spam / Not Spam?
Video: Viral / Not Viral?

Supervised learning

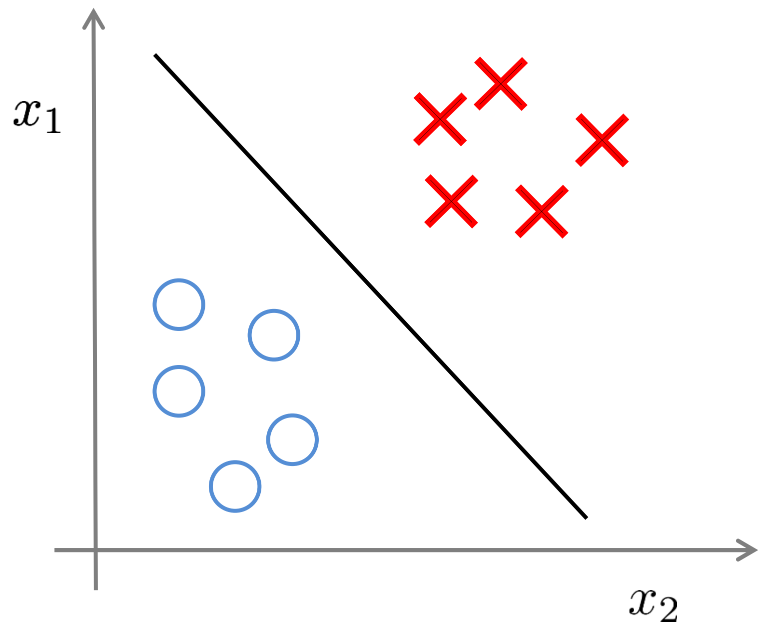


Training set: $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), (x^{(3)}, y^{(3)}), \ldots, (x^{(m)}, y^{(m)})\}$

$$y \in \{0,1\}$$

Tumor: Malignant / Benign?
Email: Spam / Not Spam?
Video: Viral / Not Viral?

Goal of Supervised learning



Training set: $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), (x^{(3)}, y^{(3)}), \ldots, (x^{(m)}, y^{(m)})\}$

$y \in \{0,1\}$

slide credit: Andrew Ng

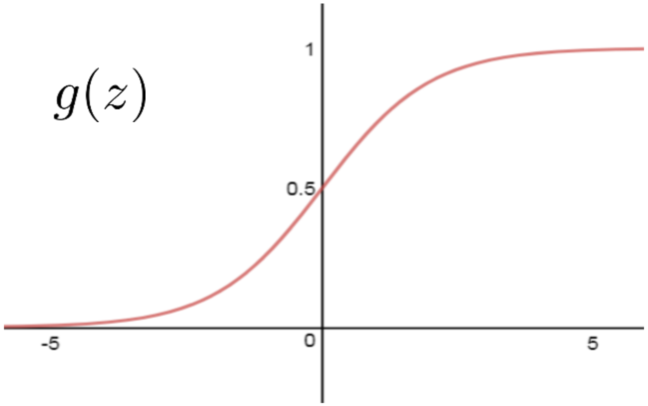
Logistic Regression

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

map to (0, 1) with "sigmoid" function

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

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



$$h_\theta(x) = p(y = 1|x)$$ "probability of class 1 given input"

Logistic Regression

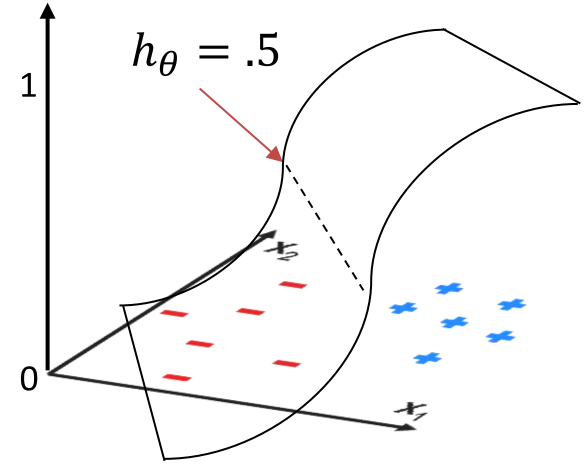
Hypothesis:

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

Predict "$y = 1$" if $h_\theta(x) \geq 0.5$

Predict "$y = 0$" if $h_\theta(x) < 0.5$

"decision boundary"



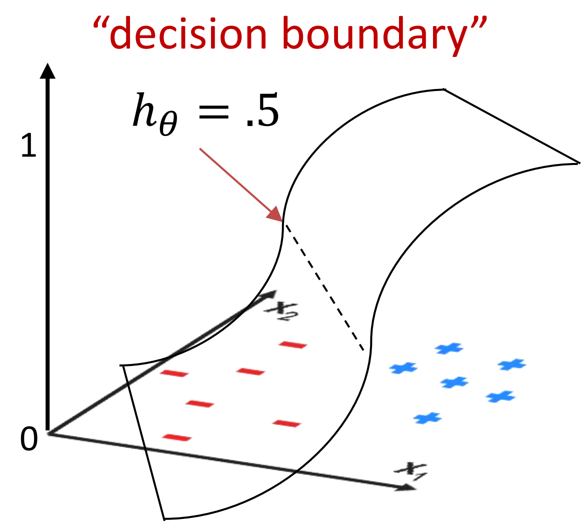
Logistic Regression Cost Function

Hypothesis:

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

θ: parameters

$D = \left(x^{(i)}, y^{(i)}\right)$: data



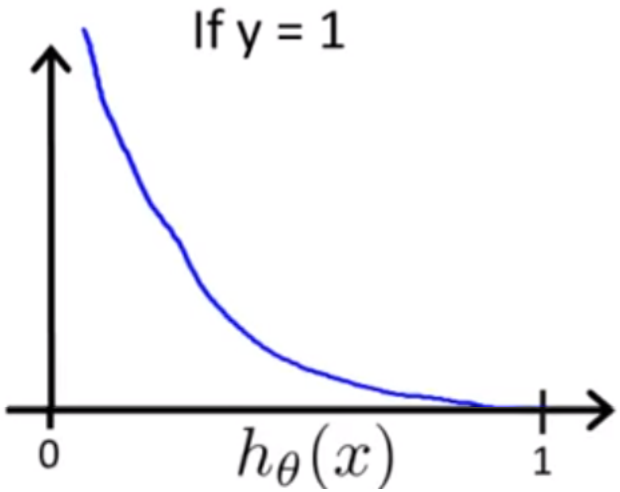
Cost Function: cross entropy

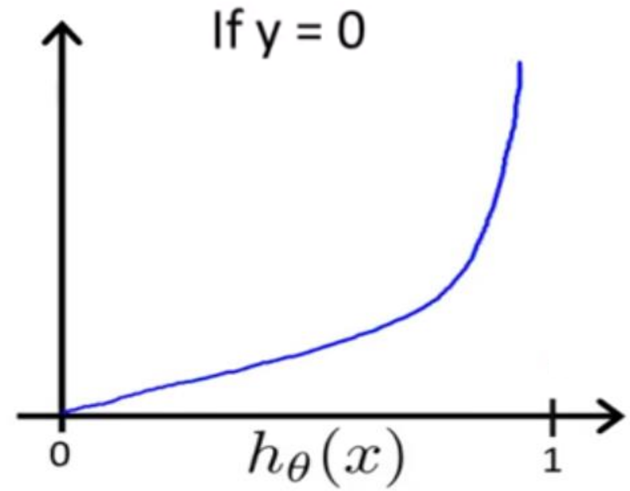
$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m} \mathrm{Cost}(h_\theta(x^{(i)}), y^{(i)})$$

Goal: minimize cost $\quad \min_\theta J(\theta)$

Logistic Regression Cost Function

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





$\text{Cost} = 0\ if\ y^{(i)} = 1,\ h_\theta\left(x^{(i)}\right) = 1$

Similarly desirable behavior

But as $h_\theta\left(x^{(i)}\right) \to 0$

$\text{Cost} \to \infty$

[Andrew Ng's Illustration]

Cross Entropy Cost

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

Can be written more compactly as:

$$\text{Cost}\left(h_\theta(x^{(i)}), y^{(i)}\right) = -y^{(i)} \log h_\theta(x^{(i)}) - (1 - y^{(i)}) \log\left(1 - h_\theta(x^{(i)})\right)$$

Cross Entropy Cost

- Cross entropy compares distribution *q* to reference *p*

$$H(p,q) = -\sum_x p(x)\, \log q(x).$$

- Here q is predicted probability of $y = 1$ given x, reference distribution is $p = y^{(i)}$, which is either *1 or 0*

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

Gradient of Cross Entropy Cost

- Cross entropy cost

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

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

- No direct closed-form solution

- its gradient w.r.t θ is:

$$\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)}$$

Gradient descent for Logistic Regression

Cost

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

Want $\min_\theta J(\theta)$:

Repeat $\{$

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

(simultaneously update all θ_j)

$\}$

Gradient descent for Logistic Regression

Cost

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

Want $\min_\theta J(\theta)$:

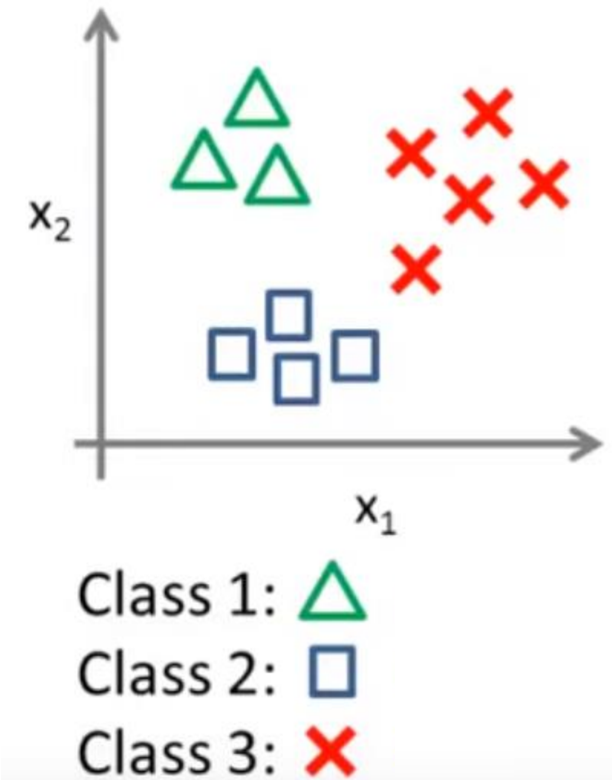
Repeat $\{$

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

(simultaneously update all θ_j)

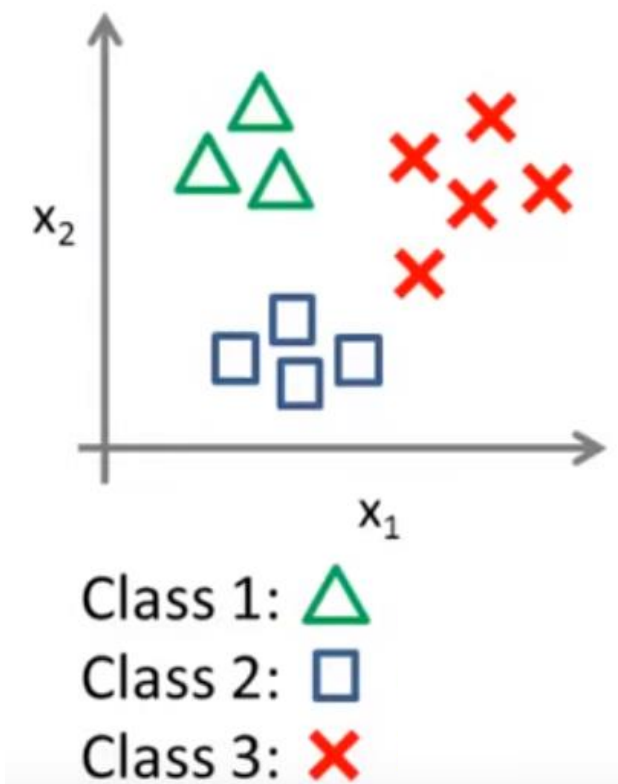
$\}$

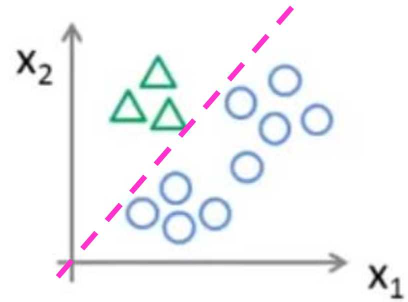
Logistic Regression for Multi-class Classification



Logistic Regression for Multi-class Classification

One-vs-all (One-vs-rest)



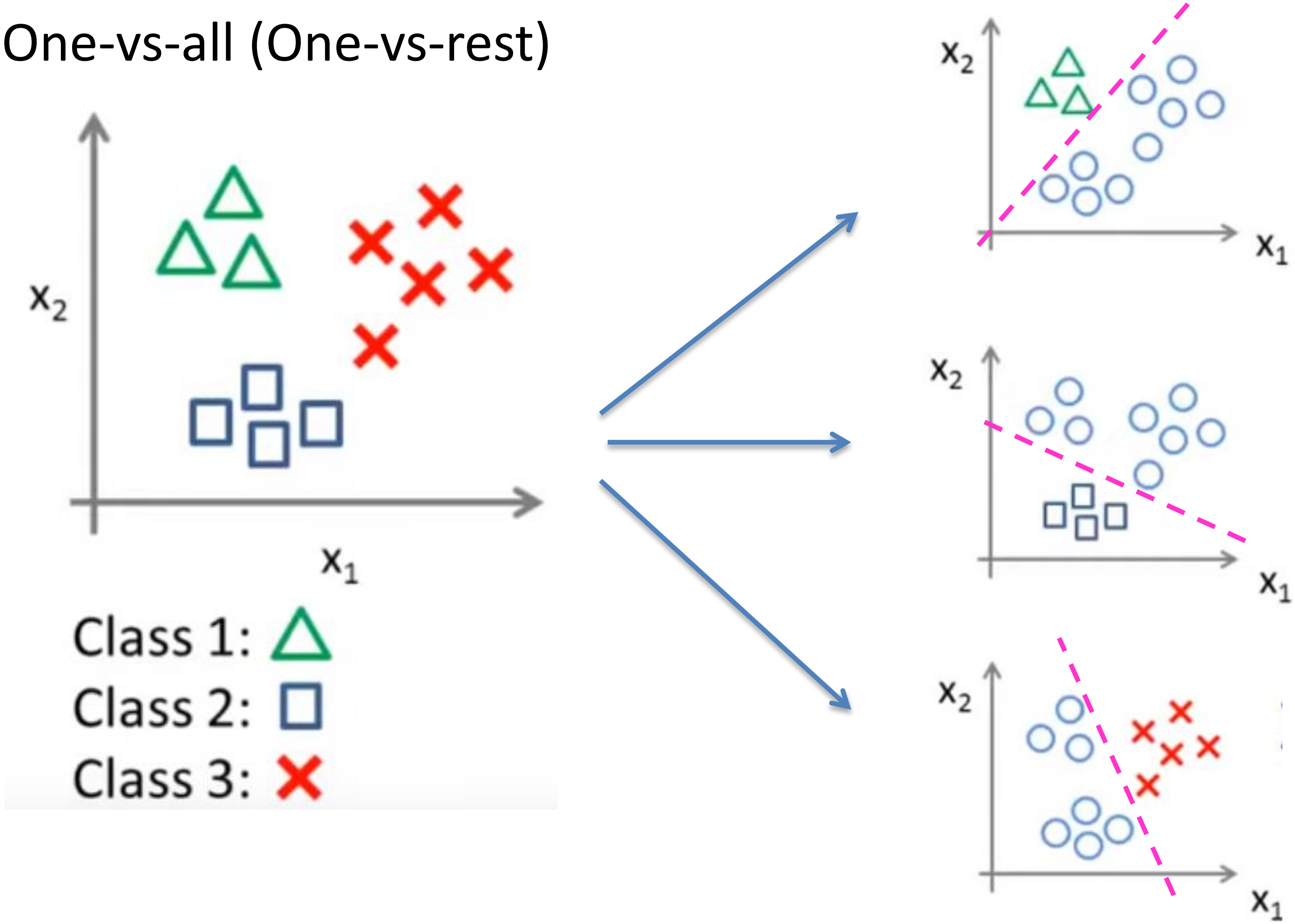


$h_\theta^1(x)$

$P(y = 1|x)$

Logistic Regression for Multi-class Classification

One-vs-all (One-vs-rest)



$h_\theta^1(x)$
$P(y=1|x)$

$h_\theta^2(x)$
$P(y=2|x)$

$h_\theta^3(x)$
$P(y=3|x)$

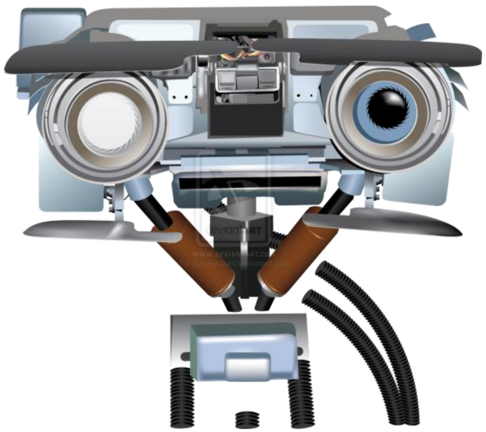
Andrew Ng

Logistic Regression for Multi-class Classification

- Trained 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:

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

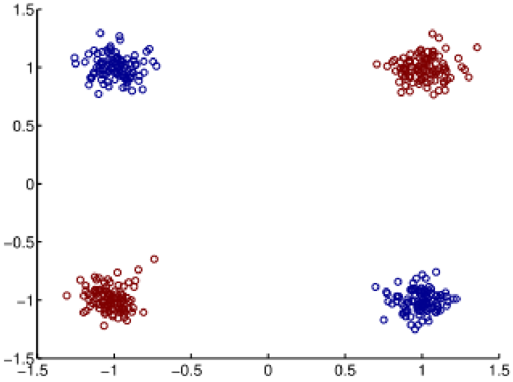


Supervised Learning:

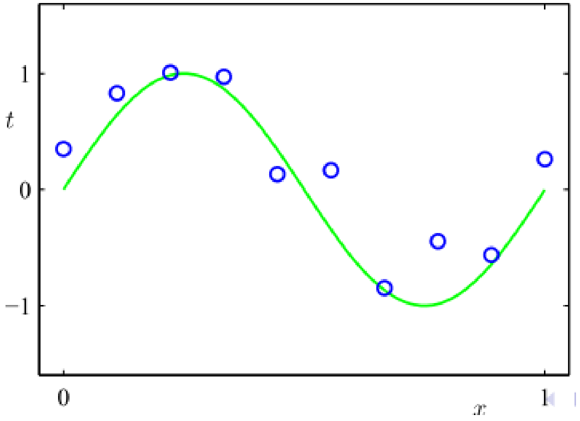
Non-linear features

What to do if data is nonlinear?

Example of nonlinear classification



Example of nonlinear regression

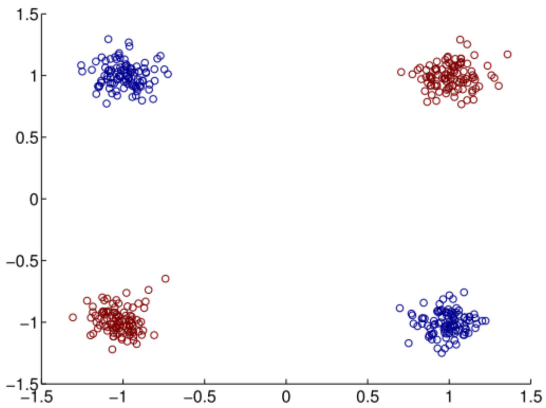


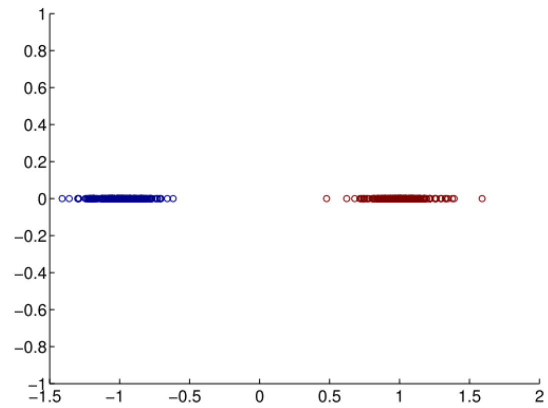
Nonlinear basis functions

Transform the input/feature

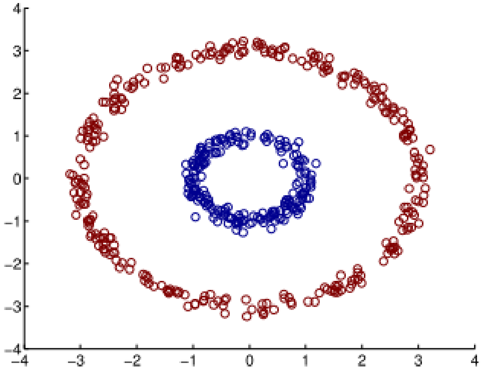
$$\phi(x) : x \in R^2 \;\rightarrow\; z \;=\; x_1 \cdot x_2$$

Transformed training data: linearly separable!



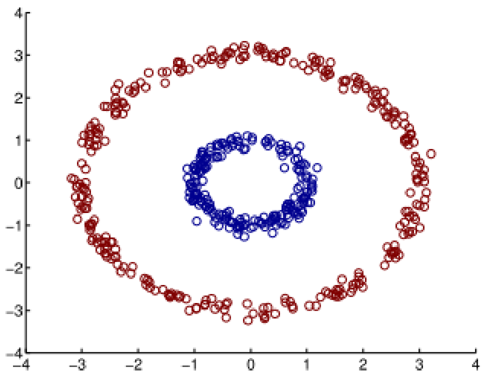


Another example



How to transform the input/feature?

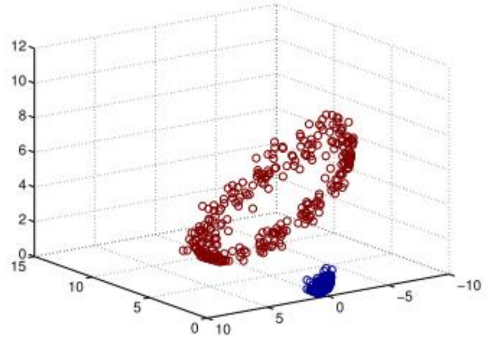
Another example



How to transform the input/feature?

$$\phi(x): x \in R^2 \;\rightarrow\; z \;= \begin{bmatrix} {x_1}^2 \\ x_1 \cdot x_2 \\ {x_2}^2 \end{bmatrix}$$

Transformed training data: linearly separable



Intuition: suppose $\theta \;= \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$

Then $\theta^T z = {x_1}^2 + {x_2}^2$

i.e., the sq. distance to the origin!

Non-linear basis functions

- We can use a nonlinear mapping, or basis function

$$\phi(x) : x \in R^N \ \rightarrow \ z \in R^M$$

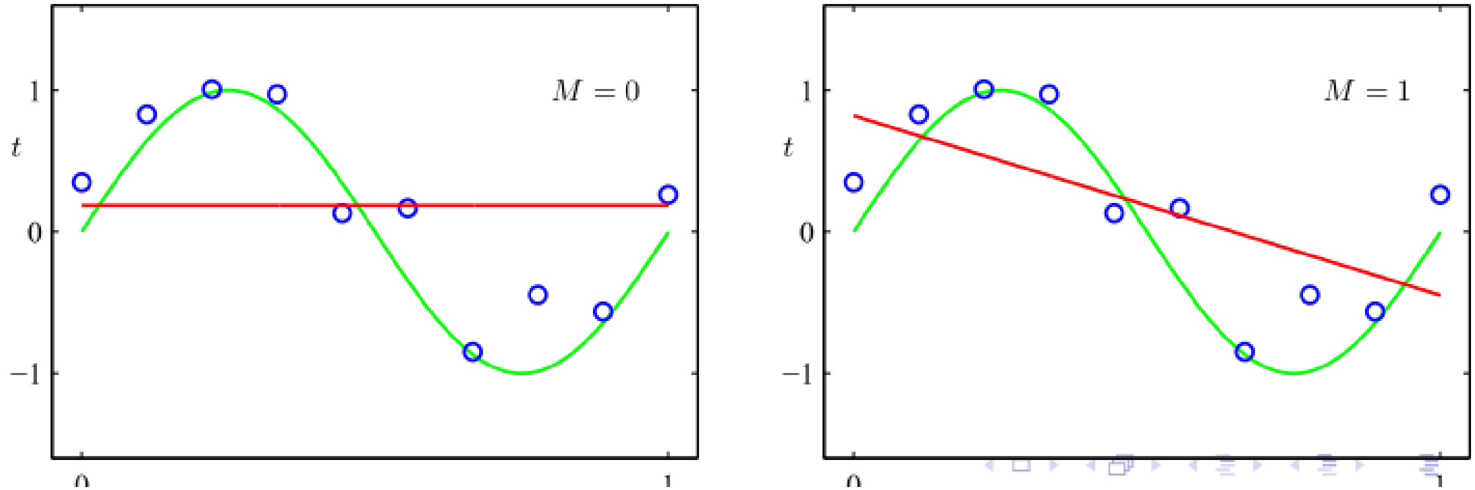
- where M is the dimensionality of the new feature/input z (or $\phi(x)$)
- Note that M could be either greater than N or less than, or the same

Example with regression

Polynomial basis functions

$$\phi(x) = \begin{bmatrix} 1 \\ x \\ x^2 \\ \vdots \\ x^M \end{bmatrix}$$

Fitting samples from a sine function: *underrfitting* as $f(x)$ is too simple



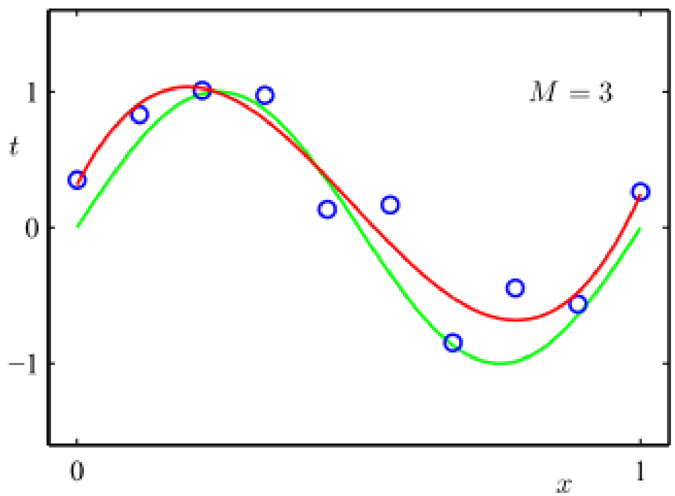
Add more polynomial basis functions

Polynomial basis functions

$$\phi(x) = \begin{bmatrix} 1 \\ x \\ x^2 \\ \vdots \\ x^M \end{bmatrix}$$

Being too adaptive leads to better results on the training data, but not so great on data that has not been seen!

M=3 *good fit*



M=9: *overfitting*