CS1138

Machine Learning

Lecture : Classification and Logistic Regression

(Slide Credits: Andrew Ng)

Arpan Gupta

Classification

Email: Spam / Not Spam?
Online Transactions: Fraudulent (Yes / No)?
Tumor: Malignant / Benign ?

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

0: "Negative Class" (e.g., benign tumor)
1: "Positive Class" (e.g., malignant tumor)

Binary Classification

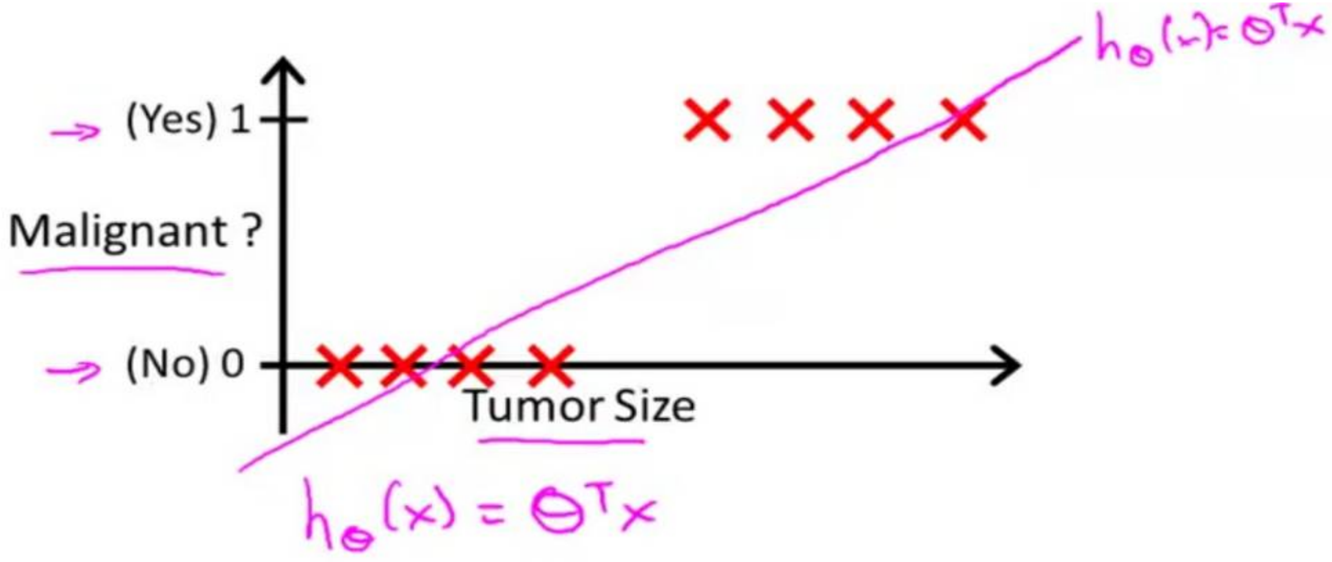
Multiclass Classification: where more than 2 classes are present.
Eg. $y \in \{0, 1, 2, 3, 4\}$

Example: Malignant/Benign Tumour based on size

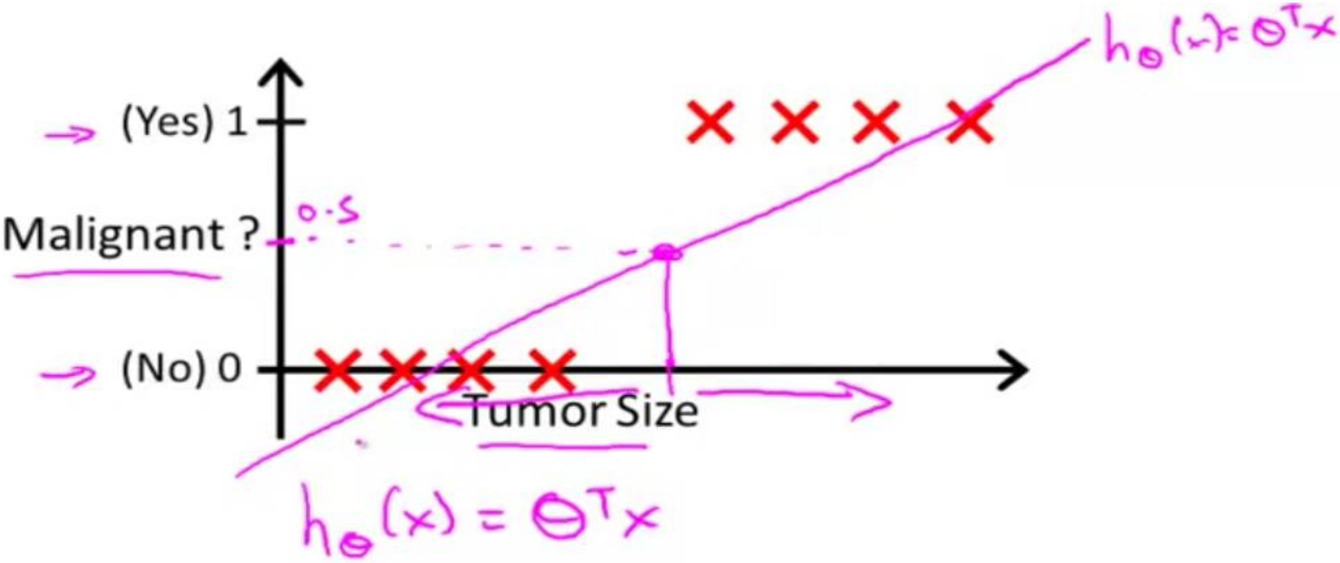
Positive class
samples

Negative class
samples

A way to use linear regression for classification?



A way to use linear regression for classification?

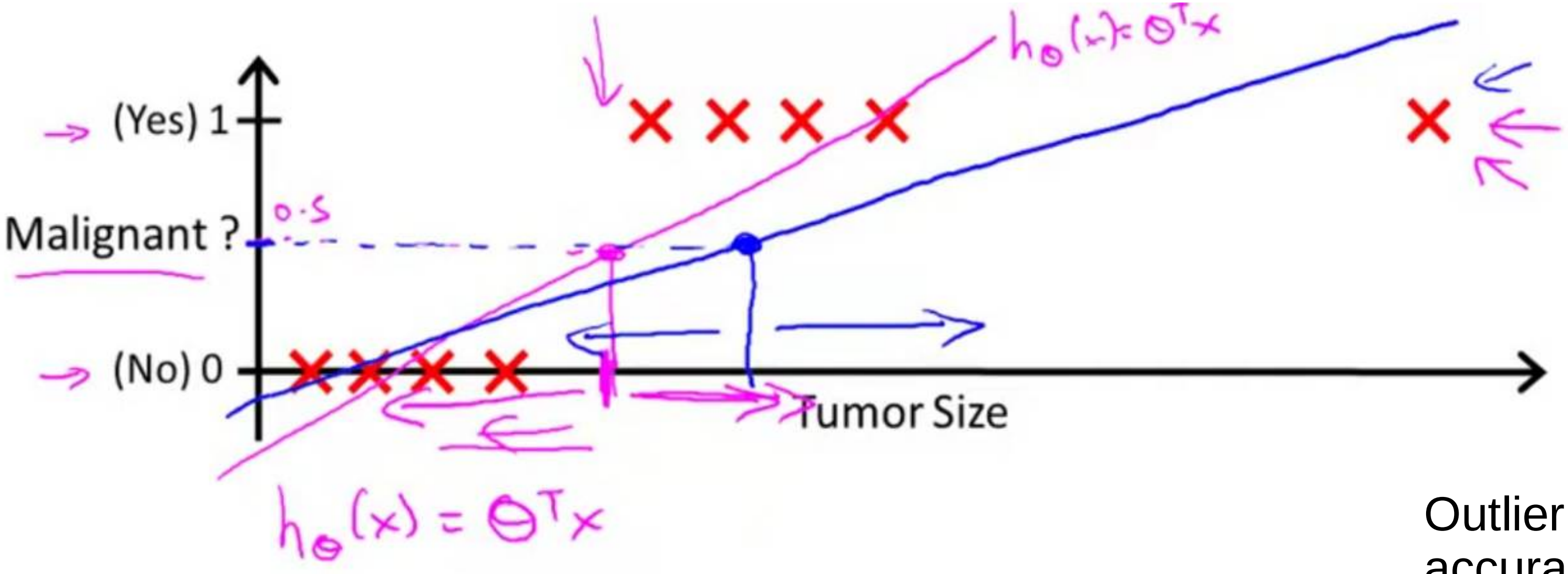


Threshold classifier output $h_\theta(x)$ at 0.5:

If $h_\theta(x) \geq 0.5$, predict "y = 1"

If $h_\theta(x) < 0.5$, predict "y = 0"

A way to use linear regression for classification?



→ Threshold classifier output $h_\theta(x)$ at 0.5:

If $h_\theta(x) \geq 0.5$, predict "y = 1"

If $h_\theta(x) < 0.5$, predict "y = 0"

Outlier effects the accuracy of classification. Therefore, not a good idea to use LR for classification.

A way to use linear regression for classification? Good or Bad?

- While using regression (for a classification task), we may not always get a hypothesis that works well. We may, but often it is not a good idea to apply linear regression hypothesis to a classification task.

- For classification: $y = 0$ or 1
  For linear regression, $h_\Theta(x)$ can be > 1 or < 0

- To overcome this, we use logistic regression:

  Logistic Regression: $0 <= h_\Theta(x) <= 1$

- **Logistic Regression is actually a classification algorithm, not a regression algorithm.**

Logistic Regression Model

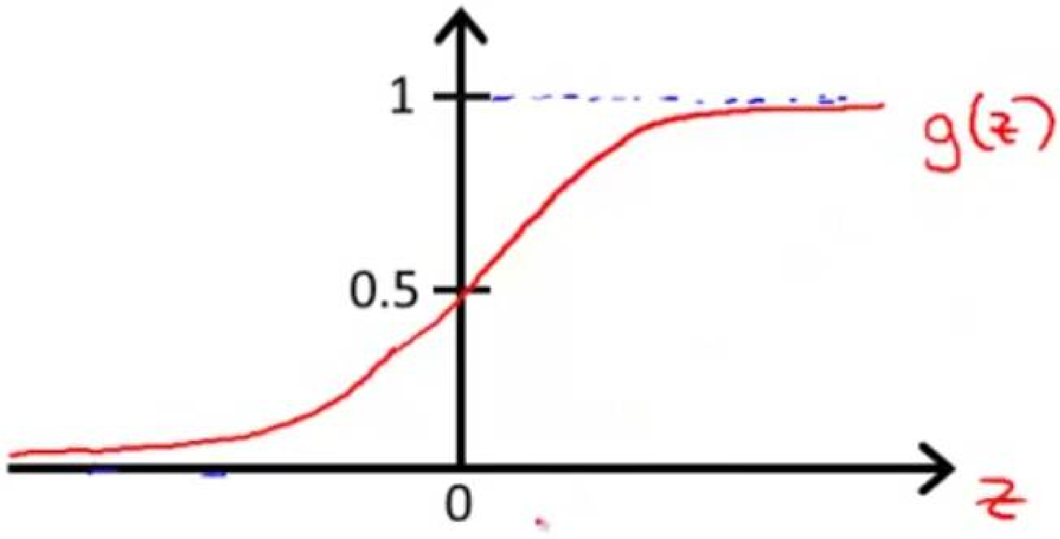
- We want: $0 \;<=\; h_\Theta(x) \;<=\; 1$
- For linear regression: $h_\Theta(x) = \Theta^T x$
- For logistic regression: $h_\Theta(x) = g(\Theta^T x)$

Where $$g(z) = \frac{1}{1+e^{-z}} \qquad h_\Theta(x) = \frac{1}{1+e^{-\Theta^T x}}$$

- g(z) is known as the **sigmoid** or **logistic** function.

Sigmoid / Logistic Function

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



The new hypothesis function h(x) can be interpreted as a probability that y = 1 on input x.

Interpretation of Hypothesis Function

$h_\theta(x)$ = estimated probability that $\boxed{y = 1}$ on input x ←

Example: If $x = \begin{bmatrix} x_0 \\ x_1 \end{bmatrix} = \begin{bmatrix} 1 \\ \text{tumorSize} \end{bmatrix}$

$$h_\theta(x) = 0.7$$

Tell patient that 70% chance of tumor being malignant

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

"probability that y = 1, given x, parameterized by θ"

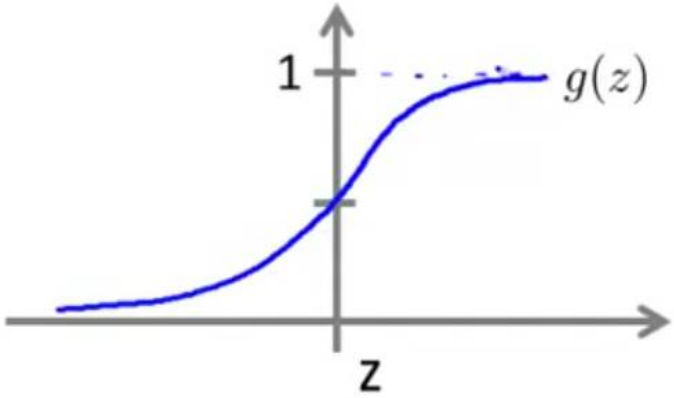
$y = 0$ or 1

$$P(y=0|x;\theta) + P(y=1|x;\theta) = 1$$
$$P(y=0|x;\theta) = 1 - P(y=1|x;\theta)$$

Logistic regression

$\rightarrow h_\theta(x) = g(\theta^T x) = P(y=1|x;\theta)$

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



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

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

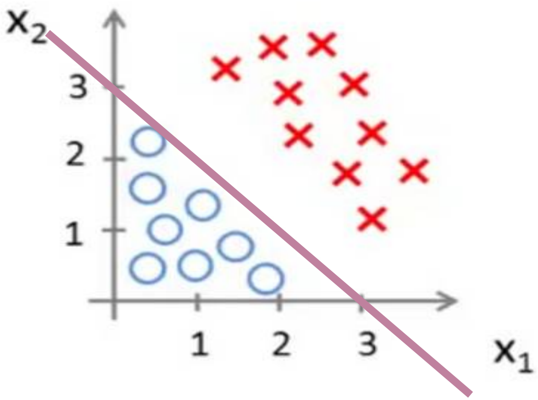
$g(z) \geq 0.5$
when $z \geq 0$
$h_\theta(x) = g(\theta^T x) \geq 0.5$
whenever $\underbrace{\theta^T x}_{z} \geq 0$

Logistic Regression

- Predict "y = 1", when $\Theta^T x >= 0$, therefore, $g(\Theta^T x) >= 0.5$

- Predict "y = 0", when $\Theta^T x < 0$, therefore, $g(\Theta^T x) < 0.5$

Decision Boundary



$$\theta = \begin{bmatrix}-3\\1\\1\end{bmatrix}$$

$$\rightarrow h_\theta(x) = g(\underset{-3}{\underset{\parallel}{\theta_0}} + \underset{1}{\underset{\parallel}{\theta_1}} x_1 + \underset{1}{\underset{\parallel}{\theta_2}} x_2)$$

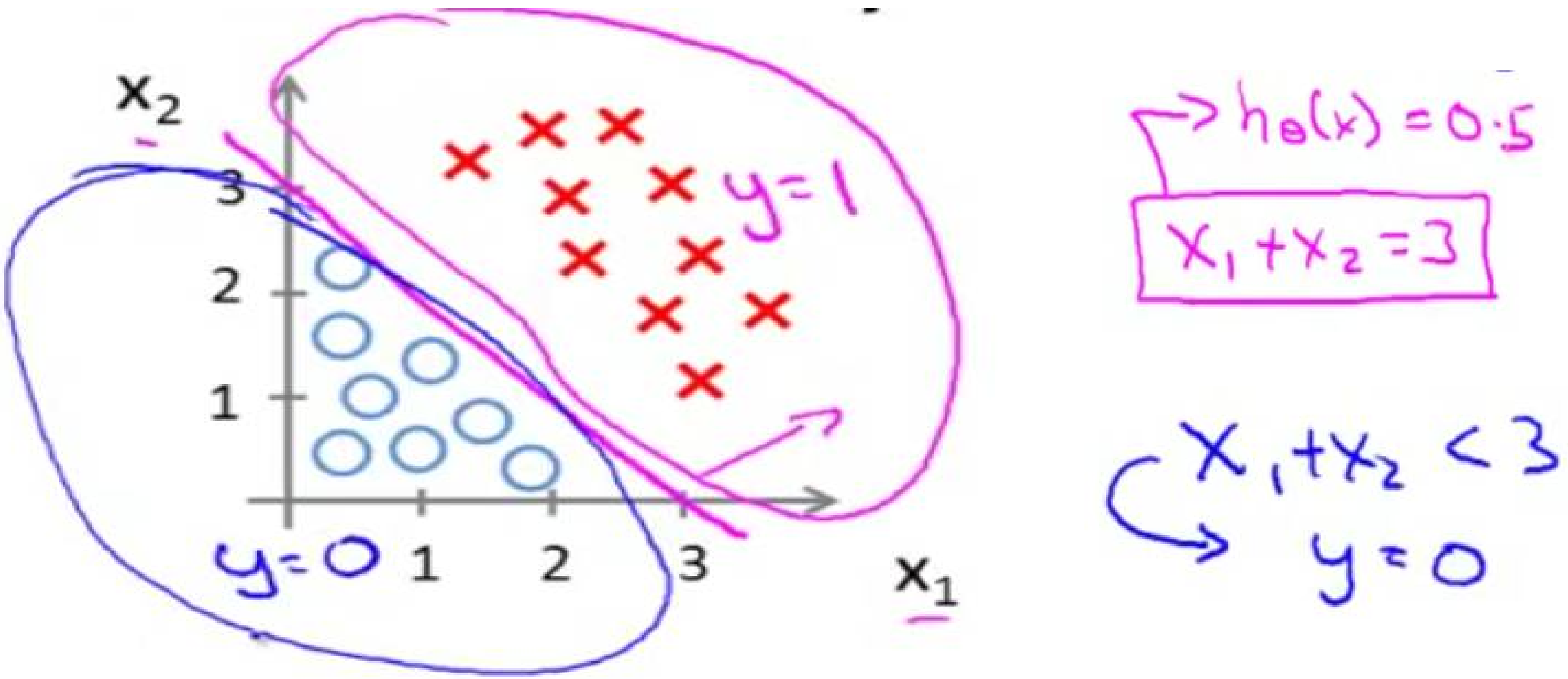
Predict "$y = 1$" if $\underbrace{-3 + x_1 + x_2}_{\theta^T x} \geq 0$

$x_1 + x_2 \geq 3$

Decision Boundary

$x_1 + x_2 = 3$

Decision Boundary: +ve and -ve regions

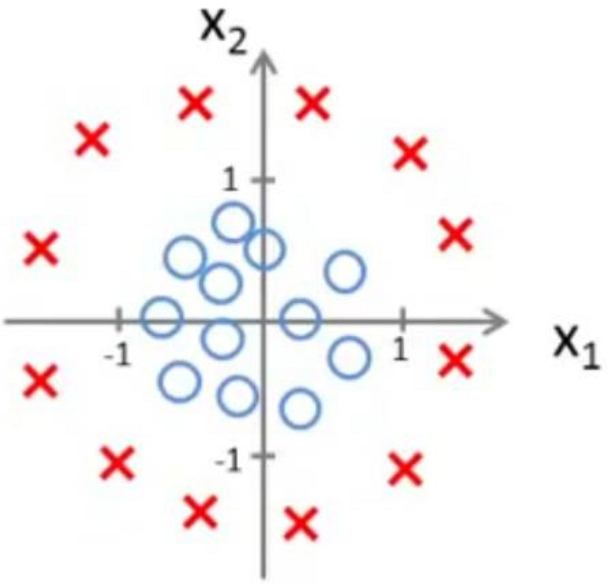


Decision Boundary

- The decision boundary is the property of the hypothesis ($h_{\Theta}(x)$) and the parameters (Θ), and not a property of the dataset.
- It helps to classify the new unseen examples.
- Therefore, we do not need to plot the training set, in order to plot the decision boundary.

Non-linear Decision Boundaries

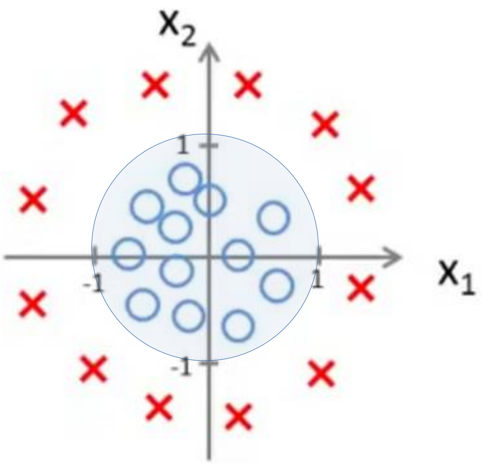
- Suppose the following dataset is given, where crosses are +ve samples and circles are the -ve samples. How can we get logistic regression to fit this data.



A good hypothesis function to model this data can be by adding higher order polynomial terms: say,

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

Non-linear Decision Boundaries



Suppose, parameters after optimization come out to be:

$$h_\theta(x) = g(\overset{=-1}{\theta_0} + \overset{=0}{\theta_1} x_1 + \overset{=0}{\theta_2} x_2 + \underset{=1}{\theta_3} x_1^2 + \underset{=1}{\theta_4} x_2^2)$$

$$\Theta = \begin{bmatrix} -1 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}$$

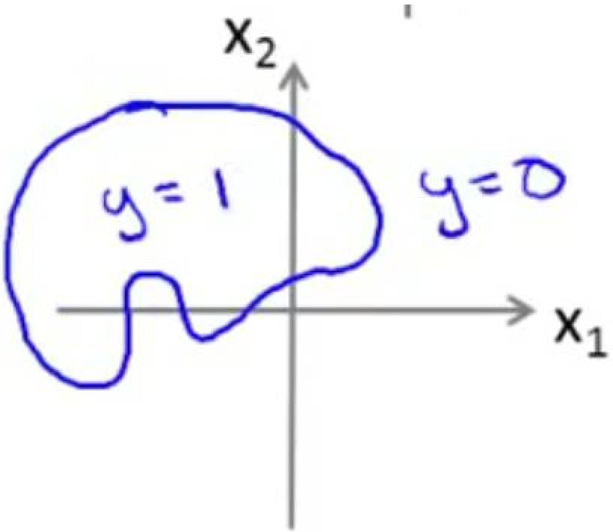
Therefore, the hypothesis will predict:

$$\text{Predict } “y = 1“ \text{ if } \underbrace{-1 + x_1^2 + x_2^2 \geq 0}_{x_1^2 + x_2^2 \geq 1}$$

Decision boundary “$x_1^2+x_2^2 = 1$” is a circle, with radius 1 around origin. Predict 0 inside, and 1 outside the circle.

Non-linear Decision Boundaries

- We can come up with more complex decision boundaries, using more higher order polynomial terms. For example, something as follows:



$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 \underline{x_1^2} + \theta_4 \underline{x_1^2 x_2} + \theta_5 \underline{x_1^2 x_2^2} + \theta_6 \underline{x_1^3 x_2} + \ldots)$$

Cost Function used to fit the parameters

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

m examples $\quad x \in \begin{bmatrix} x_0 \\ x_1 \\ \cdots \\ x_n \end{bmatrix} \quad x_0 = 1, y \in \{0, 1\}$

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

How to choose parameters θ ?

MSE Cost Function of Linear Regression: Problem using it for logistic regression

- Linear Regression Cost Function (MSE):

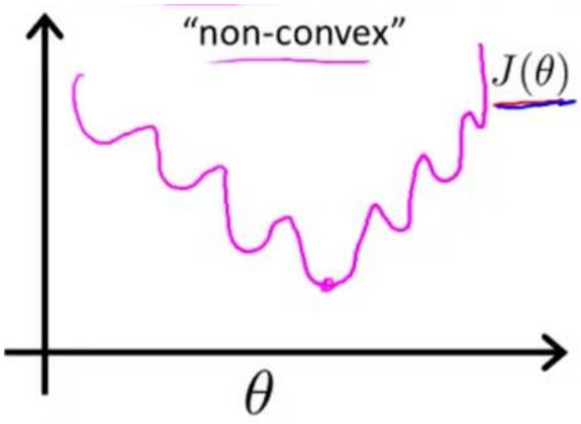
$$J_{\Theta}(x) = \frac{1}{m} \sum_{i=1}^{m} \frac{1}{2} \left(h_{\Theta}\left(x^{(i)}\right) - y^{(i)} \right)^2$$

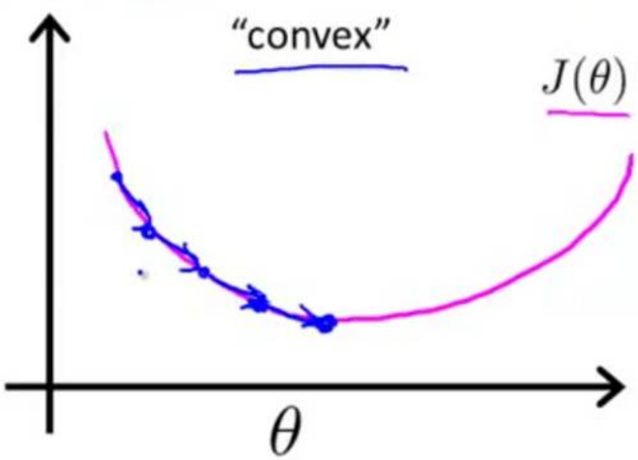
- For Linear Regression (Linear): $h_{\Theta}(x) = \Theta^T x$
- For Logistic Regression (Non-linear): $h_{\Theta}(x) = \frac{1}{1 + e^{-\Theta^T x}}$

- Let $Cost\left(h_{\Theta}\left(x^{(i)}\right), y^{(i)}\right) = \frac{1}{2} \left(h_{\Theta}\left(x^{(i)}\right) - y^{(i)} \right)^2$

Cost Function of Linear Regression: Problem using it for logistic regression

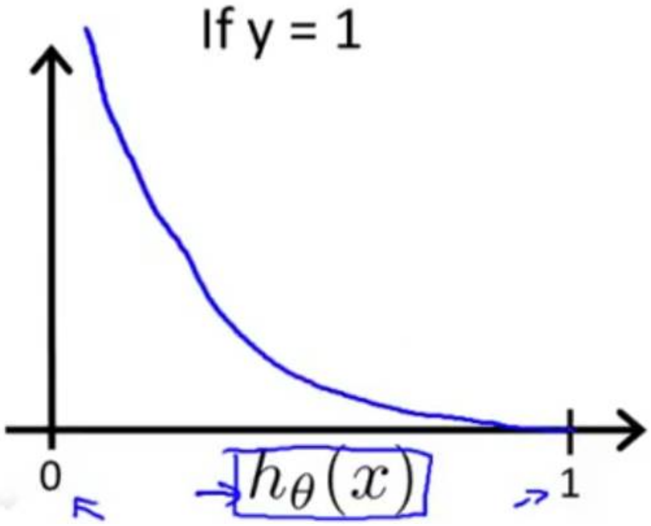
- Therefore, $J_\Theta(x)$ is non convex for logistic regression, if the same squared error cost function is used by replacing $h_\Theta(x)$ in Cost($h_\Theta(x)$, y).
- We may get a objective plot, say something like below(left) - non-convex
- And we want our cost function to be as below(right) - convex

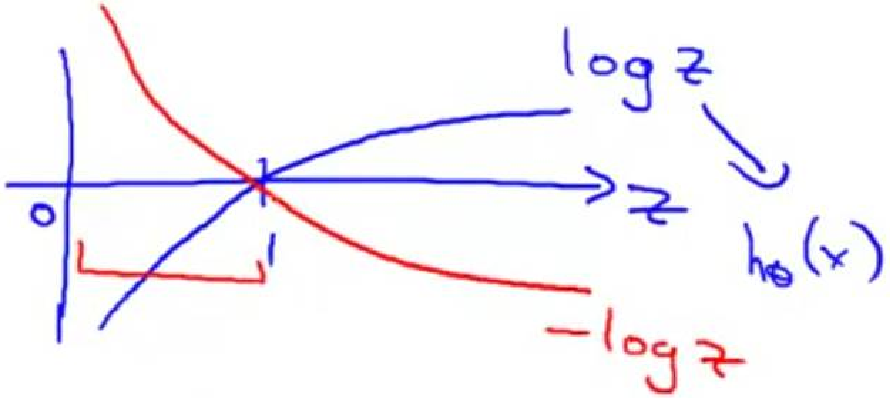




Logistic Regression Cost Function

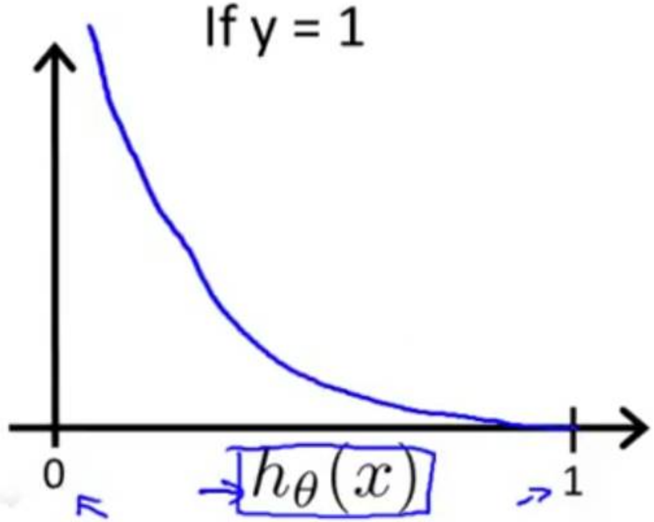
$$\text{Cost}(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & \text{if } y = 1 \\ -\log(1 - h_\theta(x)) & \text{if } y = 0 \end{cases}$$





Logistic Regression Cost Function

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



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

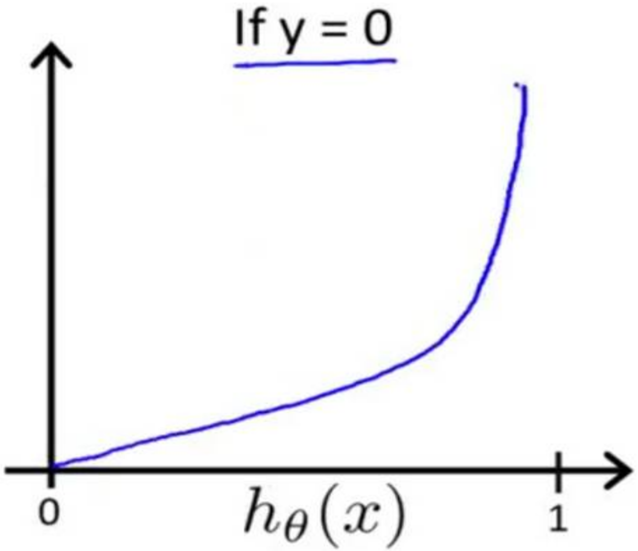
But as $h_\theta(x) \to 0$

$Cost \to \infty$

Captures intuition that if $h_\theta(x) = 0$, (predict $P(y = 1|x;\theta) = 0$), but $y = 1$, we'll penalize learning algorithm by a very large cost.

Logistic Regression Cost Function

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



Simplified Cost Function and Applying Gradient Descent

Logistic regression cost function

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

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

Note: $y = 0$ or 1 always

Simplified Cost Function and Applying Gradient Descent

- We can rewrite the Cost Function as follows:

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

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

$$\text{If } y=0: \quad \mathrm{Cost}(h_\theta(x), y) = -\log(1-h_\theta(x))$$

Simplified Cost Function and Applying Gradient Descent

Logistic regression cost function

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

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

To fit parameters θ :

$$\underbrace{\min_\theta J(\theta)}$$ Get $\underline{\theta}$

To make a prediction given new $\underline{x}$:

Output $\underline{h_\theta(x)} = \frac{1}{1+e^{-\theta^T x}}$ $\underline{p(y=1 \mid x;\theta)}$

Gradient Descent

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

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

Repeat $\{$

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

$\}$ (simultaneously update all θ_j)

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

Gradient Descent

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

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)

$\}$

$h_\theta(x) = \theta^T x$

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

Algorithm looks identical to linear regression, but the definition of $h_\Theta(x)$ is different for the two.

Multiclass Classification: One – vs – all

Email foldering/tagging: Work, Friends, Family, Hobby

$y=1$ $y=2$ $y=3$ $y=4$

Medical diagrams: Not ill, Cold, Flu

$y=1$ 2 3

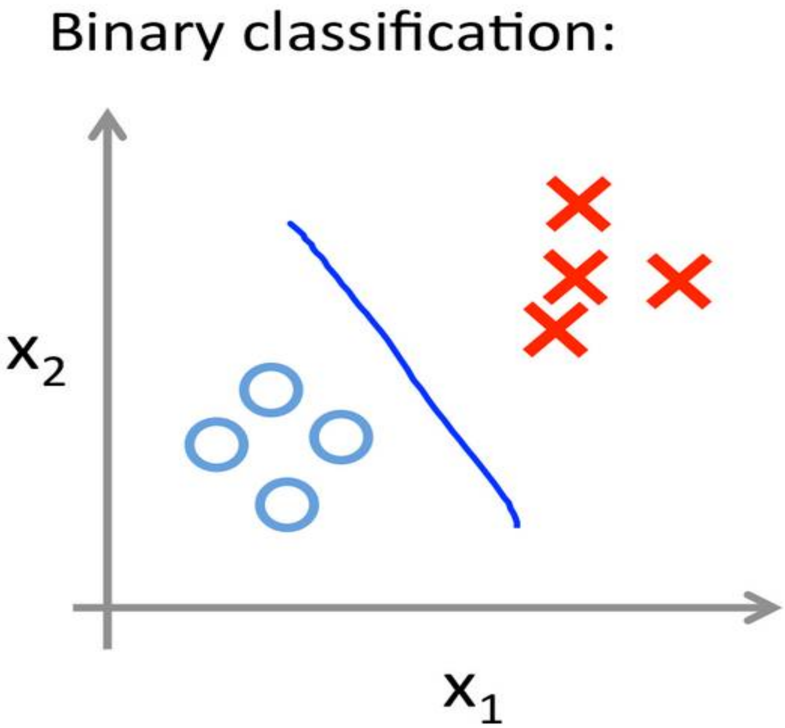
Weather: Sunny, Cloudy, Rain, Snow

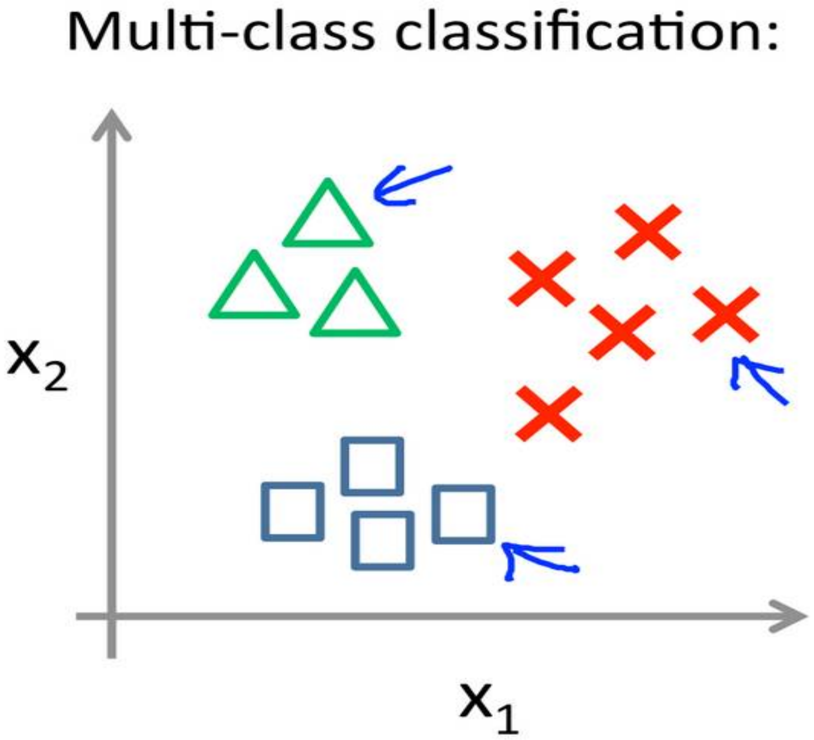
$y=1$ 2 3 4

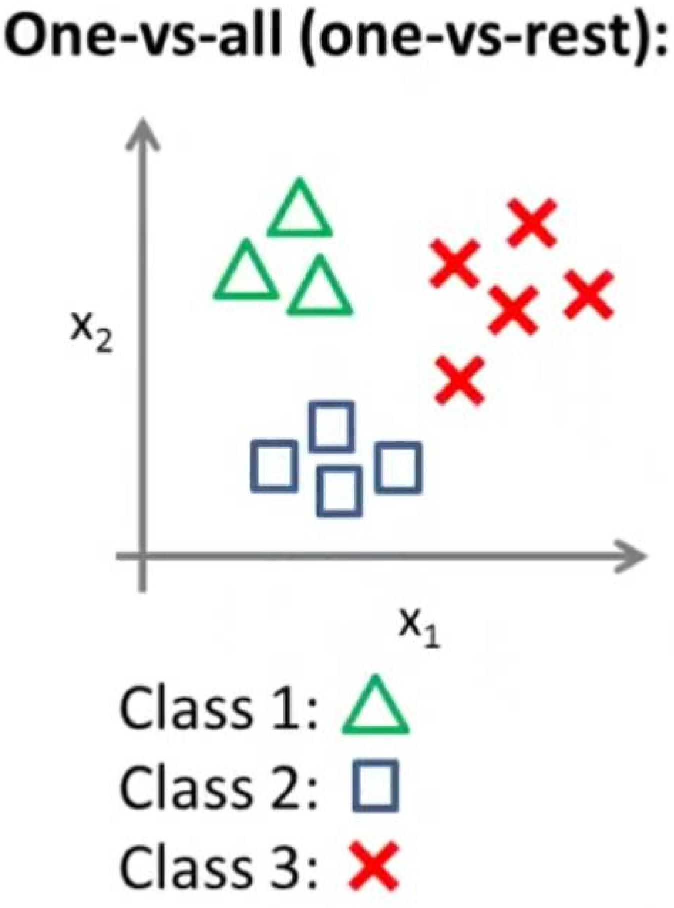
0 1 2 3

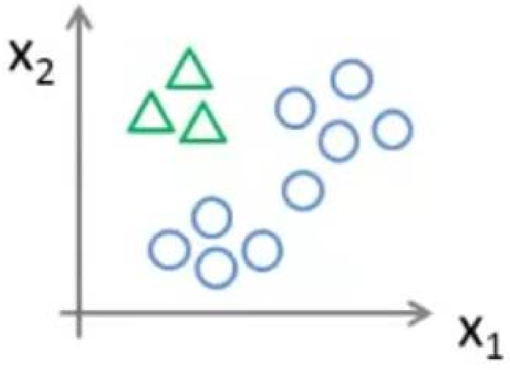
Starting labels from 0 or from 1. Both indexing schemes are fine.

Multiclass Classification: One – vs – all





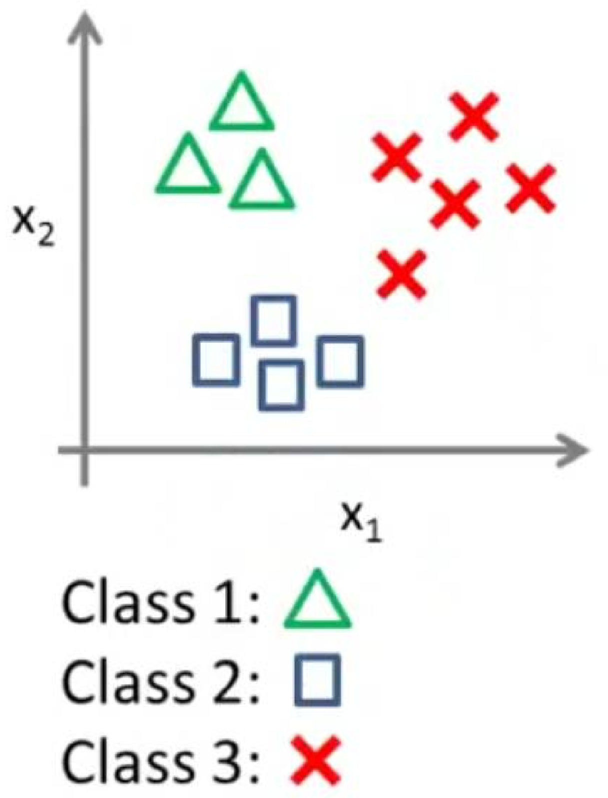


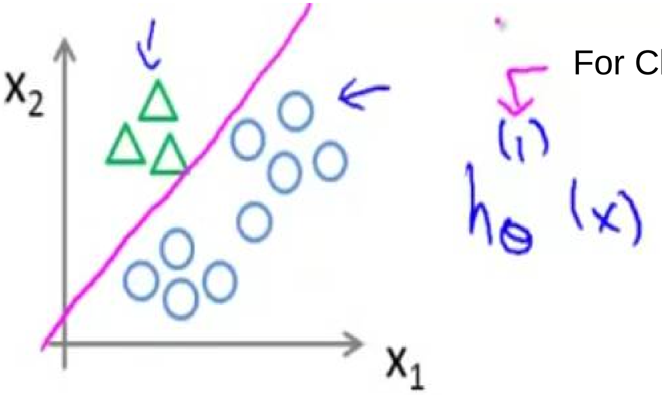


Steps:
- Create a new training set, where classes 2 and 3 are assigned to the -ve class. While triangles (or class 1) are assigned to the +ve class.

- Fit a classifier.

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





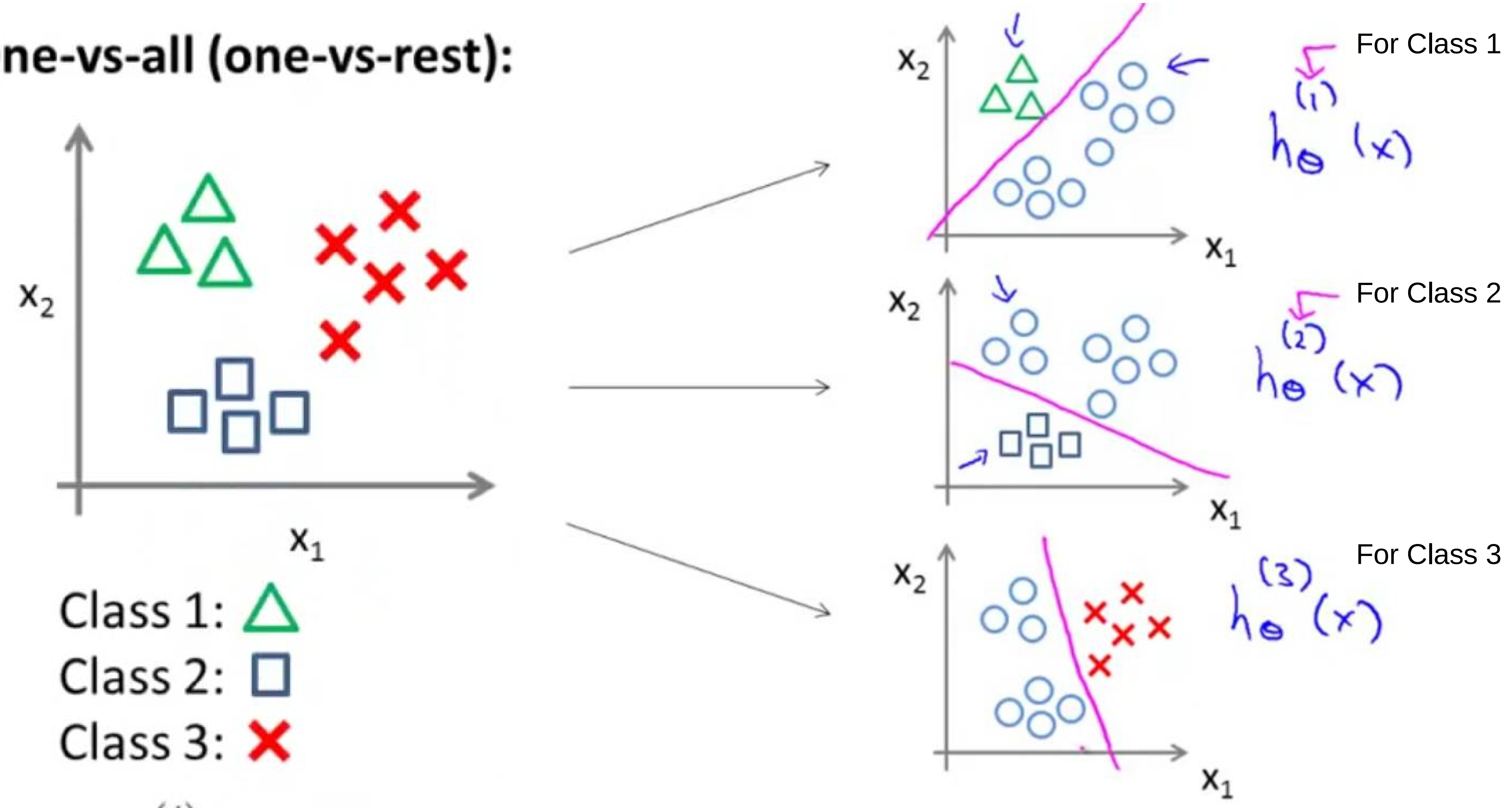
For Class 1

Steps:
- Create a new training set, where classes 2 and 3 are assigned to the -ve class. While triangles (or class 1) are assigned to the +ve class.

- Fit a classifier. $h_\Theta^{(1)}(x)$

- Similarly, do for rest of the classes.

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



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

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

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

Points to note

- There is no closed-form solution for logistic regression (similar to Normal Equation in linear regression).
- The logistic cost function is convex. It has a global optimum and has no local optima.
- The logistic regression (also linear regression) is an example of a broader class of models known as **GLM (Generalized Linear Models)**.

Points to note (contd.)

- Some texts show gradient ascent instead of gradient descent for optimizing logistic loss function. This is for maximization problem when the log likelihood is considered instead of negative log likelihood.
- There are other optimization methods for finding the best values of Θ. One such method is the Newton’s method.
- Feature scaling can help gradient descent run faster for logistic regression as well.

Homework

- Derive the gradient descent (or ascent) update rule for logistic regression, using MLE (maximum likelihood estimation that minimizes/maximizes the log likelihood of the parameters). Show that the parameter update equations are similar to that of linear regression.
  - refer CS229 notes

End of Lecture