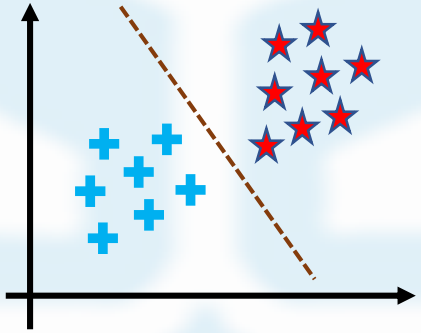
به نام او، به یاد او، برای او

Machine Learning

Classification



Dr. *Ali Valinejad*

valinejad.ir
valinejad@umz.ac.ir

Outlines:

- Classification
- Binary classification
- Logistic Regression Model
  - Sigmoid function
  - Hypothesis Representation
  - Cost Function
  - Gradient Descent to minimize $J(\theta)$
  - Gradient Ascent to minimize $J(\theta)$
- Multi-Class classification
  - Using *Binary Classification*
  - Using Softmax Regression Classifier

Classification

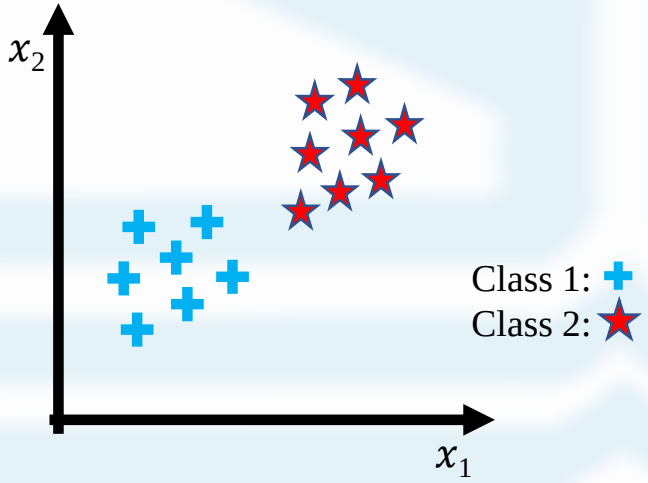
$(x^{(i)})^T$	$y^{(i)}$
$(x^{(1)})^T$	$y^{(1)}$
$(x^{(2)})^T$	$y^{(2)}$
$\vdots$	$\vdots$
$(x^{(m)})^T$	$y^{(m)}$

$$x^{(i)} = \left[x_1^{(i)}, x_2^{(i)}, \dots, x_n^{(i)}\right]^T$$

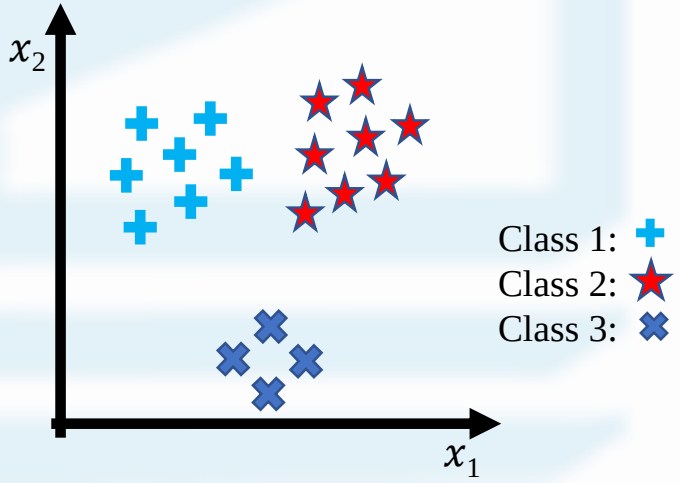
$$y^{(i)} \in \{c_1, c_2, \dots, c_k\}$$

Goal:
To categorise data points into k buckets

Binary classification, (k=2)



Multi-class classification, (k=3)



Binary classification

Binary classification

	Problem	Input Data	Output Label
1	Email Spam Detection	Email Content	Spam / Not Spam
2	Fraud detection	Online Transactions Data	Yes / No
3	Tumor Detection	Tumor Size	Malignant / Benign

Goal: To categorise data points into one of two buckets: 0 or 1, true or false, negative or positive, yes or no, etc.

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

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

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

Binary classification

$$\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \dots, (x^{(m)}, y^{(m)})\}$$

$$y^{(i)} \in \{0,1\} \qquad i = 1,2,\dots,m$$

Training set

$$X = \begin{bmatrix} (x^{(1)})^T \\ (x^{(2)})^T \\ \vdots \\ (x^{(m)})^T \end{bmatrix} = \begin{bmatrix} x_0^{(1)} & x_1^{(1)} & x_2^{(1)} & \cdots & x_n^{(1)} \\ x_0^{(2)} & x_1^{(2)} & x_2^{(2)} & \cdots & x_n^{(2)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_0^{(m)} & x_1^{(m)} & x_2^{(m)} & \cdots & x_n^{(m)} \end{bmatrix}$$

$$X_0^{(i)} := 1$$

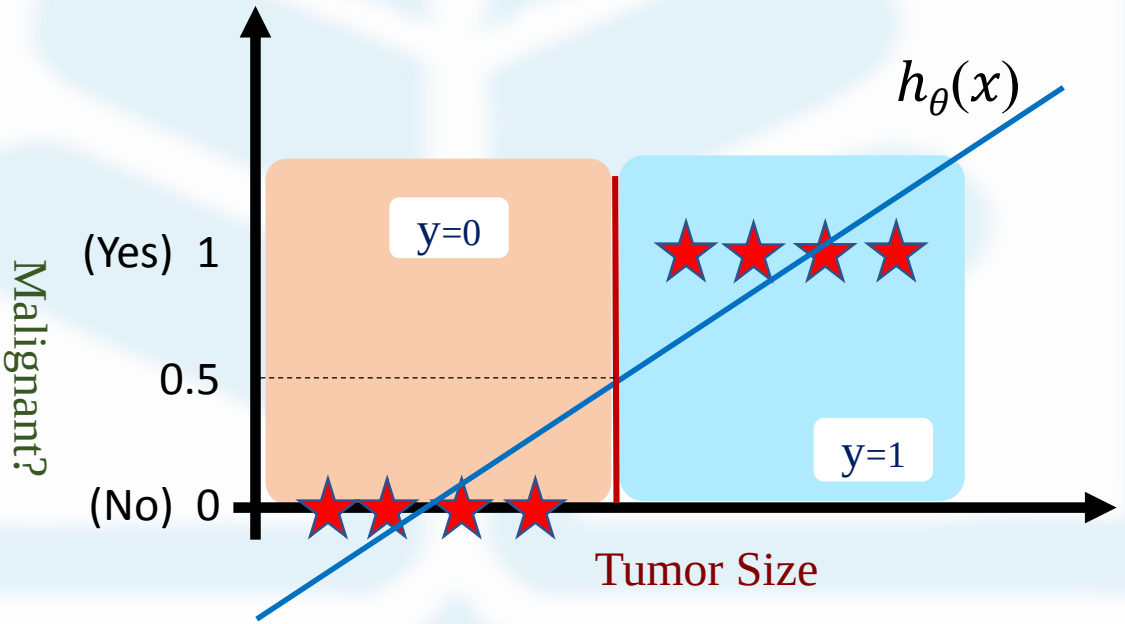
$$i = 1,2,\dots,m$$

$$y = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(m)} \end{bmatrix}$$

Binary classification

Tumor Size $x^{(i)}$	Class $y^{(i)}$
$x^{(1)}$	$y^{(1)}$
$x^{(2)}$	$y^{(2)}$
:	:
$x^{(m)}$	$y^{(m)}$

$$y^{(i)} = \begin{cases} 1 & \text{Malignant Tumor} \\ 0 & \text{Benign Tumor} \end{cases}$$



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

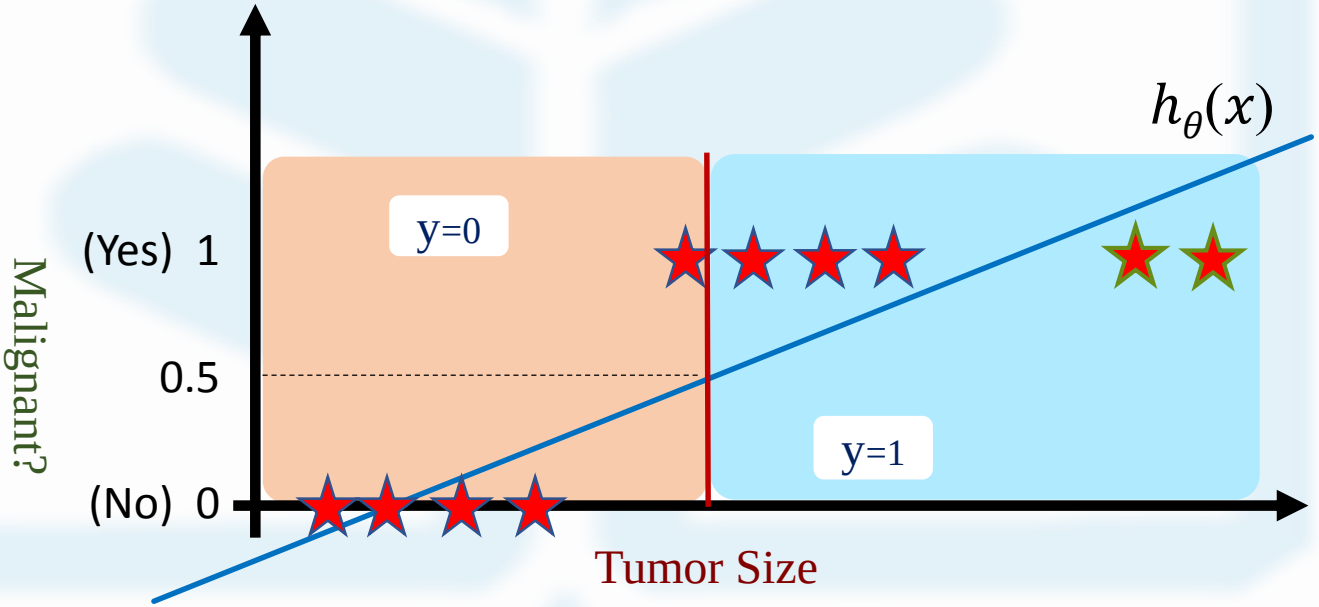
If $h_\theta(x) \geq 0.5$, predict y = 1

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

Binary classification

Tumor Size $x^{(i)}$	Class $y^{(i)}$
$x^{(1)}$	$y^{(1)}$
$x^{(2)}$	$y^{(2)}$
:	:
$x^{(m)}$	$y^{(m)}$

$$y^{(i)} = \begin{cases} 1 & \text{Malignant Tumor} \\ 0 & \text{Benign Tumor} \end{cases}$$



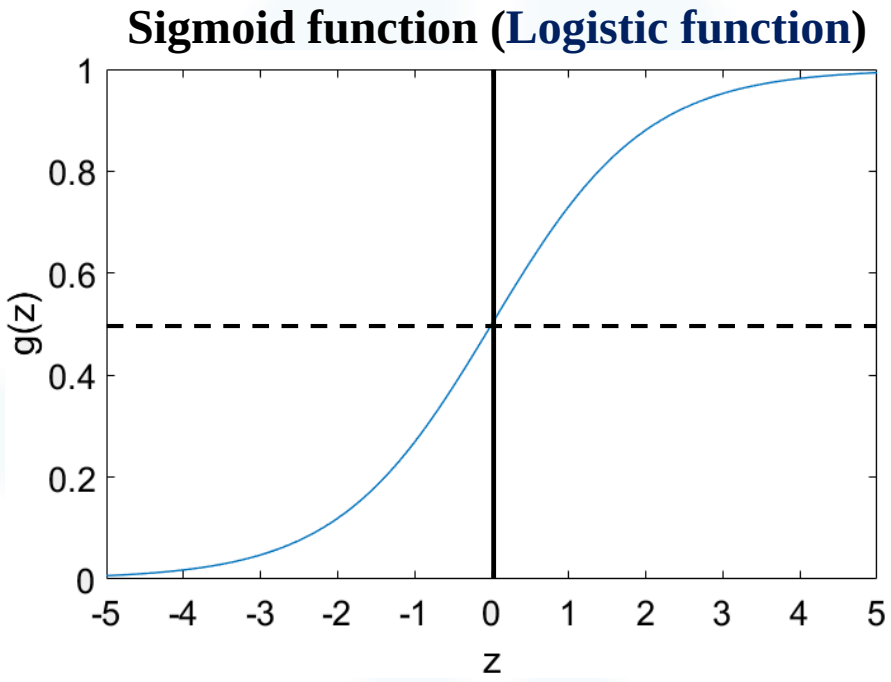
Threshold classifier output $h_\theta(x)$ at 0.5:

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

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

Logistic Regression Model (Classifier)

Logistic Regression Classifier



$$g(z) = \frac{1}{1 + e^{-z}} \qquad 0 \le g(z) \le 1$$

$$\lim_{z \to +\infty} g(z) = 1$$

$$\lim_{z \to -\infty} g(z) = 0$$

Logistic Regression Classifier

Hypothesis Representation:

Linear Regression

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

Logestic Regression

$$h_\theta(X) = g(\theta^T X)$$

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

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

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

How to choose parameters θ?

Logistic Regression Classifier

Interpretation of Hypothesis Output:

$h_\theta(x)$ = estimated probability that $y = 1$ on input x

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

$$p(y = 0|x; \theta) = 1 - h_\theta(x)$$

$$p(y|x; \theta) = h_\theta(x)^y (1 - h_\theta(x))^{1-y}$$

How to choose parameters θ?

Logistic Regression Classifier

Likelihood Function:

$$L(\theta) = p(\boldsymbol{y}|X;\theta) = \prod_{i=1}^{m} p\left(y^{(i)}\middle|x^{(i)};\theta\right)$$

$$= \prod_{i=1}^{m} h_\theta\left(x^{(i)}\right)^{y^{(i)}} \left(1 - h_\theta\left(x^{(i)}\right)\right)^{1-y^{(i)}}$$

$$l(\theta) = \log L(\theta) = \log \prod_{i=1}^{m} h_\theta\left(x^{(i)}\right)^{y^{(i)}} \left(1 - h_\theta\left(x^{(i)}\right)\right)^{1-y^{(i)}}$$

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

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

Logistic Regression Classifier

Cost Function:

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

$$Cost\left(h_\theta\left(x^{(i)}\right), y^{(i)}\right) := \begin{cases} -\log h_\theta\left(x^{(i)}\right) & \textbf{if } y = 1 \\ -\log\left(1 - h_\theta\left(x^{(i)}\right)\right) & \textbf{if } y = 0 \end{cases}$$
$$i = 1,2,\dots,m$$

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

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

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

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

Logistic Regression Classifier

Cost Function:

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

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

To fit parameters θ :

$$\theta^* = \underset{\theta}{\text{margmin}}\ J(\theta)$$

To make a prediction given x^{new} :

$$p(y = 1|x; \theta) = h_\theta(x^{new}) = \frac{1}{1 + e^{-(\theta^*)^T x^{new}}}$$

Logistic Regression Classifier

Gradient Descent to minimize $J(\theta)$

Gradient Descent to minimize $J(\theta)$

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

$$\min_{\theta} J(\theta)$$

Repeat {

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

(simultaneously update θ_j for every $j = 0,1, \dots, n$)

}

α: Learning rate

Gradient Descent to minimize $J(\theta)$

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

$$\min_\theta J(\theta)$$

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

Repeat {

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

(simultaneously update θ_j for every $j = 0,1,\dots,n$)

}

α: Learning rate

Logistic Regression Classifier

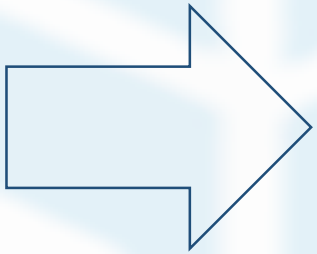
Gradient Ascent to minimize $J(\theta)$

Gradient Ascent to minimize $J(\theta)$

$$\boldsymbol{\theta} = \boldsymbol{\theta} + \alpha \nabla_{\theta} l(\boldsymbol{\theta})$$

$$\boldsymbol{\theta} = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_n \end{bmatrix}$$

$$\nabla_{\theta} l(\theta) = \begin{bmatrix} \frac{\partial l(\theta)}{\partial \theta_0} \\ \frac{\partial l(\theta)}{\partial \theta_1} \\ \vdots \\ \frac{\partial l(\theta)}{\partial \theta_n} \end{bmatrix}$$



$$\theta_j = \theta_j + \alpha \frac{\partial l(\theta)}{\partial \theta_j} \quad j = 0,1,\dots,n$$

$$\frac{\partial l(\boldsymbol{\theta})}{\partial \theta_j} = ?$$

Gradient Ascent to minimize $J(\theta)$

$$l(\boldsymbol{\theta}) = \sum_{i=1}^{m} y^{(i)} \log h_\theta\left(x^{(i)}\right) + \left(1 - y^{(i)}\right) \log\left(1 - h_\theta\left(x^{(i)}\right)\right)$$

$$\frac{\partial}{\partial \theta_j} l(\boldsymbol{\theta}) = \sum_{i=1}^{m} \left\{ y^{(i)} \frac{h'_\theta\left(x^{(i)}\right)}{h_\theta(x^{(i)})} + (1 - y^{(i)}) \frac{-h'_\theta\left(x^{(i)}\right)}{1 - h_\theta(x^{(i)})} \right\}$$

$$h'_\theta\left(x^{(i)}\right) = ?$$

Gradient Ascent to minimize $J(\theta)$

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

$$\begin{aligned} h'_\theta\left(x^{(i)}\right) &= \frac{\partial}{\partial \theta_j} h_\theta\left(x^{(i)}\right) \\ &= \frac{x_j^{(i)} e^{-\theta^T x}}{\left(1+e^{-\theta^T x}\right)^2} \\ &= x_j^{(i)} h_\theta\left(x^{(i)}\right)\left(1- h_\theta\left(x^{(i)}\right)\right) \end{aligned}$$

$$\frac{\partial}{\partial \theta_j} l(\boldsymbol{\theta}) = \sum_{i=1}^{m} \left\{ y^{(i)} \frac{h'_\theta\left(x^{(i)}\right)}{h_\theta(x^{(i)})} + (1 - y^{(i)}) \frac{-h'_\theta\left(x^{(i)}\right)}{1 - h_\theta(x^{(i)})} \right\}$$

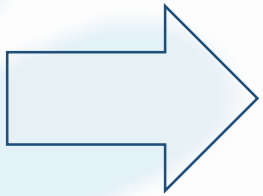
$$= \sum_{i=1}^{m} \left\{ y^{(i)} - h_\theta\left(x^{(i)}\right) \right\} x_j^{(i)}$$

Gradient Ascent to minimize $J(\theta)$

$$\boldsymbol{\theta} = \boldsymbol{\theta} + \alpha \nabla_{\theta} l(\boldsymbol{\theta})$$

$$\theta_j = \theta_j + \alpha \frac{\partial l(\theta)}{\partial \theta_j} \qquad j = 0,1, \dots, n$$

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



$$\theta_j = \theta_j + \alpha \sum_{i=1}^{m} \{y^{(i)} - h_{\theta}(X^{(i)})\} x_j^{(i)} \qquad j = 0,1, \dots, n$$

Gradient Ascent to minimize $J(\theta)$

$$\boldsymbol{\theta} = \boldsymbol{\theta} + \alpha \nabla_{\theta} l(\boldsymbol{\theta})$$

$$l(\boldsymbol{\theta}) = \sum_{i=1}^{m} y^{(i)} \log h_{\theta}\left(x^{(i)}\right) + \left(1 - y^{(i)}\right) \log\left(1 - h_{\theta}\left(x^{(i)}\right)\right)$$

$$\frac{\partial}{\partial \theta_j} l(\boldsymbol{\theta}) = \sum_{i=1}^{m} \left\{ y^{(i)} \frac{h'_{\theta}\left(x^{(i)}\right)}{h_{\theta}(x^{(i)})} + (1 - y^{(i)}) \frac{-h'_{\theta}\left(x^{(i)}\right)}{1 - h_{\theta}(x^{(i)})} \right\}$$

$$h'_{\theta}\left(x^{(i)}\right) = \frac{\partial}{\partial \theta_j} h_{\theta}\left(x^{(i)}\right)$$

$$= X_j^{(i)} h_{\theta}\left(x^{(i)}\right)(1 - h_{\theta}\left(x^{(i)}\right))$$

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

$$\theta_j = \theta_j + \alpha \sum_{i=1}^{m} \left\{ y^{(i)} - h_{\theta}\left(x^{(i)}\right) \right\} x_j^{(i)} \qquad j = 0, 1, \ldots, n$$

Gradient Ascent to minimize $J(\theta)$

Gradient Ascent Algorithm

Repeat {

for *j=0:n*

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

} *until convergence*

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

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

Multi-class classification

Logistic Regression Classifier: *Review*

In **logistic regression** we need to decide whether one *sample* belongs to one class or not.

Main Steps of Logistic regression classifier:

1- *Map* raw data into a score *z* using a linear function:

$$z = \theta^T x$$

2- *Feed z* into logistic function for *normalization*:

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

3- *Interprete* the $h_\theta(x)$ as the *probability* of the correct class (y = 1):

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

What should we do for *multi-class classification*?

- Using ***Binary Classification***
- Using ***Multinomial logistic regression***

Multi-class classification

	Problem	Input Data	Output Label
1	Email Type Detection	Email Content	Work, Friends, Family, Hobby
2	handwritten digit recognition	image of a handwritten	0,1,…,9
3	weather prediction	weather data	Sunny, Cloudy, Rain, Snow

Goal: To categorise data points into more than two buckets

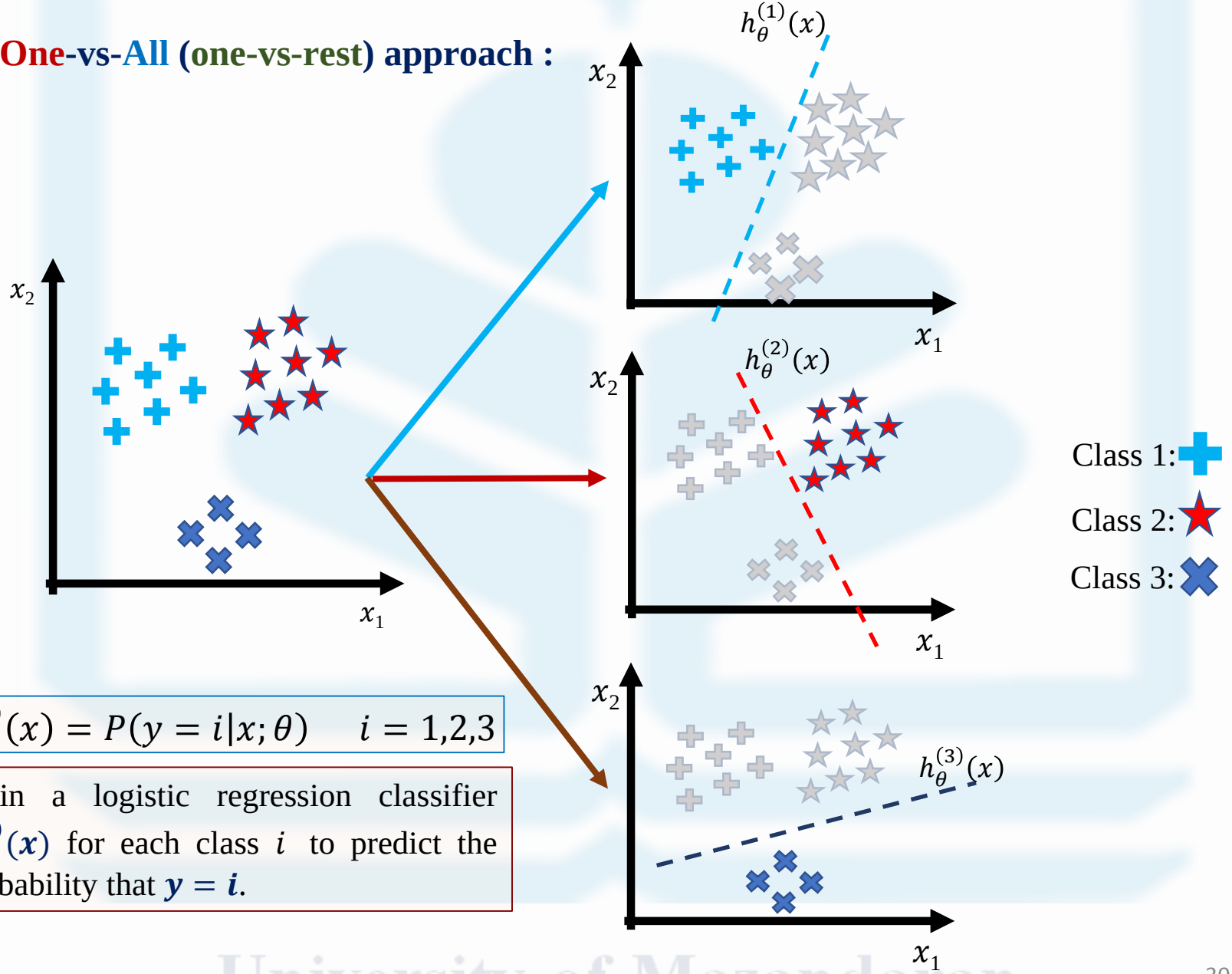
Multi-class classification

using ***Binary Classification***

One-vs-All (one-vs-rest) approach
One-vs-One (All-vs-All) approach

Multi-class classification (By *Binary Classification*)

One-vs-All (one-vs-rest) approach :



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

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

Multi-class classification (By *Binary Classification*)

One-vs-All (one-vs-rest) approach :

$$y \in \{1,2,\dots,K\}$$

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

Training Phase:

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

Testing Phase:

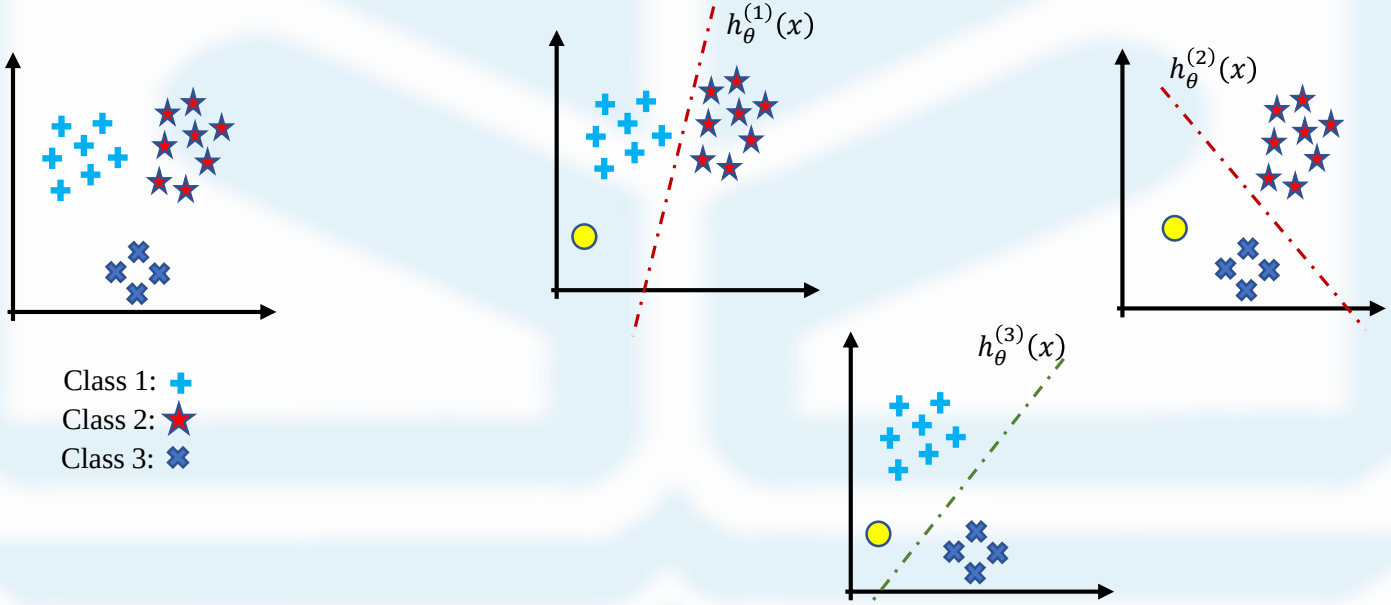
On a new input x^{new}, to make a prediction, pick the class $\boldsymbol{i}$ that maximizes $h_\theta^{(i)}(x)$

$$\boldsymbol{i}^* = \underset{i = 1,2,\dots,\mathrm{K}}{argmax}\ h_\theta^{(i)}(x)$$

Multi-class classification (By Logistic Regression Model)

One-vs-One (All-vs-All) approach :

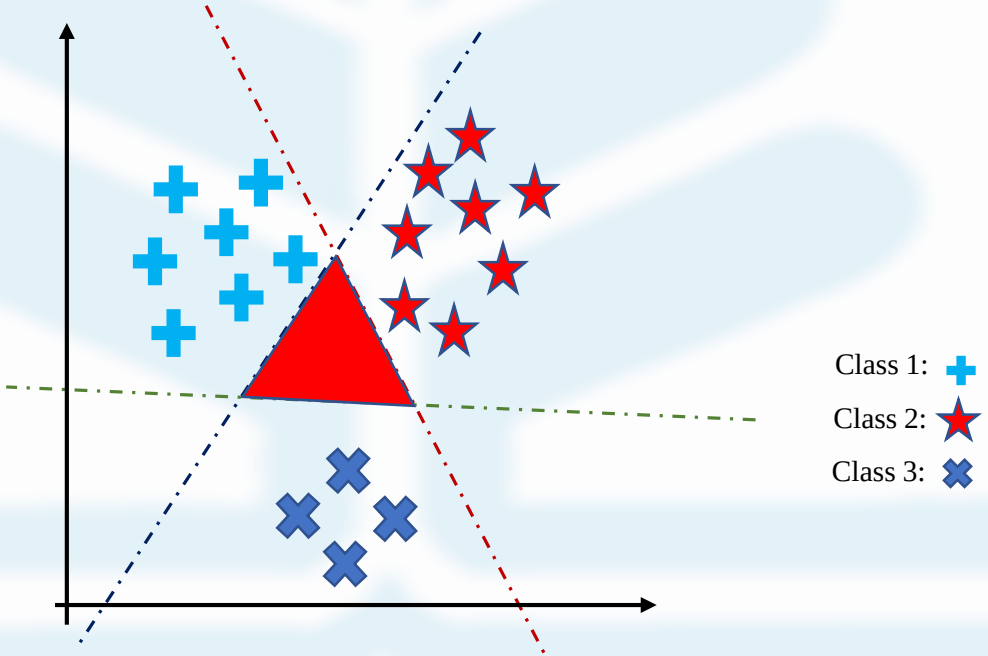
- In **one-vs-one**, we train $\binom{K}{2} = \frac{K(K-1)}{2}$ separate *binary classification* models, one for each possible pair of classes.
- To predict the class for a new example x, we run all $\binom{K}{2}$ classifiers on x and choose the class with the most "*votes*."



Multi-class classification (By *Binary Classification*)

One-vs-One (All-vs-All) approach :

- Computationally *expensive*: think of K=2000
- Problem of *ambiguous region*.
  - A major drawback is that there can exist fairly large regions in the decision space with ties for the class with the most number of votes.



Multi-class classification

By *Multinomial logistic regression*

Synonyms:

- *Softmax Regression Classifier,*
- *Maximum Entropy Classifier,*
- *Multi-class Logistic Regression,*
- *Maxent classifier.*

Multi-class classification (*Multinomial logistic regression*)

Multinomial logistic regression (*Softmax regression classifier*) is a method in machine learning which can perform multi-class classification of an input into any number of discrete classes.

- In softmax regression classifier the target y is a variable that ranges over more than two classes; we want to know the *probability* of y being in each potential class $c \in \{1,2,\dots,k\}$, $p(y = c|x)$.

The softmax regression classifier uses a generalization of the *sigmoid*, called the *softmax function* or *normalized exponential function*, to compute the probability $p(y = c|x)$.

- The *softmax function* takes a vector $z \in [z_1, z_2, \dots, z_k]$ of K arbitrary values and *maps* them to a probability distribution, with each value in the range (0,1), and all the values summing to 1.

$$softmax(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{k} e^{z_j}}$$

The Softmax Regression is preferred when we have features of different type (continuous, discrete, dummy variables etc), nevertheless given that it is a regression model, it is more vulnerable to multicollinearity problems and thus it should be avoided when our features are highly correlated.

Multi-class classification (*Multinomial logistic regression*)

Training set:

$$\mathcal{D} = \{(\boldsymbol{x}^{(1)}, y^{(1)}), (\boldsymbol{x}^{(2)}, y^{(2)}), \dots, (\boldsymbol{x}^{(m)}, y^{(m)})\}$$

$$\boldsymbol{x}^{(i)} = \left[x_1^{(i)}, x_2^{(i)}, \dots, x_n^{(i)}\right]^T$$

$$y^{(i)} \in \{1,2, \dots, K\}, \quad i=1,2,\dots, m$$

***Assumption*:**

$$p(y = j|x) = \phi_j \quad (j = 1,2, \dots, K)$$

$$\phi_1 + \phi_2 + \dots + \phi_K = 1$$

- *Parameters of the model:* $\phi_1, \phi_2, \dots, \phi_K$
- *Becouse of redundancy, we ignore* ϕ_K. $(\phi_K = 1 - \phi_1 + \phi_2 + \dots + \phi_{K-1})$

Multi-class classification (*Multinomial logistic regression*)

$$h_\theta(x) := \begin{bmatrix} p(y=1|x;\boldsymbol{\theta}) \\ p(y=2|x;\boldsymbol{\theta}) \\ \vdots \\ p(y=k-1|x;\boldsymbol{\theta}) \end{bmatrix} = \frac{1}{\sum_{j=1}^{k-1} e^{\theta_j^T x}} \begin{bmatrix} e^{\boldsymbol{\theta}^{(1)^T} x} \\ e^{\boldsymbol{\theta}^{(2)^T} x} \\ \vdots \\ e^{\boldsymbol{\theta}^{(k-1)^T} x} \end{bmatrix}$$

$$\boldsymbol{\theta}^{(j)} = \begin{bmatrix} \theta_0^{(j)} \\ \theta_1^{(j)} \\ \vdots \\ \theta_n^{(j)} \end{bmatrix}$$

$$\phi_s = \frac{e^{\boldsymbol{\theta}^{(s)^T} x}}{\sum_{j=1}^{k-1} e^{\theta_j^T x}}$$

$(s = 1,2,\dots,K-1)$

Likelihood Function:

$$L(\boldsymbol{\theta}|\mathcal{D}) = p(\boldsymbol{y}|X;\boldsymbol{\theta}) = \prod_{i=1}^{m} p\left(y^{(i)}\middle|x^{(i)};\boldsymbol{\theta}\right)$$

$$= \prod_{i=1}^{m}\prod_{j=1}^{K} p\left(y^{(i)}=j\middle|x^{(i)}\right)^{1(y^{(i)}=j)}$$

$$= \prod_{i=1}^{m} \phi_1^{1(y^{(i)}=1)}\ \phi_2^{1(y^{(i)}=1)}\ \ldots \phi_K^{1(y^{(i)}=K)}$$

$$1\left(y^{(i)}=1\right) = \begin{cases} 1 & if\ \ y^{(i)}=1 \\ 0 & o.w. \end{cases}$$

Log Likelihood Function:

$$l(\boldsymbol{\theta}|\mathcal{D}) = \log L(\boldsymbol{\theta}|\mathcal{D}) = p(\boldsymbol{y}|X;\boldsymbol{\theta})$$

$$= \sum_{i=1}^{m}\sum_{j=1}^{K} 1(y^{(i)} = j)\log\left(y^{(i)} = j\middle|x^{(i)}\right)$$

$$= \sum_{i=1}^{m}\sum_{j=1}^{K} 1(y^{(i)} = j)\left(\boldsymbol{\theta}^{(j)^T}x^{(i)} - \log\sum_{s=1}^{K} e^{\boldsymbol{\theta}^{(s)^T}x^{(i)}}\right)$$

To find optimal parameters $\boldsymbol{\theta}$ we can use gradient ascend method to maximize $l(\boldsymbol{\theta}|\mathcal{D})$

References

https://www.geeksforgeeks.org

Andrew Ng, https://www.coursera.org/learn/machine-learning

https://chrisyeh96.github.io/2018/06/11/logistic-regression.html

http://deeplearning.stanford.edu/tutorial/supervised/SoftmaxRegression/

http://fourier.eng.hmc.edu/e176/lectures/logisticRegression/node2.html