UNIVERSITÀ DEGLI STUDI DI NAPOLI
PARTHENOPE

Natural Language Processing

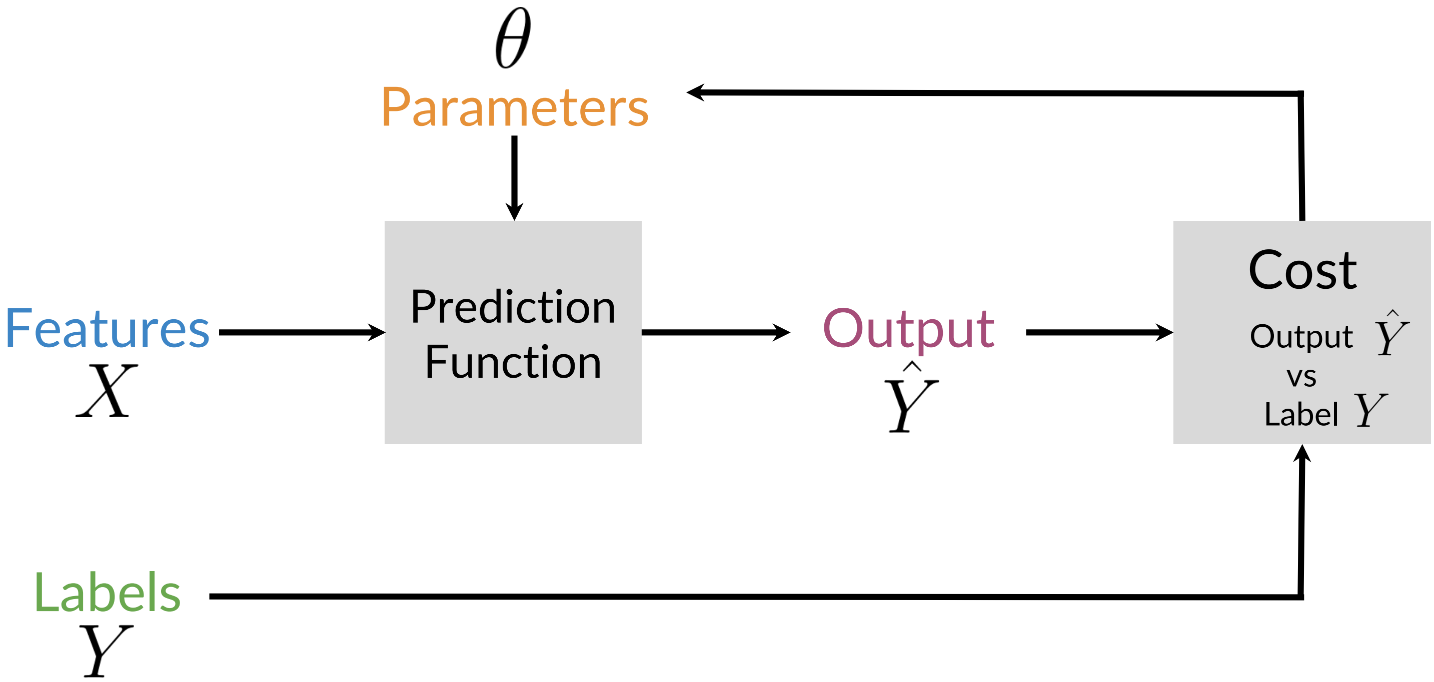
On Logistic regression

LESSON 8

prof. Antonino Staiano

M.Sc. In ''Machine Learning e Big Data'' - University Parthenope of Naples

Logistic regression

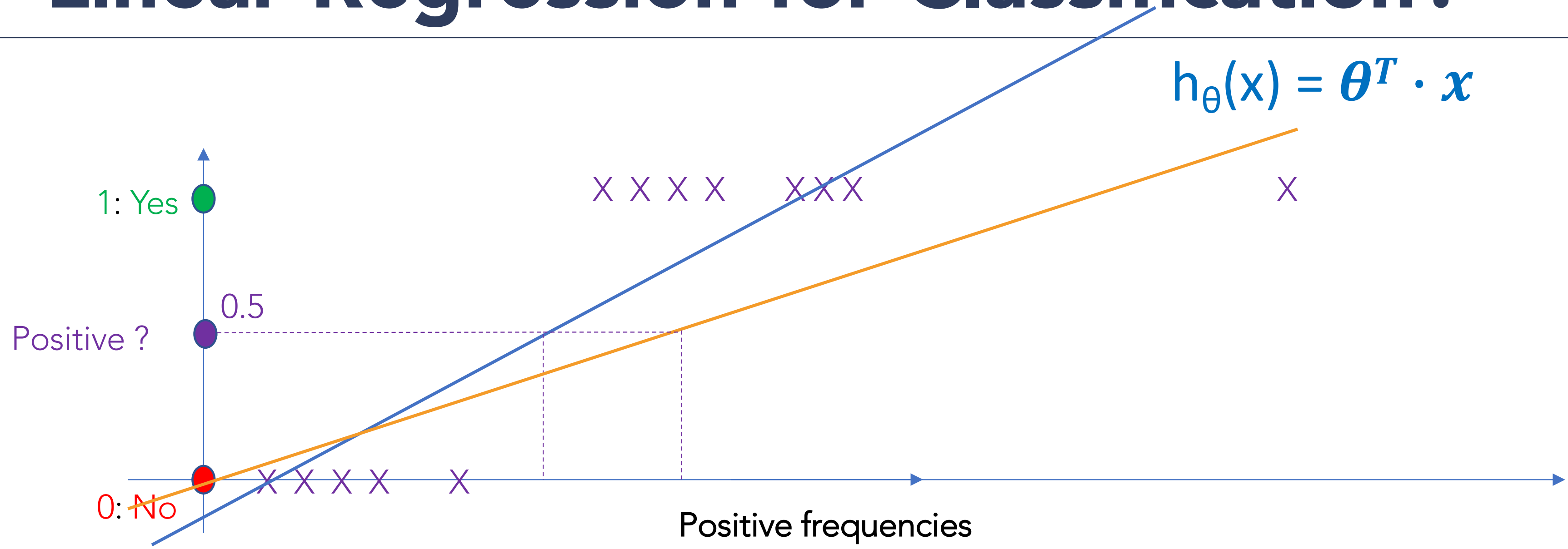


Logistic Regression

- When we work with classification problems, the output is
  - $y \in \{0,1\}$
  - For a multi-class, $y \in \{0,1,2, \dots, C\}$
- In theory, we could approach the problem as a regression problem, where the output is
  - $y \in R$
- Let's discuss how (hypothetically) to approach sentiment analysis as a linear regression task. Recall

  - $y \in \{0,1\}$ $\begin{cases} 0: \text{"Negative Class"} \\ 1: \text{"Positive Class"} \end{cases}$

Linear Regression for Classification?



- Threshold classifier output $h_{\boldsymbol{\theta}}(x)$ at 0.5
  - If $h_{\boldsymbol{\theta}}(x) >= 0.5$, predict $y = 1$
  - If $h_{\boldsymbol{\theta}}(x) < 0.5$, predict $y = 0$

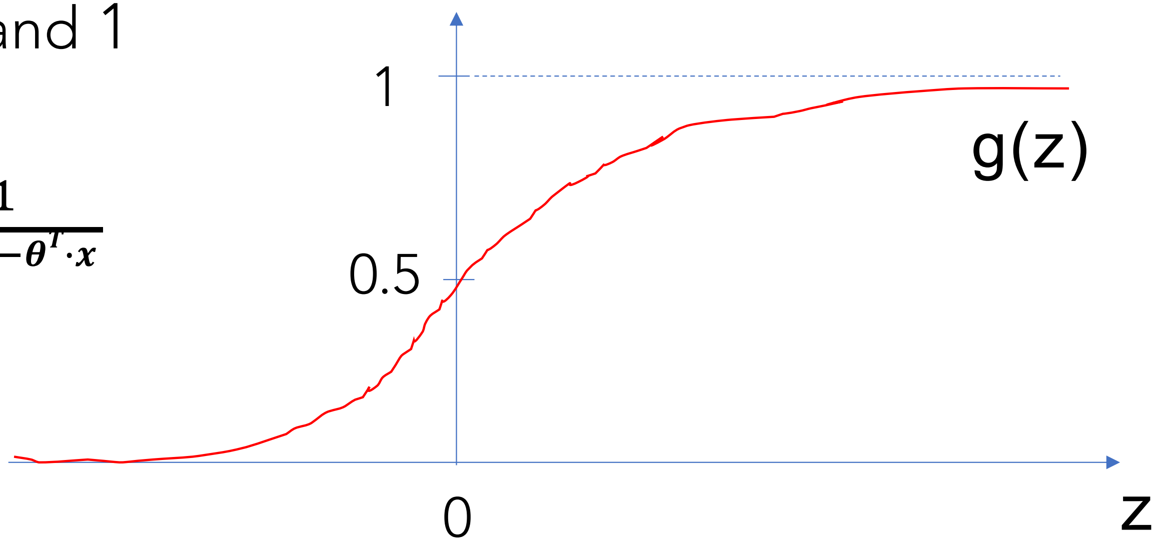
Towards Logistic Regression

- Classification y = 0 or 1
  - $h_\theta(x) = \boldsymbol{\theta}^T \cdot \boldsymbol{x}$ can be > 1 or < 0

- Logistic regression
  - $h_\theta(x) = g(\boldsymbol{\theta}^T \cdot \boldsymbol{x})$
  - $0 <= h_\theta(x) <= 1$

LR model representation

- Goal: $0 <= h_\theta(x) <= 1$
- Logistic regression makes use of the sigmoid function which outputs a probability between 0 and 1
- $h_\theta(x) = g(\boldsymbol{\theta}^T \cdot \boldsymbol{x}) = \frac{1}{1+e^{-\boldsymbol{\theta}^T \cdot \boldsymbol{x}}}$



Sigmoid or Logistic function

Interpretation of model output

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

  - $x = \begin{bmatrix} x_0 \\ x_1 \end{bmatrix} = \begin{bmatrix} 1 \\ PosFreq \end{bmatrix}$

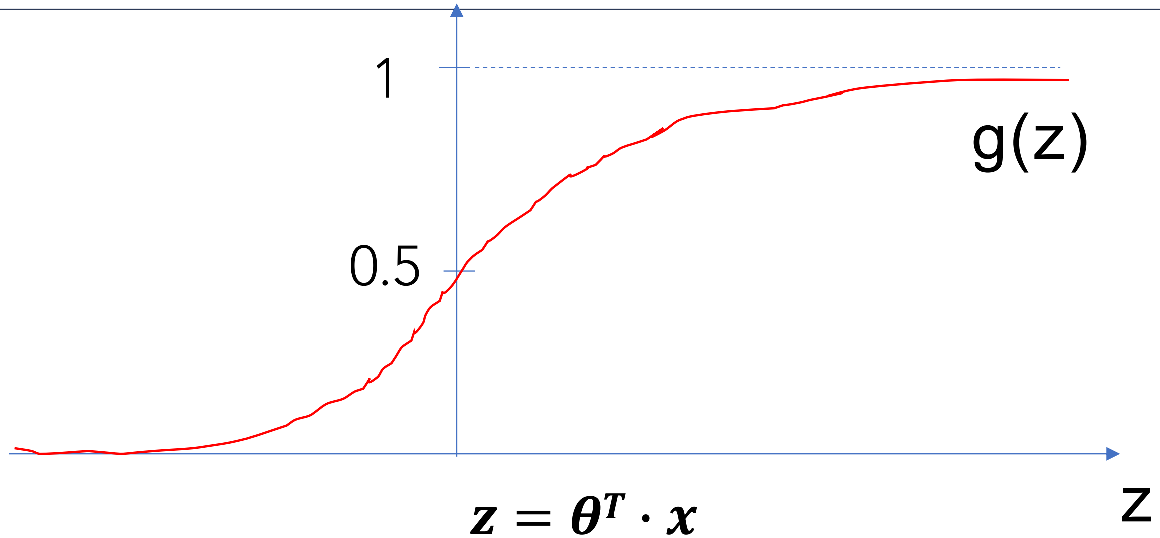
  - then if $h_{\theta}(x) = 0.7$ tell us that 70% is the chance the tweet is positive

- $h_{\theta}(x) = P(y = 1|x; \boldsymbol{\theta})$ -> probability that y = 1, given x, parametrized by $\boldsymbol{\theta}$

- $P(y = 0|x; \boldsymbol{\theta}) + P(y = 1|x; \boldsymbol{\theta}) = 1$, therefore $P(y = 0|x; \boldsymbol{\theta}) = 1 - P(y = 1|x; \boldsymbol{\theta})$

Decision boundary

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

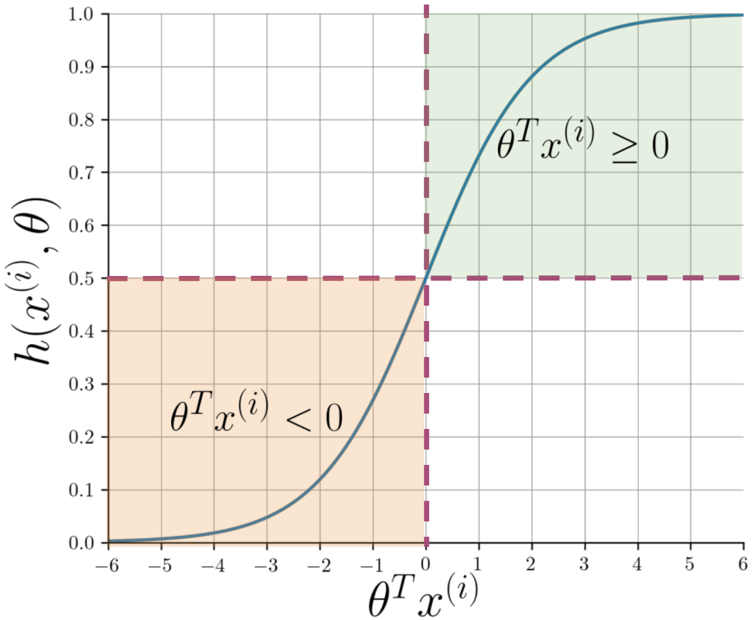


$$\boldsymbol{z = \theta^T \cdot x}$$
$$\boldsymbol{g(z) \geq 0.5 \; when \; z \geq 0}$$
$$\boldsymbol{g(z) < 0.5 \; when \; z < 0}$$

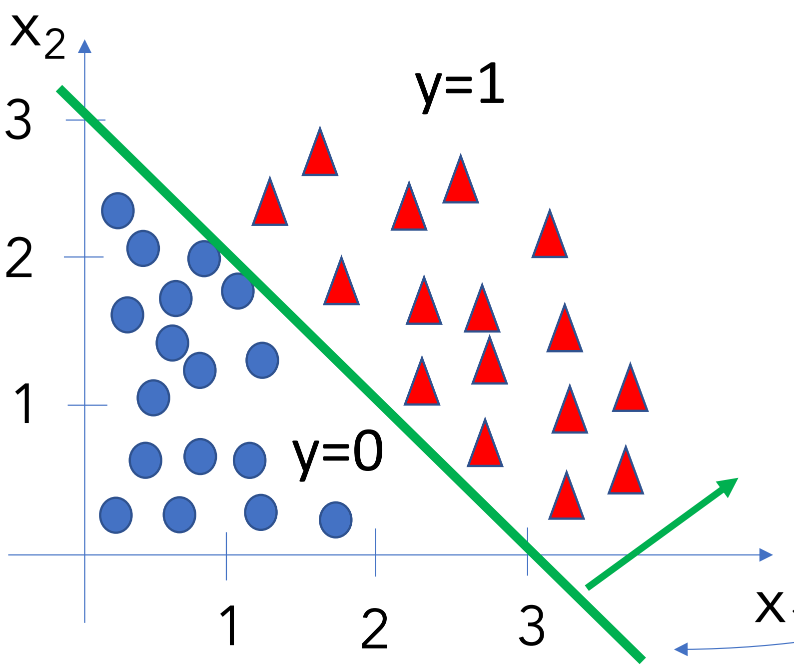
- predict y = 1 if $h_\theta(x) >= 0.5 \Longrightarrow \boldsymbol{\theta}^{\boldsymbol{T}} \cdot \boldsymbol{x} >= 0$
- Predict y = 0 if $h_\theta(x) < 0.5 \Longrightarrow \boldsymbol{\theta}^{\boldsymbol{T}} \cdot \boldsymbol{x} < 0$

Decision Boundary

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



Decision Boundary



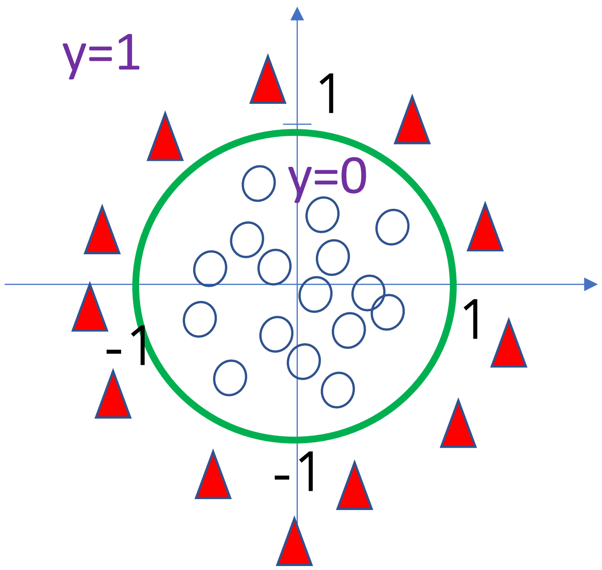
$$h_\theta(\boldsymbol{x}) = g(\boldsymbol{\theta_0} + \boldsymbol{\theta_1 x_1} + \boldsymbol{\theta_2 x_2})$$

$$\boldsymbol{\theta} = \begin{bmatrix} \boldsymbol{\theta_0} \\ \boldsymbol{\theta_1} \\ \boldsymbol{\theta_2} \end{bmatrix} = \begin{bmatrix} \mathbf{-3} \\ \mathbf{1} \\ \mathbf{1} \end{bmatrix}$$

$\boxed{x_1+x_2=3} \rightarrow \boxed{h_\theta(\boldsymbol{x}) = 0.5}$

- Predict y = 1 if $-3 + x_1 + x_2 >= 0 \Leftrightarrow x_1+x_2>=3$

Non-linear Decision Boundary



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

- $\boldsymbol{\theta} = \begin{bmatrix} \boldsymbol{\theta_0} \\ \boldsymbol{\theta_1} \\ \boldsymbol{\theta_2} \\ \boldsymbol{\theta_3} \\ \boldsymbol{\theta_4} \end{bmatrix} = \begin{bmatrix} \mathbf{-1} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{1} \\ \mathbf{1} \end{bmatrix}$

- Predict y =1 if $- \mathbf{1} + \boldsymbol{x_1^2} + \boldsymbol{x_2^2} \geq \mathbf{0}$

Cost Function

- Training set: $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})\}$
- m samples $\quad x \in \begin{bmatrix} \boldsymbol{x_0} \\ \boldsymbol{x_1} \\ \ldots \\ \ldots \\ \boldsymbol{x_n} \end{bmatrix} \in R^{n+1} \quad \boldsymbol{x_0} = 1,\ y \in \{0,1\}$
- $h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \frac{1}{1+e^{-\boldsymbol{\theta}^T \cdot \boldsymbol{x}}}$
- How to choose parameters $\boldsymbol{\theta}$?

Cost function

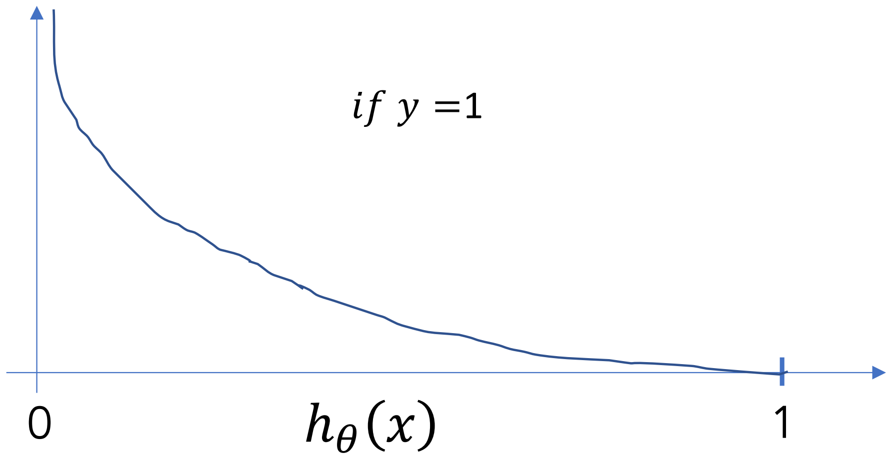
- Linear regression: $J(\boldsymbol{\theta}) = \frac{1}{m}\sum_{i=1}^{m} \underbrace{\frac{1}{2}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2}$

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

- $J(\boldsymbol{\theta})$ is non-convex with $h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \frac{1}{1+e^{-\boldsymbol{\theta}^T \cdot \boldsymbol{x}}}$

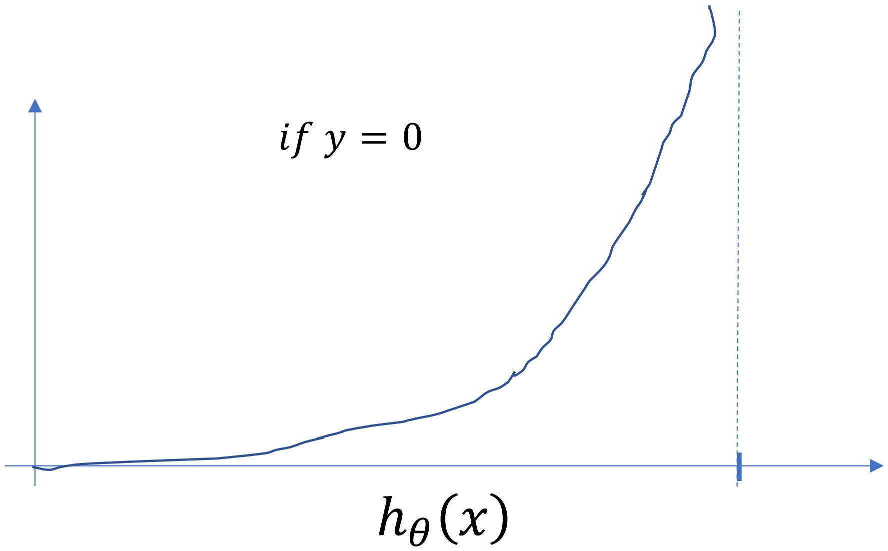
Logistic Regression cost function

- $\text{Cost}(h_\theta(x), y) = \begin{cases} \boxed{-log\left(h_\theta(x)\right)} & if\ y = 1 \\ -\log\left(1 - h_\theta(x)\right) & if\ y = 0 \end{cases}$
- $Cost = 0$ if $y = 1$ and $h_\theta(x) = 1$
  - but as $h_\theta(x) \rightarrow 0, Cost \rightarrow \infty$
  - captures the intuition that if $h_\theta(x) = 0$, it will penalize the learning algorithm by a very large cost
    - (predict $P(y = 1 \mid x; \theta) = 0$, but y =1)



Logistic regression cost function

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



Simplifying the cost function

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

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

- Since y = 0 or 1 (always):
- $\text{Cost}(h_\theta(x), y) = -y\ log(h_\theta(x)) - (1 - y) log(1 - h_\theta(x))$

Logistic regression cost function

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

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

- To fit parameters $\boldsymbol{\theta}$
  - $\min_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$
- To make a prediction given a new x
  - Output $h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \frac{1}{1+e^{-\boldsymbol{\theta}^T \cdot \boldsymbol{x}}}$

Gradient Descent

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

- $\min_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$:

  Repeat {

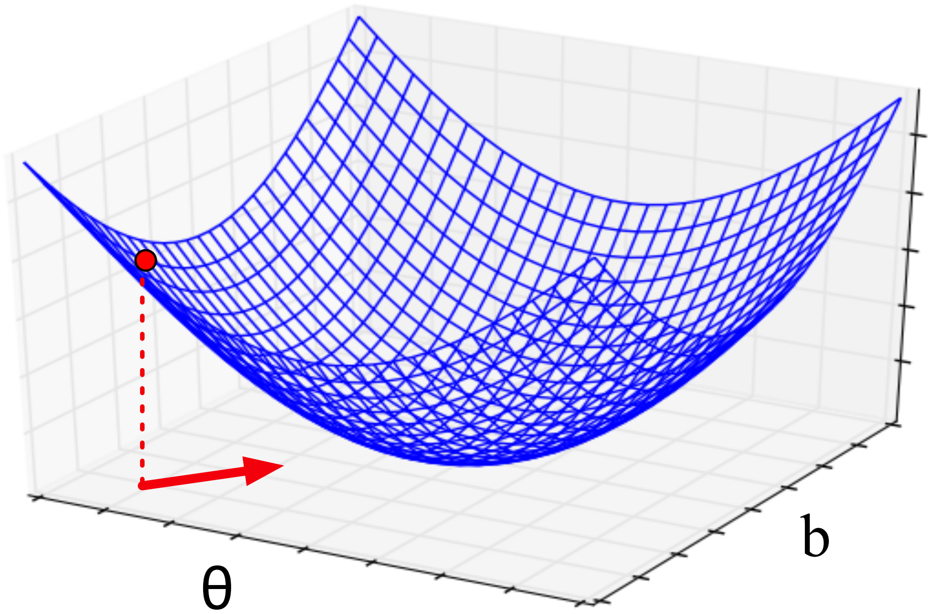
  $\theta_j := \theta_j \text{-} \eta \frac{\partial}{\partial \theta_j} J(\theta)$ (simultaneously update all θ_j)

  }

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

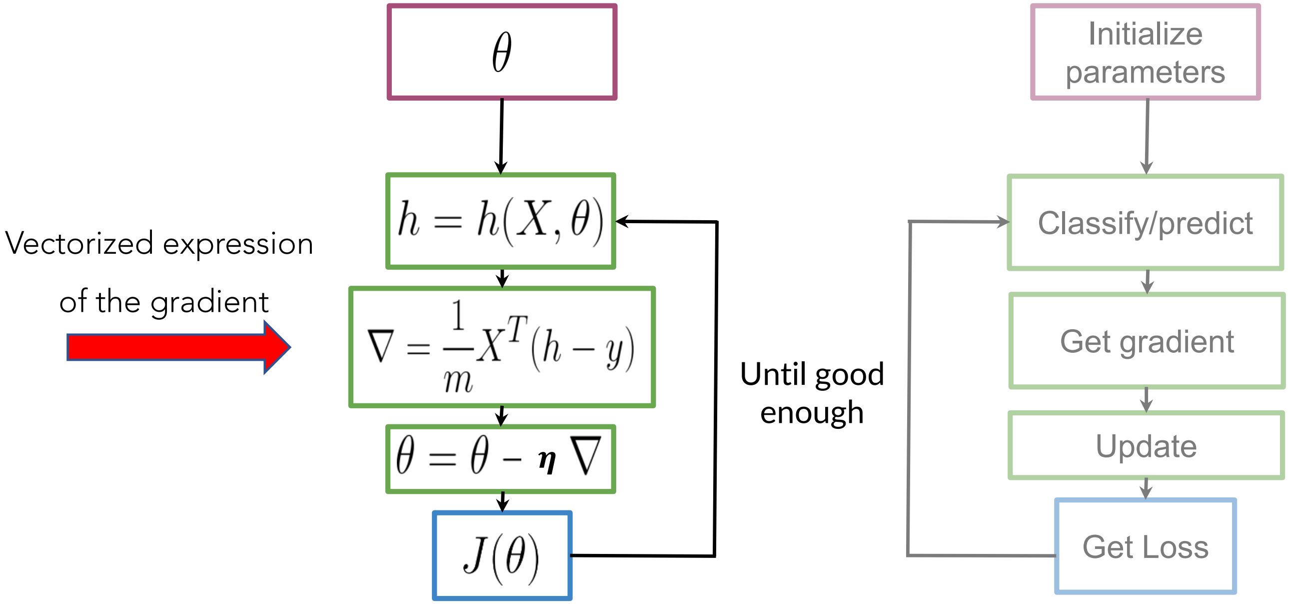
Minimizing the loss

- For logistic regression, the loss function is convex
- A convex function has just one minimum
- Gradient descent starting from any point is guaranteed to find the minimum
  - The **gradient** of a function of many variables is a vector pointing in the direction of the greatest increase in a function
- Gradient Descent: Find the gradient of the loss function at the current point and move in the **opposite** direction



LR training

- To train LR function, the following procedure is performed
  - Initialize the parameter theta
  - compute the gradient to update theta
  - calculate the cost until good enough



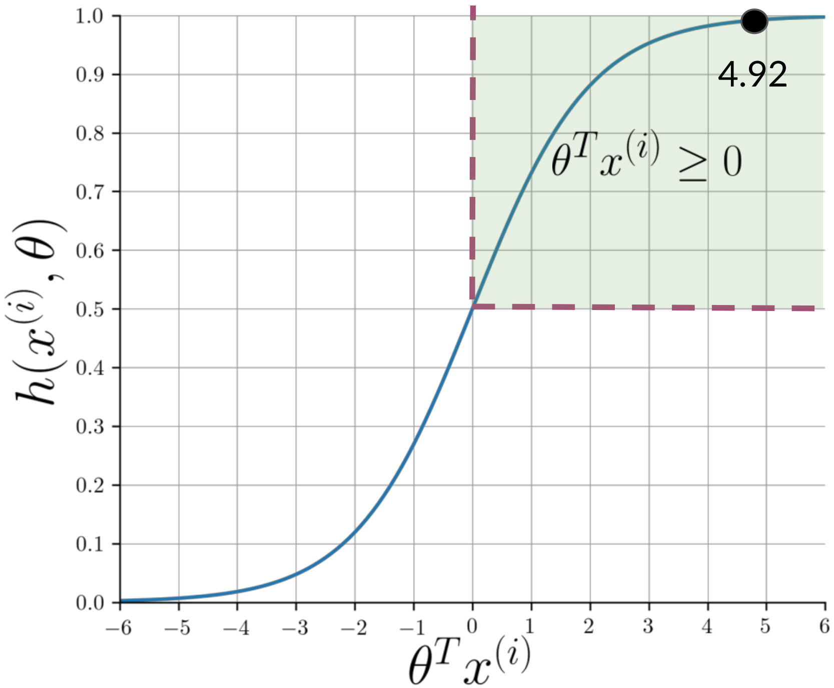
Logistic regression

- Given a tweet, you can transform it into a vector and run it through your sigmoid function to get a prediction

@AntSta and @UniParthNLPStud are tuning a GREAT AI model at https://neptunia.uniparthenope.it!!!

[tun, ai, great, model]

$$x^{(i)} = \begin{bmatrix} 1 \\ 3476 \\ 245 \end{bmatrix} \quad \theta = \begin{bmatrix} 0.00003 \\ 0.00150 \\ -0.00120 \end{bmatrix}$$



Testing the LR classifier

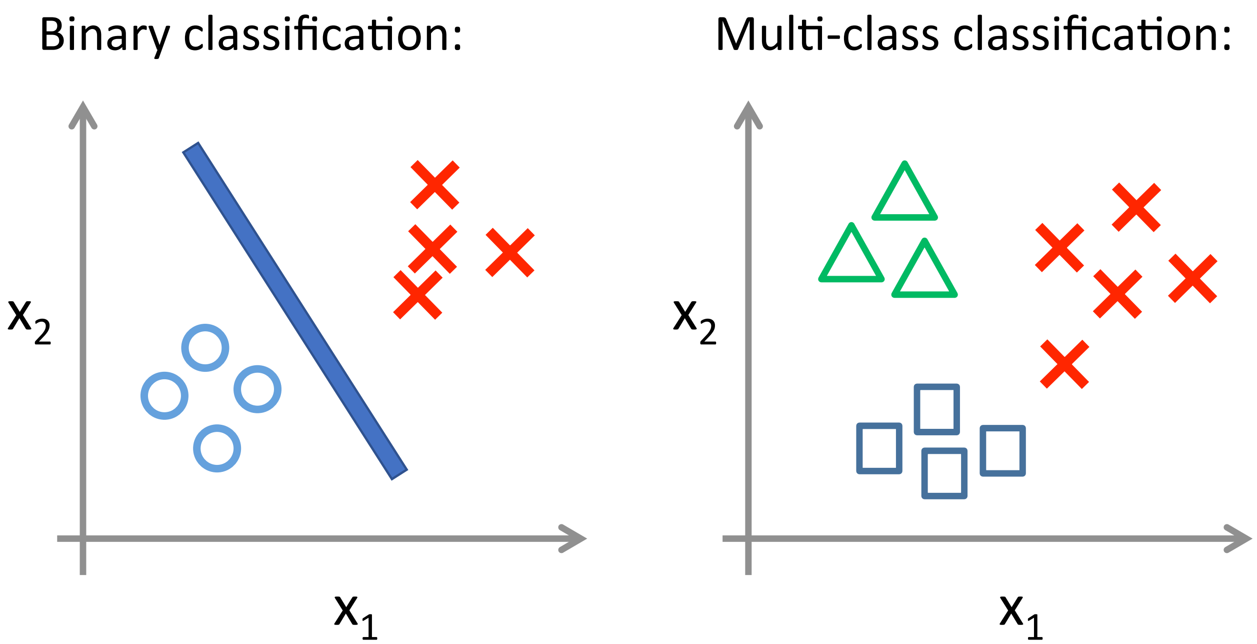
- Compute the LR prediction on each tweet from a test set and compare it to corresponding label

$$y_{val}^{(i)} = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 1 \end{bmatrix} \qquad pred^{(i)} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

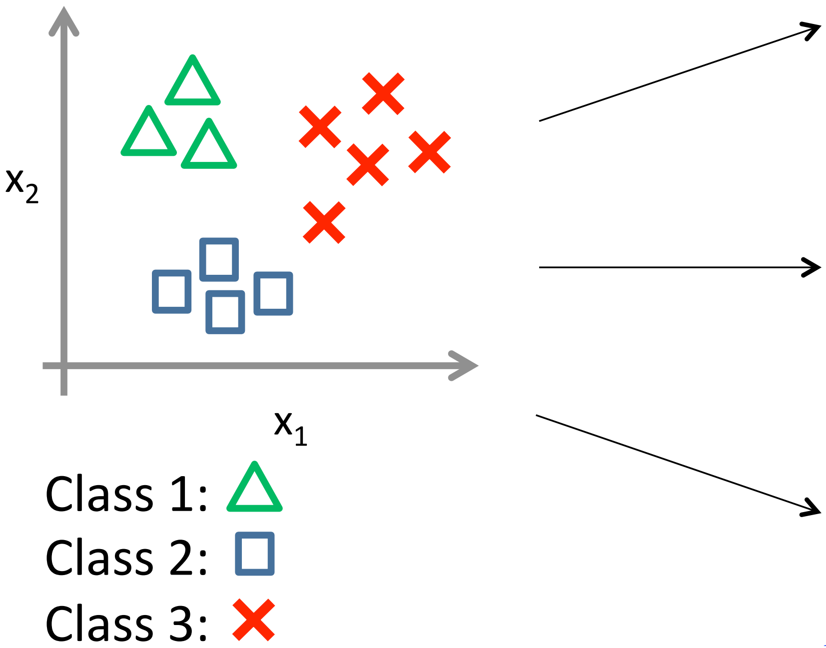
$$\text{Accuracy} \longrightarrow \sum_{i=1}^{m} \frac{(pred^{(i)} == y_{val}^{(i)})}{m}$$

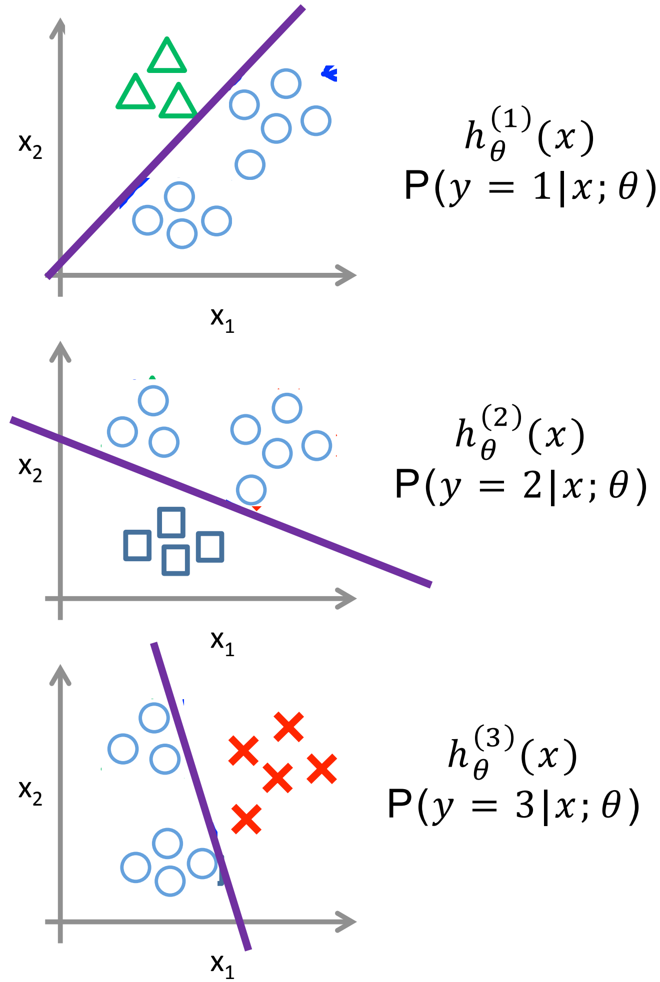
Multiclass Classification

- **Sentiment analysis**: Positive, Negative, Neutral
- **Email foldering/tagging**: Work, Friends, Family, Hobby



One-vs-all





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

One-vs-all

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