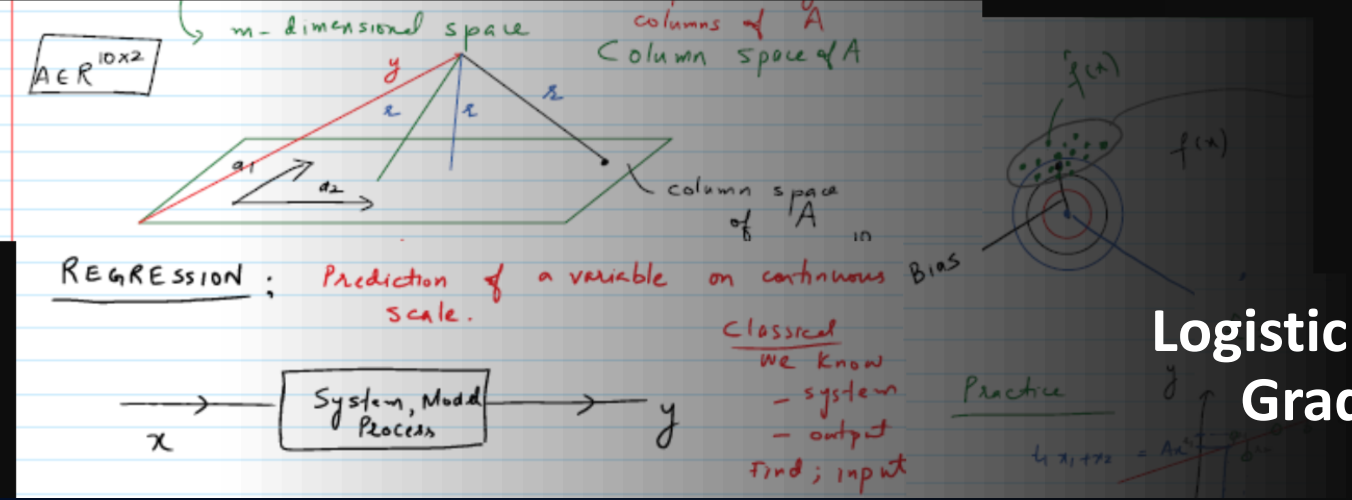

m-dimensional space
columns of A
Column space of A
$A \in R^{10\times2}$
column space of A in
REGRESSION: Prediction of a variable on continuous scale.
Classical
We know
- system
- output
Find; input
System, Model Process
x
y
Practice
Bias
f(x)

Machine Learning
EE514 – CS535

Logistic Regression: Overview, Loss Function, Gradient Descent and Multi-class case

Zubair Khalid

School of Science and Engineering
Lahore University of Management Sciences

https://www.zubairkhalid.org/ee514_2021.html

LUMS
A Not-for-Profit University

Outline

- Logistic Regression
- Decision Boundaries
- Loss/Cost Function
- Logistic Regression Gradient Descent
- Multi-class Logistic Regression

Classification

Recap:

- We assume we have training data D given by

$$D = \{(\mathbf{x_1}, y_1), (\mathbf{x_2}, y_2), \ldots, (\mathbf{x_n}, y_n)\} \subseteq \mathcal{X}^d \times \mathcal{Y}$$

Binary or Binomial Classification:

- $\mathcal{Y} = \{0, 1\}$ or $\mathcal{Y} = \{-1, 1\}$

– *Disease detection, spam email detection, fraudulent transaction, win/loss prediction, etc.*

Multi-class (Multinomial) Classification:

- $\mathcal{Y} = \{1, 2, \ldots, M\}$ (M-class classification)

– *Emotion Detection.*

– *Vehicle Type, Make, model, of the vehicle from the images streamed by road cameras.*

– *Speaker Identification from Speech Signal.*

– *Sentiment Analysis (Categories: Positive, Negative, Neutral), Text Analysis.*

– *Take an image of the sky and determine the pollution level (healthy, moderate, hazard).*

Logistic Regression

Overview:

- kNN: Instance based Classifier
- Naïve Bayes: Generative Classifier
  - Indirectly compute $P(y|x)$ as $P(x|y)$ $P(y)$ from the data using Bayes rule
- **Logistic Regression:** Discriminative Classifier
  - Estimate $P(y|x)$ directly from the data
- '**Logistic regression**' is an algorithm to carry out classification.
  - Name is misleading; the word 'regression' is due to the fact that the method attempts to fit a linear model in the feature space.
- Instead of predicting class, we compute the probability of instance being that class.

- Mathematically, model is characterized by variables $\boldsymbol{\theta}$.

$$h_{\boldsymbol{\theta}}(\mathbf{x}) = P(y|\mathbf{x})$$

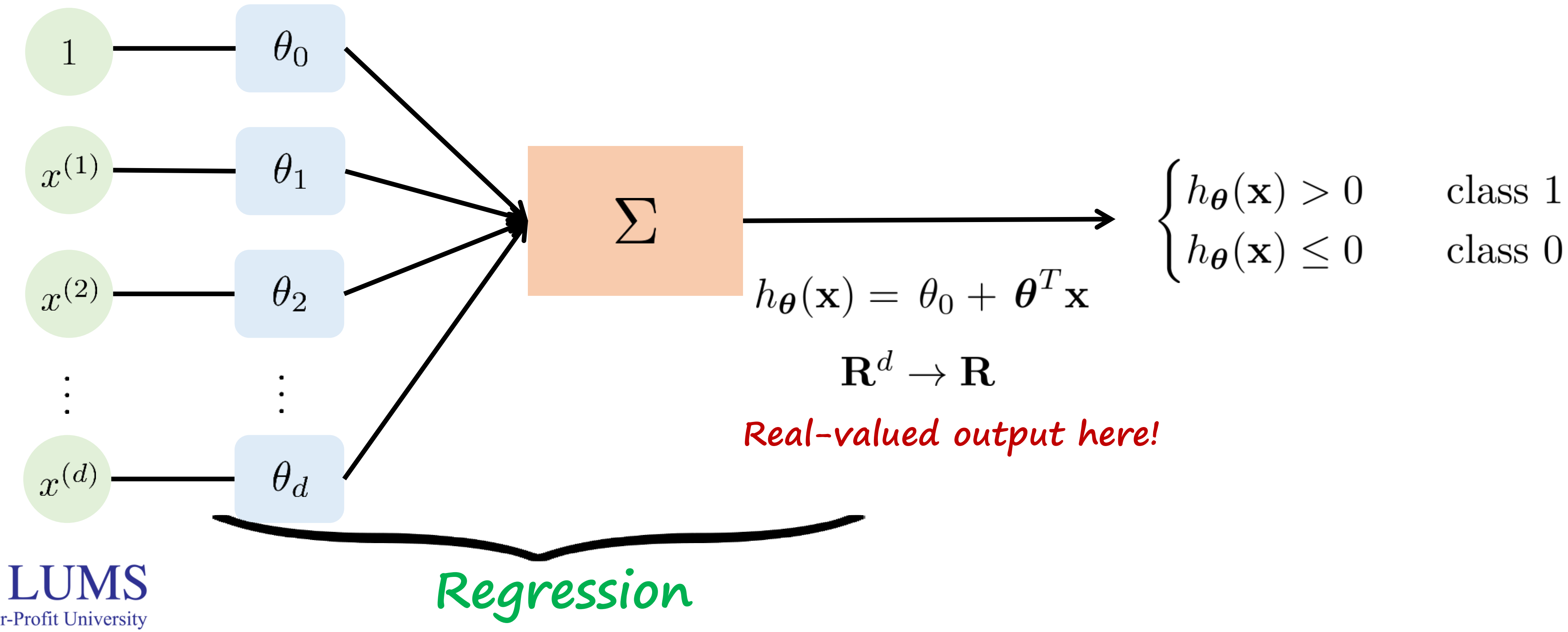
Posterior probability

- A simple form of a neural network.

Logistic Regression

Model:

- Consider a binary classification problem.
- We have a multi-dimensional feature space (d features).
- Features can be categorical (e.g., gender, ethnicity) or continuous (e.g., height, temperature).
- **Logistic regression model:**



$$\begin{cases} h_{\boldsymbol{\theta}}(\mathbf{x}) > 0 & \text{class } 1 \\ h_{\boldsymbol{\theta}}(\mathbf{x}) \leq 0 & \text{class } 0 \end{cases}$$

$$h_{\boldsymbol{\theta}}(\mathbf{x}) = \theta_0 + \boldsymbol{\theta}^T\mathbf{x}$$

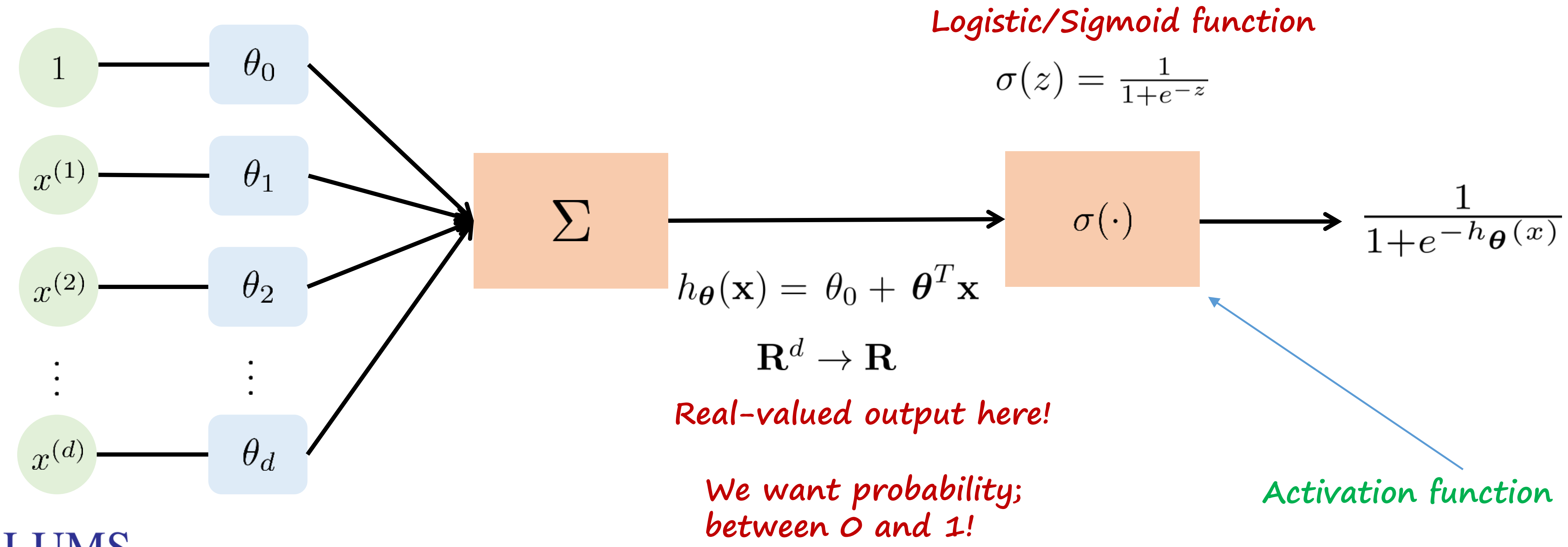
$$\mathbf{R}^d \to \mathbf{R}$$

Real-valued output here!

Logistic Regression

Model:

- Consider a binary classification problem.
- We have a multi-dimensional feature space (d features).
- Features can be categorical (e.g., gender, ethnicity) or continuous (e.g., height, temperature).
- <u>**Logistic regression model:**</u>



Logistic/Sigmoid function

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

$$h_{\boldsymbol{\theta}}(\mathbf{x}) = \theta_0 + \boldsymbol{\theta}^T\mathbf{x}$$

$$\mathbf{R}^d \rightarrow \mathbf{R}$$

Real-valued output here!

We want probability;
between 0 and 1!

Activation function

Logistic Regression

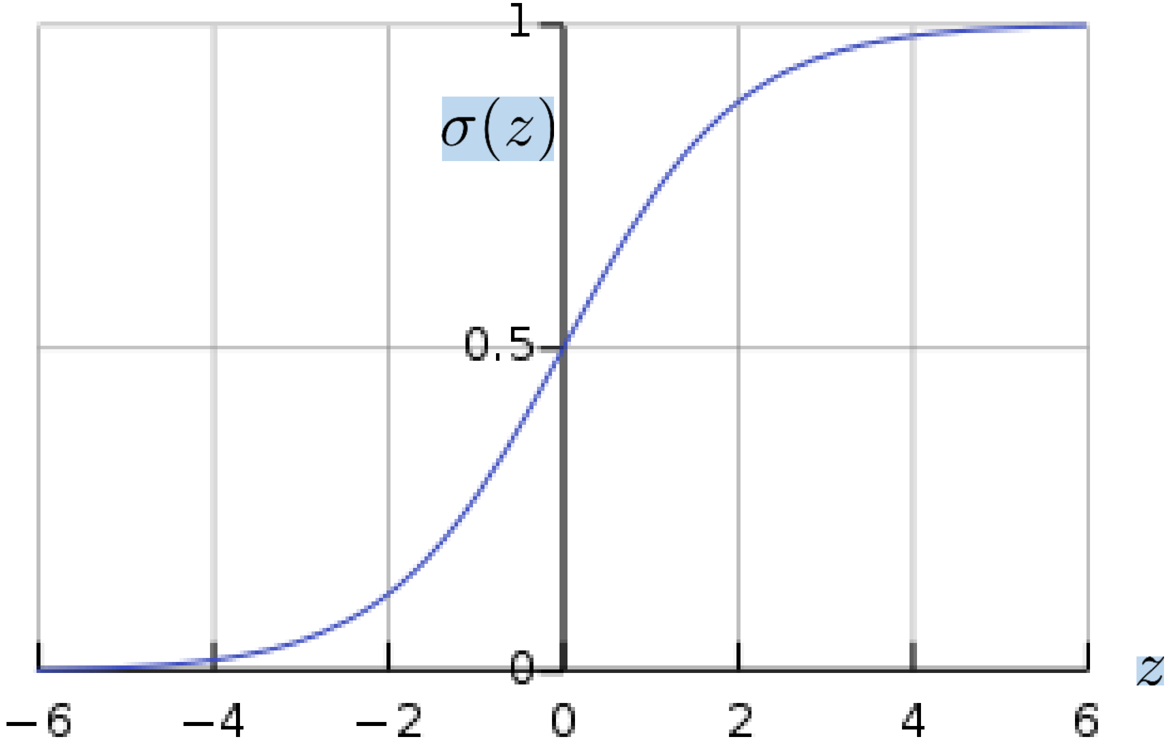
Logistic (Sigmoid) Function

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

- Interpretation: maps $(-\infty, \infty)$ to $(0, 1)$
- Squishes values in $(-\infty, \infty)$ to $(0, 1)$
- It is differentiable.
- Generalized logistic function:

$$\sigma(z) = \frac{L}{1 + e^{-k(z-z_0)}}$$

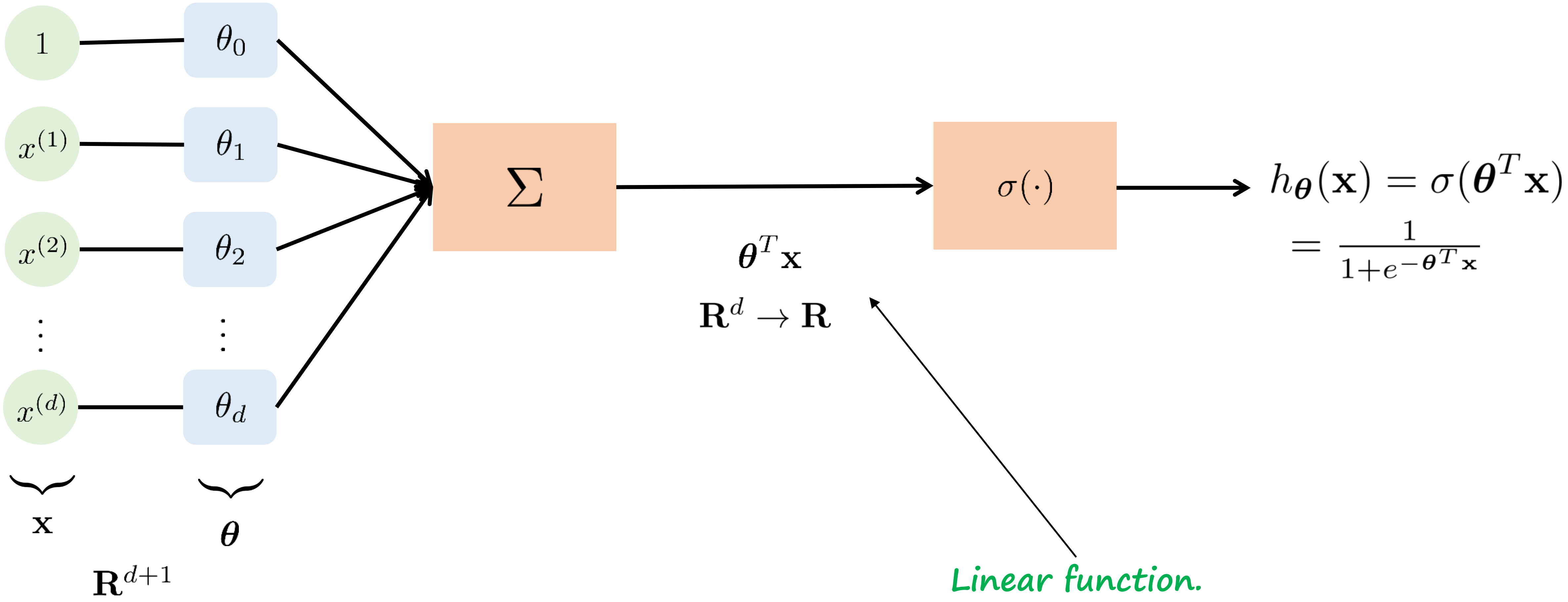
- Sigmoid: because of S shaped curve



Logistic Regression

Change in notation:

- Treat bias term as an input feature for notational convenience.



$$\boldsymbol{\theta}^T\mathbf{x}$$

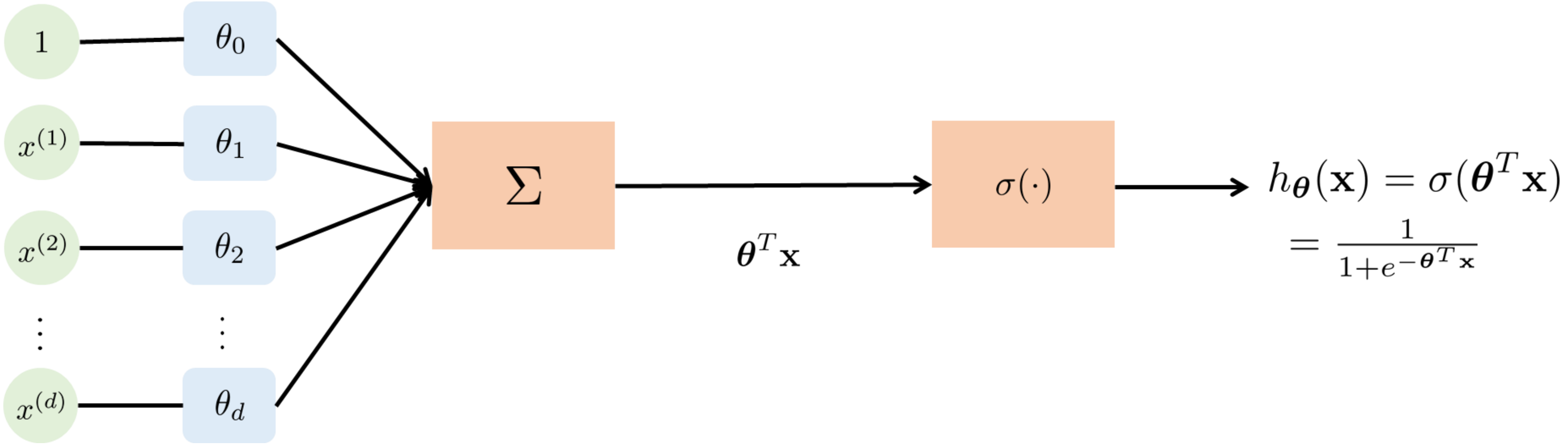
$$\mathbf{R}^d \to \mathbf{R}$$

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

Linear function.
Linear Regression.

Logistic Regression

Classification:


1
$x^{(1)}$
$x^{(2)}$
$\vdots$
$x^{(d)}$
θ_0
θ_1
θ_2
$\vdots$
θ_d
Σ
$\boldsymbol{\theta}^T\mathbf{x}$
$\sigma(\cdot)$
$h_{\boldsymbol{\theta}}(\mathbf{x}) = \sigma(\boldsymbol{\theta}^T\mathbf{x})$
$= \frac{1}{1+e^{-\boldsymbol{\theta}^T\mathbf{x}}}$

- $h_{\boldsymbol{\theta}}(\mathbf{x}) = P(y = 1|\mathbf{x})$ represents the probability of class membership.
- Assign class by applying threshold as

$$\hat{y} = \begin{cases} \text{Class 1} & \sigma(\boldsymbol{\theta}^T\mathbf{x}) > 0.5 \\ \text{Class 0} & \text{otherwise} \end{cases}$$

- 0.5 is the threshold defining decision boundary.
- We can also use values other than 0.5 as threshold.

Logistic Regression

One more interpretation:

$$P(y=1|\mathbf{x}) = h_{\boldsymbol{\theta}}(\mathbf{x}) = \frac{1}{1+e^{-\boldsymbol{\theta}^T\mathbf{x}}} \qquad P(y=0|\mathbf{x}) = 1 - h_{\boldsymbol{\theta}}(\mathbf{x}) = \frac{e^{-\boldsymbol{\theta}^T\mathbf{x}}}{1+e^{-\boldsymbol{\theta}^T\mathbf{x}}}$$

- The odds in favor of an event with probability p is $p/(1-p)$.
- Define odds of class 1. $$\frac{P(y=1|\mathbf{x})}{P(y=0|\mathbf{x})} = \frac{1}{e^{-\boldsymbol{\theta}^T\mathbf{x}}}$$
- Taking log of odds of class 1.

$$\log\frac{P(y=1|\mathbf{x})}{P(y=0|\mathbf{x})} = \log\frac{1}{e^{-\boldsymbol{\theta}^T\mathbf{x}}} = -\log e^{-\boldsymbol{\theta}^T\mathbf{x}} = \boldsymbol{\theta}^T\mathbf{x}$$

- Interpretation:
  logistic regression considers log odds as a **linear function** of $\mathbf{x}$
  logistic regression – a **linear classifier** of log of odds.

Logistic Regression

Example:

- *Disease prediction: Diagnose cancer given size of the tumor.*

- Tumor size, x
- Binary output, $y = 0$ if tumor is benign and $y = 1$ for malignant tumor.
- Linear regression model attempt

$$h_{\boldsymbol{\theta}}(x) = \boldsymbol{\theta}^T \mathbf{x} = \theta_0 + \theta_1 x$$

- output is real-valued $(-\infty, \infty)$

- Logistic regression model

$$h_{\boldsymbol{\theta}}(x) = \sigma(\theta_0 + \theta_1 x) = \frac{1}{1+e^{-(\theta_0 + \theta_1 x)}}$$

sigmoid squishes values from $(-\infty, \infty)$ to $(0, 1)$

- If $h_{\boldsymbol{\theta}}(x) = 0.65$ for any tumor size x, class label? malignant, because $h_{\boldsymbol{\theta}}(\mathbf{x}) = P(y = 1|\mathbf{x})$

Outline

- Logistic Regression
- Decision Boundaries
- Loss/Cost Function
- Logistic Regression Gradient Descent
- Multi-class Logistic Regression

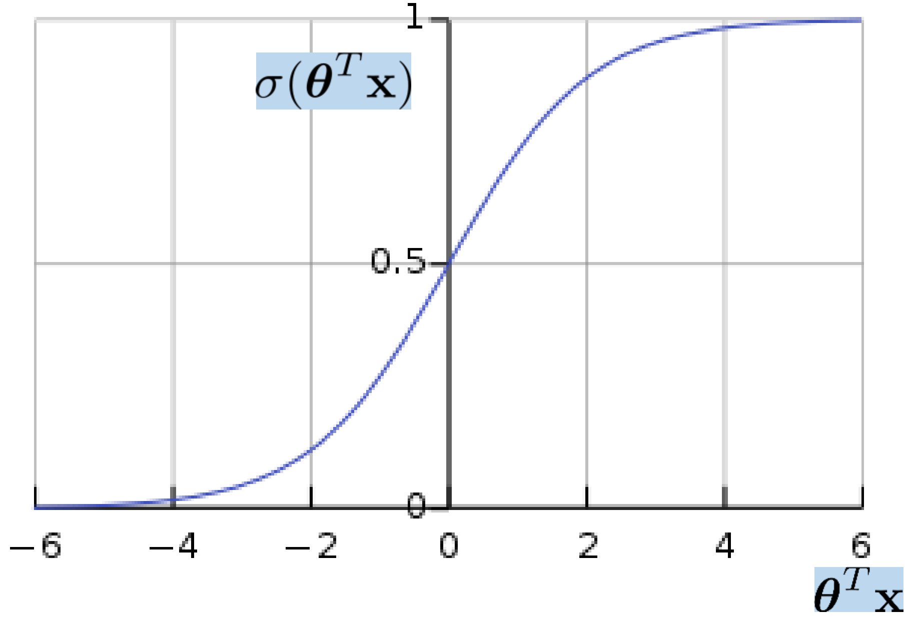
Logistic Regression

Decision Boundary:

$$P(y = 1|\mathbf{x}) = h_{\boldsymbol{\theta}}(\mathbf{x}) = \sigma(\boldsymbol{\theta}^T\mathbf{x}) = \frac{1}{1 + e^{-\boldsymbol{\theta}^T\mathbf{x}}}$$

$$\hat{y} = \begin{cases} \text{Class 1} & \sigma(\boldsymbol{\theta}^T\mathbf{x}) > 0.5 \\ \text{Class 0} & \text{otherwise} \end{cases}$$

$$\hat{y} = \begin{cases} \text{Class 1} & \boldsymbol{\theta}^T\mathbf{x} > 0 \\ \text{Class 0} & \text{otherwise} \end{cases}$$



- All $\mathbf{x}$ for which $\boldsymbol{\theta}^T\mathbf{x} > 0$ classified as Class 1.
- What does $\boldsymbol{\theta}^T\mathbf{x} > 0$ represent?
  - It represents a half-space in d-dimensional space.
  - $\boldsymbol{\theta}^T\mathbf{x} = 0$ represents a hyperplane in d-dimensional space. Need a brief explanation!

LUMS
A Not-for-Profit University

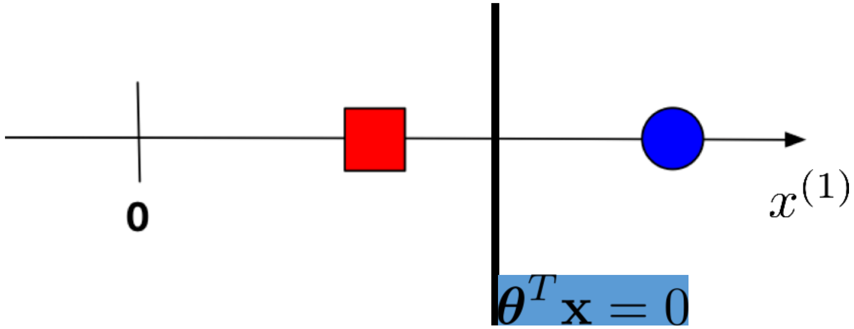
Logistic Regression

Hyper-Plane:

- $\boldsymbol{\theta}^T\mathbf{x} = 0$ represent a hyperplane in d-dimensional space.

- $d = 1$

$$\boldsymbol{\theta}^T\mathbf{x} = \theta_0 + \theta_1 x^{(1)} = 0$$

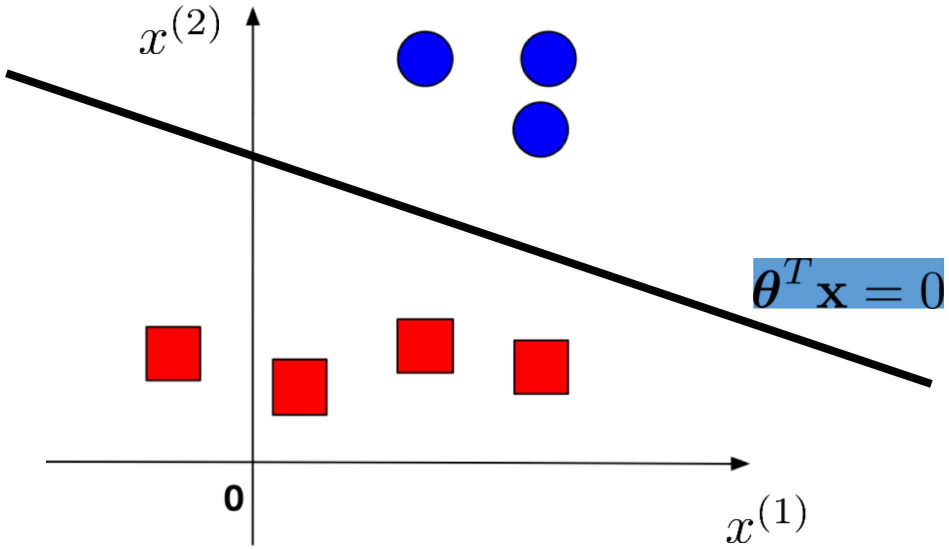


- $d = 2$

$$\boldsymbol{\theta}^T\mathbf{x} = \theta_0 + \theta_1 x^{(1)} + \theta_2 x^{(2)} = 0$$

θ_1 and θ_2 defines a normal to the hyper-plane.

- Hyper-plane $\boldsymbol{\theta}^T\mathbf{x} = 0$ divides the space into two half-spaces.
  - Half-space $\boldsymbol{\theta}^T\mathbf{x} > 0$
  - Half-space $\boldsymbol{\theta}^T\mathbf{x} < 0$



LUMS
A Not-for-Profit University

Source: https://www.cs.cornell.edu/courses/cs4780/2018sp/lectures/lecturenote03.html

Logistic Regression

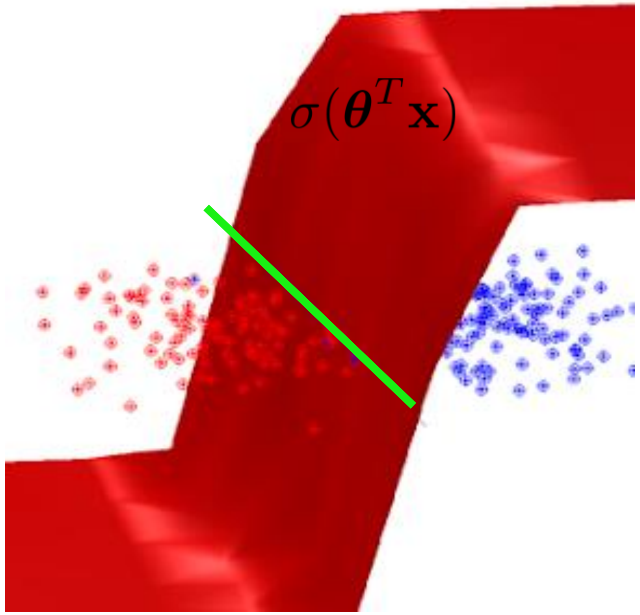
Decision Boundary - Example:

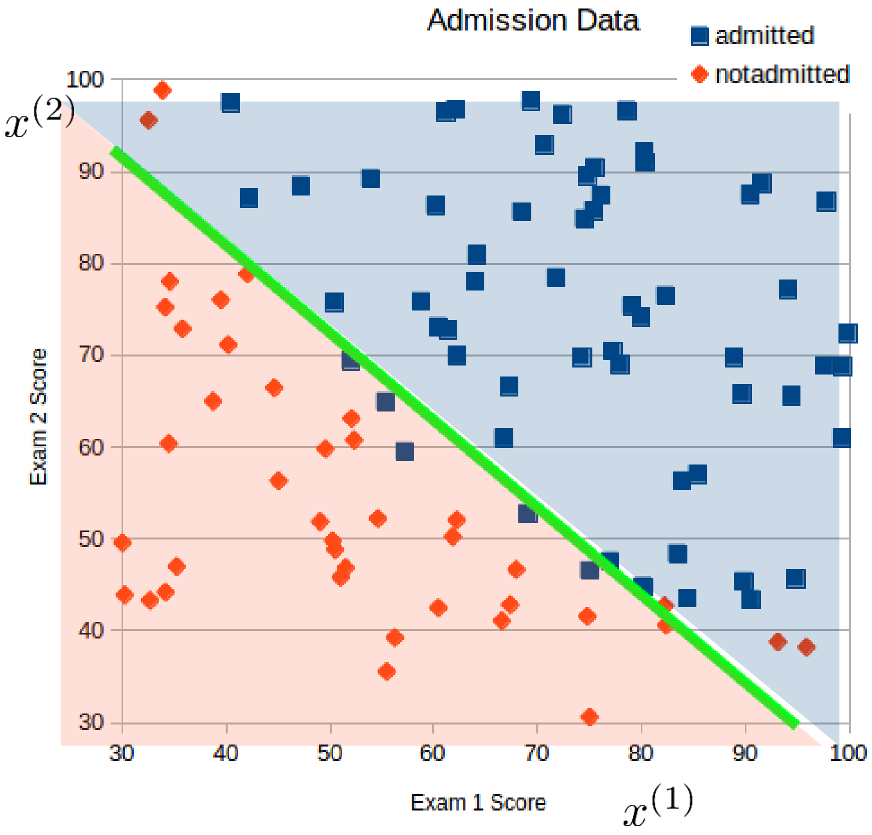
$$\hat{y} = \begin{cases} \text{Class } 1 & \boldsymbol{\theta}^T\mathbf{x} > 0 \\ \text{Class } 0 & \text{otherwise} \end{cases}$$

- Predict admission given exam 1 and exam 2 scores ($d = 2$)
- All $\mathbf{x}$ for which $\boldsymbol{\theta}^T\mathbf{x} > 0$ classified as Class 1.
- $\boldsymbol{\theta}^T\mathbf{x} = \theta_0 + \theta_1 x^{(1)} + \theta_2 x^{(2)} = 0$
- Given after learning from the data.

$$\theta_0 = -92 \qquad \theta_1 = 92/95 \qquad \theta_2 = 1$$

- Sigmoid returns close to 1 or 0
  for points farther from the boundary.





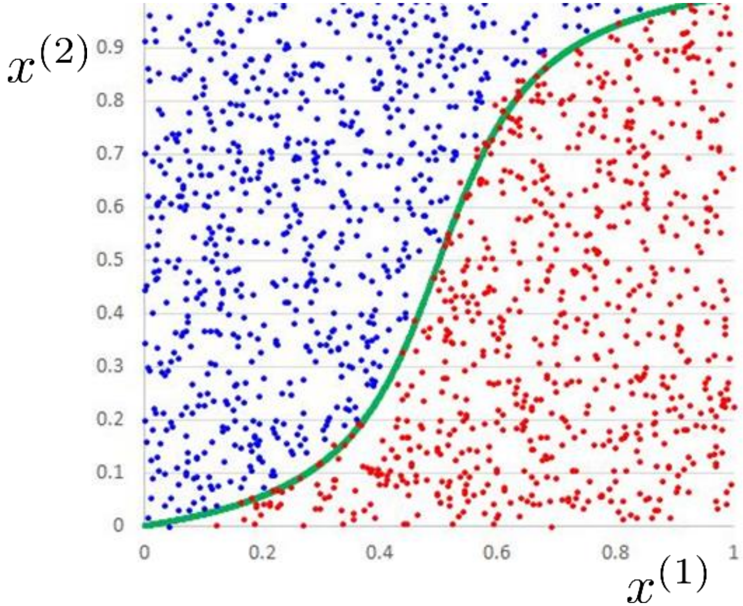
Logistic Regression

Non-linear Decision Boundary:

- Can we have non-linear decision boundaries in logistic regression?
- We first understand the origin of the linear decision boundary.
- $\boldsymbol{\theta}^T\mathbf{x} = 0$ represents a linear combination of the features.
- Connect with the concept of polynomial regression.
- Replace linear with polynomial; consider the following model, for example, for $d = 2$,

Linear boundary: $h_{\boldsymbol{\theta}}(\mathbf{x}) = \sigma(\theta_0 + \theta_1 x^{(1)} + \theta_2 x^{(2)})$

Non-linear boundary: $h_{\boldsymbol{\theta}}(\mathbf{x}) = \sigma\left(\theta_0+\theta_1 x^{(1)}+\theta_2 x^{(2)}+\theta_3\left(x^{(1)}\right)^2+\theta_4\left(x^{(2)}\right)^2\right)$



LUMS
A Not-for-Profit University

Logistic Regression

Non-linear Decision Boundary:

Non-linear boundary: $h_{\boldsymbol{\theta}}(\mathbf{x}) = \sigma\left(\theta_0 + \theta_1 x^{(1)} + \theta_2 x^{(2)} + \theta_3 \left(x^{(1)}\right)^2 + \theta_4 \left(x^{(2)}\right)^2\right)$

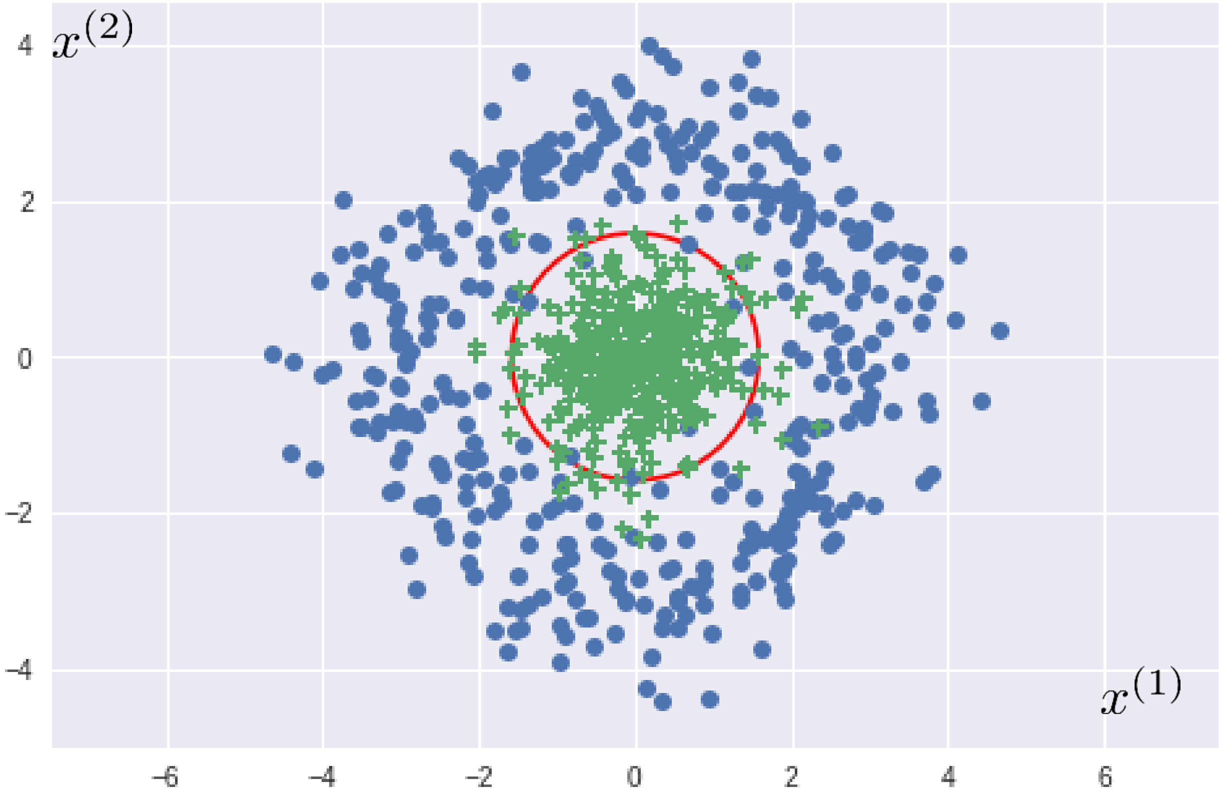
- Given after learning from the data.

$\theta_0 = -2.25$ $\quad \theta_1 = \theta_2 = 0$ $\quad \theta_3 = \theta_4 = 1$

$$h_{\boldsymbol{\theta}}(\mathbf{x}) = \sigma\left(-1 + \left(x^{(1)}\right)^2 + \left(x^{(2)}\right)^2\right)$$

Boundary: $\left(x^{(1)}\right)^2 + \left(x^{(2)}\right)^2 = 2.25$

(Circle of radius 1.5)



Outline

- Logistic Regression
- Decision Boundaries
- Loss/Cost Function
- Logistic Regression Gradient Descent
- Multi-class Logistic Regression

Logistic Regression

Model Training (Learning of Parameters):

- We assume we have training data D given by

$$D = \{(\mathbf{x_1}, y_1), (\mathbf{x_2}, y_2), \dots, (\mathbf{x_n}, y_n)\} \subseteq \mathcal{X}^d \times \mathcal{Y}$$

- $\mathcal{Y} = \{0, 1\}$

Logistic regression model:

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

$\boldsymbol{\theta} = [\theta_0, \theta_1, \dots, \theta_d]$ $\boldsymbol{\theta}$ represents $d+1$ parameters of the model.

- *Objective:* Given the training data, that is **n** training samples, we want to find the parameters of the model.
- We first formulate the loss (cost, objective) function that we want to optimize.
- We will employ gradient descent to solve the optimization problem.

Logistic Regression

Loss/Cost Function:

- **Candidate 1:** Squared-error, the one we used in regression.

$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{2}\sum_{i=1}^{n}\left(h_{\boldsymbol{\theta}}(\mathbf{x}_i) - y_i\right)^2 = \frac{1}{2}\sum_{i=1}^{n}\left(\sigma(\boldsymbol{\theta}^T\mathbf{x}_i) - y_i\right)^2$$

$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{2}\sum_{i=1}^{n}\left(\frac{1}{1+e^{-\boldsymbol{\theta}^T\mathbf{x}_i}} - y_i\right)^2$$

- We wish to have a loss function that is differentiable and convex.
- The squared-error is not a convex function due to sigmoid operation.
- Due to non-convexity, we cannot numerically solve to find the global minima.
- Furthermore, the hypothesis function is estimating probability and we do not use difference operation to determine the distance between the two probability distributions.

Logistic Regression

Loss/Cost Function:

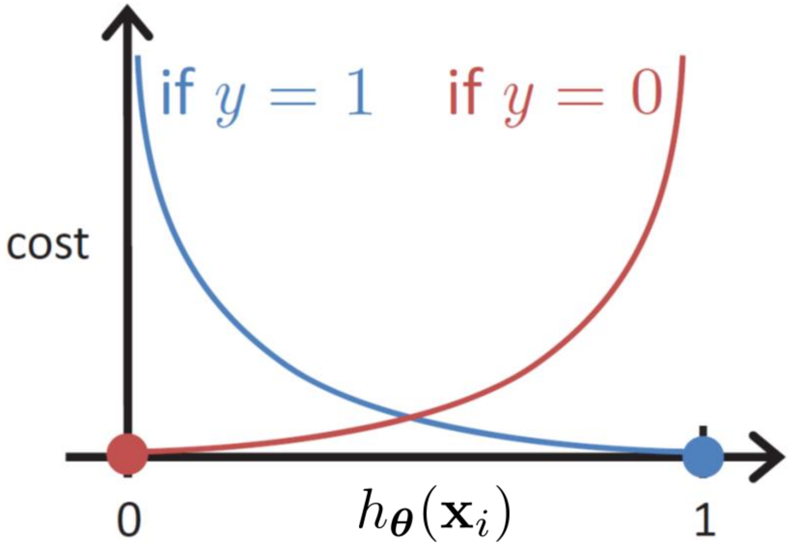
- **Candidate 2:** *Cross entropy loss or Log loss* function is used when classifier output is in terms of probability.
- Idea: Cross-entropy loss increases when the predicted probability diverges from the actual label.
  - If the actual class is 1 and the model predicts 0, we should highly penalize it and vice-versa.

- **Loss/cost function** for single training example:

$$\text{cost}\big(h_{\boldsymbol{\theta}}(\mathbf{x}_i), y_i\big) = \begin{cases} -\log(h_{\boldsymbol{\theta}}(\mathbf{x}_i)) & y = 1 \\ -\log(1 - h_{\boldsymbol{\theta}}(\mathbf{x}_i)) & y = 0 \end{cases}$$

For $y_i = 1$,

- cost=0 when $h_{\boldsymbol{\theta}}(\mathbf{x}_i) = 1$
- cost=∞ when $h_{\boldsymbol{\theta}}(\mathbf{x}_i) = 0$



- *Mismatch is penalized:* larger mistakes get larger penalties

Logistic Regression

Loss/Cost Function:

- We can also express the loss/cost for one training sample as

$$\text{cost}\big(h_{\boldsymbol{\theta}}(\mathbf{x}_i), y_i\big) = \begin{cases} -\log(h_{\boldsymbol{\theta}}(\mathbf{x}_i)) & y = 1 \\ -\log(1 - h_{\boldsymbol{\theta}}(\mathbf{x}_i)) & y = 0 \end{cases}$$

$$\text{cost}\big(h_{\boldsymbol{\theta}}(\mathbf{x}_i), y_i\big) = -y_i \log(h_{\boldsymbol{\theta}}(\mathbf{x}_i)) - (1 - y_i) \log(1 - h_{\boldsymbol{\theta}}(\mathbf{x}_i))$$

- Using this formulation, we define the loss function:

$$\mathcal{L}(\boldsymbol{\theta}) = -\sum_{i=1}^{n} y_i \log(h_{\boldsymbol{\theta}}(\mathbf{x}_i)) + (1 - y_i) \log(1 - h_{\boldsymbol{\theta}}(\mathbf{x}_i))$$

- Since cost for each sample penalizes mismatch, this loss function prefers the correct class label to be more likely.
- Finding parameters that minimizes loss function or maximizes negative of the loss function is, in fact, maximum likelihood estimation (MLE). How?

Logistic Regression

Loss/Cost Function:

- We can also reformulate the loss/cost for one training sample as

$$\text{cost}\big(h_{\boldsymbol{\theta}}(\mathbf{x}_i), y_i\big) = -y_i \log(h_{\boldsymbol{\theta}}(\mathbf{x}_i)) - (1 - y_i) \log(1 - h_{\boldsymbol{\theta}}(\mathbf{x}_i))$$

$$\text{cost}\big(h_{\boldsymbol{\theta}}(\mathbf{x}_i), y_i\big) = -\log\left(h_{\boldsymbol{\theta}}(\mathbf{x}_i))^{y_i} \; (1 - h_{\boldsymbol{\theta}}(\mathbf{x}_i))^{(1-y_i)} \right)$$

Inside the log; we have a

- likelihood function since $h_{\boldsymbol{\theta}}(\mathbf{x}_i)$ gives us probability of $y_i = 1$.
- probability mass function, $(p^{y_i})(1-p)^{1-y_i}$, of Bernoulli random variable.
- Cost is the negative log-likelihood function, also referred to as cross-entropy loss.
- Minimizing cost; equivalent to maximization of log-likelihood or likelihood.
- Therefore, $\boldsymbol{\theta}$ that minimizes $\mathcal{L}(\boldsymbol{\theta})$, maximizes likelihood.

Logistic Regression

Model Training (Learning of Parameters):

- We have following optimization problem in hand:

$$\underset{\boldsymbol{\theta}}{\text{minimize}} \quad \mathcal{L}(\boldsymbol{\theta}) = -\sum_{i=1}^{n} y_i \log(h_{\boldsymbol{\theta}}(\mathbf{x}_i)) + (1 - y_i) \log(1 - h_{\boldsymbol{\theta}}(\mathbf{x}_i))$$

- We do not attempt to find analytical solution.
- We can use properties of convex functions, composition rules and concavity of log to show that the loss function is a convex function.
- We use gradient descent to numerically solve the optimization problem.

Outline

- Logistic Regression
- Decision Boundaries
- Loss/Cost Function
- Logistic Regression Gradient Descent
- Multi-class Logistic Regression

Logistic Regression

Gradient Descent:

- For gradient descent, we defined the following update in each iteration:

$$\theta_j \leftarrow \theta_j - \alpha \frac{\partial \mathcal{L}}{\partial \theta_j}, \quad \alpha > 0$$

- $\frac{\partial \mathcal{L}}{\partial \theta_j}$: Rate of change in the loss function with respect to θ_j
- α is referred to as step size or learning rate.
- Idea: step size in the direction of negative of the derivative.

Algorithm (we have seen this before):

Overall:

- Start with some $\boldsymbol{\theta} \in \mathbf{R}^d$ and keep updating to reduce the loss function until we reach the minimum. Repeat until convergence

Pseudo-code:

- Initialize $\boldsymbol{\theta} \in \mathbf{R}^d$.
- Repeat until convergence:

$$\theta_j \leftarrow \theta_j - \alpha \frac{\partial \mathcal{L}}{\partial \theta_j}, \quad \text{for each} \quad i = 0, 1, 2, \ldots, d \qquad \boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \alpha \nabla \mathcal{L}(\boldsymbol{\theta})$$

Note: Simultaneous update.

Logistic Regression

Gradient Descent Computation:

- How to compute $\frac{\partial \mathcal{L}}{\partial \theta_j}$?

$$\mathcal{L}(\boldsymbol{\theta}) = -\sum_{i=1}^{n} y_i \log(h_{\boldsymbol{\theta}}(\mathbf{x}_i)) + (1 - y_i) \log(1 - h_{\boldsymbol{\theta}}(\mathbf{x}_i))$$

- Derivative is linear; drop subscript i and compute for each training sample.

$$\frac{\partial}{\partial \theta_j}\Big(y \log(h_{\boldsymbol{\theta}}(\mathbf{x})) + (1 - y) \log(1 - h_{\boldsymbol{\theta}}(\mathbf{x}))\Big) = \left(y \frac{1}{h_{\boldsymbol{\theta}}(\mathbf{x})} - (1 - y) \frac{1}{1 - h_{\boldsymbol{\theta}}(\mathbf{x})}\right) \frac{\partial}{\partial \theta_j}(h_{\boldsymbol{\theta}}(\mathbf{x}))$$

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

- We can write

$$\frac{\partial}{\partial \theta_j}(h_{\boldsymbol{\theta}}(\mathbf{x})) = \frac{e^{-\boldsymbol{\theta}^T \mathbf{x}}}{(1 + e^{-\boldsymbol{\theta}^T \mathbf{x}})^2} \frac{\partial}{\partial \theta_j}(\boldsymbol{\theta}^T \mathbf{x}) = \frac{e^{-\boldsymbol{\theta}^T \mathbf{x}}}{1 + e^{-\boldsymbol{\theta}^T \mathbf{x}}} \frac{1}{1 + e^{-\boldsymbol{\theta}^T \mathbf{x}}} x^{(j)} \qquad = h_{\boldsymbol{\theta}}(\mathbf{x})(1 - h_{\boldsymbol{\theta}}(\mathbf{x})) \, x^{(j)}$$

Logistic Regression

Gradient Descent Computation:

$$\frac{\partial}{\partial\theta_j}\Big(y\log(h_{\boldsymbol\theta}(\mathbf{x})) + (1-y)\log(1-h_{\boldsymbol\theta}(\mathbf{x}))\Big)$$

$$= \left(y\frac{1}{h_{\boldsymbol\theta}(\mathbf{x})} - (1-y)\frac{1}{1-h_{\boldsymbol\theta}(\mathbf{x})}\right)\frac{\partial}{\partial\theta_j}(h_{\boldsymbol\theta}(\mathbf{x}))$$

$$\frac{\partial}{\partial\theta_j}(h_{\boldsymbol\theta}(\mathbf{x})) = h_{\boldsymbol\theta}(\mathbf{x})(1-h_{\boldsymbol\theta}(\mathbf{x}))\; x^{(j)}$$

$$= \frac{y(1-h_{\boldsymbol\theta}(\mathbf{x})) - (1-y)h_{\boldsymbol\theta}(\mathbf{x})}{h_{\boldsymbol\theta}(\mathbf{x})(1-h_{\boldsymbol\theta}(\mathbf{x}))}\; h_{\boldsymbol\theta}(\mathbf{x})(1-h_{\boldsymbol\theta}(\mathbf{x}))\; x^{(j)}$$

$$= (y - h_{\boldsymbol\theta}(\mathbf{x}))\, x^{(j)} \quad = -(h_{\boldsymbol\theta}(\mathbf{x}) - y)\; x^{(j)}$$

Overall:

$$\frac{\partial\mathcal{L}(\boldsymbol\theta)}{\partial\theta_j} = -\sum_{i=1}^{n}\frac{\partial}{\partial\theta_j}\Big(y_i\log(h_{\boldsymbol\theta}(\mathbf{x}_i)) + (1-y_i)\log(1-h_{\boldsymbol\theta}(\mathbf{x}_i))\Big)$$

$$\frac{\partial\mathcal{L}(\boldsymbol\theta)}{\partial\theta_j} = \sum_{i=1}^{n}(h_{\boldsymbol\theta}(\mathbf{x}_i) - y_i)\; x_i^{(j)}$$

Outline

- Logistic Regression
- Decision Boundaries
- Loss/Cost Function
- Logistic Regression Gradient Descent
- Multi-class Logistic Regression

Logistic Regression

Multi-Class (Multinomial) Classification:

- $\mathcal{Y} = \{0, 1, 2, \dots, M-1\}$ (M-class classification)

Option 1: Build a one-vs-all (OvA) one-vs-rest (OvR) classifier:

- Train M different binary logistic regression classifiers $h_0(\mathbf{x}), h_1(\mathbf{x}), \dots, h_{M-1}(\mathbf{x})$.
- Classifier $h_i(\mathbf{x})$ is trained to classify if $\mathbf{x}$ belongs to i-th class or not.
- For a new test point $\mathbf{z}$, get scores for each classifier, that is, $s_i = h_i(\mathbf{z})$.
- s_i represents the probability that $\mathbf{z}$ belongs to class i.
- Predict the label as $\hat{y} = \max\limits_{i=0,1,2,\dots,M-1} \quad s_i$

Logistic Regression

Multi-Class (Multinomial) Classification:

- $\mathcal{Y} = \{0, 1, 2, \ldots, M-1\}$ (M-class classification)

Option 2: Build an all-vs-all classifier (commonly known as one-vs-one classifier):

- Train $\binom{M}{2} = \frac{(M)(M-1)}{2}$ different binary logistic regression classifiers $h_{i,j}(\mathbf{x})$.
- Classifier $h_{i,j}(\mathbf{x})$ is trained to classify if $\mathbf{x}$ belongs to i-th class or j-th class.
- For a new test point $\mathbf{z}$, get scores for each classifier, that is, $s_{i,j} = h_{i,j}(\mathbf{z})$.
- $s_{i,j}$ gives the probability of $\mathbf{z}$ being from class i and not in class j.
- Predict the label $\hat{y}$ for which the sum of probabilities is maximum.

Select label for which the sum is maximum

Example:

- Consider a problem with 3 classes, A, B and C.

Classifier 1
A vs B

$P_1(A)$
$P_1(B)$

Classifier 2
B vs C

$P_2(B)$
$P_2(C)$

Classifier 3
A vs C

$P_3(A)$
$P_3(C)$

$P_1(A) + P_3(A)$

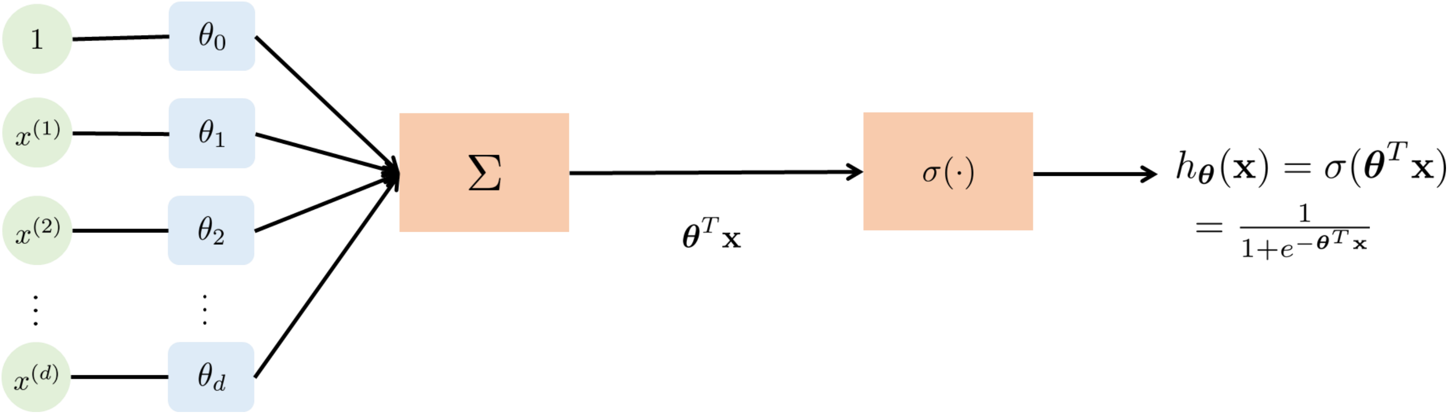
$P_1(B) + P_2(B)$

$P_2(C) + P_3(C)$

Logistic Regression

Multi-Class (Multinomial) Logistic Regression:

- **Idea:** Extend logistic regression using softmax instead of logistic (sigmoid).
- We have following logistic regression model for binary classification case (M=2).



1
$x^{(1)}$
$x^{(2)}$
$x^{(d)}$
θ_0
θ_1
θ_2
θ_d
Σ
$\boldsymbol{\theta}^T\mathbf{x}$
$\sigma(\cdot)$
$h_{\boldsymbol{\theta}}(\mathbf{x}) = \sigma(\boldsymbol{\theta}^T\mathbf{x}) = \frac{1}{1+e^{-\boldsymbol{\theta}^T\mathbf{x}}}$

- $h_{\boldsymbol{\theta}}(\mathbf{x}) = P(y=1|\mathbf{x})$ represents the probability of membership of class 1.
- Model: weighted sum of features followed by sigmoid for squishing the values of weighted sum between 0 and 1.

$$P(y=1|\mathbf{x}) = h_{\boldsymbol{\theta}}(\mathbf{x}) = \frac{1}{1+e^{-\boldsymbol{\theta}^T\mathbf{x}}}$$

$$P(y=0|\mathbf{x}) = 1 - h_{\boldsymbol{\theta}}(\mathbf{x}) = \frac{e^{-\boldsymbol{\theta}^T\mathbf{x}}}{1+e^{-\boldsymbol{\theta}^T\mathbf{x}}}$$

$$\Rightarrow$$

$$P(y=1|\mathbf{x}) = \frac{e^{\boldsymbol{\theta}^T\mathbf{x}}}{e^{\boldsymbol{\theta}^T\mathbf{x}}+1} \qquad P(y=1|\mathbf{x}) = \frac{e^{\boldsymbol{\theta}^T\mathbf{x}}}{e^{\boldsymbol{\theta}^T\mathbf{x}}+e^0}$$

$$P(y=0|\mathbf{x}) = \frac{1}{e^{\boldsymbol{\theta}^T\mathbf{x}}+1} \qquad P(y=0|\mathbf{x}) = \frac{e^0}{e^{\boldsymbol{\theta}^T\mathbf{x}}+e^0}$$

Logistic Regression

Multi-Class (Multinomial) Logistic Regression:

- For M classes, we extend the formulation of the logistic function.
- Again, note that the model gives us probability of class membership.
- We assign the label that is more likely.

- Noting this, we build a model for m-th class as

$$P(y = m|\mathbf{x}) = h_{\boldsymbol{\theta}_m}(\mathbf{x}) = \frac{e^{{\boldsymbol{\theta_m}}^T\mathbf{x}}}{\sum_{k=0}^{M-1} e^{{\boldsymbol{\theta_k}}^T\mathbf{x}}}$$

$\boldsymbol{\theta}_m-$ model parameters

- Model: weighted sum of features followed by softmax function.
- Softmax - extension of logistic function:

$$\sigma(z) = \frac{1}{1+e^{-z}} = \frac{e^z}{e^z + e^0}$$

Logistic function for 2 classes.

$$\text{softmax}(z_m) = \frac{1}{1+e^{-z}} = \frac{e^{z_m}}{\sum_{k=0}^{M-1} e^{z_k}}$$

Softmax for M classes.

Logistic Regression

Multi-Class (Multinomial) Logistic Regression:

$$P(y = m|\mathbf{x}) = h_{\boldsymbol{\theta}_m}(\mathbf{x}) = \frac{e^{\boldsymbol{\theta_m}^T \mathbf{x}}}{\sum_{k=0}^{M-1} e^{\boldsymbol{\theta_k}^T \mathbf{x}}}$$

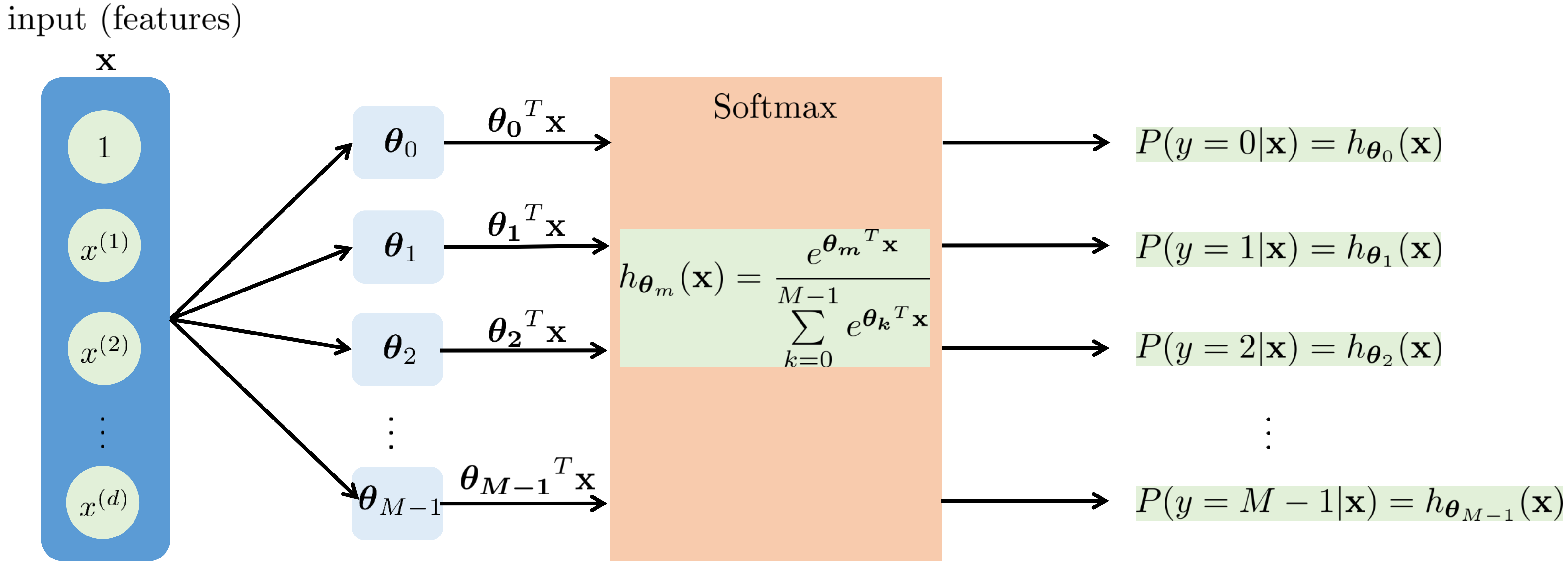
$\boldsymbol{\theta}_m -$ model parameters

- A critical assumption here: no ordinal relationship between the classes.
- Linear function for each of the m classes.
- The softmax function
  - Input: a vector of M real numbers
  - Output: M probabilities proportional to the exponentials of the input numbers.

- We have $\boldsymbol{\theta}_m = [\theta_{m,0}, \theta_{m,1}, \ldots, \theta_{m,d}]$ for each class $m = \{0, 1, \ldots, M-1\}$.
- In total, we have $(d+1) \times M$ parameters.

Logistic Regression

Multi-Class Logistic Regression – Graphical Representation of the Model:


input (features)
$\mathbf{x}$
1
$x^{(1)}$
$x^{(2)}$
$\vdots$
$x^{(d)}$
$\boldsymbol{\theta}_0$
$\boldsymbol{\theta}_1$
$\boldsymbol{\theta}_2$
$\vdots$
$\boldsymbol{\theta}_{M-1}$
$\boldsymbol{\theta_0}^T\mathbf{x}$
$\boldsymbol{\theta_1}^T\mathbf{x}$
$\boldsymbol{\theta_2}^T\mathbf{x}$
$\boldsymbol{\theta_{M-1}}^T\mathbf{x}$
Softmax
$P(y=0|\mathbf{x}) = h_{\boldsymbol{\theta}_0}(\mathbf{x})$
$P(y=1|\mathbf{x}) = h_{\boldsymbol{\theta}_1}(\mathbf{x})$
$P(y=2|\mathbf{x}) = h_{\boldsymbol{\theta}_2}(\mathbf{x})$
$\vdots$
$P(y=M-1|\mathbf{x}) = h_{\boldsymbol{\theta}_{M-1}}(\mathbf{x})$

$$h_{\boldsymbol{\theta}_m}(\mathbf{x}) = \frac{e^{\boldsymbol{\theta_m}^T\mathbf{x}}}{\sum_{k=0}^{M-1} e^{\boldsymbol{\theta_k}^T\mathbf{x}}}$$

- Prediction:

$$\hat{y} = \max_{m=0,1,2,\ldots,M-1} \quad h_{\boldsymbol{\theta}_m}(\mathbf{x})$$

LUMS
A Not-for-Profit University

Logistic Regression

Multi-Class (Multinomial) Logistic Regression – Cost Function

- For binary classification, we have:

$$\mathcal{L}(\boldsymbol{\theta}) = -\sum_{i=1}^{n} y_i \log(h_{\boldsymbol{\theta}}(\mathbf{x}_i)) + (1 - y_i)\log(1 - h_{\boldsymbol{\theta}}(\mathbf{x}_i))$$

- Extending the same for multi-class logistic regression:

$$\mathcal{L}(\boldsymbol{\theta}) = -\sum_{i=1}^{n}\sum_{m=0}^{M-1} \delta(y_i - m)\ \log\left(h_{\boldsymbol{\theta}_m}(\mathbf{x}_i)\right)$$

$$\mathcal{L}(\boldsymbol{\theta}) = -\sum_{i=1}^{n}\sum_{m=0}^{M-1} \delta(y_i - m)\ \log\left(\frac{e^{\boldsymbol{\theta_m}^T \mathbf{x}_i}}{\sum_{k=0}^{M-1} e^{\boldsymbol{\theta_k}^T \mathbf{x}_i}}\right)$$

Logistic Regression

Summary:

- Employs regression followed by mapping to probability using logistic function (binary case) or softmax function (multinomial case).
- Do not make any assumptions about distributions of classes in feature space.
- Decision boundaries separating classes are linear.
- It provides a natural probabilistic view of class predictions.
- Loss function is formulated using cross entropy loss.
- Can be trained quickly using gradient descent.
- Computationally efficient at classifying (needs inner product only)
- Model coefficients can be interpreted as indicators of importance of the features.

Feedback: Questions or Comments?

Email: zubair.khalid@lums.edu.pk