CS M146: Introduction to Machine Learning
Logistic Regression

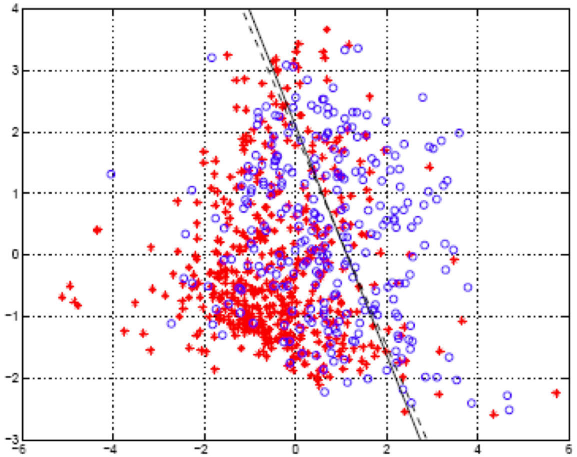
Aditya Grover
UCLA

The instructor gratefully acknowledges Sriram Sankararaman (UCLA) and Andrew Ng (Stanford) for some of the materials and organization used in these slides, and many others who made their course materials freely available online.

https://aditya-grover.github.io/ @adityagrover_

Classification Based on Probability

- Instead of just predicting the class, give the probability of the instance being that class i.e., learn $p(y|\boldsymbol{x})$
- Comparison to perceptron:
  - Perceptron doesn't produce a probability estimate
- Recall that:
  - For any event $E \in \mathcal{E}$, $0 \leq p(E) \leq 1$
  - Sum of probabilities $\sum_{E \in \mathcal{E}} p(E) = 1$
- For binary classification, we will assume $y = 1$ and $y = 0$ as the two events for an input $\boldsymbol{x}$



Logistic Regression

- Takes a probabilistic approach to learning functions (i.e., a classifier) i.e. $h_{\boldsymbol{\theta}}(\boldsymbol{x})$ outputs a probability $p_{\boldsymbol{\theta}}(y = 1 \mid \boldsymbol{x})$
  - Want $0 \leq h_{\boldsymbol{\theta}}(\boldsymbol{x}) \leq 1$ for all $\boldsymbol{x}$
- **Logistic regression model**:

$$h_{\boldsymbol{\theta}}(\boldsymbol{x}) = g(\boldsymbol{\theta}^T \boldsymbol{x})$$

  where g(z) is logistic function

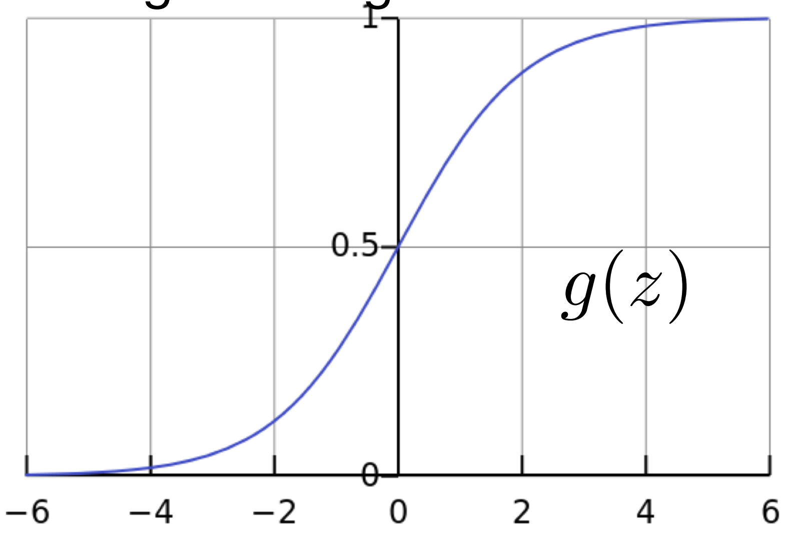
$$g(z) = \frac{1}{1+\mathrm{e}^{-z}}$$

- Hence,

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

  Not a regression model (despite the name!)

Logistic / Sigmoid Function



Interpretation of Hypothesis Output

$h_{\boldsymbol{\theta}}(\boldsymbol{x})$ should give $p_{\boldsymbol{\theta}}(y = 1 \mid \boldsymbol{x})$

Example: Cancer diagnosis from tumor size with y=1 as malignant

$$\boldsymbol{x} = \begin{bmatrix} x_0 \\ x_1 \end{bmatrix} = \begin{bmatrix} 1 \\ \text{tumorSize} \end{bmatrix}$$

$$h_{\boldsymbol{\theta}}(\boldsymbol{x}) = 0.7$$

Tell patient that 70% chance of tumor being malignant as per model

Note that: $p_{\boldsymbol{\theta}}(y = 1 \mid \boldsymbol{x}) + p_{\boldsymbol{\theta}}(y = 0 \mid \boldsymbol{x}) = 1$

Therefore, $p_{\boldsymbol{\theta}}(y = 0 \mid \boldsymbol{x}) = 1 - \ p_{\boldsymbol{\theta}}(y = 1 \mid \boldsymbol{x})$

Another Interpretation

Side Note: the odds in favor of an event is the quantity
p / (1 – p), where p is the probability of the event
E.g., If I toss a fair dice, what are the odds that I will have a 6?

- Equivalently, logistic regression assumes that

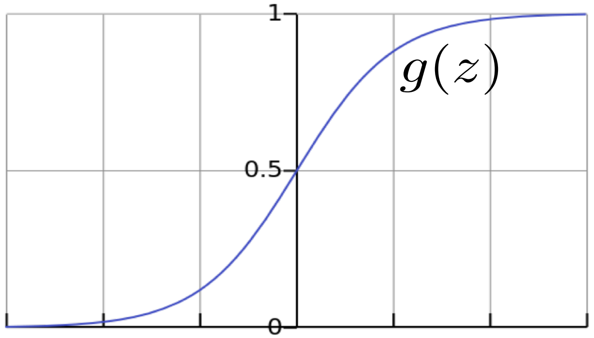
$$\log \underbrace{\boxed{\frac{p(y=1 \mid \boldsymbol{x}; \boldsymbol{\theta})}{p(y=0 \mid \boldsymbol{x}; \boldsymbol{\theta})}}}_{\text{odds of } y = 1} = \boldsymbol{\theta}^T \boldsymbol{x}$$

- In other words, logistic regression assumes that the log odds is a linear function of $\boldsymbol{x}$

Logistic Regression

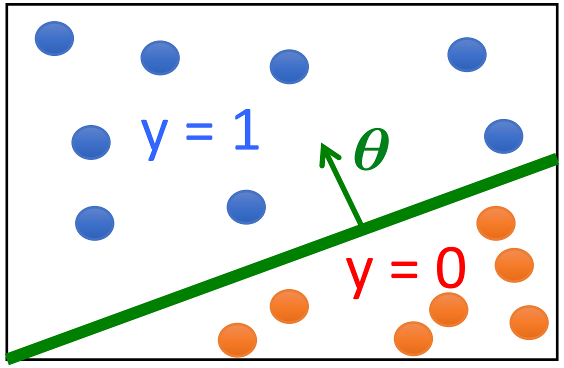
$$h_{\boldsymbol{\theta}}(\boldsymbol{x}) = g(\boldsymbol{\theta}^T \boldsymbol{x})$$

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



$\boldsymbol{\theta}^T \boldsymbol{x}$ should be large <u>negative</u> values for negative instances	$\boldsymbol{\theta}^T \boldsymbol{x}$ should be large <u>positive</u> values for positive instances

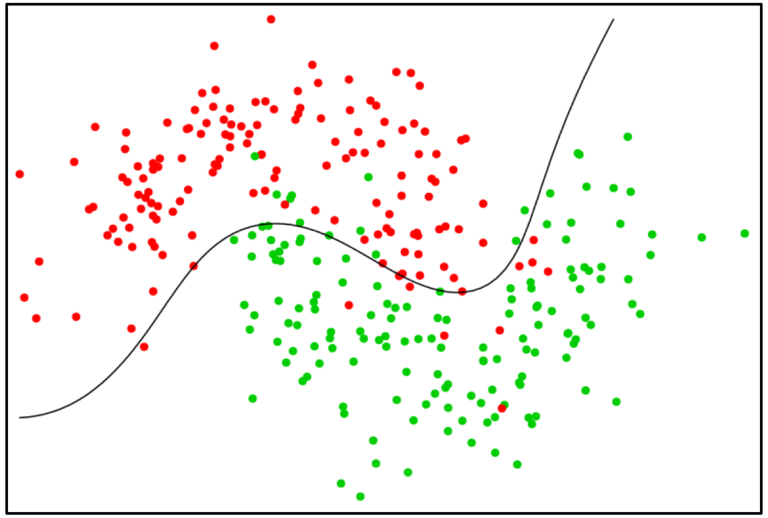
- To make hard predictions, assume a threshold t (e.g., 0.5)
  - Predict $y = 1$ if $h_{\boldsymbol{\theta}}(\boldsymbol{x}) \geq t$
  - Predict $y = 0$ if $h_{\boldsymbol{\theta}}(\boldsymbol{x}) < t$



Non-Linear Decision Boundary

- Can apply basis function expansion to features, same as with linear regression

$$\boldsymbol{x} = \begin{bmatrix} 1 \\ x_1 \\ x_2 \end{bmatrix} \rightarrow \begin{bmatrix} 1 \\ x_1 \\ x_2 \\ x_1 x_2 \\ x_1^2 \\ x_2^2 \\ x_1^2 x_2 \\ x_1 x_2^2 \\ \vdots \end{bmatrix}$$



Loss Function

- Loss of a single instance:

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

- Total loss over n training instances:

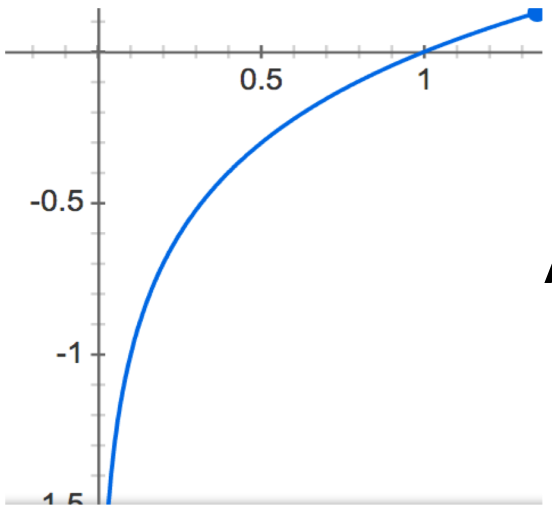
$$J(\boldsymbol{\theta}) = \sum_{i=1}^{n} \ell(y^{(i)}, \boldsymbol{x}^{(i)}, \boldsymbol{\theta})$$

Logistic regression loss:

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

Intuition Behind the Objective

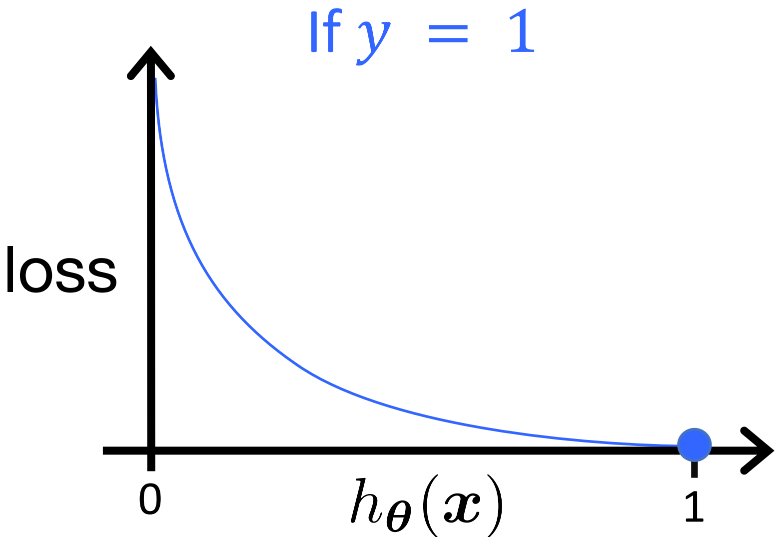
$$\ell(y^{(i)}, \boldsymbol{x}^{(i)}, \boldsymbol{\theta}) = \begin{cases} -\log h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) & \text{if } y^{(i)} = 1 \\ -\log\left(1 - h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right)\right) & \text{if } y^{(i)} = 0 \end{cases}$$



Aside: Recall the plot of log(z)

Intuition Behind the Objective

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

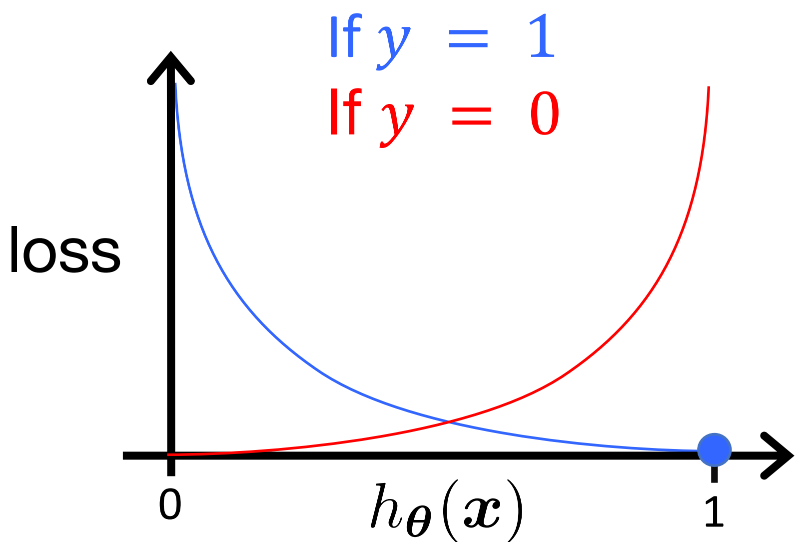


If $y^{(i)} = 1$

- As $h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) \rightarrow 1$, loss$\rightarrow 0$
- As $h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) \rightarrow 0$, loss$\rightarrow \infty$
- Captures intuition that larger mistakes should get larger penalties
  - e.g., predict $h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) = 0$, but $y^{(i)} = 1$

Intuition Behind the Objective

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



If $y^{(i)} = 0$

- As $h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) \rightarrow 0$, loss$\rightarrow 0$
- As $h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) \rightarrow 1$, loss$\rightarrow \infty$
- Captures intuition that larger mistakes should get larger penalties
  - e.g., predict $h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) = 1$, but $y^{(i)} = 0$

Regularized Logistic Regression

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

- We can regularize logistic regression exactly as before:

$$\begin{aligned} J_{\text{reg}}(\boldsymbol{\theta}) &= J(\boldsymbol{\theta}) + \frac{\lambda}{2}\textstyle\sum_{j=1}^{d}\theta_j^2 \\ &= J(\boldsymbol{\theta}) + \frac{\lambda}{2}\ \|\boldsymbol{\theta}_{1:d}\|_2^2 \end{aligned}$$

- $\lambda > 0$ is the regularization coefficient

Gradient Descent for Logistic Regression

$$J_{\text{reg}}(\boldsymbol{\theta}) = -\sum_{i=1}^{n}\left[y^{(i)} \log h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) + \left(1 - y^{(i)}\right) \log\left(1 - h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right)\right)\right] + \frac{\lambda}{2} \| \boldsymbol{\theta}_{1:d} \|_{\mathbf{2}}^{\mathbf{2}}$$

- Initialize $\boldsymbol{\theta}$ randomly
- Repeat until convergence

$$\theta_j \leftarrow \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\boldsymbol{\theta})$$

simultaneous update for j = 0 ... d

Gradient Descent for Logistic Regression

$$J_{\text{reg}}(\boldsymbol{\theta}) = -\sum_{i=1}^{n}\left[y^{(i)} \log h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) + \left(1 - y^{(i)}\right)\log\left(1 - h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right)\right)\right] + \frac{\lambda}{2}\|\,\boldsymbol{\theta}_{1:d}\|_{\mathbf{2}}^{\mathbf{2}}$$

- Initialize $\boldsymbol{\theta}$ randomly
- Repeat until convergence

[simultaneous update for j = 0 ... d]

$$\theta_0 \leftarrow \theta_0 - \alpha \sum_{i=1}^{n}\left(h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)$$

$$\theta_j \leftarrow \theta_j - \alpha \left[\sum_{i=1}^{n}\left(h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right) x_j^{(i)} + \lambda\theta_j\right]$$

Gradient Descent for Logistic Regression

- Initialize $\boldsymbol{\theta}$ randomly
- Repeat until convergence

[simultaneous update for j = 0 ... d]

$$\theta_0 \leftarrow \theta_0 - \alpha \sum_{i=1}^{n} \left(h_{\boldsymbol{\theta}} \left(\boldsymbol{x}^{(i)} \right) - y^{(i)} \right)$$

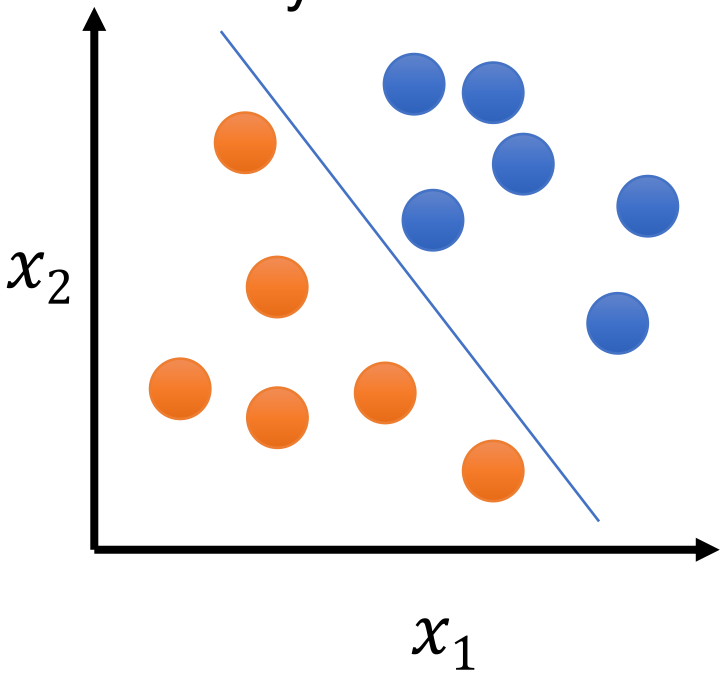
$$\theta_j \leftarrow \theta_j - \alpha \left[\sum_{i=1}^{n} \left(h_{\boldsymbol{\theta}} \left(\boldsymbol{x}^{(i)} \right) - y^{(i)} \right) x_j^{(i)} + \lambda \theta_j \right]$$

This looks IDENTICAL to linear regression!

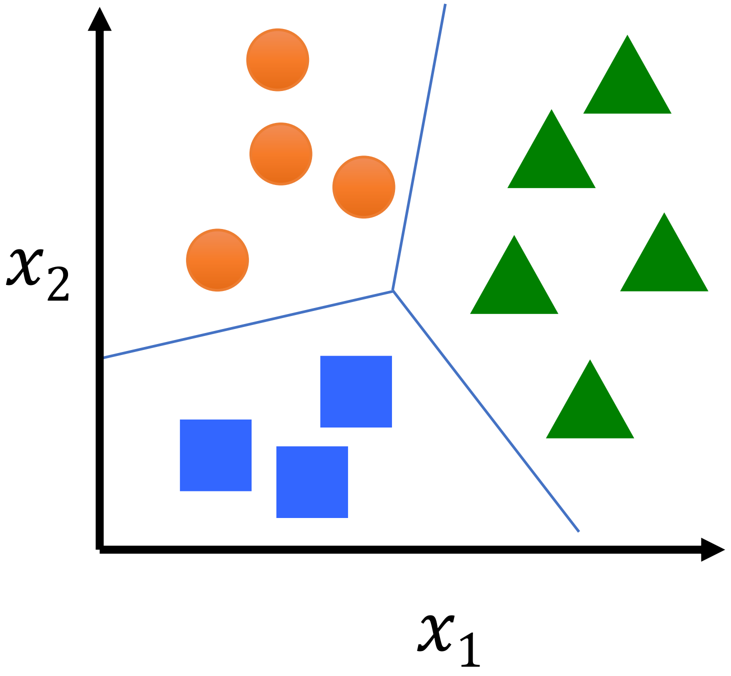
- Ignoring the 1/n constant
- However, the form of the hypothesis $h_{\boldsymbol{\theta}}(\boldsymbol{x})$ is very different

Multi-Class Classification

Binary classification



Multi-class classification



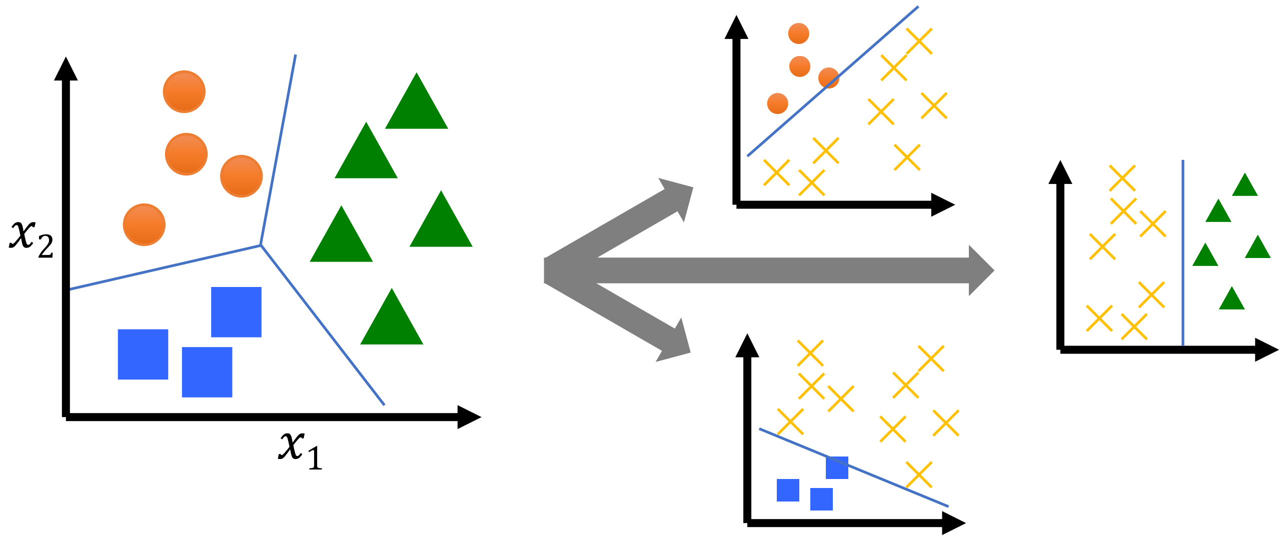
Disease diagnosis: healthy / cold / flu / pneumonia ..

Object classification: desk / chair / monitor / bookcase …

ChatGPT: next word prediction

Multi-Class Logistic Regression

Split into one v.s. rest:



- Expensive! Solving c separate classification problems

Multi-Class Logistic Regression

- For 2 classes:

$$h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \frac{1}{1+\exp(-\boldsymbol{\theta}^T \boldsymbol{x})} = \frac{\exp(\boldsymbol{\theta}^T \boldsymbol{x})}{\underbrace{\boxed{1}}_{\text{weight assigned to } y\,=\,0}+\underbrace{\boxed{\exp(\boldsymbol{\theta}^T \boldsymbol{x})}}_{\text{weight assigned to } y\,=\,1}}$$

- For C classes:

$$h_{\boldsymbol{\theta}_{1:C}}^{(c)}(\boldsymbol{x}) = p_{\boldsymbol{\theta}_{1:C}}(y = c \mid \boldsymbol{x}) = \frac{\exp(\boldsymbol{\theta}_c^T \boldsymbol{x})}{\sum_{k=1}^{C} \exp(\boldsymbol{\theta}_k^{\mathrm{T}} \boldsymbol{x})}$$

  - Here $\boldsymbol{\theta}_c \in \mathbb{R}^{d+1}$ is a parameter vector for class $c \in \{1, \ldots, C\}$
  - Hypothesis also called the **softmax** function
  - Note that sum of class probabilities equals 1

Multi-Class Logistic Regression

- The hypothesis for class c

$$h_{\boldsymbol{\theta}_{1:C}}^{(c)}(\boldsymbol{x}) = \frac{\exp(\boldsymbol{\theta}_c^T \boldsymbol{x})}{\sum_{k=1}^{C} \exp(\boldsymbol{\theta}_k^{\mathrm{T}} \boldsymbol{x})}$$

- Gradient descent simultaneously updates all parameters for c
  - Same derivative as before, just with the above $h_{\boldsymbol{\theta}_{1:C}}^{(c)}(\boldsymbol{x})$
- Predict class label as the most probable label

$$\hat{y} = \underset{c \in \{1,\ldots,C\}}{\arg\max}\, h_{\boldsymbol{\theta}_{1:C}}^{(c)}(\boldsymbol{x})$$

Summary

Logistic Regression

- A probabilistic linear model for classification (despite the name)

Loss function

- Binary/softmax cross-entropy loss

Basis Function, Optimization, Regularization

- Analogous to linear regression