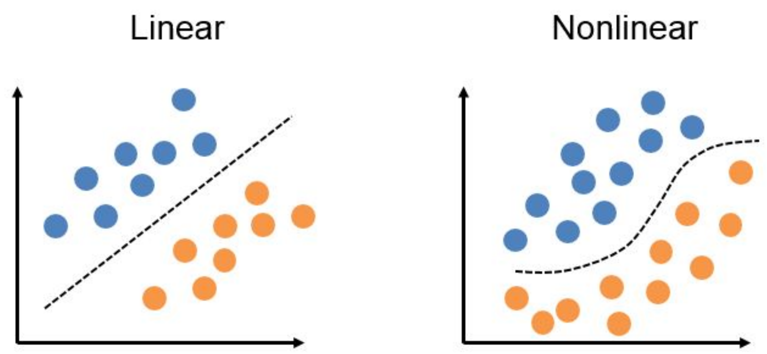


Generalised Linear Classifier Cheat Sheet (Discriminative Models)

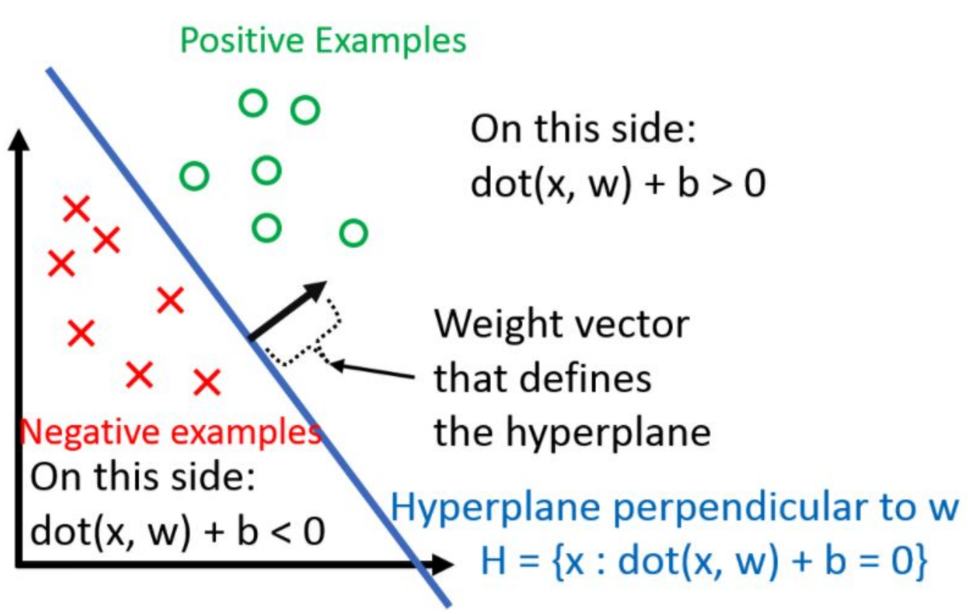
V2020.12.19

(Dr Yan Xu)

Generalised Linear Classifier

- Discriminative Models
  - Perceptron
  - SVM (*popular*)
  - Logistic regression (*popular*)
- Generative models
  - Naive Bayes classifier
  - Linear Discriminant Analysis (LDA)

Perceptron



➔ decision surface: Linear

➔ classifier:

$$h(x_i) = \text{sign}(\mathbf{w}^\top \mathbf{x}_i + b)$$

➔ unique solution? No

➔ estimating **w**: [Iterative adjustment method]

1: Perceptron is not used in practice, good to know conceptually.

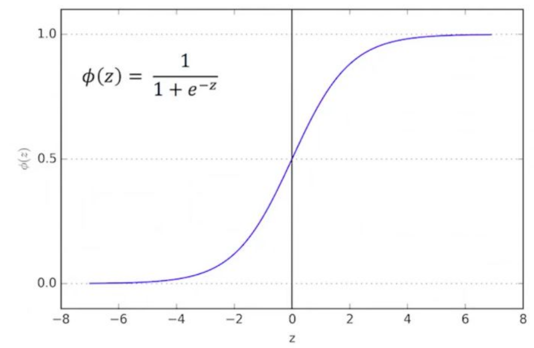
Logistic regression

➔ Decision Surface: Linear

➔ Classifier:

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

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



➔ Why Logit?

- It is difficult to use a linear model to predict a variable which has restricted range [0, 1]
- Logit(x) converts range from [0, 1] -> (-∞, +∞)

$$logit(p) = log(\frac{p}{1-p}) = \beta_0 + \beta_1 x_1 + \cdots + \beta_k x_k.$$

➔ Cross-entropy

- a measure of dissimilarity between p and q

$$H(p,q) = -\sum_{x \in \mathcal{X}} p(x)\, \log q(x)$$

➔ Cross-entropy loss (or log-loss)

$$J(\mathbf{w}) = \frac{1}{N}\sum_{n=1}^{N} H(p_n, q_n) = -\frac{1}{N}\sum_{n=1}^{N}\left[y_n \log \hat{y}_n + (1-y_n)\log(1-\hat{y}_n)\right]$$

➔ Relation to log-likelihood ([reference])

- MLE of ***p*** = minimizing the cross-entropy

➔ Estimating ***w***: Gradient methods

➔ Bayesian Logistic regression ([reference])

Information Theory Review

➔ Entropy:

given distribution ***p(x)***, the min lossless encoding size:

$$H = -\sum p(x) \log p(x)$$

➔ KL Divergence:

measures how well the probability distribution Q approximates the probability distribution P

$$D_{KL}(P||Q) = H(P,Q) - H(P)$$

The KL divergence is not symmetric

Support Vector Machine (SVM)

➔ Decision Surface: Linear

➔ Distance of a point x to the hyperplane:

$$\frac{|\mathbf{w}^T\mathbf{x}+b|}{\|\mathbf{w}\|_2}$$

➔ Formation I (with unequal constraints):

$$\min_{\mathbf{w},b} \mathbf{w}^T\mathbf{w}$$

$$\text{s.t.} \quad \forall i \; y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1$$

➔ Support Vectors:

training points with the following constraint as equal:

$$y_i(\mathbf{w}^T\mathbf{x}_i + b) = 1$$

➔ Formulation II (Unconstrained):

$$\min_{\mathbf{w},b} \underbrace{\mathbf{w}^T\mathbf{w}}_{l_2-regularizer} + C \underbrace{\sum_{i=1}^{n} \max\left[1 - y_i(\mathbf{w}^T\mathbf{x}+b), 0\right]}_{hinge-loss}$$

where, C is a scalar that needs to be tuned

➔ Solution: [Classic Optimisation Method]

Kernels

➔ A powerful idea that used a lot together with SVM.

➔ Motivation:

Project X into a much higher space before classification

➔ Kernel:

- generalized dot product: dot(x, y)
- computing the dot product of two vectors **x** and **y** in a high dimensional feature space, without explicitly defining those dimensionals (like magic)
- also be considered as a similarity function

➔ Kernel-SVM is kind of non-parametric, so it tends to be slow for large training data

➔ Common Kernels:

- Polynomial Kernel
- Radial Basis Function (or Gaussian Kernel) (popular)

➔ Regularization with Kernel SVMs

- Gaussian Kernel: ***C*** and gamma