Non-Parametric Classifiers

K-Nearest Neighbors

- For each test point, find the K nearest samples in the training data: a distance metric such as the 2-norm $\|x_{test} - x_i\|_2^2$ $i = 1, \ldots, N$ may be used.
- Classify the point according to the majority of their class labels

Classification Rule (Naive)

① $\vec{d} = \|x_{test} - x_i\| \quad i = 1, \ldots, N$

② Assume the target function (that assigns labels) is

$$g: \mathbb{R}^n \rightarrow \{1, \ldots, C\}$$

Find position of smallest elements of $\vec{d}$: $\{d_j\}, j \in \tau$ where $\tau \subseteq \{1, \ldots, N\}$ is the ordered set of the positions of the smallest K entries of $\vec{d}$.

③ Define set $S \triangleq \{(x_{\tau(1)}, g(x_{\tau(1)})), \ldots (x_{\tau(k)}, g(x_{\tau(k)}))\}$ where $\tau(i)$ is the i^{th} element of set τ. The class of x_{test} is

$$\hat{g}(x_{test}) = \text{mode}\{g(x_{\tau(1)}), \ldots g(x_{\tau(k)})\}$$

Rule of thumb : $k < \sqrt{N}$, N being the no. of training samples

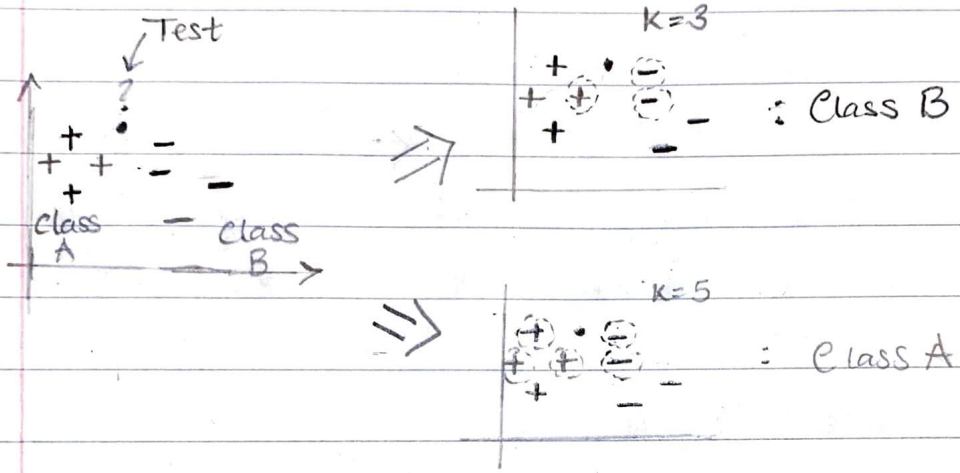


Figure above illustrates how the selection of k may affect classification

In general, the KNN scheme is intuitive and simple to implement. But it may become very expensive for large datasets. Notwithstanding it is easy to parallelize.

Parzen Window (Probability Density Estimation)

In this approach, we split our domain of interest (n-dimensional space) into hypercubes (bins or intervals). For each bin, we 'count' or measure how many samples (per total samples counted) fall into that bin; and divide by the volume of the hypercube.

This approach helps build a probability density estimate over the domain:

$$P_{est}(x) = \frac{K_N/N}{\Delta x}$$

K_N: number of training samples in the interval Δx

N: total number of samples counted

K_N/N: frequency

Δx: volume, size of interval

In particular, we can write this probability estimate using a windowing function

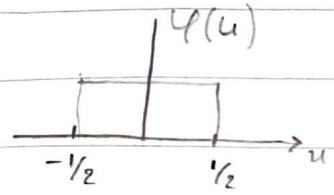
$$P_N(x) = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{V_N} \varphi\left(\frac{x - x_i}{h_N}\right)$$

h_N: width of bin

V_N: h_N^d volume of bin

Window function: measures if or how much x_i is in the bin with width h_N centered at x

$$\varphi(u) = \begin{cases} 1 & |u_i| < 1/2 \quad i = 1, \ldots, n \\ 0 & \text{otherwise} \end{cases}$$



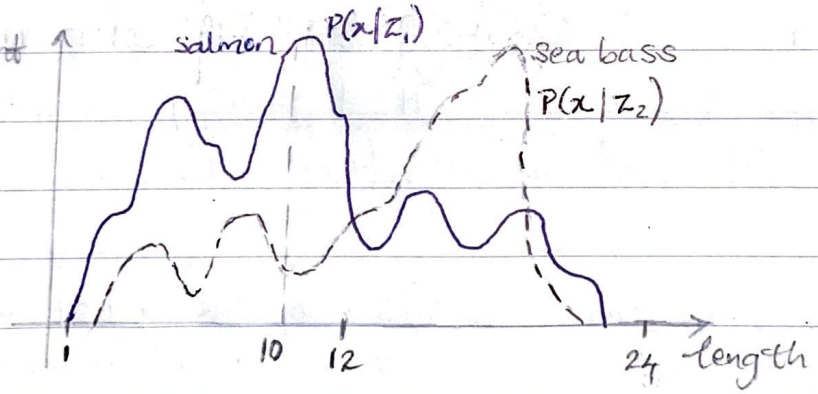
Note that we may also use a normal distribution window

$$\varphi(u) = \frac{1}{\sqrt{2\pi}} e^{-u^2/2}$$

and define $h_N = h_1/\sqrt{N}$ where h_1 becomes the parameter at our disposal. This definition is to ensure that

$$\lim_{n \to \infty} V_n = 0$$

the estimate converges to the true probability density function



Decision Rule: $\underset{i \in \{1,\ldots,c\}}{\text{argmax}} \; p(x|\omega_i)$

PARAMETRIC CLASSIFIERS

We would like to have parametric methods for classification that summarize the data with a fixed number of parameters that does not scale with the number of training data available
We focus on what we call discriminant functions.

BINARY CLASSIFICATION

Given training data $\{x_i, z_i\}$ for $i = 1, \ldots, N$, $x \in \mathbb{R}^n$, $z_i \in \{-1, 1\}$
we want a discriminant function

$$g(x_i) \begin{cases} \geq 0 & \text{if } z_i = 1 \\ < 0 & \text{if } z_i = -1 \end{cases}$$

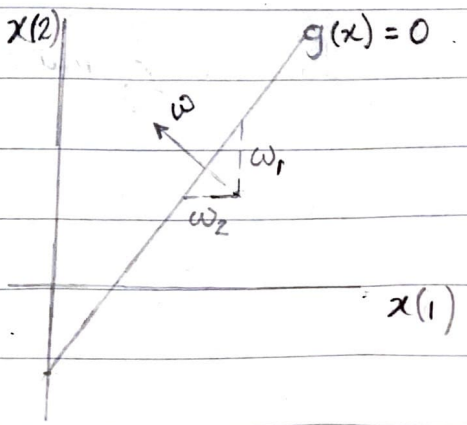
i.e $z_i g(x_i) > 0$ implies correct classification

LINEAR CLASSIFIER

If we assume that the classes are linearly seperable then we can write

$$g(\hat{x}) = \hat{\omega}^T \hat{x} + \omega_0$$

In 2D, the contour $g(x)$ is a line since $x(1) = \frac{-\omega_1}{\omega_2} x(2) + \frac{b}{\omega_2}$

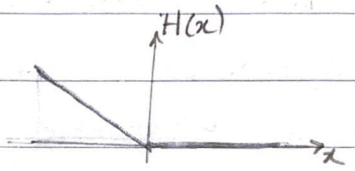


In n-dimensional space, it is a hyperplane.

We can write $\omega = \begin{bmatrix} \hat{\omega} \\ \omega_0 \end{bmatrix}$, $x = \begin{bmatrix} \hat{x} \\ 1 \end{bmatrix}$, so that $g(x) = \omega^T x$

Recall that we want $z_i g(x_i) > 0$, so we are only interested in cases where

$$\underbrace{\max(0, -z_i g(x_i))}_{\text{Perceptron loss function: generalizes misclassification in one term}} > 0$$



for when $z_i g(x_i) > 0$: $\max(0, -z_i g(x_i)) = 0$

only when $z_i g(x_i) < 0$: $\max(0, -z_i g(x_i)) > 0$
i.e misclassification

PERCEPTRON ALGORITHM

We want to minimize misclassifications. This is equivalent to minimizing

$$J(\omega) = \sum_{i=1}^{N} \max(0, -z_i \omega^T x_i)$$

which we can re-write as

$$J(\omega) = -\sum_{i \in \mathcal{M}} z_i \omega^T x_i$$

where $\mathcal{M} \subseteq \{1, \ldots, N\}$ is the set of misclassified samples

To find ω that minimizes the cost function J, i.e

$$\underset{\omega}{\operatorname{argmin}}\, J(\omega)$$

we use gradient descent

Algorithm

for $k = 1, 2, \ldots$

Update $M_k = \{i \mid z_i \omega^T x_i < 0\}$ for $\omega = \omega_{(k)}$

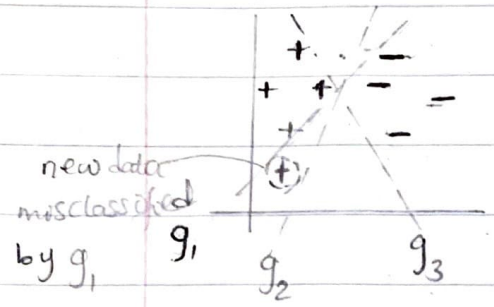
Gradient Descent

$$\omega_{(k+1)} \leftarrow \omega_{(k)} + \alpha_k \sum_{i \in M_k} z_i x_i$$

End for

This algorithm typically converges for linearly seperable data.
Note that, in fact, J is convex.

Support Vector Machine



But what is the best hyperplane $g(x,\omega)$ that separates the data? If the data is linearly seperable, there will likely be several (that are optimal based on the perceptron algorithm), but it may quickly misclassify on unseen (test) data.

So we want an hyperplane (characterized with the orthogonal vector ω) that will maintain a margin between the closest element of a class and the dividing hyperplane.

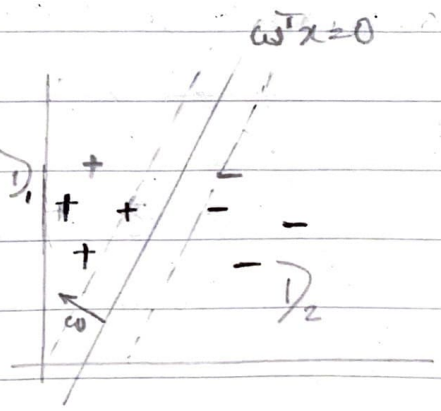
In fact, $g(x,\omega)$ should be such that it is equidistant from the closest vector (in terms of projection onto ω) in class $\mathcal{D}_1 \triangleq \{x \mid z_i = 1\}$ and the closest vector in class $\mathcal{D}_2 \triangleq \{x \mid z_i = -1\}$

So we can think of 3 parallel hyperplanes

$\hat{\omega}^T\hat{x} + \omega_0 = d$ Supporting $\mathcal{D}_1$

$\hat{\omega}^T\hat{x} + \omega_0 = 0$ dividing hyperplane

$\hat{\omega}^T\hat{x} + \omega_0 = -d$ Supporting $\mathcal{D}_2$



We can normalize by dividing by d to get

$\omega^T x = 1$ (for $\mathcal{D}_1$) $\quad \omega^T x = -1$ (for $\mathcal{D}_2$)

where $w = \frac{1}{d}\begin{bmatrix}\tilde{w}\\ w_0\end{bmatrix}$, $x = \begin{bmatrix}x\\ 1\end{bmatrix}$

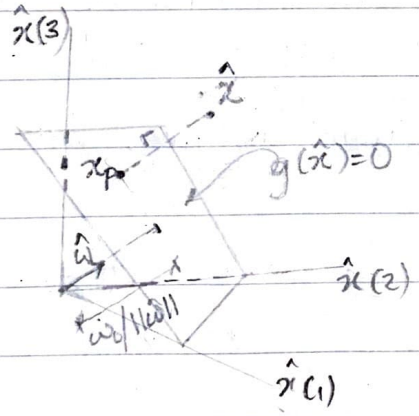
So,

$w^T x_i \geq 1$ if $z_i = 1$

$w^T x_i \leq 1$ if $z_i = -1$

which can be written as

$z_i w^T x_i \geq 1 \quad \forall i$



Now for any $\hat{x} \in \mathbb{R}^n$, we can write

$$\hat{x} = x_p + r\frac{\hat{w}}{\|\hat{w}\|}$$

since $g(x_p) = 0$

$$g(\hat{x}) = \hat{w}^T\hat{x} + \hat{w}_0$$

$$= w^T x_p + r\|w\| + w_0$$

$$= r\|\hat{w}\|$$

$$r = \frac{g(\hat{x})}{\|\hat{w}\|}$$

The distance of the hyperplane from $\hat{x}=0 = \frac{w_0}{\|\hat{w}\|}$

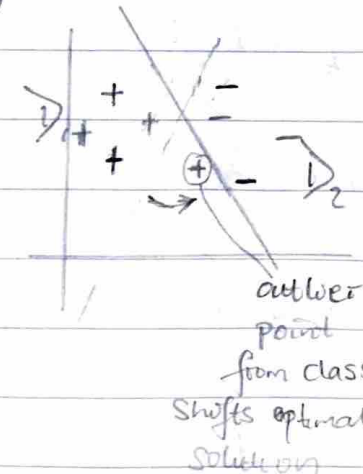
It follows that the distance from any vector x_i on the plane $\vec{w}^T x_i = 1$ from any vector x_j on the plane $\vec{w}^T x_j = -1$ is $\frac{2}{\|\hat{w}\|}$

We want to maximize this distance while ensuring that the classification criterion ($z_i \vec{w}^T x_i \geq 1 \; \forall i$) holds:

$$\min \|\hat{w}\|_2^2$$

$$\text{s.t.} \quad z_i \vec{w}^T x_i \geq 1 \quad \forall i \in \{1, N\}$$

This is a quadratic problem with linear constraints, hence there is a unique minimum.
But the constraint ($z_i \vec{w}^T x_i \geq 1$) may limit us from getting 'good' margins



To make the constraint softer, we introduce slack variables $\xi_i \geq 0$

if $0 < \xi_i \leq 1$ the point is between margin and the correct side of hyperplane. This is called margin violation
if $\xi > 1$, point is misclassified.

The optimization problem now becomes

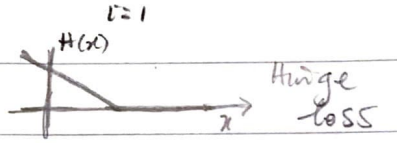
$$\min_{\omega, \xi} \|\hat{\omega}\|_2^2 + C \sum_{i=1}^{N} \xi_i$$

$$z_i \omega^T x_i \geq 1 - \xi_i \quad \forall i$$

Equivalent expression

$$\xi_i = \underbrace{\max(0, 1 - z_i \omega^T x_i)}_{\text{Hinge loss function}}$$

$$\min_{\omega} \|\hat{\omega}\|_2^2 + C \sum_{i=1}^{N} \max(0, 1 - z_i \omega^T x_i)$$



Large C makes constraint hard to ignore

Small C allows constraint to be easily ignored → large margin

In SVM's, it is common to first transform or map x to a higher dimension

$$y_i = \varphi(x_i)$$

generally nonlinear mapping

in expectation that in the higher dimensional space, a separating hyperplane will exist

Multiclass

In multiclass cases,

$$g_i(x) = \omega_i^T x$$

We assign x to z_i if $g_i(x) > g_j(x) \; \forall \; j \neq i$

We can write the multi-class SVD problem as (without slack variables)

$$\max_{\omega_1, \ldots, \omega_C} \sum_k \|\hat{\omega}_k\|_2^2 + C \sum_{i=1}^{N} \xi_i$$

$$\text{s.t.} \quad \omega_{z_i}^T x_i - \omega_k^T x_i \geq 1 - \xi_i$$

$$\forall \; k \in (1, 2, \ldots, C), \; k \neq z_i$$

$$\forall \; i$$