IAML 2016-2017

1. a. Support Vector Machines

i. Define the maximum margin hyperplane

non-kernelised - linear
kernelised - linear / non-linear

Margin → distance between the decision boundary and the closest training point.

Maximum margin hyperplane
→ the decision boundary that has the max. distance towards the closest training point.

i.e. the hyperplane $w^T x + w_0$ such that

$$\min_i |w^T x_i + w_0| = 1$$

ii. Solution for the $\underline{w}$ for maximum margin hyperplane, explain support vectors wrt this eqn.
Describe how a class prediction is made, using SVM, for a new test input vector $\underline{x}_0$.

The SVM weights are determined by solving the OP

$$\min_w \|w\|^2 \quad \text{where } \|w\| = \sqrt{w^T w}$$

max margin optimisation problem:

$$\text{s.t. } y_i(w^T x_i + w_0) \geq +1 \quad \text{for all } i.$$

$$\underline{w} = \sum_i \alpha_i y_i x_i$$

Support vectors are the vectors where α_i is not zero.

Prediction on new datapoint $\underline{x}$

$$f(\underline{x}) = \text{sign}((\underline{w}^T \underline{x}) + w_0)$$

$$= \text{sign}\left(\sum_{i=1}^{n} \alpha_i y_i (x_i^T x) + w_0\right)$$

(α_i: the lagrange multiplier)

$\text{sgn}(x)$: $< 0 \rightarrow -1$, $\geq 0 \rightarrow 1$

If $\text{sgn}(f(x))$: $\geq 0, 1$; $< 0, -1$

iii. Non-linear SVMs map $\underline{x}$ (← input space) to feature space $\phi(\underline{x})$. The kernel trick can be used to compute $\phi(x) \cdot \phi(y)$ efficiently.

$$\phi(x) = \begin{bmatrix} x_1^2 \\ \sqrt{2}\, x_1 x_2 \\ x_2^2 \end{bmatrix}^T \qquad x = (x_1, x_2)^T$$

Equation for $K(x, y)$.

$$\phi(x)^T \cdot \phi(y) = \begin{bmatrix} x_1^2 & \sqrt{2}\, x_1 x_2 & x_2^2 \end{bmatrix} \begin{bmatrix} y_1^2 \\ \sqrt{2}\, y_1 y_2 \\ y_2^2 \end{bmatrix}$$

$$= x_1^2 y_1^2 + 2 x_1 x_2 y_1 y_2 + x_2^2 y_2^2$$

$$= (x \cdot y)^2$$

$$\star \left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \cdot \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} \right)^2 = (x_1 y_1 + x_2 y_2)^2$$

$$= x_1^2 y_1^2 + 2 x_1 y_1 x_2 y_2 + x_2^2 y_2^2$$

so $K(x, y) = (x \cdot y)^2$

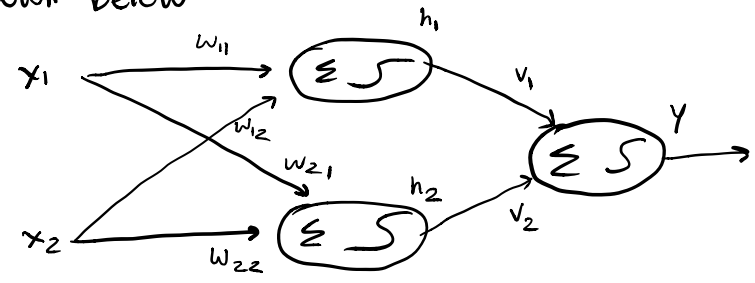
b. Artificial Neural Networks

Training data

x_1	x_2	y
0.0	0.0	0
4.0	5.0	0
0.0	5.0	1
4.0	0.0	1

Here x_1 and x_2 are real-valued features, and the class label y can take values of 0 and 1.

Consider the feed-forward neural network with one hidden layer shown below:



In this network

$$y \leftarrow \sigma(v_1 h_1 + v_2 h_2 + v_0)$$
$$h_1 \leftarrow \sigma(w_{11} x_1 + w_{12} x_2 + w_{10})$$
$$h_2 \leftarrow \sigma(w_{21} x_1 + w_{22} x_2 + w_{20})$$

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

we will also use this weight setting in the network

$w_{11} = 1.0$	$w_{12} = -0.8$	$w_{10} = 2.0$
$w_{21} = -1.0$	$w_{22} = 0.8$	$w_{20} = 1.5$
$v_1 = -1.0$	$v_2 = 1.5$	$v_0 = -0.5$

i. For every instance in the training set, compute the network's values of h_1, h_2, and y.

$x_1 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \end{bmatrix}$ $x_2 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}$

$$h_1 = \sigma(w_{11} x_1 + w_{12} x_2 + w_{10})$$
$$= \sigma(1 \times 0 + (-0.8) \times 0 + 2.0)$$
$$= \sigma(2.0) = 0.88$$

$$h_2 = \sigma(w_{21} x_1 + w_{22} x_2 + w_{20})$$
$$= \sigma(-1.0 \times 0 + 0.8 \times 0 + 1.5)$$
$$= \sigma(1.5) = 0.82$$

$y \begin{cases} \geq 0.5 \rightarrow 1 \\ < 0.5 \rightarrow 0 \end{cases}$

$$y = \sigma(v_1 h_1 + v_2 h_2 + v_0)$$
$$= \sigma(1 \times 0.88 + 1.5 \times 0.82 - 0.5)$$
$$= \sigma(1.61) = 0.83 \longrightarrow 1$$

... do for all

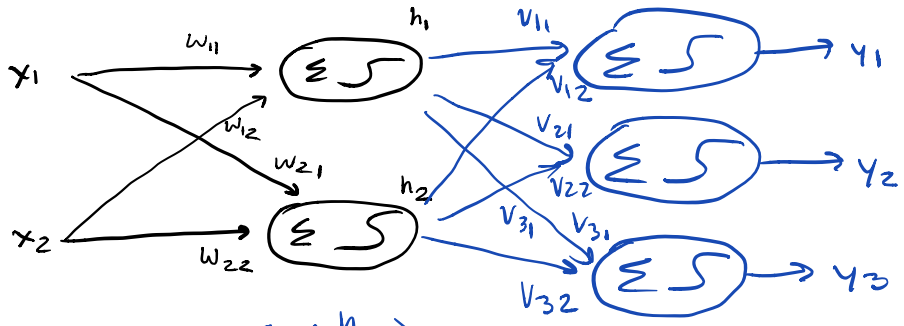
ii. Accuracy

$$\text{Accuracy} = \frac{\text{total correct}}{\text{total}}$$

iii. 3 possible labels

x_1	x_2	y
1.7	2.1	A
0.3	6.4	B
0.5	4.6	A
4.3	0.8	C

- change the output node to 1 for each class



$$y_m \leftarrow v_m^T \begin{pmatrix} h_1 \\ h_2 \end{pmatrix} + v_{m0}$$

$$= v_{m1} h_1 + v_{m2} h_2 + v_{m0}$$

$$y_1 \leftarrow v_{11} h_1 + v_{12} h_2 + v_{10}$$

$$y_2 \leftarrow v_{21} h_1 + v_{22} h_2 + v_{20}$$

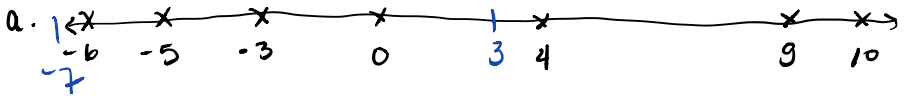
$$y_3 \leftarrow v_{31} h_1 + v_{32} h_2 + v_{30}$$

Prediction $f(x)$ is the class w/ the highest probability

$$p(y = m \mid x) = \frac{e^{y_k}}{\sum_{k=1}^{M} e^{y_k}}$$

$$f(x) = \max_{m=1}^{M} p(y = m \mid x)$$

2. K-Means and Decision Trees

a. 

i. K-means algorithm

- Initialise K random centroids
- Assign each point to the cluster w/ closest centroids
- Compute mean of cluster → centroid'
- Keep doing this until means are stationary

ii. $K=2$. $\mu_1=-7$ $\mu_2=3$

Do.

iii. Pick K

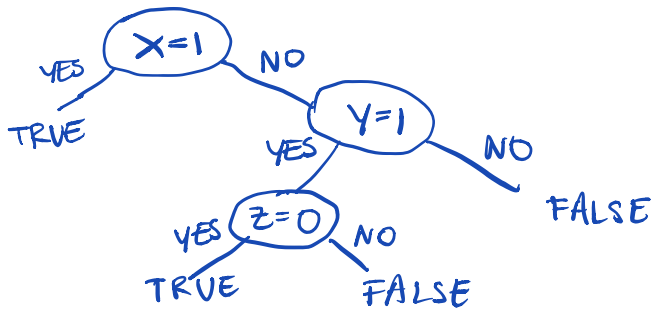
Optimise V → aggregate intra-cluster distance

- visually from scree plot
  - point where mountain ends and rubble begins
  - elbow method: max. 2nd derivative of V.
    - point where rate of decline changes most

b. i. ID3 algorithm $Gain(S,A) = H(S) - \sum_{v \in Values A} \frac{|S_v|}{|S|} H(S_v)$

split(node, {examples}):
1. A ← the best attribute for splitting {ex's}
   ↳ most I.G.
2. Decision attribute for this node ← A
3. For each att/value of A, create new child node
4. Split trn. examples to child nodes
5. For each child node:
   if subset is pure: STOP
   else: split (child node, {subset's}).

iii. DT to split $X=1$ OR $(Y=1$ AND $Z=0)$



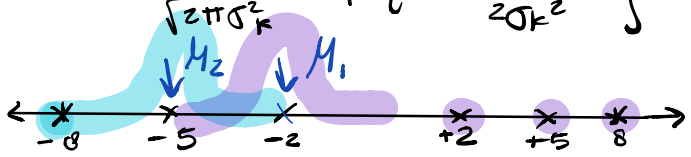
ii. DT vs. Log. Regr

- DT allows to split the data into quadrants.
- DT allows lines that are parallel to the axes.
- Log. Regr can handle non-linear db w/ kernelisation.
- Log. Regr single / linear d.b.
- DT interpretable

3. a. Gaussian Mixtures and EM.

$$P(x) = \sum_{k=1}^{K} P(x|k)\,P_k$$

$$P(x|k) = \frac{1}{\sqrt{2\pi\sigma_k^2}} \exp\left\{ - \frac{(x-\mu_k)^2}{2\sigma_k^2} \right\}$$



i. Describe how EM can be used to fit Gaussian MM. eqn for $P(k|x)$, provde update eqn for μ_k, σ_k, P_k.

start with K Gaussians and initialise μ_k, σ_k^2, for each K. and start with random priors $P(k)$.

* Usually $P(k) = \frac{1}{K}$

Use μ_k, σ_k^2 to find the likelihood and use $P(x|k)$ and $P(k)$ to find $P(k|x)$

Find probability that x belongs to class k.

Weight that associates instance i w/ gaussian c.

$$w_{i,c} = \frac{P(c|x_i)}{\sum_i P(c|x_i)}$$ → posterior / → sum of posteriors

→ tells you how important that posterior is

Mean of att. a in items assigned to c

$$\mu_{ca} = \underbrace{w_{c1}}_{\text{value skewed by weight}} x_{1a} + \cdots + w_{cn} x_{na}$$

$$P(C) = \frac{1}{n}(P(C|x_1) + \dots + P(C|x_n))$$

→ update the prior

ii. Assume you fit 2-component. $\mu_1 = -2$, $\mu_2 = -5$, $\sigma_1 = \sigma_2 = 1$

Assume a uniform prior.

Compute posterior probabilities of the 2 Gaussians for $x = -5$.

$$P(x=-5|k=1) = \frac{1}{\sqrt{2\pi \cdot 1}} \exp\left(-\frac{(-5+2)^2}{2 \cdot 1}\right)$$

$$= \frac{1}{\sqrt{2\pi}} \exp(-9/2) = 0.0044$$

$$P(x=-5|k=2) = \frac{1}{\sqrt{(2\pi)\cdot 1}} \exp\left(-\frac{(-5+5)^2}{2 \cdot 1}\right)$$

$$= \frac{1}{\sqrt{2\pi}} = 0.3990$$

$P(k=2|\mu_1) = 0.01$

$$P(k=1|x=-5) = \frac{0.0044 \times 0.5}{0.0044 \times 0.5 + 0.3990 \times 0.5} = 0.01$$

$P(k=1|\mu_1) = 0.99$

$$P(k=2|x=-5) = 1 - 0.01 = 0.99$$

$\mu_2 = -5$

iii. Estimate pp of the first Gaussian for every point in the dataset.

$\mu_1 = -2$

$P(k=1\|x=-8) = 0.00$		$P(k=2\|x=-8) = 1.00$
$P(k=1\|x=-5) = 0.01$		$P(k=2\|x=-5) = 0.99$
$P(k=1\|x=-2) = 0.99$		$P(k=2\|x=-2) = 0.01$
$P(k=1\|x=2) = 1.00$		$P(k=2\|x=2) = 0.00$
$P(k=1\|x=5) = 1.00$		$P(k=2\|x=5) = 0.00$
$P(k=1\|x=8) = 1.00$		$P(k=2\|x=8) = 0.00$

iv. Compute new means

$$\mu_1 = \frac{-8-5}{2} = -6.5 \qquad \mu_2 = \frac{-2+2+5+8}{4} = 3.25$$

$$\sigma_1 = \frac{\sqrt{(-8+6.5)^2 + (-5+6.5)^2}}{2}$$

$$\sigma_2 = \frac{\sqrt{(-2-3.25)^2 + (2-3.25)^2 + (5-3.25)^2 + (8-3.25^2)}}{4}$$

$P(K=1) = 2/6$

$P(K=2) = 4/6$