1. (a) The images were of '2'.
The components appear to be capturing vertical (more PC 1) and horizontal (more PC 2) translation.

(b) PC 1: $\frac{5.43 \times 10^6}{3.04 \times 10^7} \simeq 18\%$

PC 2: $\frac{4.66 \times 10^6}{3.04 \times 10^7} \simeq 15\%$

(c) Two degrees of freedom: horizontal and vertical translation. First 2 components capture only 33% of variance, because they are linear functions of pixel values, whereas translations are non-linear in pixel positions. So non-linear dimensionality reduction methods may give better results.

2. (a) $p(x) = \sum_{k=1}^{K} \pi_k \mu_k^x (1-\mu_k)^{1-x}$

A mixture of K Bernoulli distributions.

(b) $E(x) = \sum_x x\, p(x) = 0 \cdot p(x=0) + 1 \cdot p(x=1)$

$= p(x=1) = \sum_{k=1}^{K} \pi_k \mu_k$

$$\text{var}(x) = E(x^2) - E(x)^2$$

$$= \sum_x x^2 \cdot p(x) - E(x)^2$$

$$= 0^2 \cdot p(0) + 1^2 \cdot p(1) - E(x)^2$$

$$= p(1) - E(x)^2 = E(x) - E(x)^2$$

$$= E(x)\,(1 - E(x))$$

$$= \sum_k \pi_k \mu_k \left(1 - \sum_k \pi_k \mu_k\right)$$

2.(c) Latent varible $\underline{z} \in \mathbb{R}^K$ denotes the choice of coin, $z_{k'} = 1$, all other $z_k = 0 (k \neq k')$, where k' denotes the chosen coin.

$$\Rightarrow p(z_k = 1) = \pi_k \;,\; p(\underline{z}) = \prod_{k=1}^{K} \pi_k^{z_k}$$

$$p(x \mid z_k = 1) = \mu_k^x (1-\mu_k)^{1-x}$$

$$p(x \mid \underline{z}) = \prod_{k=1}^{K} \left[\mu_k^x (1-\mu_k)^{1-x}\right]^{z_k}$$

$$\Rightarrow p(x, \underline{z}) = \prod_{k=1}^{K} \left[\pi_k \mu_k^x (1-\mu_k)^{1-x}\right]^{z_k}$$

2.(d) complete-data log likelihood:

$$p(X, Z) = \prod_{n=1}^{N} \prod_{k=1}^{K} \left[\pi_k \mu_k^{x_n} (1-\mu_k)^{1-x_n}\right]^{z_{nk}}$$

$$\ln p(X, Z) = \sum_{n=1}^{N} \sum_{k=1}^{K} z_{nk} \left\{\ln \pi_k + x_n \ln \mu_k + (1-x_n) \ln (1-\mu_k)\right\}$$

2.(e) E step

(posteriors on $z_{nk}=1$)

$\triangleq \gamma(z_{nk})$

$$\gamma(z_{nk}) = p(z_{nk}=1 \mid x_n) = \frac{p(x_n \mid z_{nk}=1)\cdot p(z_{nk}=1)}{p(x_n)}$$

$$\Rightarrow \gamma(z_{nk}) = \frac{\mu_k^{x_n}(1-\mu_k)^{1-x_n}\cdot \pi_k}{\sum_{k'=1}^{K} \mu_{k'}^{x_n}(1-\mu_{k'})^{1-x_n}\pi_{k'}}$$

M step

Expected complete-data log likelihood

$$= \sum_z p(z) \ln p(X,Z)$$

$$= \sum_{n=1}^{N}\sum_{k=1}^{K} \gamma(z_{nk}) \left\{ \ln \pi_k + x_n \ln \mu_k + (1-x_n)\ln(1-\mu_k) \right\}$$

Maximise w.r.t. μ_k:

$$\underbrace{\sum_{n=1}^{N} \gamma(z_{nk})}_{\triangleq N_k} \left\{ \frac{x_n}{\mu_k} - \frac{1-x_n}{1-\mu_k} \right\} = 0$$

$$\Rightarrow \frac{1}{\mu_k}\sum_{n=1}^{N}\gamma(z_{nk})x_n = \frac{1}{1-\mu_k}\sum_{n=1}^{N}\gamma(z_{nk})(1-x_n)$$

$$(1-\mu_k)\sum_{n=1}^{N}\gamma(z_{nk})x_n = \mu_k\left(N_k - \sum_{n=1}^{N}\gamma(z_{nk})x_n\right)$$

$$\sum_{n=1}^{N} \gamma(z_{nk}) x_n = \mu_k \left(N_k - \sum_{n=1}^{N} \gamma(z_{nk}) x_n + \sum_{n=1}^{N} \gamma(z_{nk}) x_n \right)$$

$$\Rightarrow \mu_k = \frac{\sum_{n=1}^{N} \gamma(z_{nk}) x_n}{N_k}$$

'Average' of data points weighted by responsibility of component k

Maximise w.r.t. π_k:

[Use Lagrange multiplier λ for $\sum_k \pi_k = 1$]

$$\sum_{n=1}^{N} \gamma(z_{nk}) \frac{1}{\pi_k} + \lambda = 0$$

$$\Rightarrow \pi_k = -\frac{N_k}{\lambda}$$

To get λ:

$$\sum_{n=1}^{N} \gamma(z_{nk}) + \lambda \pi_k = 0$$

$$\sum_{n=1}^{N} \underbrace{\sum_{k=1}^{K} \gamma(z_{nk})}_{=1 \text{ (by defn.)}} = -\sum_{k=1}^{K} \lambda \pi_k$$

$$\Rightarrow N = -\lambda \underbrace{\sum_{k=1}^{K} \pi_k}_{=1}$$

$$\Rightarrow \lambda = -N \quad \Rightarrow \pi_k = \frac{N_k}{N}$$

Fraction of data points 'belonging' to k^{th} component

3. (a) Min. L w.r.t. $\underline{w}$

$$\nabla_{\underline{w}} L = \underline{w} - \sum_{i=1}^{N} \mu_i y_i \underline{x}_i = 0$$

$$\Rightarrow \underline{w} = \sum_{i=1}^{N} \mu_i y_i \underline{x}_i \quad -(1)$$

Min. L w.r.t. b

$$\frac{\partial L}{\partial b} = -\sum_{i=1}^{N} \mu_i y_i = 0 \quad -(2)$$

Plugging $\underline{w}$ from (1) into L:

$$L = \frac{1}{2}\left(\sum_{i=1}^{N} \mu_i y_i \underline{x}_i^T\right)\left(\sum_{j=1}^{N} \mu_j y_j \underline{x}_j\right)$$

$$+ \sum_{i=1}^{N} \mu_i \left[1 - y_i\left(\sum_{j=1}^{N} \mu_j y_j \underline{x}_j^T \underline{x}_i + b\right)\right]$$

$$= \frac{1}{2} \sum_{i,j=1}^{N} \mu_i \mu_j y_i y_j \underline{x}_i^T \underline{x}_j + \sum_{i=1}^{N} \mu_i$$

$$- \sum_{i,j=1}^{N} \mu_i \mu_j y_i y_j \underline{x}_j^T \underline{x}_i - \sum_{i=1}^{N} \mu_i y_i b$$

($= 0$ from (2))

$$\Rightarrow \tilde{L}(\mu) = \sum_{i=1}^{N} \mu_i - \frac{1}{2} \sum_{i,j=1}^{N} \mu_i \mu_j y_i y_j \underline{x}_i^T \underline{x}_j$$

[Dual]

(b) Training inputs appear only as dot products, which can be replaced by general kernel:

$$\tilde{L}(\mu) = \sum_{i=1}^{N} \mu_i - \frac{1}{2} \sum_{i,j=1}^{N} \mu_i \mu_j y_i y_j K(\underline{x}_i, \underline{x}_j)$$

3. (c) $K(\underline{x}_1, \underline{x}_2) = (f(\underline{x}_1) + f(\underline{x}_2))^2$

$= f(\underline{x}_1)^2 + f(\underline{x}_2)^2 + 2 f(\underline{x}_1) f(\underline{x}_2)$

Suppose I define $f(\cdot)$ such that

$f(\underline{x}_1) = 0$ and $f(\underline{x}_2) = 1$,

where $\{\underline{x}_1, \underline{x}_2\}$ are 2 data points given to me

∴ The kernel matrix for these two points is $K = \begin{bmatrix} 0 & 1 \\ 1 & 4 \end{bmatrix}$

This matrix is <u>not</u> p.s.d., as can be seen if we compute:

$$(-3 \;\; 1)\begin{pmatrix} 0 & 1 \\ 1 & 4 \end{pmatrix}\begin{pmatrix} -3 \\ 1 \end{pmatrix} = (1 \;\; 1)\begin{pmatrix} -3 \\ 1 \end{pmatrix} = \underline{-2}$$

Therefore it is not a valid kernel

4. $p(t|\underline{x}, \underline{w}) = y^t (1-y)^{1-t}$

$E(\underline{w}) = -\ln p = -\{t \ln y + (1-t)\ln(1-y)\}$

$= -\{t \ln \sigma(a) + (1-t)\ln(1-\sigma(a))\}$

$\Rightarrow \delta = \frac{\partial E}{\partial a} = -\left\{\frac{t}{\sigma(a)}\sigma(a)(1-\sigma(a)) + \frac{1-t}{1-\sigma(a)}(-\sigma(a))(1-\sigma(a))\right\}$

If $t=0$: $\delta = y = p(t=1|\underline{x}) = (1-t)\sigma(a) - t(1-\sigma(a))$

If $t=1$: $\delta = -(1-y) = p(t=0|\underline{x}) = \underline{(1-t)y - t(1-y)} = \underline{y - t}$

So the interpretation of δ is that it is the classification error, or the probability of assigning the wrong class, with the sign denoting the direction of the error in y.

δ is useful in computing the gradient of E with respect to the network weights:

$$\frac{\partial E}{\partial w_j} = \frac{\partial E}{\partial a} \cdot \frac{\partial a}{\partial w_j} = \delta z_j$$

(δ: Error at output of link w_j, z_j: times input to the link)

This gradient can then be used to update the weight value via gradient descent:

$$w_j^{(\tau+1)} = w_j^{(\tau)} - \eta \frac{\partial E}{\partial w_j^{(\tau)}}$$