Feb 23, 2022
2020094
OJUS SINGHAL
SML - Assignment 2

A1) a) We generate the samples using scipy.

b) Let's derive MLE for multivariate bernoulli distribution.

no. of samples $\longrightarrow n$
no. of dimensions $\longrightarrow d$

$x_{kj} \longrightarrow j^{th}$ dimension of k^{th} sample

$1 \leq j \leq d \; , \; 1 \leq k \leq n$

$\theta_j \longrightarrow \theta_{MLE}$ of j^{th} dimension

$D \longrightarrow$ All data of training set

Assuming independence of dimensions,

$$P(D|\theta) = \prod_{k=1}^{n} P(x_k|\theta)$$

where $$P(x_k|\theta) = \prod_{j=1}^{d} P(x_{kj}|\theta_j)$$

and $$P(x_{kj}|\theta_j) = \theta_j^{x_{kj}} (1-\theta_j)^{1-x_{kj}}$$

Hence,

$$P(D|\theta) = \prod_{k=1}^{n} \prod_{j=1}^{d} \theta_j^{x_{kj}} (1-\theta_j)^{1-x_{kj}}$$

Let log-likelihood be $l(\theta) = \ln P(D|\theta)$

$$l(\theta) = \sum_{k=1}^{n} \sum_{j=1}^{d} x_{kj} \ln(\theta_j) + (1-x_{kj}) \ln(1-\theta_j)$$

$$\cancel{\frac{d}{d\theta}} \nabla_\theta l(\theta) = \left[\frac{\partial l(\theta)}{\partial \theta_1} \quad \frac{\partial l(\theta)}{\partial \theta_2} \cdots \frac{\partial l(\theta)}{\partial \theta_d} \right]^T$$

Where $\frac{\partial l(\theta)}{\partial \theta_j} = \sum_{k=1}^{n} \frac{x_{kj}}{\theta_j} - \frac{(1-x_{kj})}{1-\theta_j}$

Solving for $\nabla_\theta l(\theta) = 0$,

$$\boxed{\theta_j = \frac{\sum_{k=1}^{n} x_{kj}}{n}}$$

← we use this to compute θ_{MLE} in the code

After computing MLE for $n = 1, 2, \ldots, 50$ it is clear that θ_{MLE} gets closer to the actual θ as n increases.

d) We ~~p~~lot using matplotlib.pyplot.scatter

e) Discriminant derivation for multivariate bernoulli:

no. of dimensions $\longrightarrow d$
no. of classes $\longrightarrow c$ $\{w_1, \ldots w_c\}$

$$x = [x_1 \; x_2 \ldots x_d]^t$$

$\theta_{ij} \longrightarrow$ parameter for the i^{th} class, j^{th} dimension

$$1 \leq i \leq c$$
$$1 \leq j \leq d$$

We have the discriminant for the i^{th} class as:

$$g_i(x) = \ln P(x \mid w_i) + \ln P(w_i)$$

where $P(x|w_i) = \prod_{j=1}^{d} \theta_{ij}^{x_j} (1-\theta_{ij})^{1-x_j}$

Hence,

$$g_i(x) = \sum_{j=1}^{d} x_j \ln \theta_{ij} + (1-x_j)\ln(1-\theta_{ij})$$

$$\cancel{+ \ln P(\omega_i)}$$

We can use this to find $g_i(x)$ $\forall\ 1 \leq i \leq c$, and chose ω_i for which $g_i(x)$ is maximum.

~~Also, we can easily calculate the prior $P(\omega_i)$ using the 50 training samples.~~

We ignore the prior since we have 50 samples each of ω_1 and ω_2

A2) We have,

d dimensions
C classes
n Samples

$\theta_j \leftarrow$ parameter for ~~ith class~~ j^{th} dimension

$x_{kj} \leftarrow$ value for k^{th} sample
j^{th} dimension

We have,

$$P(x_{kj} \mid \theta_j) = \theta_j^{x_{kj}} (1-\theta_j)^{1-x_{kj}}$$

$$P(x_k \mid \theta) = \prod_{j=1}^{d} \theta_j^{x_{kj}} (1-\theta_j)^{1-x_{kj}}$$

Let D be the entire dataset

$$P(D \mid \theta) = \prod_{j=1}^{d} \prod_{k=1}^{n} \theta_j^{x_{kj}} (1-\theta_j)^{1-x_{kj}}$$

Also,

$$P(\theta) = \exp\left\{-\sum_{j=1}^{d} \theta_j\right\} \prod_{j=1}^{d} \theta_j$$

Using conditional indipendence,

$$P(\theta_j) = \theta_j e^{-\theta_j}$$

for j^{th} dimension, we calculate $\theta_{MAP,j}$ as:

first maximise

$$l(\theta_j) = \ln P(D_j \mid \theta_j) P(\theta_j)$$

$$= -\theta_j + \ln \theta_j + \sum_{k=1}^{n} x_{kj} \ln(\theta_j)$$

$$+ \sum (1 - x_{kj}) \ln(1 - \theta_j)$$

$$\frac{\partial l(\theta_j)}{\partial \theta_j} = \frac{1}{\theta_j} - 1 + \sum_{k=1}^{n} \frac{x_{kj}}{\theta_j} - \frac{(1 - x_{kj})}{1 - \theta_j}$$

$$= 0$$

$$\frac{1-\theta_j}{\theta_j} \quad \frac{1}{\theta_j(1-\theta_j)} \sum_{k=1}^{n} x_{kj} - \theta_j = 0$$

$$\frac{(1-\theta_j)^2 - n\theta_j + \sum_{k=1}^{n} x_{kj}}{\theta_j(1-\theta_j)} = 0$$

$$\Rightarrow \boxed{\theta_j = \frac{n+2 \pm \sqrt{n^2+4n-4\sum_{k=1}^{n} x_{kj}}}{2}}$$

b) $d=2$, $N=4$

$$X = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

$$\theta_1 = \frac{6 \pm \sqrt{16+16-4(3)}}{2}$$

$$\theta_1 = 0.763$$

$$\theta_2 = \frac{6 \pm \sqrt{16+16-4}}{2}$$

$$\theta_2 = 0.354$$

A3) a) Let $X = \begin{bmatrix} 1 & 2 \\ 7 & 5 \end{bmatrix}$

$$\mu_x = \begin{bmatrix} 1.5 \\ 6 \end{bmatrix}$$

$$X_c = X - \mu_x = \begin{bmatrix} -0.5 & 0.5 \\ 1 & -1 \end{bmatrix}$$

Covariance Matrix $\leftarrow S$

$$S = \cancel{\begin{bmatrix} 0.25 & -0.5 \\ -0.5 & 1 \end{bmatrix}} \begin{bmatrix} 0.25 & -0.5 \\ -0.5 & 1 \end{bmatrix}$$

$$\text{Eigenvalues}_S = 5/4, \ 0$$

$$\text{Eigenvectors}_S = \begin{bmatrix} +1/\sqrt{5} \\ 2/\sqrt{5} \end{bmatrix}, \begin{bmatrix} 2/\sqrt{5} \\ 1/\sqrt{5} \end{bmatrix}$$

PCA computation

$$U = \begin{bmatrix} -1/\sqrt{5} & 2/\sqrt{5} \\ 2/\sqrt{5} & 1/\sqrt{5} \end{bmatrix}$$

$$Y = U^T X_c = \begin{bmatrix} -1/\sqrt{5} & 2/\sqrt{5} \\ 2/\sqrt{5} & 1/\sqrt{5} \end{bmatrix} \begin{bmatrix} -0.5 & 0.5 \\ 1 & -1 \end{bmatrix}$$

$$Y = \frac{1}{\sqrt{5}} \begin{bmatrix} 2.5 & -2.5 \\ 0 & 0 \end{bmatrix}$$

$$Y = \frac{\sqrt{5}}{2} \begin{bmatrix} 1 & -1 \\ 0 & 0 \end{bmatrix}$$

$$UY + \mu_x = \frac{1}{\sqrt{5}} \begin{bmatrix} -1 & 2 \\ 2 & 1 \end{bmatrix} \frac{\sqrt{5}}{2} \begin{bmatrix} 1 & -1 \\ 0 & 0 \end{bmatrix}$$

$$UY + \mu_x = \begin{bmatrix} 1 & 2 \\ 7 & 5 \end{bmatrix}$$

$$UY + \mu_x - X = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

b)

Hence MSE $= 0$

$\left(MSE = \frac{1}{2}(UY + \mu_x - X)^T (UY + \mu_x - X)\right)$

c) We use numpy and apply the same formulae.

From code also we get the same MSE value of 0.

d) We generate random mean $(d \times 1)$ and covariance matrices $(d \times d)$, generate samples and then recalculate the mean and covariance matrix for the data