ML/DL을 위한 수학 1,2

Q1.

(1) $A = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 2 & 3 \\ 0 & 0 & 3 \end{bmatrix}$

$$D_A = \begin{vmatrix} 1 & 0 & 1 \\ 0 & 2 & 3 \\ 0 & 0 & 3 \end{vmatrix} = 1\begin{vmatrix} 2 & 3 \\ 0 & 3 \end{vmatrix} - 0 \cdot \cancel{\begin{vmatrix} 0 & 3 \\ 0 & 3 \end{vmatrix}} + 1\begin{vmatrix} 0 & 2 \\ 0 & 0 \end{vmatrix}$$

$$= 2\times3 - 3\times\cancel{0} + 0\times\cancel{0} - 2\times\cancel{0} = \boxed{6}$$

determinant는 선형변환될 때 단위면적이 얼마나 늘어나는지 보여준다.

(2) $$\det(A - \lambda I) = \begin{vmatrix} 1-\lambda & 0 & 1 \\ 0 & 2-\lambda & 3 \\ 0 & 0 & 3-\lambda \end{vmatrix}$$

$$= (1-\lambda)\begin{vmatrix} 2-\lambda & 3 \\ 0 & 3-\lambda \end{vmatrix} - 0 \cdot \cancel{\begin{vmatrix} 0 & 3 \\ 0 & 3-\lambda \end{vmatrix}} + 1 \cdot \cancel{\begin{vmatrix} 0 & 2-\lambda \\ 0 & 0 \end{vmatrix}}$$

$$= (1-\lambda)(2-\lambda)(3-\lambda)$$

eigenvalue → $\boxed{1, 2, 3}$

$\lambda=1$ 선택시, $A - \lambda I = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 3 \\ 0 & 0 & 2 \end{bmatrix}$ 이며,

$\begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 3 \\ 0 & 0 & 2 \end{pmatrix}\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$ 이 되는 X, Y, Z가 필요

$X=1$ 대입 시, $0\cdot1 + 0\cdot Y + 1Z = 0 \quad \therefore Z=-1$

$0\cdot1 + 1\cdot Y + 3Z = 0 \quad \therefore Y=3$

따라서 $\boxed{\text{eigenvector} \rightarrow \begin{pmatrix} 1 \\ 3 \\ -1 \end{pmatrix}}$

벡터 A에 대해 n차 정방행렬 eigenvector를 곱한 결과와 상수인 eigenvalue를 곱한 결과와 같음을 의미한다.

Q2

우선 eigenvalue를 구한다.

$$\det(B - \lambda I) = \begin{vmatrix} 0-\lambda & 0 & 0 \\ 0 & 0-\lambda & 0 \\ 3 & 0 & 1-\lambda \end{vmatrix} = -\lambda\begin{vmatrix} 0-\lambda & 0 \\ 0 & 1-\lambda \end{vmatrix} = \lambda^2(1-\lambda)$$

$\lambda=0$ 선택시, $B - \lambda I = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 3 & 0 & 1 \end{bmatrix}$

$\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 3 & 0 & 1 \end{pmatrix}\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \rightsquigarrow \cancel{3X+Z}=0.\ V_1 = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}, V_2 = \begin{pmatrix} -1 \\ 0 \\ 3 \end{pmatrix}$

$\lambda=1$ 선택시, $B - \lambda I = \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 3 & 0 & 0 \end{bmatrix}$ $\quad P = \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 3 & 1 \end{pmatrix}$

$\begin{pmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 3 & 0 & 0 \end{pmatrix}\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \rightsquigarrow X=0,\ Y=0,\ Z=1 \quad V_3 = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$

And, $\boxed{D = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}}$, $P^{-1} = \begin{pmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 3 & 0 & 1 \end{pmatrix}$

$B = PDP^{-1}$ is verified.

Q3. '당첨' 이라는 단어를 포함하는데 스팸일 확률

$$P(\text{Spam} \mid \text{'당첨'}) = \frac{\overset{0.5}{\overset{\shortparallel}{P(\text{'당첨'} \mid \text{Spam})}} \cdot \overset{0.2}{\overset{\shortparallel}{P(\text{Spam})}}}{P(\text{'당첨'})}$$

$$= \frac{P(\text{'당첨'} \mid \text{Spam})\, P(\text{Spam})}{P(\text{'당첨'} \mid \text{Spam})\, P(\text{Spam}) + P(\text{'당첨'} \mid \text{Not Spam})\, P(\text{Not Spam})}$$

$$= \frac{0.5 \times 0.2}{0.5 \times 0.2 + 0.01 \times (1-0.2)} = \frac{0.1}{0.1 + 0.08} = \frac{0.1}{0.18} = \boxed{0.5556}$$

Q4

(2) $p(\text{Hit}) = \frac{9}{15} = 0.6$, $p(\text{Out}) = \frac{6}{15} = 0.4$

$$H = -(0.4 \log_2(0.4) + 0.6 \log_2(0.6))$$
$$= -(0.4 \times (-1.3219) + 0.6 \times (-0.7370))$$
$$= 0.4 \times 1.3219 + 0.6 \times 0.7370$$
$$= 0.971$$

(3) $p(\text{Hit}) = \frac{4}{15}$, $p(\text{Out}) = \frac{11}{15}$ (각각 0.267, 0.733으로 계산)

$$H = -(0.267 \times \log_2(0.267) + 0.733 \times \log_2(0.733))$$
$$= -(0.267 \times (-1.9050) + 0.733 \times (-0.4481))$$
$$= 0.267 \times 1.905 + 0.733 \times 0.4481 = 0.837$$

(2)에서의
entropy 값이
0.134 더 높으므로
(3)에서 나온 타자가
더 치우친 값을 가진다고 볼 수 있다.

Q5. Logistic Regression에서 Convex Optimization.

⇒ 경사하강법 사용

Logistic Regression의 cost function:

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}\log h_\theta(x) + (1-y^{(i)})\log(1-h_\theta(x))\right]$$

$J(\theta)$을 최소화하는 θ를 찾으면서 convex optimization이 진행됨.

$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$ 을 계속 반복하면서 최적의 θ를 찾음

$$\left(\frac{\partial}{\partial \theta_j} J(\theta) = \frac{1}{m}\sum_{i=1}^{m}\left[h_\theta(x^{(i)}) - y^{(i)}\right] x_j^{(i)} \right)$$