CS542 Machine Learning

Homework 2

Chih Wei Tung (U17550453)

3.3

$ED(w) = \frac{1}{2}\sum_{n=1}^{N} r_n\{t_n - w^T\phi(x_n)\}^2$

$\nabla E_D(w) = \sum_{n=1}^{N} r_n\{t_n - w^T\phi(x_n)\}\phi(x_n)^T$

$0 = \sum_{n=1}^{N} r_n t_n \phi(x_n)^T - \sum_{n=1}^{N} w^T\phi(x_n)\phi(x_n)^T$

$\Rightarrow \sum_{n=1}^{N} w^T\phi(x_n)\phi(x_n)^T = \sum_{n=1}^{N} r_n t_n \phi(x_n)^T$

$\phi = (\phi_0, \phi_1 \cdots, \phi_{m-1})^T$

$\Rightarrow \sum_{n=1}^{N} w^T(\phi_0, \phi_1 \cdots, \phi_{m-1})((\phi_0, \phi_1 \cdots, \phi_{m-1})^T)^T = \sum_{n=1}^{N} r_n t_n \phi(x_n)^T$

$r_1, \cdots, r_n$ $\Rightarrow$ represented by a diagonal matrix

Use the summing from 1 to N $((\phi_0, \phi_1 \cdots, \phi_{m-1})^T)^T \Rightarrow \Phi$

$\mathrm{w}^T\Phi^T R\Phi = \sum_{n=1}^{N} r_n t_n \phi(x_n)^T$

$= \sum_{n=1}^{N} r_n t_n \phi((\phi_0, \phi_1 \cdots, \phi_{m-1})^T)^T$

$\mathrm{w}^T = \Phi^T Rt$

$= \Phi^{-1}\mathrm{R}^{-1}(\Phi^T)^{-1}\Phi^T Rt$

$= (\mathrm{R}\Phi)^{-1}(\Phi^T)^{-1}\Phi^T Rt$

$\mathrm{w}^* = (\Phi^T\mathrm{R}\Phi)^{-1}\Phi^T Rt$

(i) Data dependent noise variation r_n can be seen as an inverse variance parameter to a data point (x_n, t_n) which modifies the precision matrix

(ii) r_n can be seen as an effective number of replicated observation of a data point (x_n, t_n)

3.11

(3.59): $\sigma_N^2(x) = \frac{1}{\beta} + \phi(x)^T S_N \phi(x)$

$$
\begin{aligned}
S_N^{-1} &= S_0^{-1} + \beta \Phi^T \Phi \\
&= S_0^{-1} + \beta \textstyle\sum_{n=1}^{N} \phi(x_n)\phi(x_n)^T \\
\Rightarrow S_{N+1}^{-1} &= S_0^{-1} + \beta \textstyle\sum_{n=1}^{N} \phi(x_n)\phi(x_n)^T \\
&= S_N^{-1} + \beta\phi(x_{N+1})\phi(x_{N+1})^T
\end{aligned}
$$

From (3.59)

$$\Rightarrow \sigma_{N+1}^2(x) = \frac{1}{\beta} + \phi(x)^T S_{N+1}\phi(x)$$

$$= \sigma_N^2(x) + \phi(x)^T(S_{N+1} - S_N)\phi(x)$$

Set $M = S_N^{-1}$ and $\sqrt{\beta}\phi(x_{N+1})$

$$\Rightarrow S_{N+1} = S_N - \frac{\beta S_N \phi(x_{N+1})\phi(x_{N+1})^T (S_N)}{1+\beta\phi(x_{N+1})^T S_N \phi(x_{N+1})}$$

$$\Rightarrow \sigma_{N+1}^2(x) = \sigma_N^2(x) - \frac{\beta\phi(x)^T S_N \phi(x_{N+1})\phi(x_{N+1})^T S_N \phi(x)}{1+\beta\phi(x_{N+1})^T S_N \phi(x_{N+1})}$$

S_N: positive definite

$S_N\phi(x_{N+1})\phi(x_{N+1})^T S_N$: positive semidefinite, nonnegative

$\Rightarrow \sigma_{N+1}^2(x) \le \sigma_N^2(x)$

3.14

$\alpha = 0, \; k(x, x') = \psi(x)^T \psi(x')$

When $\alpha = 0, \; {S_N}^{-1} = \beta \phi^T \Phi$

Since $\psi_j(x)$ is new orthonormal basis passing the same space

$\Rightarrow \psi(x) = V\phi(x)$

V, matrix that represents the function that transforms original basis to the new one

(V has an inverse)

First, transform Φ to the new orthonormal basis

$\Rightarrow \Phi V^T = \Psi$

$\Rightarrow \Phi = \Psi (V^T)^{-1}$

$\Rightarrow {S_N}^{-1} = \beta \phi^T \Phi$

$$
\begin{aligned}
\Rightarrow S_N &= (\beta \phi^T \Phi)^{-1} \\
&= \beta^{-1} (\phi^T \Phi)^{-1} \\
&= \beta^{-1} ((\Psi (V^T)^{-1})^T (\Psi (V^T)^{-1}))^{-1} \\
&= \beta^{-1} (\Psi (V^T)^{-1})^{-1} ((\Psi (V^T)^{-1})^T)^{-1} \\
&= \beta^{-1} (((V^T)^{-1})^{-1} \Psi^{-1}) (((V^T)^{-1})^T \Psi^T)^{-1} \\
&= \beta^{-1} ((V^T \Psi^{-1}) (\Psi^T)^{-1} V) \\
&= \beta^{-1} (V^T V)
\end{aligned}
$$

$$
\begin{aligned}
\Rightarrow \; K(x, x') &= \beta \phi(x)^T S_N \phi(x') \\
&= \beta \phi(x)^T \beta^{-1} (V^T V) \phi(x') \\
&= \phi(x)^T (V^T V) \phi(x')
\end{aligned}
$$

Apply ϕ to linear transformation V and V^T

$\Rightarrow \; K(x, x') = \psi(x)^T \psi(x')$

$$
\begin{aligned}
\Rightarrow \sum_{n=1}^{N} k(x, x_n) &= \sum_{n=1}^{N} \psi(x)^T \psi(x_n) \\
&= \sum_{n=1}^{N} \sum_{i=1}^{M} \psi_i(x)^T \psi_i\,(x_n)
\end{aligned}
$$

Use 3.115

$\Rightarrow \sum_{n=1}^{N} k(x, x_n) = \; \sum_{i=1}^{M} \psi_i(x_n) \psi_i(x_n) = 1$

3.21

Let A be real, symmetric matrix A

The transformation of A can be recreated by multiplying the original vector by constant λ_i

Let $\{u_i\}$ be the set of orthonormal vectors

$\Rightarrow Au_i = \lambda_i u_i$

$\Rightarrow \ln|A| = \ln \prod_{i=1}^{M} \lambda_i$

$= \sum_{i=1}^{M} \ln \lambda_i$

α: random optimized value

$\Rightarrow \frac{\delta \ln|A|}{\delta\alpha} = \sum_{i=1}^{M} \frac{1}{\lambda_i}\frac{\delta\lambda_i}{\delta\alpha}$

Recreate matrixes and their inverse with their eigenvalues

$A = \sum_{i=1}^{M} \lambda_i u_i {u_i}^T$

$A^{-1} = \sum_{i=1}^{M} \frac{1}{\lambda_i} u_i {u_i}^T$

$\Rightarrow \frac{\delta}{\delta\alpha} A = \sum_{i=1}^{M} \frac{\delta\lambda_i}{\delta\alpha} u_i {u_i}^T + \lambda_i (\frac{\delta u_i}{\delta\alpha} {u_i}^T + u_i \frac{\delta {u_i}^T}{\delta\alpha})$

If length of u_i is always constant

$\Rightarrow$ the derivative of a vector is orthogonal to the vector

$\Rightarrow u_i \left(\frac{\delta u_i}{\delta\alpha}\right) = 0$

$\Rightarrow \frac{\delta}{\delta\alpha} A = \sum_{i=1}^{M} \frac{\delta\lambda_i}{\delta\alpha} u_i {u_i}^T$

$\Rightarrow Tr\left(A^{-1} \frac{\delta}{\delta\alpha} A\right) = Tr(\sum_{i=1}^{M} \frac{1}{\lambda_i} u_i {u_i}^T \sum_{j=1}^{M} \frac{\delta\lambda_j}{\delta\alpha} u_j {u_j}^T)$

$= tr(\sum_{i=1}^{M} \sum_{j=1}^{M} \frac{1}{\lambda_i}\frac{\delta\lambda_j}{\delta\alpha} u_i {u_i}^T u_j {u_j}^T)$

Multiplication of orthogonal vectors = 0, $\sum_{i=1}^{M} u_i {u_i}^T = I$

$\Rightarrow Tr\left(\frac{1}{\lambda_i}\frac{\delta\lambda_j}{\delta\alpha}\right) = \sum_{i=1}^{M} \frac{1}{\lambda_i}\frac{\delta\lambda_j}{\delta\alpha}$

$\ln(t|\alpha,\beta) = \frac{M}{2}\ln\alpha + \frac{N}{2}\ln\beta - \mathrm{E}(m_N) - \frac{1}{2}\ln|A| - \frac{N}{2}\ln(2\pi)$

$\Rightarrow \frac{d\ln(t|\alpha,\beta)}{\alpha} = \frac{M}{2}\frac{1}{\alpha} - \frac{1}{2} m_N^T m_N - \frac{1}{2} Tr(A^{-1}\frac{d}{d\alpha}A)$

$$= \frac{1}{2}(\frac{M}{\alpha} - m_N^T m_N - Tr(A^{-1}\frac{d}{d\alpha}A))$$

$A = S_N^{-1}$

$\Rightarrow \frac{d}{d\alpha}A = 1$

$$\Rightarrow \frac{d \ln(t|\alpha, \beta)}{\alpha} = \frac{1}{2}(\frac{M}{\alpha} - m_N^T m_N - Tr(A^{-1}))$$

$$= \frac{1}{2}(\frac{M}{\alpha} - m_N^T m_N - \sum \frac{1}{\lambda_i + \alpha})$$

$\Rightarrow$ 3.117 can be used to derive 3.92

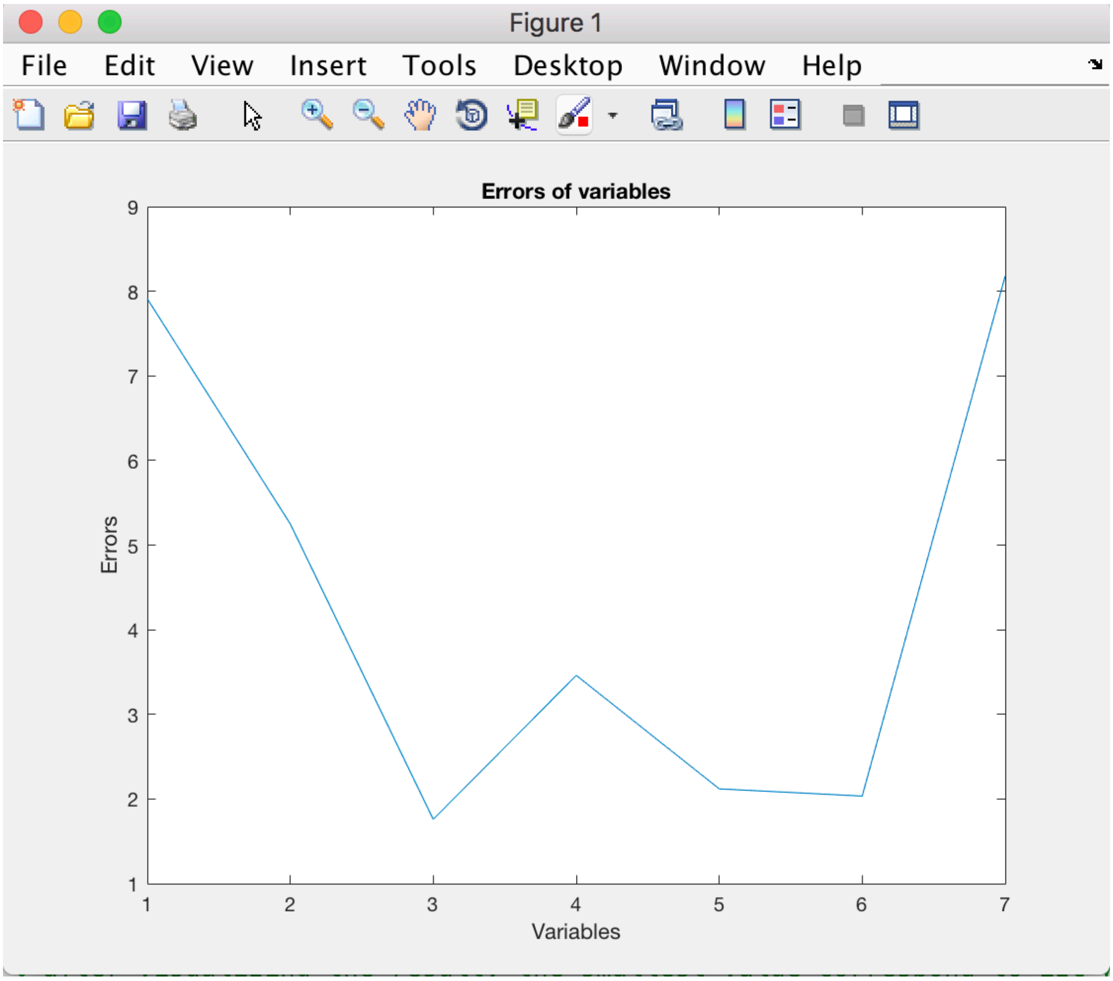
2(a)

Before finding the third variable in determining HOM, we must to create a matrix with size of 1*13 that fill it with 1 and combine it with FTP and WE ⇒ [1, FTP, WE] (Matrix1)

The way to find the third variable in determining HOM is shown as below:

1. Combine the matrix of targeted variable with Matrix1 ⇒ [1, FTP, WE, variable] (Matrix2)
2. Get the lowest error for each variable:
    1. Get the sum of (Matrix2*((((Matrix2')*Matrix2)^(-1))*(Matrix2')*HOM))-HOM
    2. Lowest error will be "sum/(2*13)"
3. Find the index of the minimum value from Step 2
4. The minimum of those lowest error is LIC, so **LIC is the third variable in determining HOM**.

The errors of each variables can be seen as below:



2(b)

(i) The unknown non-number features are replaced by the median value of all the other features. And the unknown number features are replaced by label-conditioned mean (sum of the feature of “+” or “-“ / number of ”+” or “-“).

(iii) Accuracy Table

K value	Lenses	CRX
1	5/6 = 83.33%	114/138 = 82.6%
2	4/6 = 66.66%	113/138 = 81.88%
3	5/6 = 83.33%	113/138 = 81.88%
5	3/6 = 50%	117/138 = 84.78%
10	3/6 = 50%	119/138 = 86.23%
100	3/6 = 50%	108/138 = 78.26%
300	3/6 = 50%	97/138 = 70.28%
500	3/6 = 50%	83/138 = 60.14%

PS. I couldn’t generate executable file for both of the program of problem 2. But the first one works on MATLAB app, and the second one works with standard commands (python xxx.py ….)