SANJANA AGARWAL

ASSIGNMENT 2.

1 a.) $X_1 \ldots X_n \Rightarrow$ Gaussian random variables $\sim N(\theta, \sigma^2)$

Prior : $N(\mu, \sigma^2)$

$$M_{MAP} = \underset{M \in \mathcal{M}}{\operatorname{argmax}} \{ p(D|M) \cdot p(M) \}$$

$$\theta_{MAP} = \underset{\theta}{\operatorname{argmax}}\ p(\theta|D) = \underset{\theta}{\operatorname{argmax}}\ p(D|\theta) \cdot p(\theta$$

$$= \underset{\theta}{\operatorname{argmax}} (\log p(D|\theta) + \log p(\theta))$$

$$p(D|\theta) = p(x_1, \ldots x_n|\theta) = \prod_{i=1}^{n} p(x_i|\theta)$$

$$= \left(\frac{1}{\sqrt{2\pi\sigma^2}}\right)^n \exp\left(\frac{-1}{2\sigma_0^2} \sum_{i=1}^{n} (x_i - \theta)^2\right)$$

$$\log p(D|\theta) = \frac{-n}{2} \log(2\pi\sigma_0^2) - \frac{1}{2\sigma^2} \sum_{i=1}^{n} (x_i - \theta)^2$$

$$0 = \frac{\partial}{\partial\theta} \log p(D|\theta) = \frac{1}{2\sigma_0^2} \left(\sum_{i=1}^{n} x_i - n\theta\right)$$

$$= \frac{1}{\sigma^2} \left(\sum_{i=1}^{n} x_i - n\theta\right)$$

$$\Rightarrow \theta = \frac{1}{n} \sum_{i=1}^{n} x_i$$

$$\frac{\partial}{\partial\theta} \log p(\theta) = \frac{\partial}{\partial\theta} \left(\log \left(\frac{1}{\sqrt{2\pi}} e^{\frac{-1}{2}\left(\frac{\theta-\mu}{\sigma}\right)^2}\right)\right)$$

$$= \frac{\partial}{\partial \theta}\left(\frac{-1}{2}\log 2\pi - \frac{1}{2}\left(\frac{\theta-\mu}{\sigma}\right)^2\right)$$

$$= \frac{-1}{2} \times 2\left(\frac{\theta-\mu}{\sigma^2}\right) = \frac{\mu-\theta}{\sigma^2}$$

$$\frac{\partial}{\partial \theta}\left(\log p(D|\theta) + \log p(\theta)\right) = 0$$

$$\therefore \frac{1}{\sigma_0^2}\left(\sum_{i=1}^{n} x_i - n\theta\right) + \left(\frac{\mu-\theta}{\sigma^2}\right) = 0$$

$$\frac{\sum_{i=1}^{n} x_i}{\sigma_0^2} + \frac{\mu}{\sigma^2} - \left(\frac{n}{\sigma_0^2} + \frac{1}{\sigma^2}\right)\theta = 0$$

$$\therefore \frac{\sigma^2 \sum_{i=1}^{n} x_i + \sigma_0^2 \cdot \mu}{\sigma_0^2 \cdot \sigma^2} = \left(\frac{\sigma^2 n + \sigma_0^2}{\sigma_0^2 \sigma^2}\right)\theta$$

$$\therefore \theta_{MAP} = \frac{n\sigma_0^2}{\sigma^2 n + \sigma_0^2}\left(\frac{1}{n}\sum_{j=1}^{n} x_i\right) + \left(\frac{\sigma_0^2}{\sigma_n^2 + \sigma_0^2}\right)\cdot \mu$$

1.b) $X_1 \ldots X_n \Rightarrow$ Gaussian random variable

Prior : $\mathcal{L}(u, b)$, known variance.

$$p(x) = \frac{1}{2b} \exp\left(-\frac{|x-u|}{b}\right)$$

$$\theta_{MAP} = \underset{\theta}{\operatorname{argmax}}\ (\theta|D) = \underset{\theta}{\operatorname{argmax}}\ p(D|\theta) \cdot p(\theta)$$

$$\theta_{MAP} = \underset{\theta}{\operatorname{argmax}}\ (\log p(D|\theta) + \log p(\theta))$$

~~$\theta_{MAP} = \underset{\theta}{\operatorname{argmax}}\ (\log p(D|\theta) + \log$~~

$$0 = \frac{\partial}{\partial\theta} (\log p(D|\theta) + \log p(\theta)) - (1)$$

$$\frac{\partial}{\partial\theta} \log p(D|\theta) = \frac{1}{\sigma_0^2} \left(\sum_{i=1}^{n} x_i - n\theta\right) - (2)$$

$$\frac{\partial}{\partial\theta} \log p(\theta) = \frac{\partial}{\partial\theta} \left(-\log 2b + \left(-\frac{|\theta-u|}{b} \log e\right)\right)$$

$$= \frac{\partial}{\partial\theta} \left(-\log 2b - \frac{|\theta-u|}{b}\right)$$

$$= -\frac{\partial}{\partial\theta} \frac{|\theta-u|}{b} \quad - (3.)$$

$$= -\frac{\partial}{\partial\theta} \frac{|\theta|}{b} \quad \text{Since} \quad u = 0$$

Substituting (2) & (3) in (1),

$$0 = \frac{1}{\sigma_0^2}\left(\sum_{i=1}^{n} x_i - n\theta\right) - \frac{2}{\partial\theta}\frac{|\theta|}{b}$$

It is not possible to obtain a closed form solution.

So to get the solution, we start at some point & go in the negative gradient descent direction until we reach a local minimum.

Thus, using gradient descent as the iterative approach, we can obtain the solution!

O.C

1c) $X_1 \ldots X_n \Rightarrow$ Gaussian random variable $N(\theta, \Sigma_0)$
$\theta \in R^d$, mean is unknown.
$\Sigma_0 = I \in R^{d \times d}$

Prior : Zero-mean multivariate Gaussian.

$$\theta_{MAP} = \underset{\theta}{\operatorname{argmax}}\ p(\theta|D) = \underset{\theta}{\operatorname{argmax}}\ p(D|\theta) \cdot p(\theta)$$

$$\theta_{MAP} = \underset{\theta}{\operatorname{argmax}}\ (\log P(D|\theta) + \log P(\theta))$$

$$\frac{\partial}{\partial \theta} (\log P(D|\theta) + \log P(\theta)) = 0 \quad -(1.)$$

$$\frac{\partial}{\partial \theta} (\log P(D|\theta)) = \frac{\partial}{\partial \theta} \left(\log \left(\frac{1}{(2\pi)^{n/2} \cdot |\Sigma_0|^{1/2}} \cdot \exp\left(\frac{-1}{2}(x-\theta)^T \Sigma_0^{-1} (x-\theta)\right)\right)\right)$$

$$= \frac{\partial}{\partial \theta} \left(-\frac{Nd}{2} \log(2\pi) - \frac{N}{2} \log|\Sigma_0| - \frac{1}{2} \sum_{i=1}^{N} (x_i - \theta)^T \Sigma_0^{-1} (x_i - \theta) \right)$$

Using trace trick,

$$\sum_{i=1}^{N} (x_i - \theta)^T \Sigma_0^{-1} \cdots = \sum_{i=1}^{N} \Sigma_0^{-1} (x - \theta), \quad -(2.)$$

$$\frac{\partial}{\partial \theta} (\log P(\theta)) = \frac{\partial}{\partial \theta} \left(\log \left(\frac{1}{2\pi^{d/2} \cdot |\Sigma|^{1/2}} \exp\left(\frac{-1}{2}(\theta - \mu)^T \Sigma^{-1} (\theta - \mu)\right)\right)\right)$$

$$= \frac{\partial}{\partial \theta}\left(-\frac{d}{2}\log\left(\frac{1}{2\pi}\right) - \frac{1}{2}\log|\Sigma| - \frac{1}{2}(\theta-\mu)^T(\theta-\mu)\log e\right)$$

Substituting $\mu = 0$

$$= \frac{\partial}{\partial \theta}\left(-\frac{1}{2}(\theta-\mu)^T\Sigma^{-1}(\theta-\mu)\right)$$

$$= -\frac{1}{2}\frac{\partial}{\partial \theta}\left(\theta^T\Sigma^{-1}\theta\right)$$

$$= -\frac{1}{2} \times 2 \cdot \Sigma^{-1}\theta = -\Sigma^{-1}\theta \quad -(3.)$$

Substituting (2) & (3) in (1.), we get,

$$0 = \sum_{i=1}^{N} \Sigma_0^{-1}(x-\theta) + (-\Sigma^{-1}\theta)$$

$$\Sigma^{-1}\cdot\theta = \sum_{i=1}^{N} \Sigma_0^{-1} x - N\theta\Sigma_0^{-1}$$

$$\therefore \theta = \frac{\sum_{i=1}^{N} \Sigma_0^{-1} x}{\Sigma^{-1} + N\Sigma_0^{-1}}$$

$$= \left(\sum_{i=1}^{N} \Sigma_0^{-1} x\right)\left(\Sigma^{-1} + N\Sigma_0^{-1}\right)^{-1}$$

2.a. When we increase the number of features, we get an error: numpy.linalg.linalg.LinAlgError : Singular matrix. This error is thrown when we calculate the value of X^T, since calculating X^T ~~that~~ involves finding the determinant of Matrix X. (1x1) Thus, we cannot invert it (also why singular matrix is called non-invertible)

This is happening because as the number of features increases, X has more features that are identical, similar or nearly dependent.

To remedy this, we can use Singular Value Decomposition. Thus, X can be transformed as $U\Sigma V^T$ for orthonormal matrices $U \in \mathbb{R}^{n \times n}$, $V \in \mathbb{R}^{d \times d}$ & non-negative diagonal matrix $\Sigma \in \mathbb{R}^{n \times d}$

Thus, $X^T X = V\Sigma U^T U \Sigma V^T$

$= V \Sigma_d^2 V^T$ $(\Sigma_d = \Sigma(1:d, 1:d))$

$\because$ U is orthonormal $(U^T U = I)$

$\therefore w = V \Sigma_d^{-2} V^T \cdot y$

2(b) The code runs for 2 runs for each predictor. The split that I have decided to do is 80-20 for training & test data.
It makes the calculations simple and have used the split of 80-20 for several experiments before. I did not use only the first half of the data as there is a possibility that the data is ordered in a specific way.

2(c) I had tried to create a list of list of the meta-parameters and even tried to ~~create~~ add the features to the regressionAlgs, but it made looping through difficult. It was difficult to understand the code flow, so created a new table to map key-value pairs of the feature & its value.
Thus, the average mean & standard errors have been reported in the code.

2(d.) The error after comparing with FSLinear-Regression is lesser clearly.
We also observe that as λ increases, error decreases more. Thus, the regularization parameter λ helps reduce the error. It happens because the model has lesser chances of overfitting becaus

of increasing λ.

2.(e) As we increase the number of features we get the same error as Singular Matrix. As we increase number of features, till 69,000 we get the error to be reducing. The implementation is in the code.

2.(f) Lasso Regression takes 2 parameters, namely, soft threshold parameter & tolerance.

α is used for shrinkage operator. Thus, error decreases with when α increases as the overall weight is more.

I have used batch gradient descent & stop when the Δe reduces below the threshold value. This is because the mode trains longer with decreasing tolerance.

2(g) For stochastic gradient descent, for every epoch, we calculate δ and w as

$$\delta = (x^T w - y) * x$$

& $w = w - \text{incr} * \delta$.

Thus the value of incr is kept small for better results.

The weight is updated in the direction of the gradient descent.

2(h) Stochastic gradient descent gives better results with more epochs. I observed the following values:

~~As tolerance decreased, the mean err~~
As tolerance decreased, we needed more time to run, thus the error was reduced.

I observed a few examples:

Tolerance	Epoch	Mean Error	Time
10e-6	1449, 2	0.193, 0.505	0.609, 0.015
10e-5	1782, 1, 1	0.186, 0.488, 0.488	0.567, 0.002, 0.002