1. Refreshing Math [20 Points]

$w \in \mathbb{R}^n$ (column vector) $\quad f(w) \in \mathbb{R} \quad \nabla f(w) \in \mathbb{R}^n \quad H \in \mathbb{R}^{n \times n}$

(a) $f(w) = w^T x$

$1 \times n \quad n \times 1$

$$f(w) = \begin{bmatrix} w_1, w_2, \cdots w_{n-1}, w_n \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n-1} \\ x_n \end{bmatrix}$$

$f(w) = w_1 x_1 + w_2 x_2 + \cdots + w_{n-1} x_{n-1} + w_n x_n$ // Expand to sum

$$\nabla f(w) = \begin{bmatrix} \frac{\partial f}{\partial w_1} \\ \vdots \\ \frac{\partial f}{\partial w_n} \end{bmatrix}$$ // Gradient definition

$$\nabla f(w) = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n-1} \\ x_n \end{bmatrix}$$

$\nabla f(w) = x$ // Convert to matrix form.

(b) $f(w) = tr(ww^T A)$ where $A \in \mathbb{R}^{n \times n}$

$f(w) = tr(ww^T A) = tr(w^T A w)$ // using trace property $tr(AB) = tr(BA)$

$f(w) = w^T A w$ // since $w^T A w$ is a 1×1 matrix, it is equal to its trace.

$$f(w) = w^T \underbrace{\begin{bmatrix} \sum_{j=1}^{n} a_{1j} w_j \\ \sum_{j=1}^{n} a_{2j} w_j \\ \vdots \\ \sum_{j=1}^{n} a_{dj} w_j \end{bmatrix}}_{Aw}$$ // Summation Form

$$f(w) = \sum_{i=1}^{d} \sum_{j=1}^{d} w_i a_{ij} w$$ // More summation form (including w^T)

$$= \sum_{i=1}^{d} \left(a_{ii} w_i^2 + \sum_{j \neq i} w_i a_{ij} w_j \right)$$ // Convert to easier form

$$\frac{\partial}{\partial w_k}\left[\sum_{i=1}^{d}\left(a_{ii}w_i^2 + \sum_{j\neq i} w_i a_{ij} w_j\right)\right]$$ // Take partial derivative w.r.t a generic element k.

$$\frac{\partial}{\partial w_k} = 2a_{kk}w_k + \sum_{j\neq k} w_j a_{jk} + \sum_{j\neq k} a_{ki} w_j$$

$$\frac{\partial}{\partial w_k} = \sum_{j=1}^{d} w_j a_{jk} + \sum_{j=1}^{d} a_{kj} w_j$$

$$\nabla f(w) = \begin{bmatrix} \frac{\partial f}{\partial w_1} \\ \vdots \\ \frac{\partial f}{\partial w_d} \end{bmatrix} = \begin{bmatrix} \sum_{j=1}^{d} w_j a_{j1} + \sum_{j=1}^{d} a_{1j} w_j \\ \vdots \\ \sum_{j=1}^{d} w_j a_{jd} + \sum_{j=1}^{d} a_{dj} w_j \end{bmatrix} = \begin{bmatrix} \sum_{j=1}^{d} = w_j a_{j1} \\ \vdots \\ \sum_{j=1}^{d} w_j a_{jd} \end{bmatrix} + \begin{bmatrix} \sum_{j=1}^{d} a_{1j} w_j \\ \vdots \\ \sum_{j=1}^{d} a_{dj} w_j \end{bmatrix} = A^T w + Aw$$

(c) Hessian of $f(w) = tr(ww^TA)$

From above, $\frac{\partial f}{\partial w_k} = \sum_{j=1}^{d} w_j a_{jk} + \sum_{j=1}^{d} a_{kj} w_j$

Thus, $\frac{\partial^2 f}{\partial w_k \partial w_{k'}} = a_{k'k} + a_{kk'}$

$$\therefore \nabla^2 f(w) = \begin{bmatrix} a_{11}+a_{11} & a_{21}+a_{12} & \cdots & a_{d1}+a_{1d} \\ a_{12}+a_{21} & a_{22}+a_{22} & \cdots & a_{d2}+a_{2d} \\ \vdots & \vdots & \ddots & \\ a_{1d}+a_{d1} & a_{2d}+a_{d2} & \cdots & a_{dd}+a_{dd} \end{bmatrix}$$

$$\therefore \nabla^2 f(w) = A + A^T$$

(d) $\sigma(a) = \frac{1}{1+e^{-a}}$, $f(w) = \log(\sigma(w^T x))$

Let $f(w) = \log(v)$

Let $v = \sigma(a) = (1+e^{-a})^{-1}$

Let $a = w^T x$

$$\frac{df}{dw_n} = \frac{df}{dv} \cdot \frac{dv}{da} \cdot \frac{da}{dw_n}$$

$$\frac{df}{dv} = \frac{1}{v} = \frac{1}{\sigma(w^T x)}$$

$$\frac{dv}{da} = (1+e^{-a})^{-1} = \frac{e^a}{(e^a+1)^2} = \frac{e^{w^T x}}{(e^{w^T x}+1)^2}$$

$$\frac{da}{dw_n} = x_n$$

$$w^T x = w_1 x_1 + \cdots + w_n x_n$$

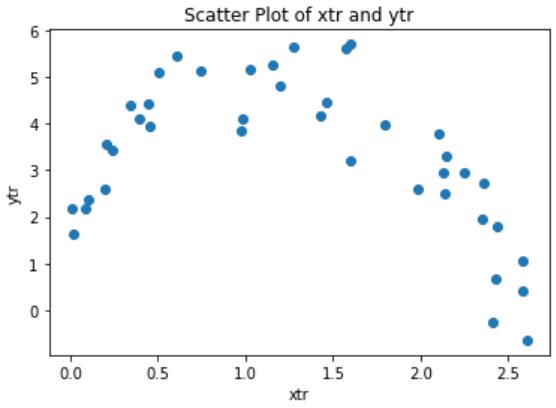
$$\frac{\partial f}{\partial w_1} = \frac{df}{dv} \cdot \frac{dv}{da} \cdot \frac{da}{dw_1} = \left(\frac{1}{\sigma(w^T x)}\right)\left(\frac{e^{w^T x}}{(e^{w^T x}+1)^2}\right)(x_1)$$

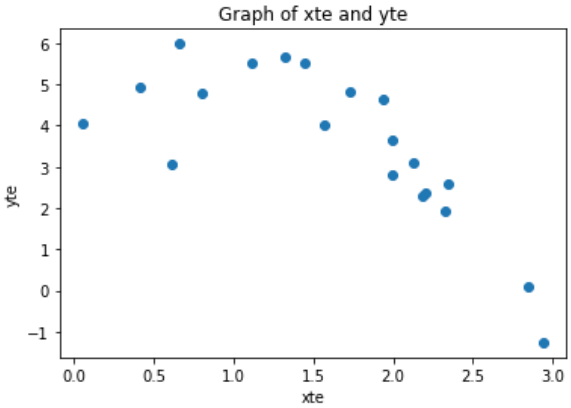
$$\frac{\partial f}{\partial w_n} = \frac{df}{dv} \cdot \frac{dv}{da} \cdot \frac{da}{dw_n} = \left(\frac{1}{\sigma(w^T x)}\right)\left(\frac{e^{w^T x}}{(e^{w^T x}+1)^2}\right)(x_n)$$

$$\therefore \nabla f(w) = \begin{bmatrix} \frac{\partial f}{\partial w_1} \\ \vdots \\ \frac{\partial f}{\partial w_n} \end{bmatrix} = \begin{bmatrix} \frac{e^{w^T x} x_1}{\sigma(w^T x)(e^{w^T x}+1)^2} \\ \vdots \\ \frac{e^{w^T x} x_n}{\sigma(w^T x)(e^{w^T x}+1)^2} \end{bmatrix}$$

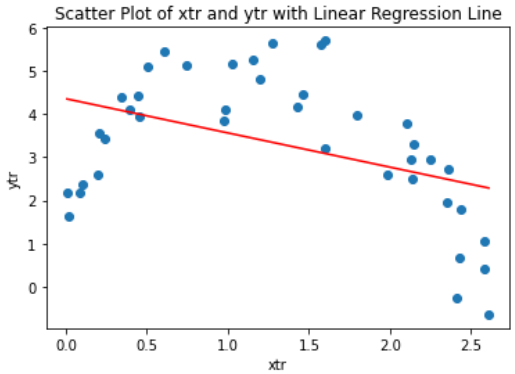
Linear and Polynomial Regression

2(a)



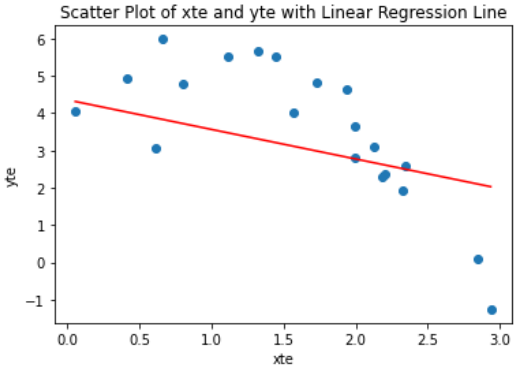


2(b)



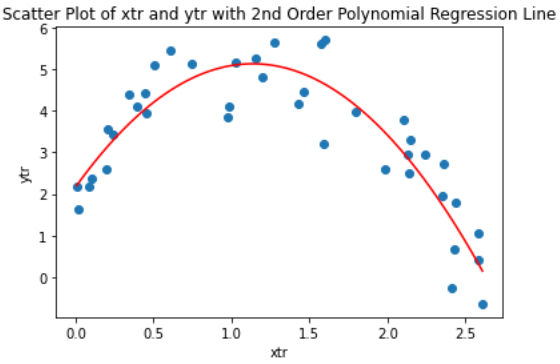
Average Error (Training): 2.173945579049259

2(c)

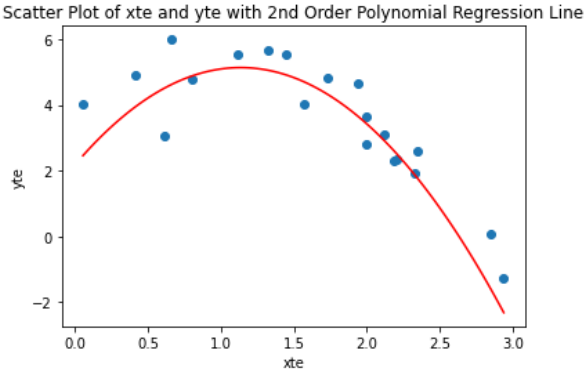


Average Error (Test): 2.3118753456727985

2(d)



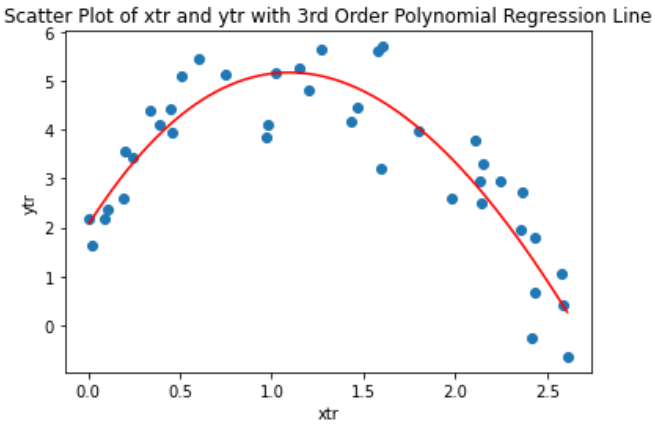
Average Error (Training): 0.484684503127155

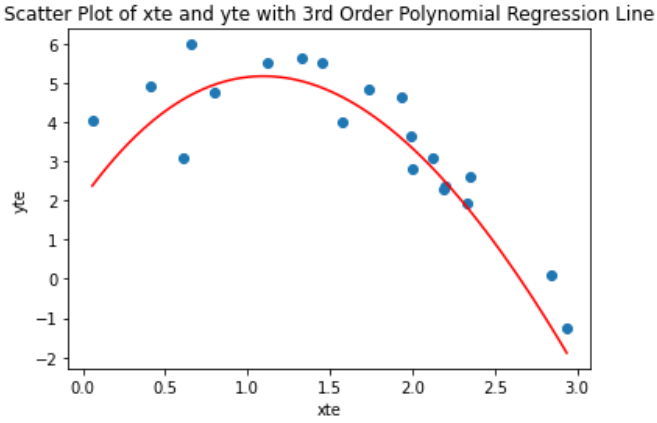


Average Error (Test): 0.7573635655518017

For 2nd order polynomial, the training error is less than the test error, but they are relatively close. Both look like they fit the data well. 2nd order polynomial regression is a better fit than linear regression. Both the training and test error are lower than linear regression errors. Therefore, it fits the training and test data better.

2(e)

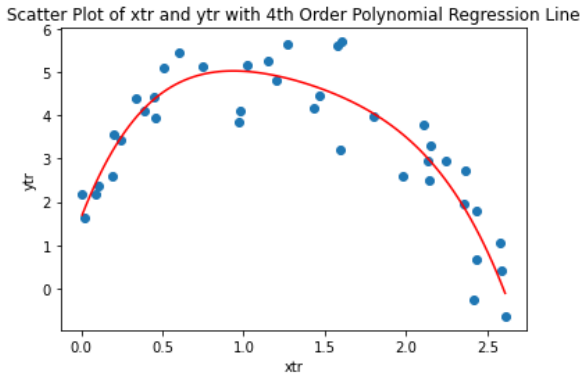


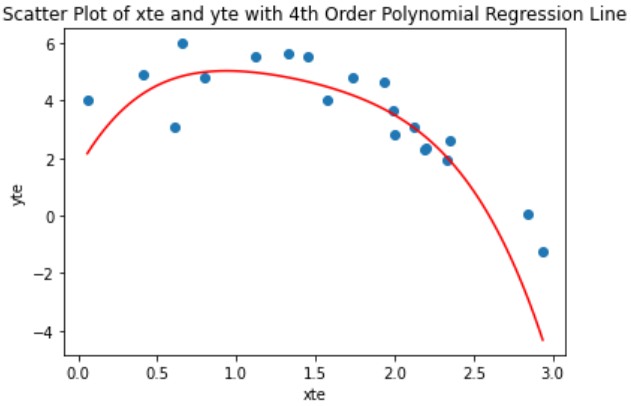


Average Error (Training): 0.48055213344532516 Average Error (Test): 0.6911245362890185

For 3rd order polynomial, the training error is less than the test error, meaning it fits the training data better. They are not too far apart. The training error is approximately the same as 2nd order polynomial. Thus, it has a better fit than linear and similar fit as 2nd order. However, the test error is lower than both linear and 2nd order. Therefore 3rd order fits the best.

2(f)





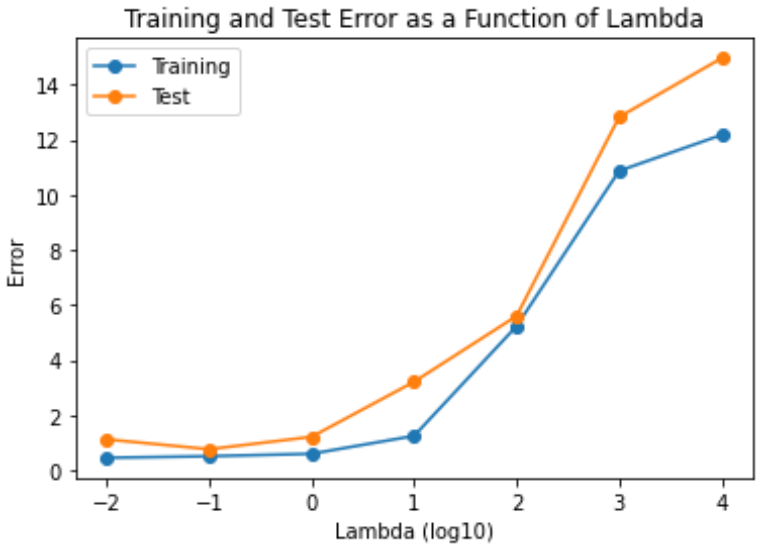
Average Error (Training): 0.4366476340997149 Average Error (Test): 1.5584694832601127

For 4th order polynomial regression, the training error is significantly lower than the test error. This seems to be due to over fitting of the training data. When compared to the other orders of regression, 4th

order has the best fit to the training data, but the 2nd worst fit to the test data. Based on both training and test error, 3rd order is the best fit (low training error, lowest test error).

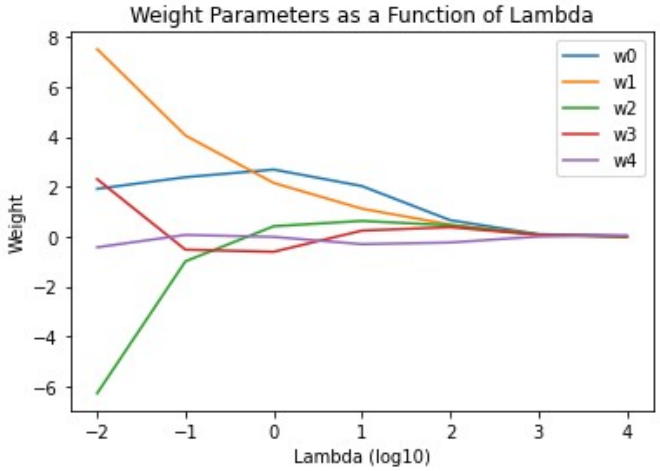
Regularization and Cross-validation

3(a)



When lambda=0.1 (-1 on the graph) is has a low training error and the lowest test error. Thus, it is the best for fitting the data.

3(b)



3(c)

Lambda 0.01 train error: 0.4377717030770172 validation error: 0.5677062928552145
Lambda 0.1 train error: 0.4995684263361276 validation error: 0.612600028938294
Lambda 1.0 train error: 0.5994666449326319 validation error: 0.7086804611397775
Lambda 10.0 train error: 1.409355036723435 validation error: 1.5105268234400362
Lambda 100.0 train error: 5.8841710454316924 validation error: 6.011980449531493
Lambda 1000.0 train error: 11.120323333442272 validation error: 11.309312830378898
Lambda 10000.0 train error: 12.191244945412908 validation error: 12.390229359958067

From the above results, the average validation error for lambda=0.01 is the lowest. Thus, it fits the data the best. It is different from the results from part 3(a) (lambda=0.1). The errors for both lambda values are really close, and the implementation of the code (i.e. packages used) may have given them slightly different values (sklearn vs Numpy).