$\boxed{\text{CS 4442 Assignment \#1}}$ $\boxed{\text{Matvey Skripchenko}}$

$\boxed{\#1}$

a) $f(w) = w^T x$, where $x \in \mathbb{R}^n$, $w \in \mathbb{R}^n$, $f(w) \in \mathbb{R}$.

Thus, $f(w) = [w_1, w_2, w_3, \dots w_n] \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix} = w^T x$.

$$f(w) = w_1 x_1 + w_2 x_2 + w_3 x_3 + \dots + w_n x_n$$

Then, $\nabla f(w) = \left[\frac{\partial f}{\partial w_1}, \frac{\partial f}{\partial w_2}, \frac{\partial f}{\partial w_3}, \dots, \frac{\partial f}{\partial w_n} \right]$

$= [x_1, x_2, x_3, \dots, x_n]$

$= x^T$.

$\boxed{\text{Therefore, } \nabla f(w) = x^T}$ $\therefore$

b) $f(w) = tr(w w^T A)$

Using $tr(AB) = tr(BA)$ property, then:

$f(w) = tr(w^T A w)$

$$= tr\left([w_1, w_2, \dots, w_n] \begin{bmatrix} \sum_{i=1}^{n} a_{1j} w_j \\ \sum_{i=1}^{n} a_{2j} w_j \\ \vdots \\ \sum_{i=1}^{n} a_{nj} w_j \end{bmatrix} \right)$$

$$= \sum_{i=1}^{n} a_{1j} w_1 w_j + \sum_{i=1}^{n} a_{2j} w_2 w_j + \dots + \sum_{i=1}^{n} a_{nj} w_n w_j$$

$$f(w) = a_{11} w_1^2 + a_{12} w_1 w_2 + \dots + a_{1n} w_1 w_n +$$
$$+ a_{21} w_1 w_2 + a_{22} w_2^2 + \dots + a_{2n} w_2 w_n + \dots +$$
$$+ a_{n1} w_1 w_n + a_{n2} w_2 w_n + \dots + a_{nn} w_n^2$$

Then, $\nabla f(w) = \left[\frac{\partial f}{w_1}, \frac{\partial f}{w_2}, \frac{\partial f}{w_3}, \dots, \frac{\partial f}{w_n} \right]$

$$= \Big[2a_{11} w_1 + a_{12} w_2 + \dots + a_{1n} w_n + a_{21} w_2 + \dots + a_{n1} w_n,$$
$$a_{21} w_1 + 2a_{22} w_2 + \dots + a_{2n} w_n + a_{12} w_1 + a_{32} w_3 + \dots +$$
$$+ a_{n2} w_n, \dots\dots\dots,$$
$$a_{n1} w_1 + a_{n2} w_2 + \dots + 2a_{nn} w_n + a_{1n} w_1 + a_{2n} w_2 + \dots$$
$$+ a_{n-1\,n} w_{n-1} \Big]$$

$$= \Big[2a_{11} w_1 + (a_{12} + a_{21}) w_2 + (a_{13} + a_{31}) w_3 + \dots +$$
$$+ (a_{1n} + a_{1n}) w_n, (a_{12} + a_{21}) w_1 + 2a_{22} w_2 +$$
$$+ (a_{23} + a_{32}) w_3 + \dots + (a_{2n} + a_{n2}) w_n, \dots\dots$$
$$\dots\dots, (a_{n1} + a_{1n}) w_1 + (a_{n2} + a_{2n}) w_2 + \dots +$$
$$+ 2a_{nn} w_n \Big]$$

$$\nabla f(w) = \left(\begin{bmatrix} 2a_{11} & a_{12}+a_{21} & a_{13}+a_{31} & \cdots & a_{1n}+a_{n1} \\ a_{21}+a_{12} & 2a_{22} & a_{23}+a_{32} & \cdots & a_{2n}+a_{n2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1}+a_{1n} & a_{n2}+a_{2n} & a_{n3}+a_{3n} & \cdots & 2a_{nn} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix} \right)^T$$

$$\boxed{\text{Therefore, } \nabla f(w) = \left([A + A^T] \, w \right)^T = w^T [A + A^T]} \quad \therefore$$

c) So, from b) we have:

$$\frac{\partial f}{\partial w_1} = 2a_{11} w_1 + a_{12} w_2 + \ldots + a_{1n} w_n + a_{21} w_2 + \ldots + a_{n1} w_n$$

$$\frac{\partial f}{w_2} = a_{21} w_1 + 2a_{22} w_2 + \ldots + a_{2n} w_n + a_{12} w_1 + \ldots + a_{n2} w_n$$

$$\frac{\partial f}{w_n} = a_{n1} w_1 + a_{n2} w_2 + \ldots + 2a_{nn} w_n + a_{1n} w_1 + a_{2n} w_2 + a_{n-1n} w_{n}$$

Then:

$$\frac{\partial^2 f}{\partial w_1^2} = 2a_{11} \Bigg\} \frac{\partial^2 f}{\partial w_1 \partial w_2} = a_{12} + a_{21} \Bigg\} \frac{\partial^2 f}{\partial w_1 \partial w_n} = a_{1n} + a_{n1}$$

$$\frac{\partial^2 f}{\partial w_2^2} = 2a_{22} \Bigg\} \frac{\partial^2 f}{\partial w_2 \partial w_1} = a_{21} + a_{12} \Bigg\} \frac{\partial^2 f}{\partial w_2 \partial w_n} = a_{2n} + a_{n2}$$

$$\frac{\partial^2 f}{\partial w_n^2} = 2a_{nn} \Bigg\} \frac{\partial^2 f}{\partial w_n \partial w_1} = a_{n1} + a_{1n} \Bigg\} \frac{\partial^2 f}{\partial w_n \partial w_2} = a_{n2} + a_{2n}$$

Thus, our $H = \begin{bmatrix} \frac{\partial^2 f}{\partial w_1^2} & \frac{\partial^2 f}{\partial w_1 \partial w_2} & \cdots & \frac{\partial^2 f}{\partial w_1 \partial w_n} \\ \frac{\partial^2 f}{\partial w_2 \partial w_1} & \frac{\partial^2 f}{\partial w_2^2} & \cdots & \frac{\partial^2 f}{\partial w_2 \partial w_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial w_n \partial w_1} & \frac{\partial^2 f}{\partial w_n \partial w_2} & \cdots & \frac{\partial^2 f}{\partial w_n^2} \end{bmatrix}$

$$\boxed{\text{Therefore, } H = \begin{bmatrix} 2a_{11} & a_{12}+a_{21} & \cdots & a_{1n}+a_{n1} \\ a_{21}+a_{12} & 2a_{22} & \cdots & a_{2n}+a_{n2} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1}+a_{1n} & a_{n2}+a_{2n} & \cdots & 2a_{nn} \end{bmatrix} \quad .:}$$

d) So, we have → $\sigma(a) = \frac{1}{1+e^{-a}}$, $f(w) = \ln(\sigma(w^T x))$.

Need to find $\nabla f(w)$.

Let $a = w^T x$, then using the chain rule we compute:

$$\frac{d}{da}\left(\ln(\sigma(a))\right) = \frac{1}{\sigma(a)} * \frac{d}{da}(\sigma(a))$$

Then, $\nabla f(w) = \frac{1}{\sigma(a)} * \frac{d}{da}(\sigma(a)) * \frac{da}{dw}$

$$= \frac{1}{\sigma(a)} * \left[-\frac{1}{(1+e^{-a})^2} * (-a e^{-a})\right] * \frac{da}{dw}$$

$$= \frac{1}{\sigma(a)} * \left[\frac{a e^{-a}}{(1+e^{-a})^2}\right] * \frac{da}{dw}$$

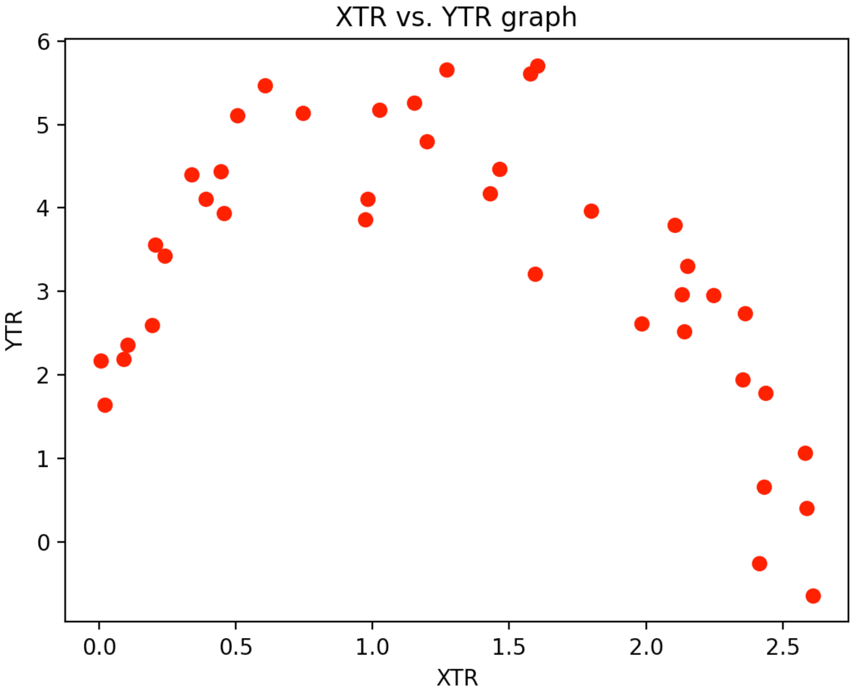
$$= \cancel{(1+e^{-a})} * \left[\frac{a e^{-a}}{(1+e^{-a})^{\cancel{2}}} \right] * \frac{da}{dw}$$

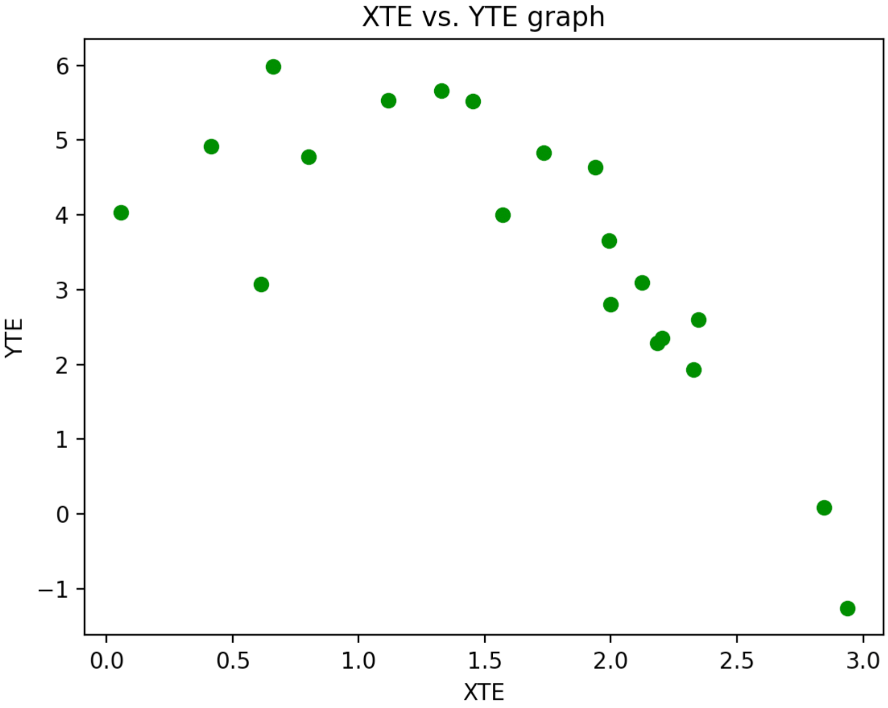
$$= \left[\frac{a e^{-a}}{(1+e^{-a})} \right] * \frac{\partial w^T}{\partial w} \cancel{x}$$

$$\boxed{\nabla f(w) = \frac{w^T x e^{-w^T x}}{1+e^{-w^T x}} x^T} \quad \therefore$$

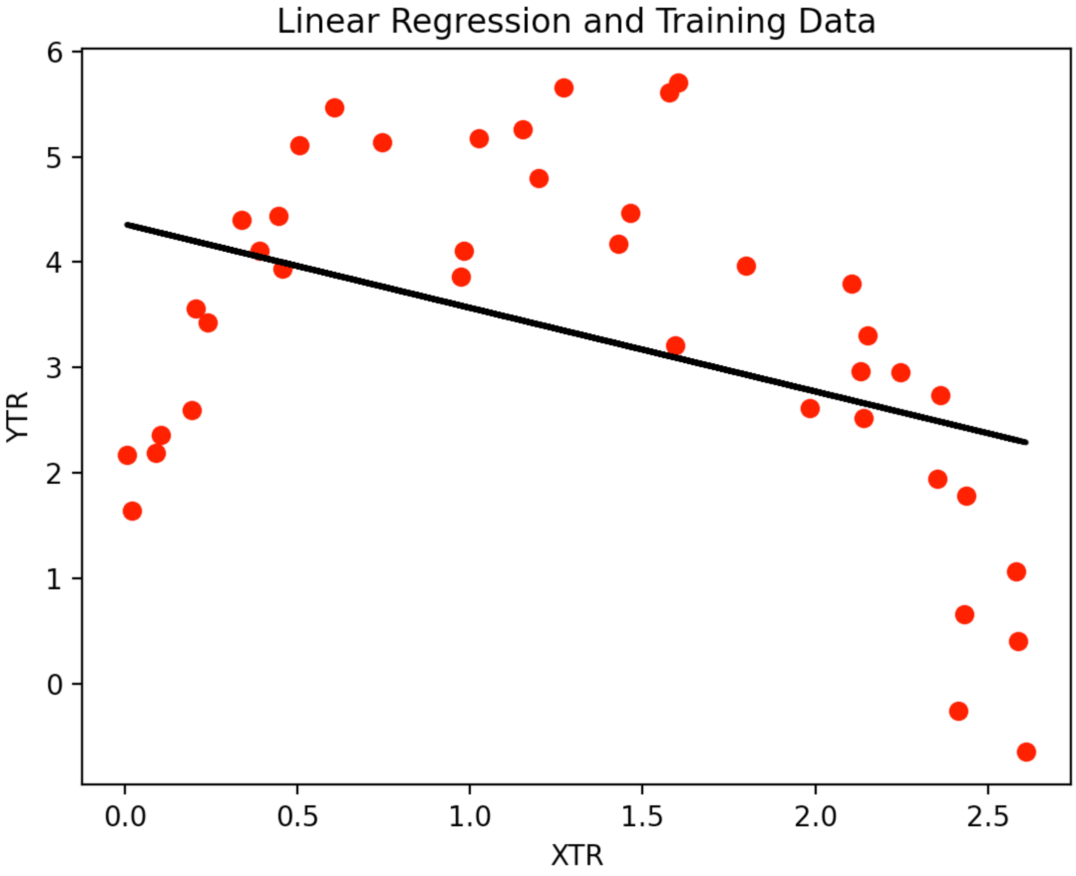
Question 2.

a) Here are the 2 graphs required to be plotted:





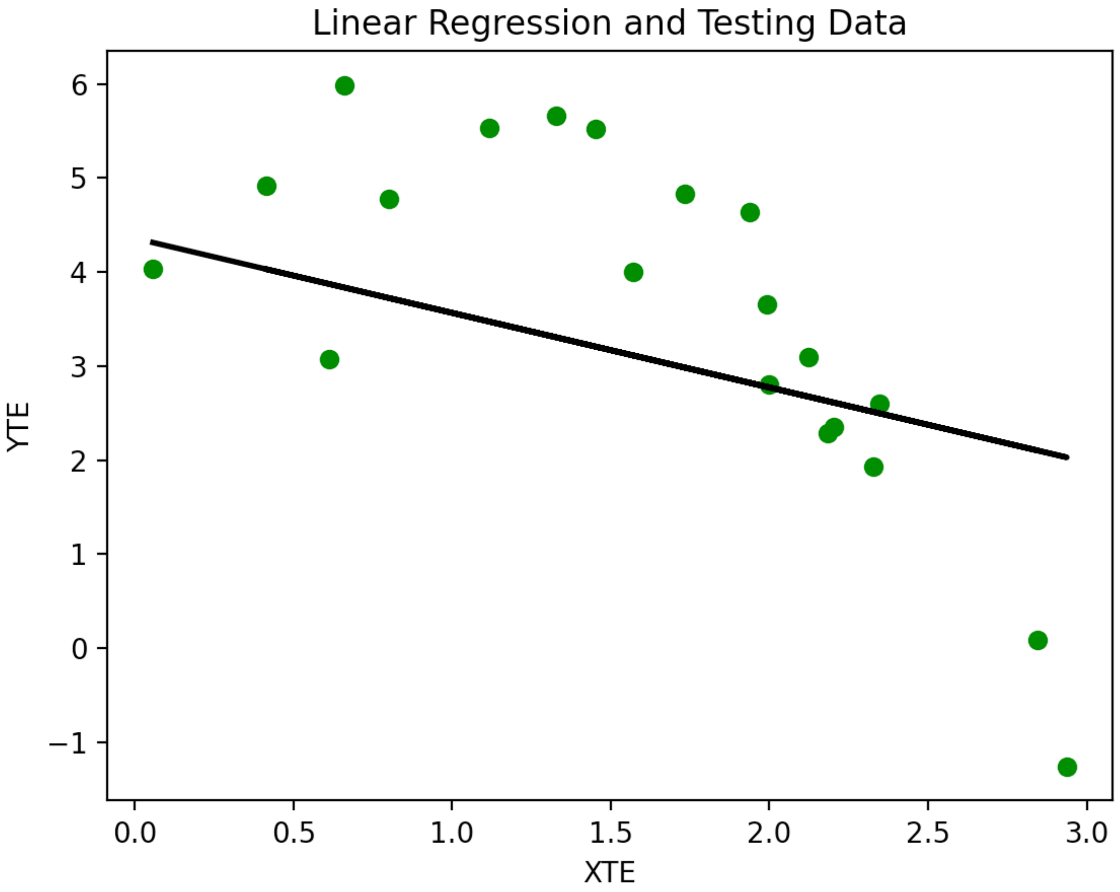
b) Here is the graph of the regression line and the training data:



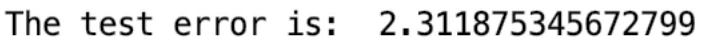
The training error was calculated using the given formula (1) and is shown below:

```
The train error is:  2.1739455790492586
```

c) Here is the graph of the regression line and the test data:



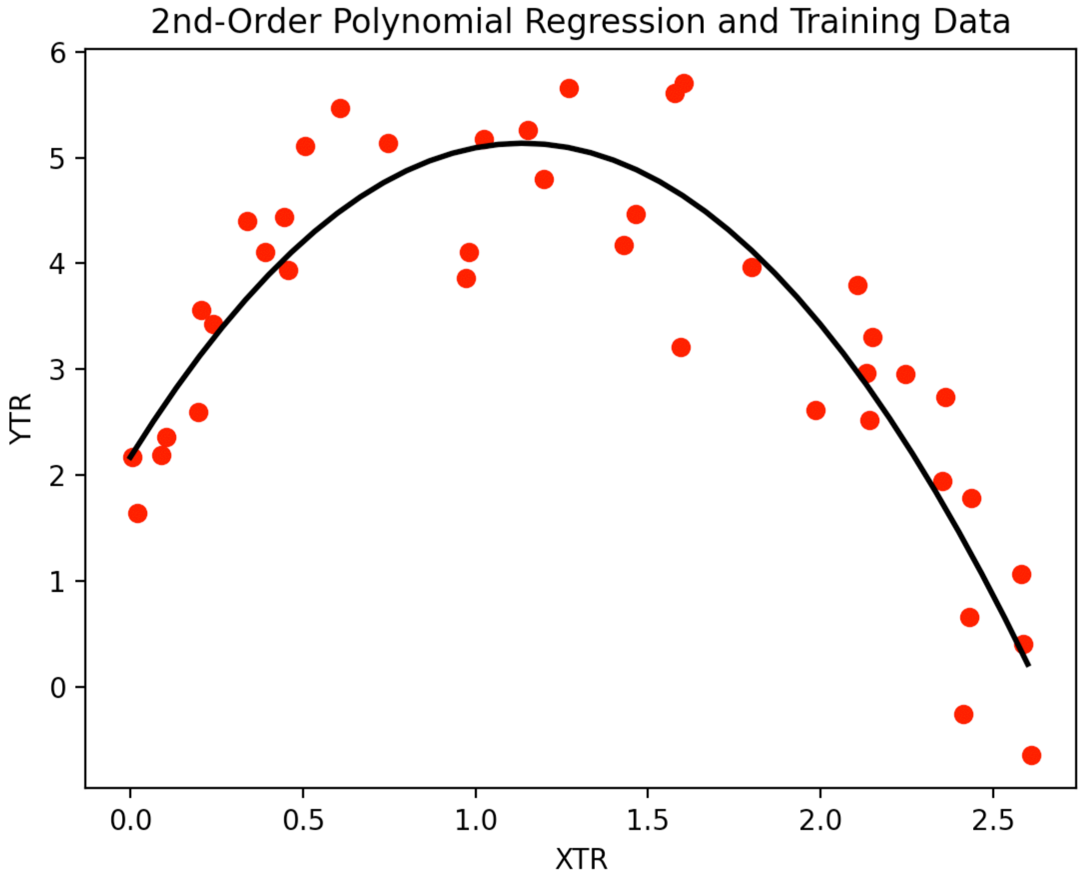
The test error was calculated using the given formula (1) and is shown below:



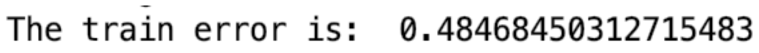
```
The test error is:  2.311875345672799
```

d)

Here is the graph of the second-order polynomial regression and the training data:

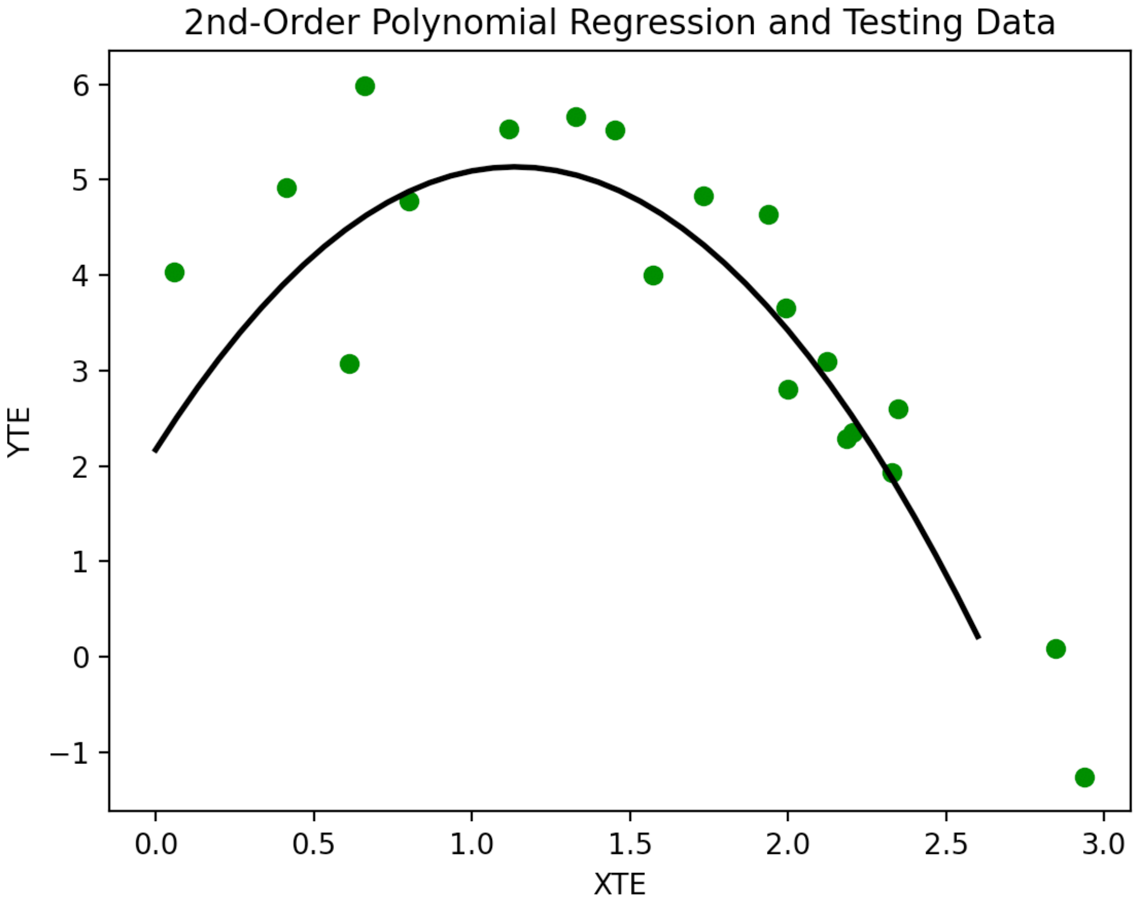


The training error is shown below:

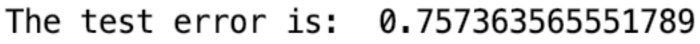


```
The train error is:  0.48468450312715483
```

Here is the graph of the second-order polynomial regression and the test data:



The test error is shown below:

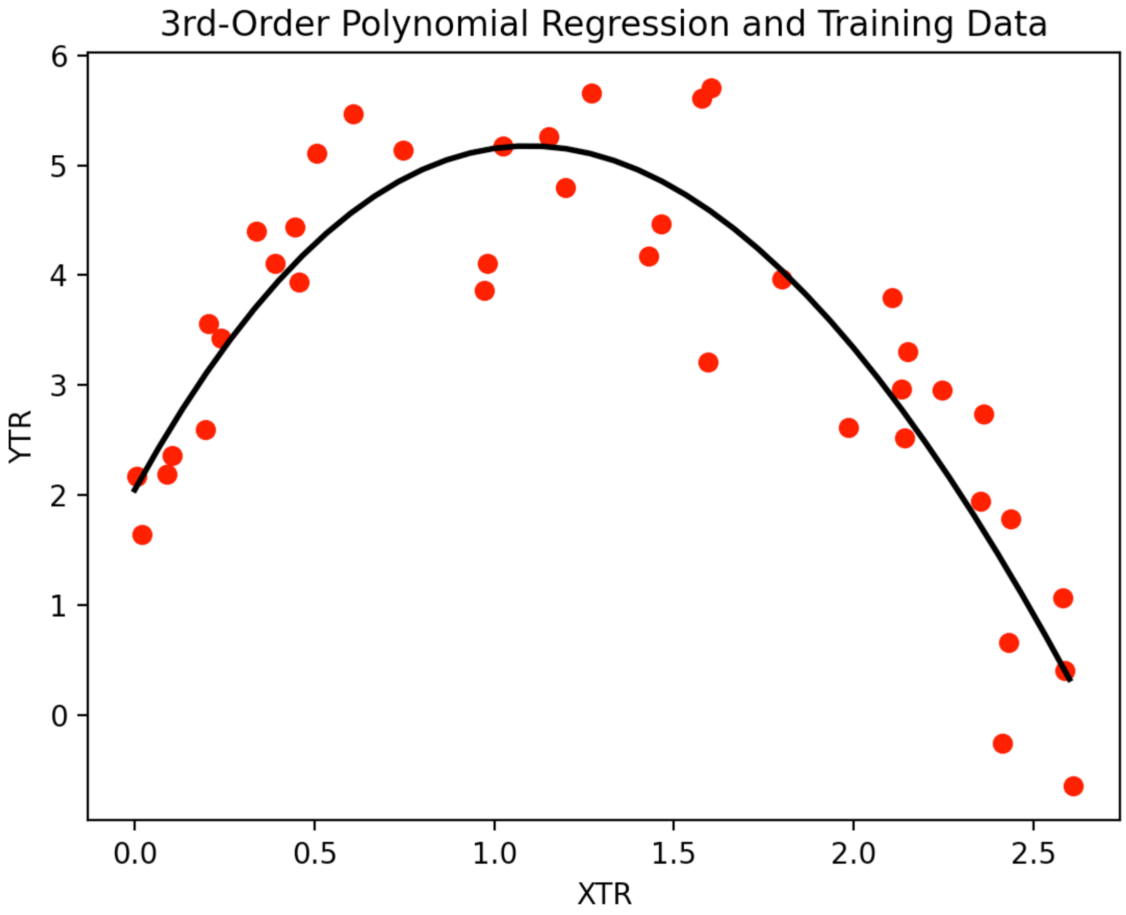


```
The test error is:  0.757363565551789
```

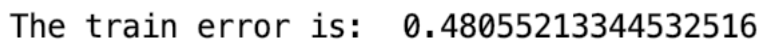
The test error is larger than the train error, which is a usual occurrence. The testing error is quite close to the train error, the difference is 0.27267906. This is usually can be the case when the model is near perfect or the model is underfitted (has a high bias). It is likely that our model is still a little underfitted judging by the errors, however it is definitely a better fit than the linear regression. Linear regression errors are much higher.

e)

Here is the graph of the third-order polynomial regression and the training data:

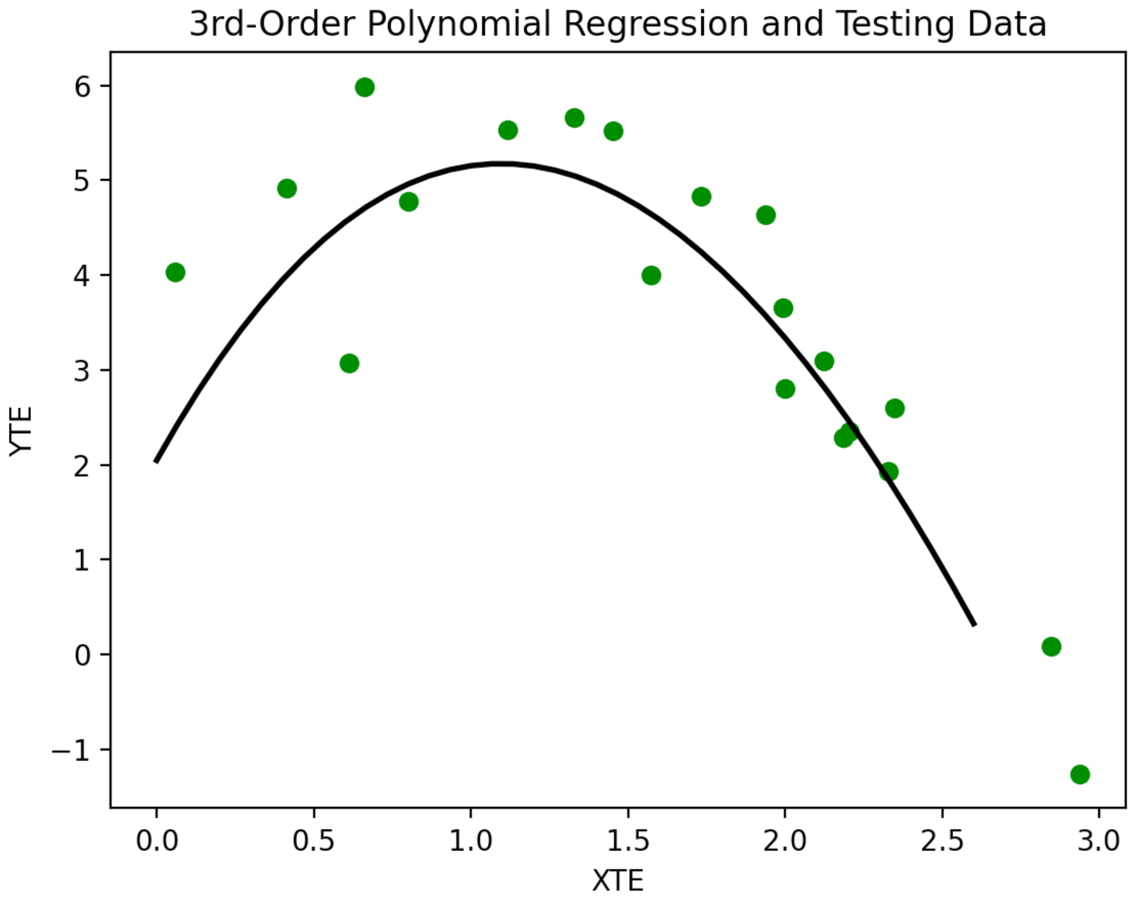


The train error is shown below:

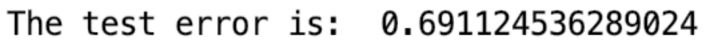


```
The train error is:  0.48055213344532516
```

Here is the graph of the third-order polynomial regression and the test data:



The test error is shown below:

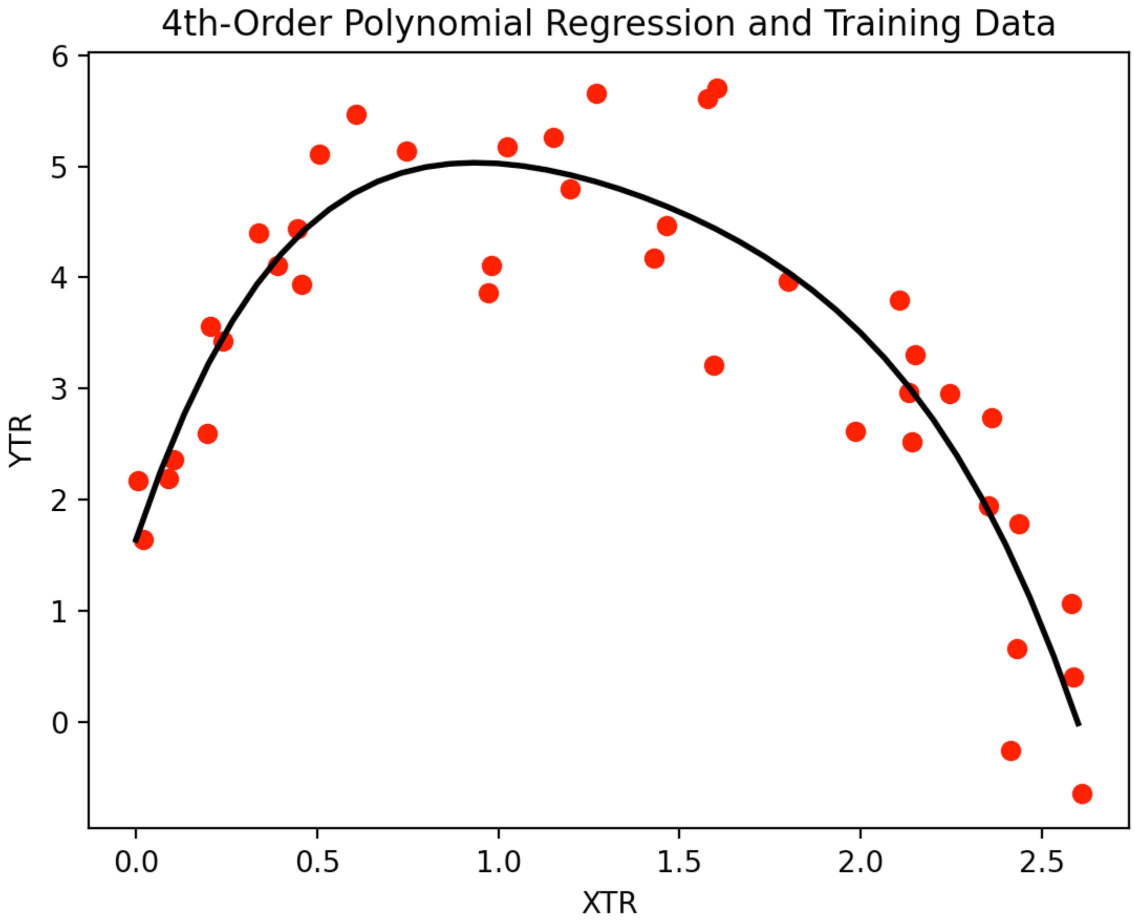


```
The test error is:  0.691124536289024
```

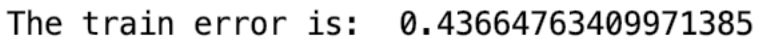
Again, the test error is larger than the train error, which is good and consistent with previous results. The difference between the test error and the training error has shrunk a little, meaning that the test error is now a little closer to the train error. The difference is now 0.2105724. The test and training errors have also decreased compared to the errors from the 2nd-degree polynomial regression. Thus, 3rd-degree polynomial regression is a better fit than the 2nd-degree polynomial regression and obviously better than the linear regression.

f)

Here is the graph of the fourth-order polynomial regression and the training data:

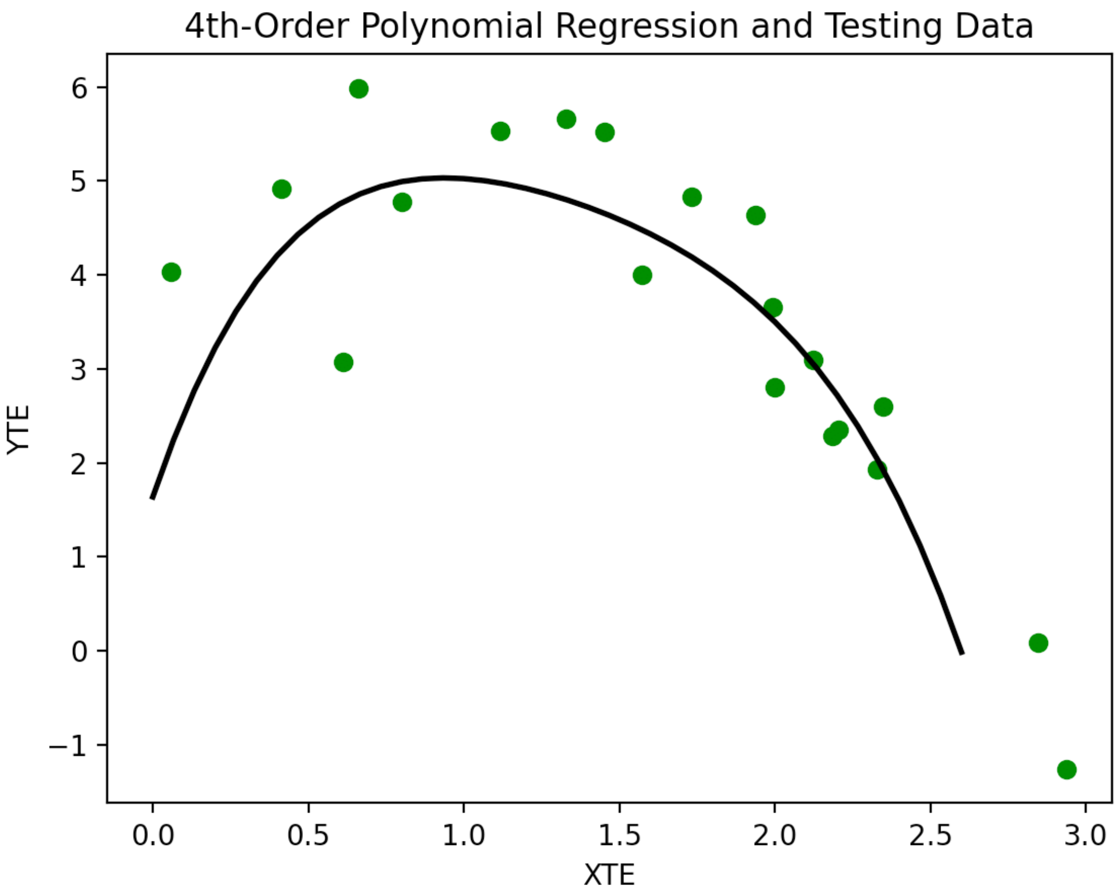


The training error is shown below:

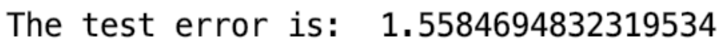


```
The train error is:  0.43664763409971385
```

Here is the graph of the fourth-order polynomial regression and the test data:



The test error is shown below:



```
The test error is:  1.5584694832319534
```

Here, we lose our consistency compared to the previous results. Our training error did decrease, but the test error skyrocketed. The difference between the errors is now 1.12182185. Also, **minimizing the training loss does not indicate a good test performance**. Thus, judging by the test error, our 4th-order polynomial regression model is not a good fit for our test data in this case (cannot generalize to all new data). If we take a look at all the test errors from previous models, we can say that the 3rd-degree polynomial regression had the lowest test error and, therefore, is the best fit for our data. The 4th-order polynomial turns out to be a worse fit than 2nd and 3rd degree polynomials, but better than linear. Therefore, if we rank our models in terms of testing errors, this is what we get:

3rd-order polynomial **>** 2nd-order polynomial **>** 4th-order polynomial **>** linear

where 3rd-order is the best fit and linear is the worst.

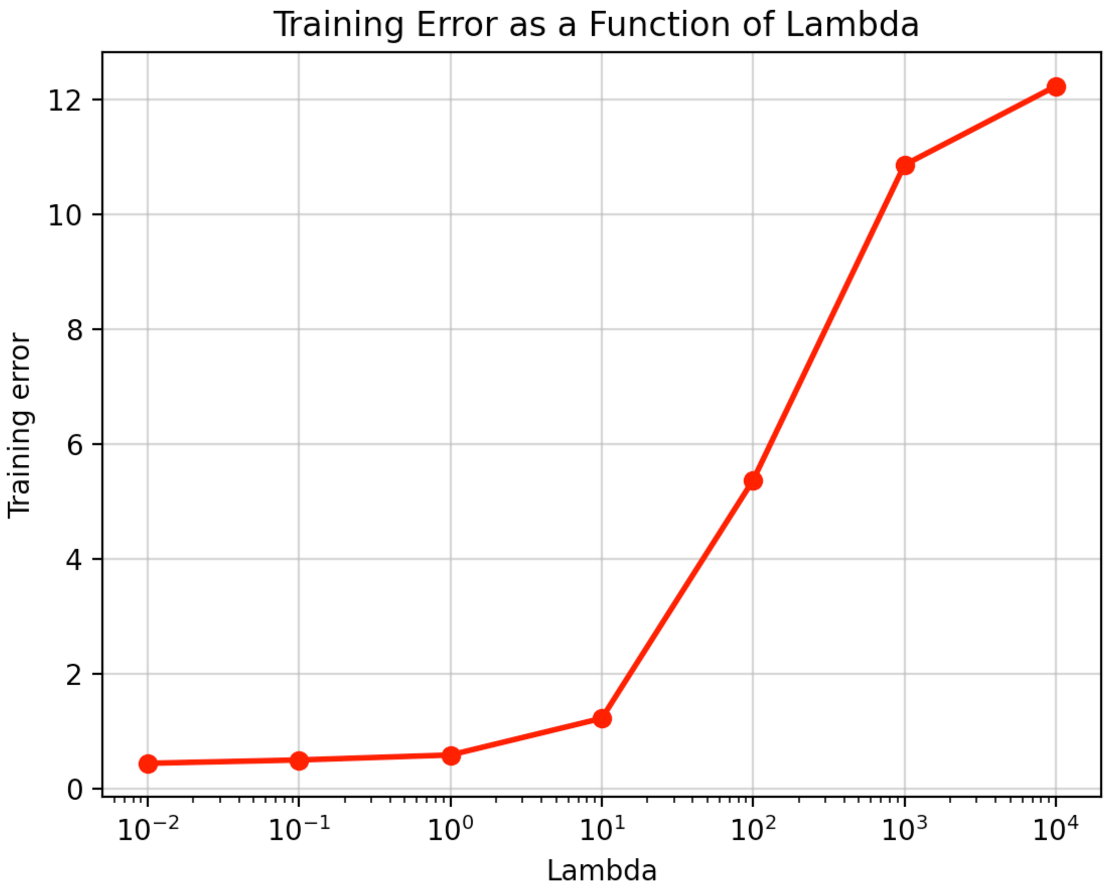
***** On the side note, for the graphs, I could have made the polynomials and lines infinitely continuous (instead of cutting them off a little), but decided to keep them on the interval [0, 2.6] for a better look and the consistency throughout. *****

Question 3.

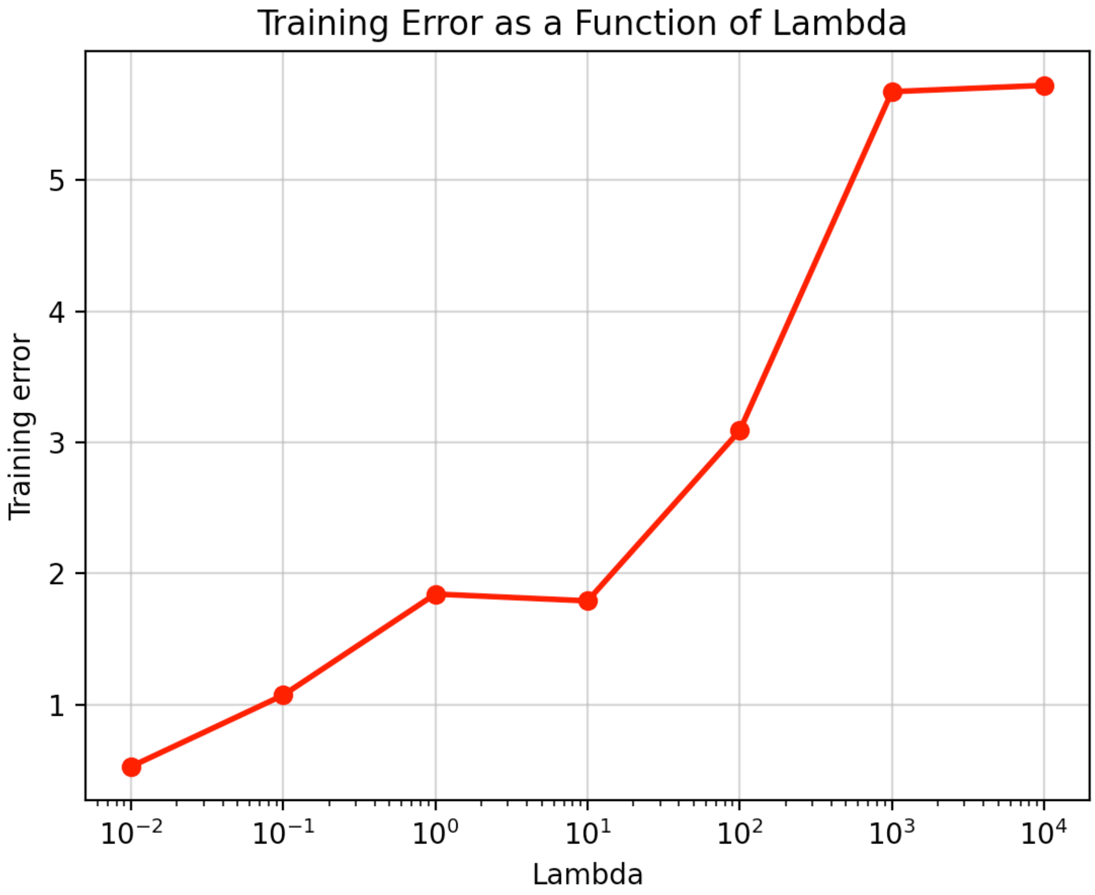
a)

Since the bias term should not be penalized, the way I understand it is we keep the same bias term as we got from our 4th-order polynomial regression, but penalize the rest of the terms. I decided to plot both graphs (with penalized bias term and non-penalized bias term, just in case).

Here is the graph of the training error as a function of λ (here bias term is penalized):

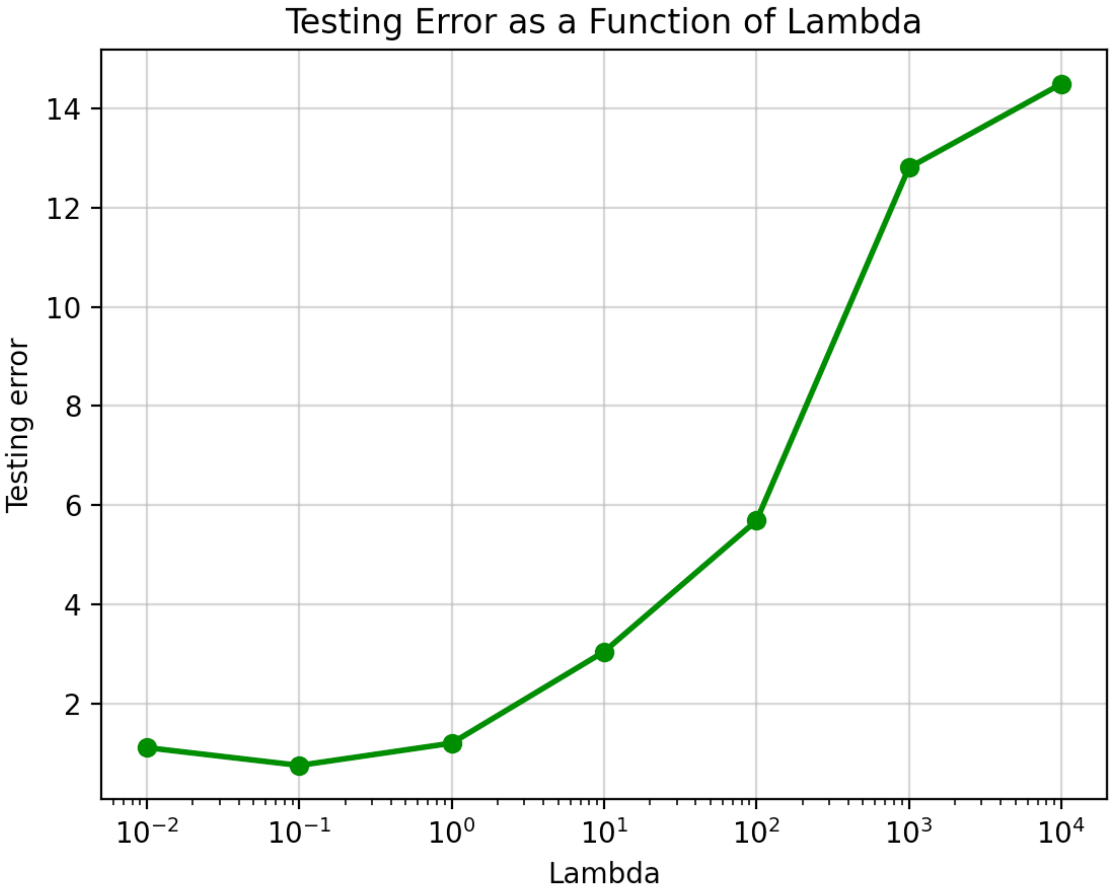


Here is the graph of the training error as a function of λ (here bias term is **not** penalized):

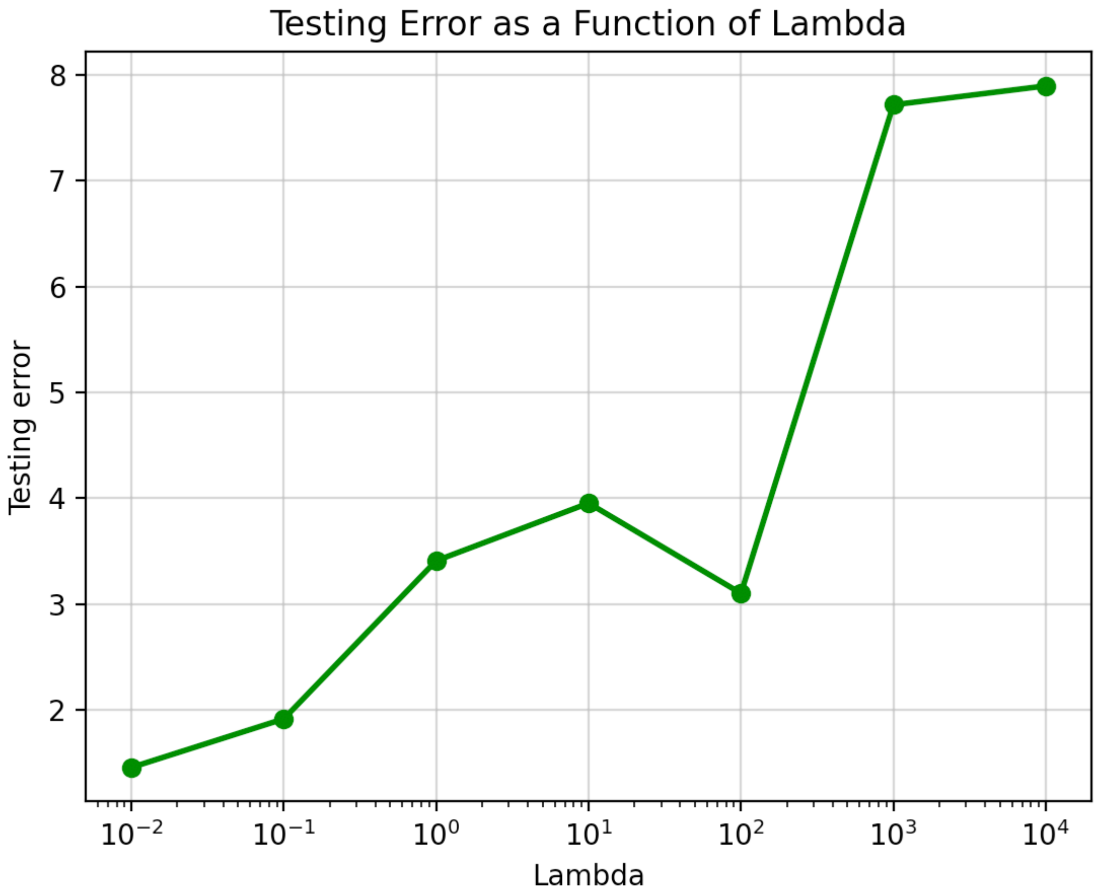


I did the same thing as described above for the testing data.

Here is the graph of the test error as a function of λ (here bias term is penalized):



Here is the graph of the test error as a function of λ (here bias term is **not** penalized):



Judging by the results of the testing errors where the bias term is penalized, λ = 0.1 gave the lowest testing error. But, since we do not penalize the bias term in l_2-regularization, we look at the testing errors where the bias term is **not** penalized, where λ = 0.01 gave the lowest testing error. Thus, λ = 0.01 is the best fit for our training data.

** I did both scenarios due to my interest in what can happen when the bias term is penalized. **

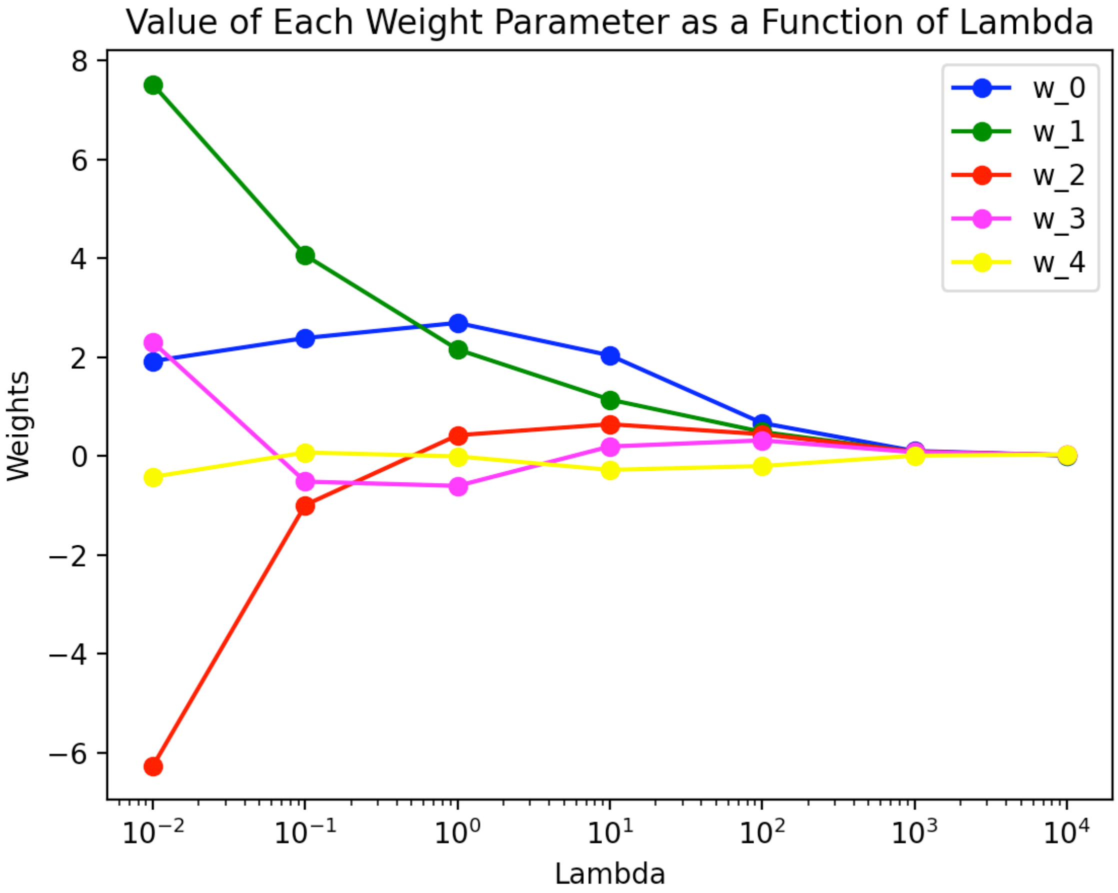
***Also, it does not really matter too much, but I have only made the $\log_{10}$ scale for λ (x-axis) and kept the y-axis as it is (as I assume the assignment was asking to only use the $\log_{10}$ scale for λ). ***

b)

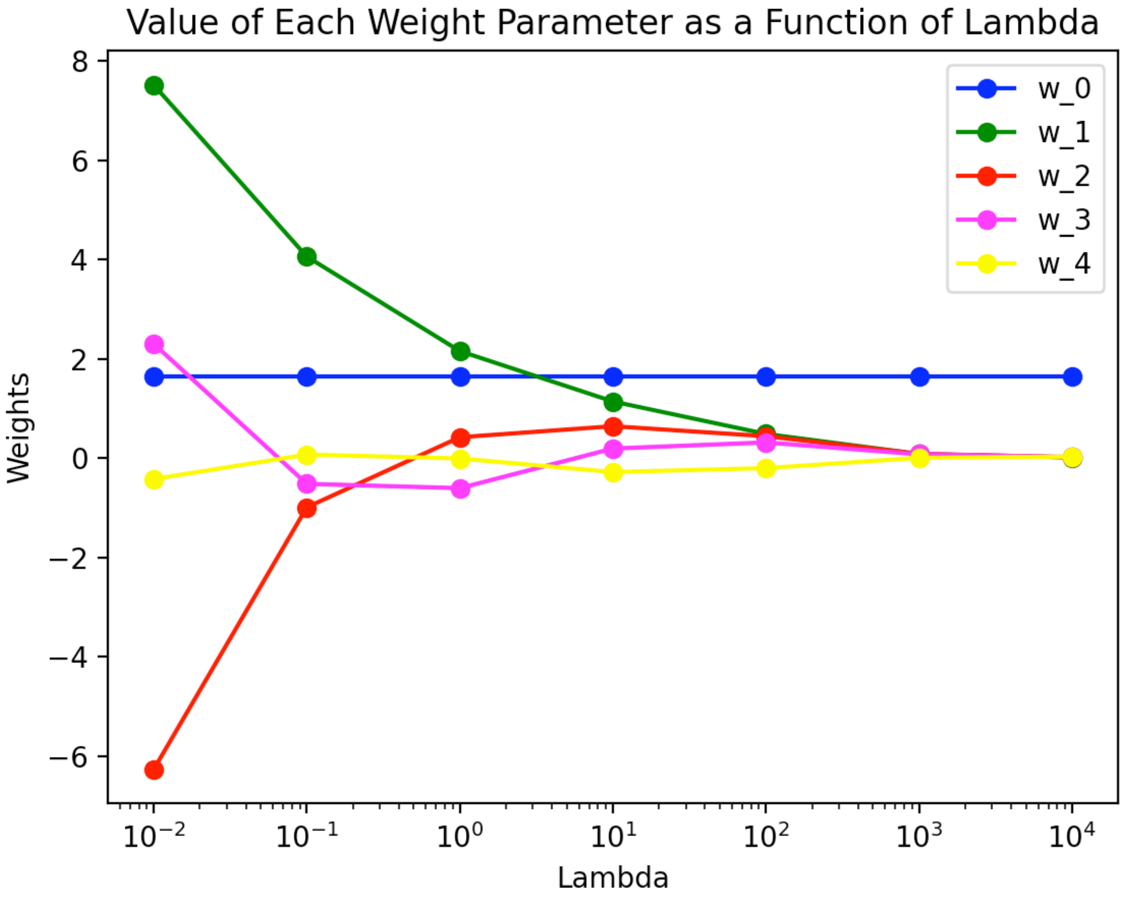
From how I understand the question, we need to plot each of the weight coefficients we got as a function of λ. So, we have seven w_0 coefficients from seven lambdas and we plot them as a function of λ, then we have seven w_1 coefficients from seven lambdas and we plot them as a function of λ, etc.

Same way as before, I will have one graph with penalized and one graph non-penalized bias term.

Here is the graph of the weight coefficients as a function of λ (bias term is penalized):

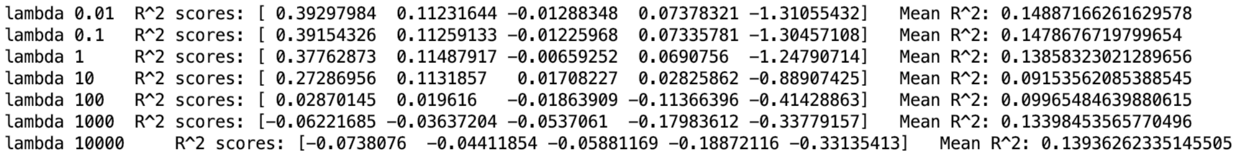


Here is the graph of the weight coefficients as a function of λ (bias term is **not** penalized):



c)

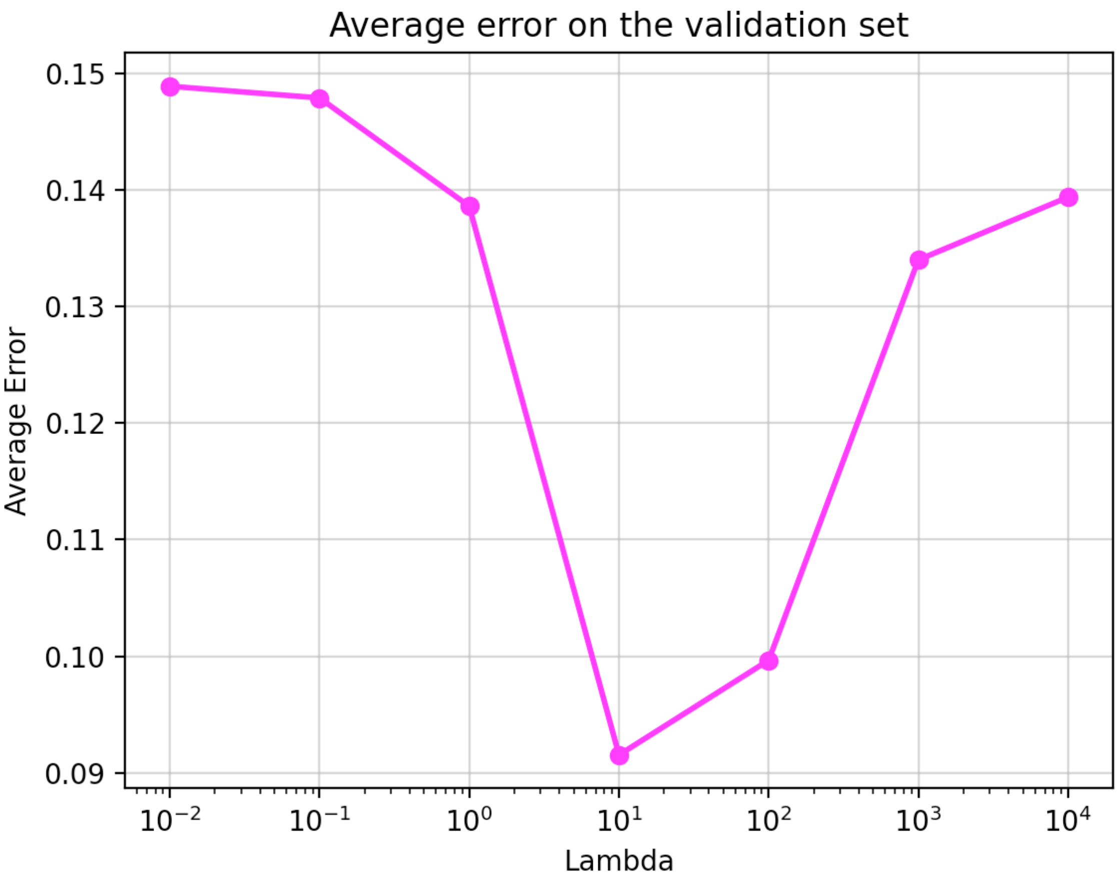
After performing the cross-validation procedure and calculating errors, the following data was obtained:



```
lambda 0.01  R^2 scores: [ 0.39297984  0.11231644 -0.01288348  0.07378321 -1.31055432]   Mean R^2: 0.14887166261629578
lambda 0.1   R^2 scores: [ 0.39154326  0.11259133 -0.01225968  0.07335781 -1.30457108]   Mean R^2: 0.1478676719799654
lambda 1     R^2 scores: [ 0.37762873  0.11487917 -0.00659252  0.0690756  -1.24790714]   Mean R^2: 0.13858323021289656
lambda 10    R^2 scores: [ 0.27286956  0.1131857   0.01708227  0.02825862 -0.88907425]   Mean R^2: 0.09153562085388545
lambda 100   R^2 scores: [ 0.02870145  0.019616   -0.01863909 -0.11366396 -0.41428863]   Mean R^2: 0.09965484639880615
lambda 1000  R^2 scores: [-0.06221685 -0.03637204 -0.0537061  -0.17983612 -0.33779157]   Mean R^2: 0.13398453565770496
lambda 10000     R^2 scores: [-0.0738076  -0.04411854 -0.05881169 -0.18872116 -0.33135413]   Mean R^2: 0.13936262335145505
```

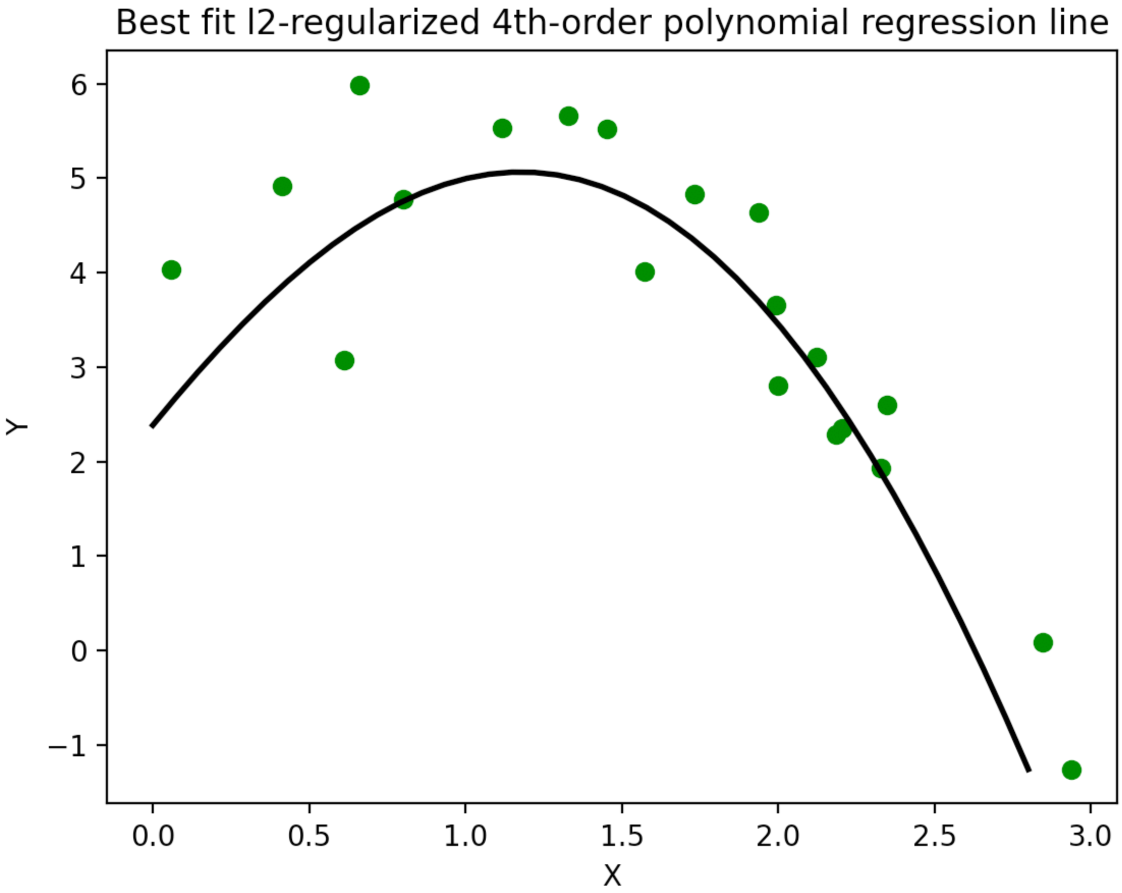
Judging by the above values, it looks like λ=10 is the best value due to having the smallest error from the information above.

Here is the graph of the average error on the validation set as a function of λ:



Thus, the best λ here is different from the λ in part a). They are not same.

Finally, here is the plot of the test data and the l_2-regularized 4-th order polynomial:


Best fit l2-regularized 4th-order polynomial regression line
Y
X