McMaster University
Dept. of Electrical and Computer Engineering
COE 4SL4- Term I (Fall) 2023

Assignment 1

Trade-off between Overfitting and Underfitting

Due Date: September 29, 11:30 pm

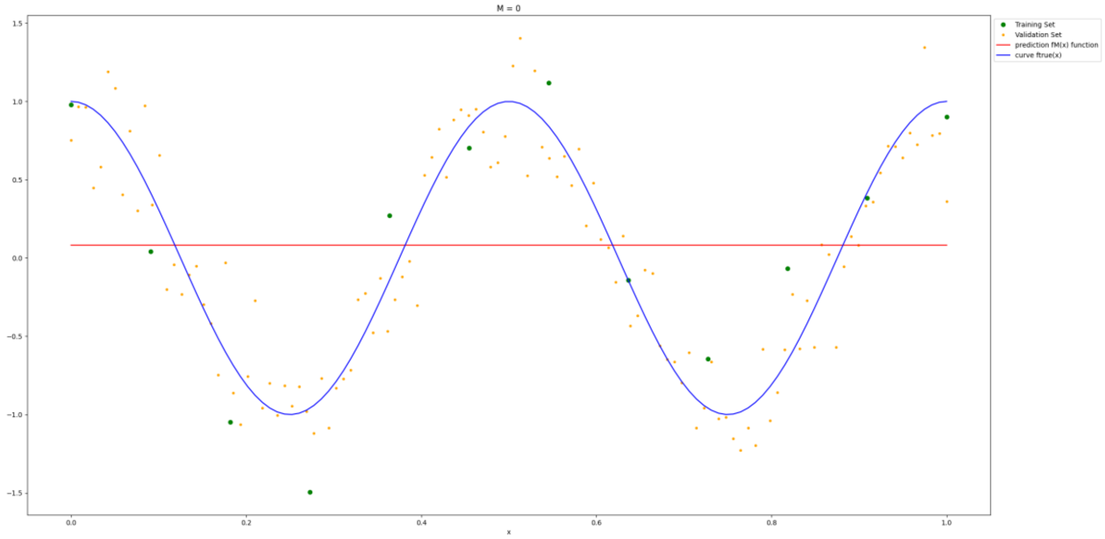
Tianze zhang

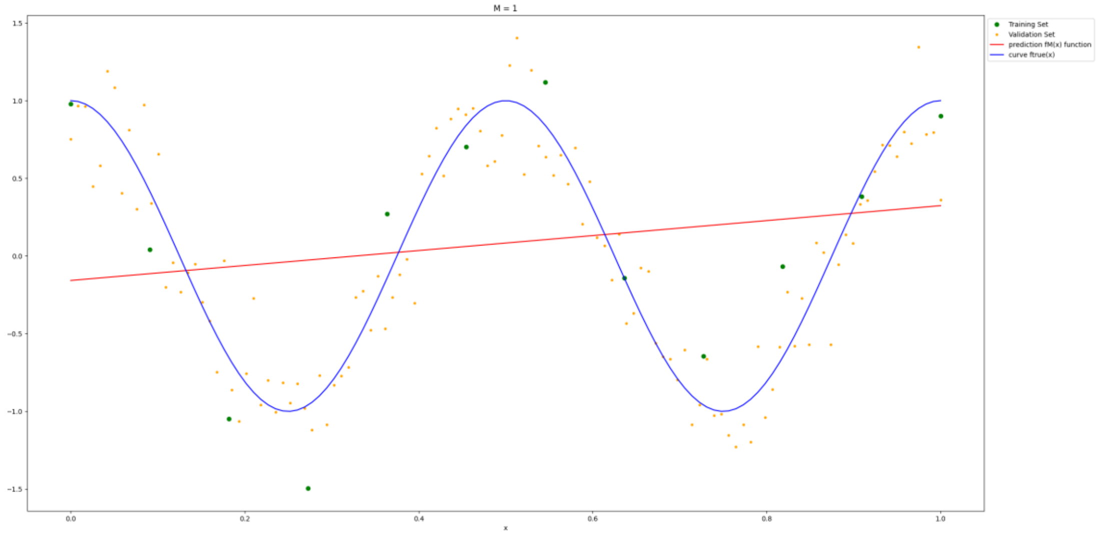
400208135

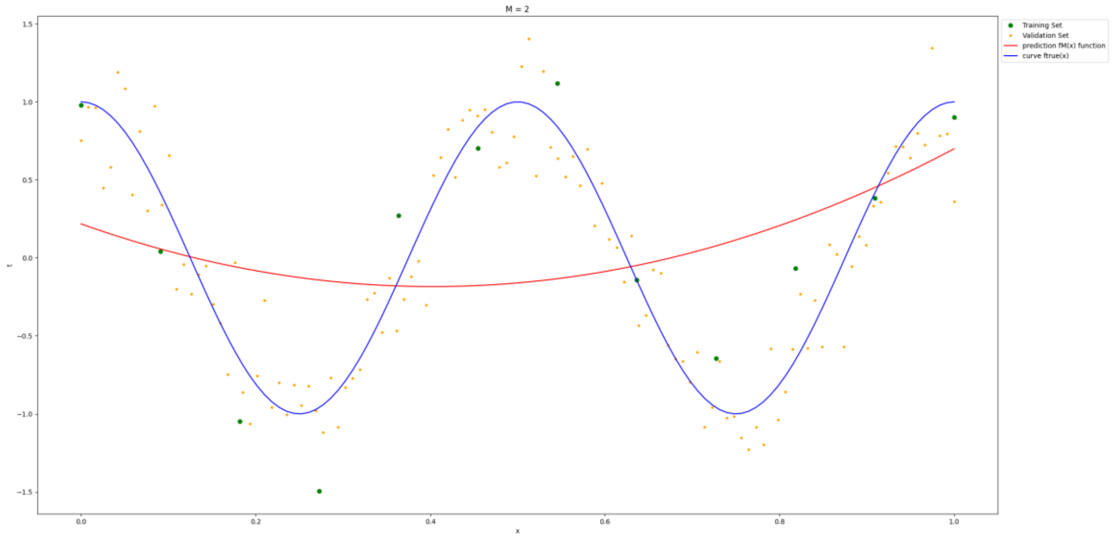
zhant22@mcmaster.ca

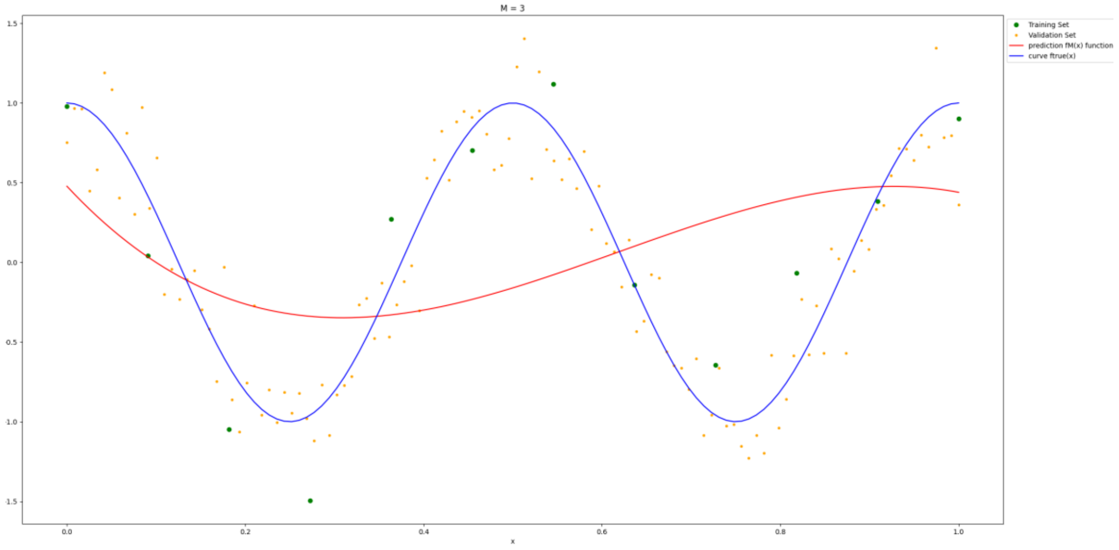
Part 1: adjusting model's capacity reaches

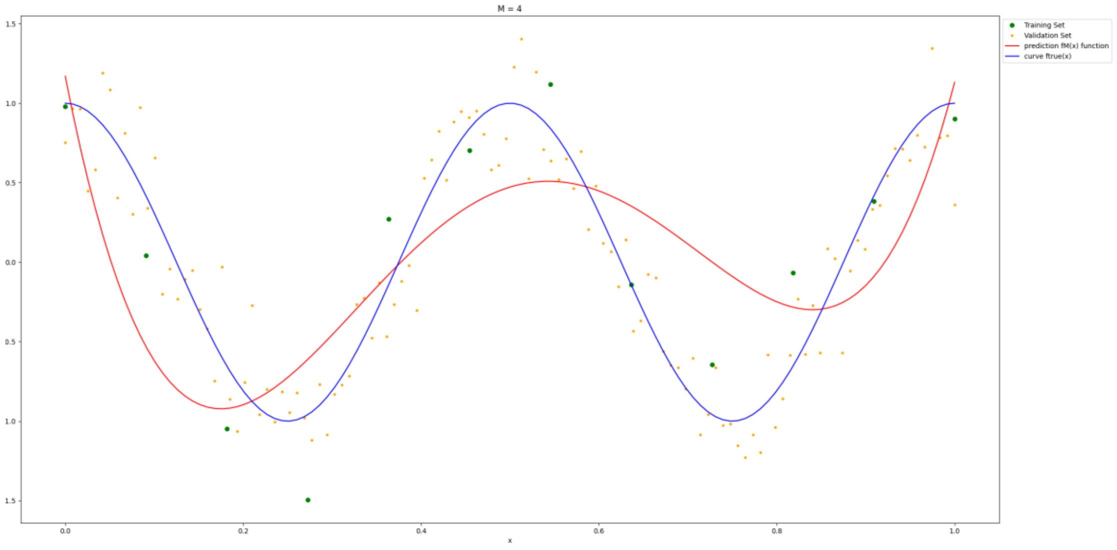
In the first part, I program the code so that it will plot the prediction fM(x) function and the curve f true(x) versus x =[0; 1] for each M. below is the plot:

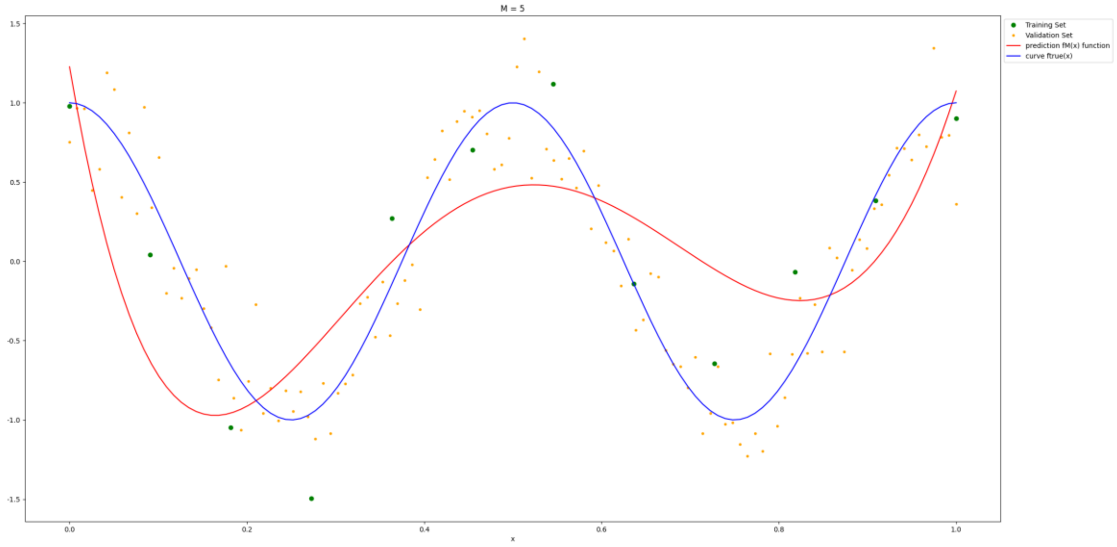


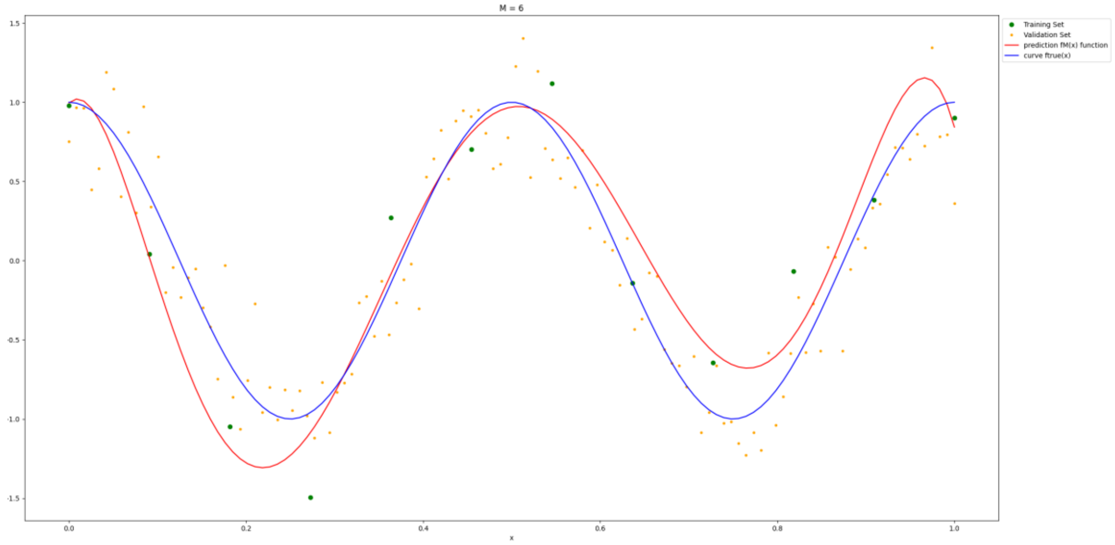


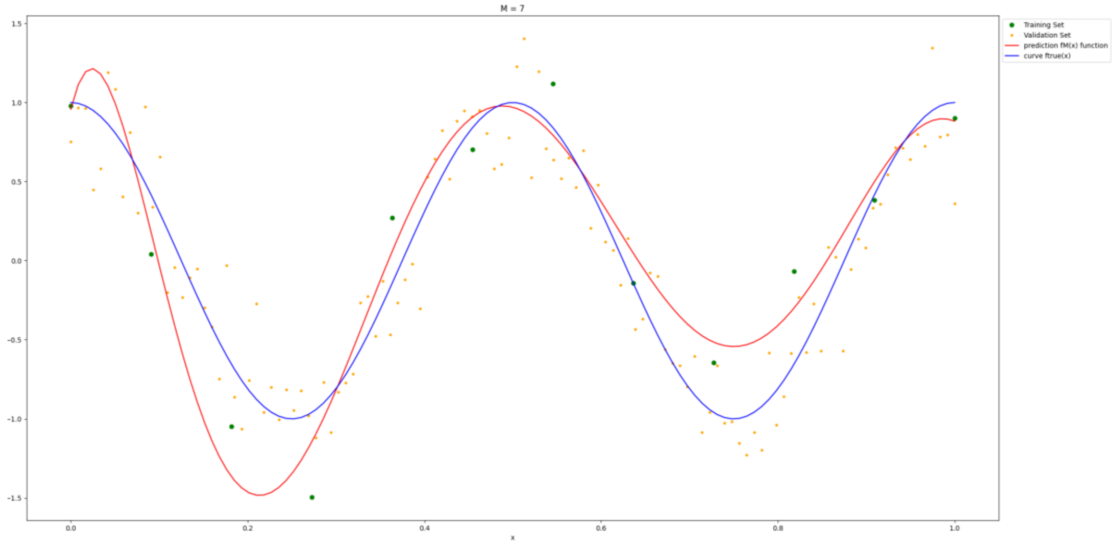

M = 2
Training Set
Validation Set
prediction fM(x) function
curve ftrue(x)
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
t
0.0
0.2
0.4
0.6
0.8
1.0
x

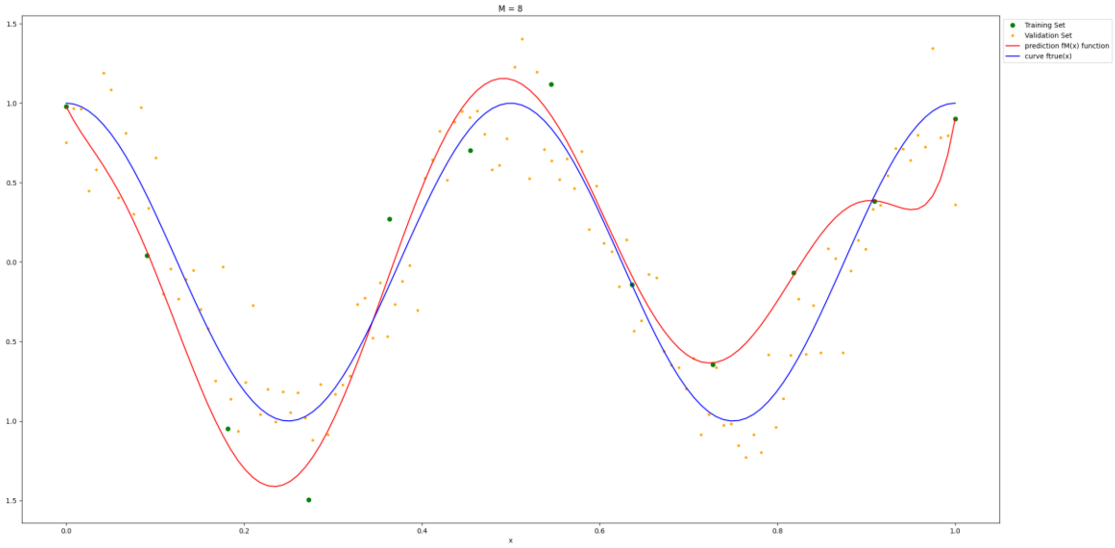

M = 3
Training Set
Validation Set
prediction fM(x) function
curve ftrue(x)
1.5
1.0
0.5
0.0
0.5
1.0
1.5
0.0
0.2
0.4
0.6
0.8
1.0
x


M = 4
Training Set
Validation Set
prediction fM(x) function
curve ftrue(x)
1.5
1.0
0.5
0.0
0.5
1.0
1.5
0.0
0.2
0.4
0.6
0.8
1.0
x

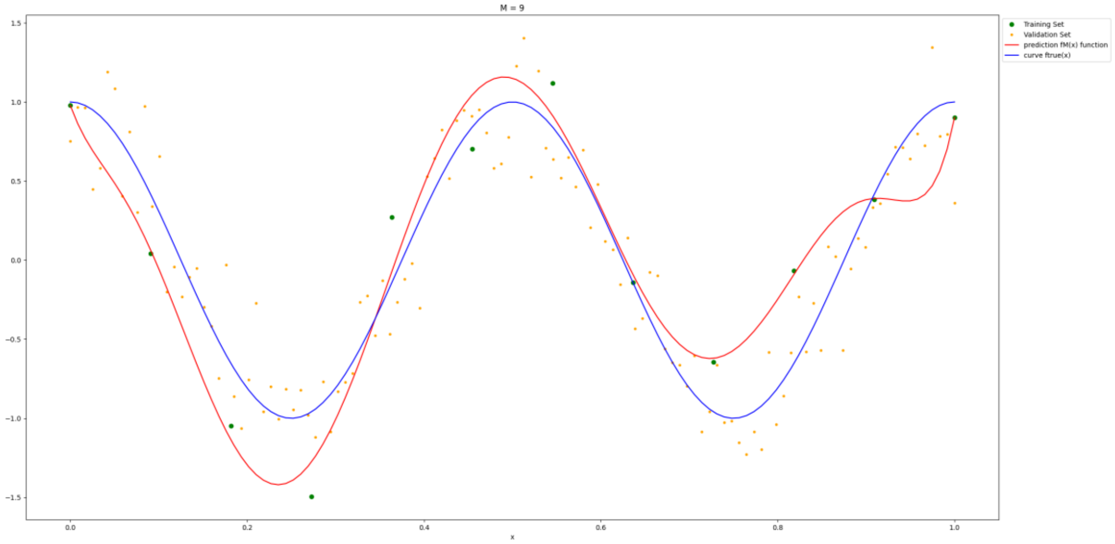

M = 5
1.5
1.0
0.5
0.0
-0.5
-1.0
-1.5
0.0
0.2
0.4
0.6
0.8
1.0
x
Training Set
Validation Set
prediction fM(x) function
curve ftrue(x)


M = 6
1.5
1.0
0.5
0.0
-0.5
-1.0
-1.5
0.0
0.2
0.4
0.6
0.8
1.0
x
Training Set
Validation Set
prediction fM(x) function
curve ftrue(x)

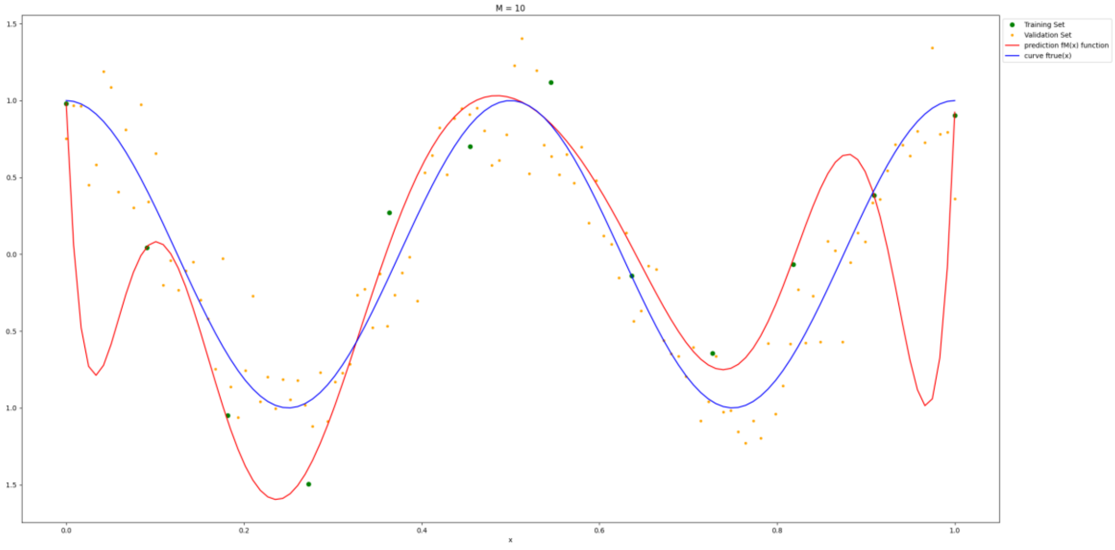

M = 7
1.5
1.0
0.5
0.0
-0.5
-1.0
-1.5
0.0
0.2
0.4
0.6
0.8
1.0
x
Training Set
Validation Set
prediction fM(x) function
curve ftrue(x)


M = 8

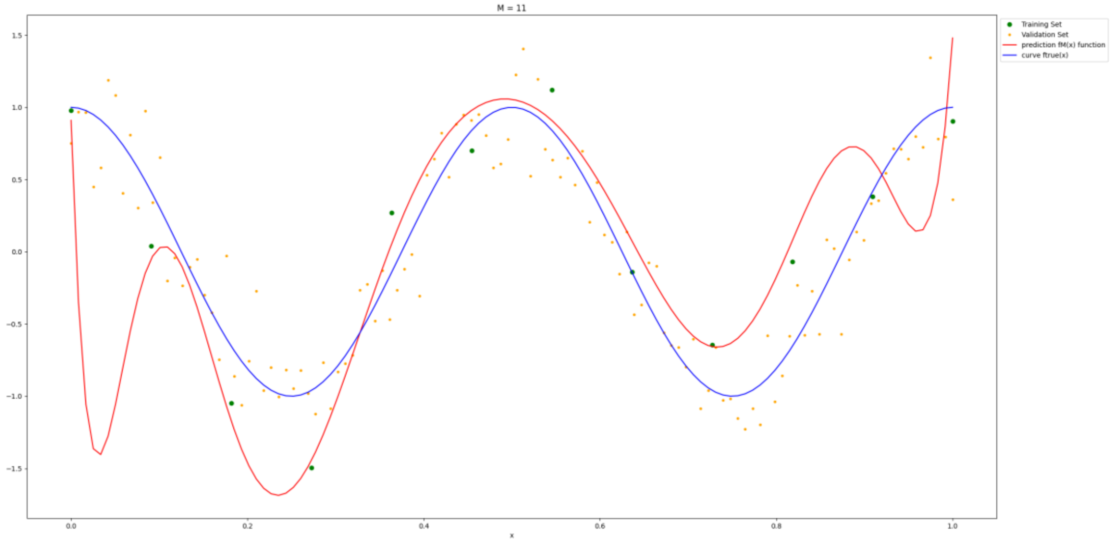
1.5
1.0
0.5
0.0
0.5
1.0
1.5
0.0
0.2
0.4
0.6
0.8
1.0
x
Training Set
Validation Set
prediction fM(x) function
curve ftrue(x)


M = 9

1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
0.0
0.2
0.4
0.6
0.8
1.0
x
Training Set
Validation Set
prediction fM(x) function
curve ftrue(x)


M = 10

1.5
1.0
0.5
0.0
0.5
1.0
1.5
0.0
0.2
0.4
0.6
0.8
1.0
x
Training Set
Validation Set
prediction fM(x) function
curve ftrue(x)

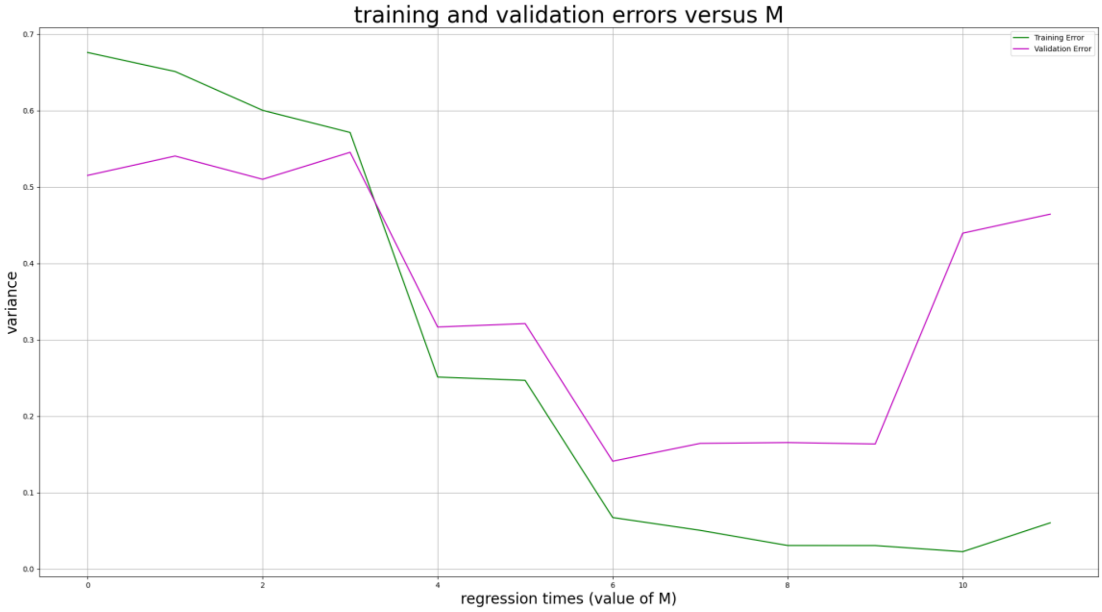


Below is the training error and validation error at each M:

Program generated:

```
when M =    0: training error = 0.6761684568763675, validation error = 0.5152473982095603
when M =    1: training error = 0.6512628234234306, validation error = 0.5406595501975593
when M =    2: training error = 0.6004599838212049, validation error = 0.510161973796316
when M =    3: training error = 0.5714425717855695, validation error = 0.5455176570975521
when M =    4: training error = 0.2513070488176721, validation error = 0.3168098359418833
when M =    5: training error = 0.24693890753668316, validation error = 0.3212819413093336
when M =    6: training error = 0.06746890057912841, validation error = 0.14114313300245454
when M =    7: training error = 0.050640199913564804, validation error = 0.16451691590344245
when M =    8: training error = 0.030856306629806683, validation error = 0.16559281079182497
when M =    9: training error = 0.030791161359958005, validation error = 0.16371810558037805
when M =    10: training error = 0.022729540824933125, validation error = 0.4396318991017979
when M =    11: training error = 0.060510963924473366, validation error = 0.4645856541045858
The lowest training error was when M =    10 and the lowest validation error was when M = 6
```

Training and validation errors versus M plot:



Discussion

Based on the provided plots, it becomes evident that elevating the model's capacity leads to a substantial reduction in the training error. This reduction is in line with our objective of minimizing the disparity between the actual and predicted targets when training against the training dataset.

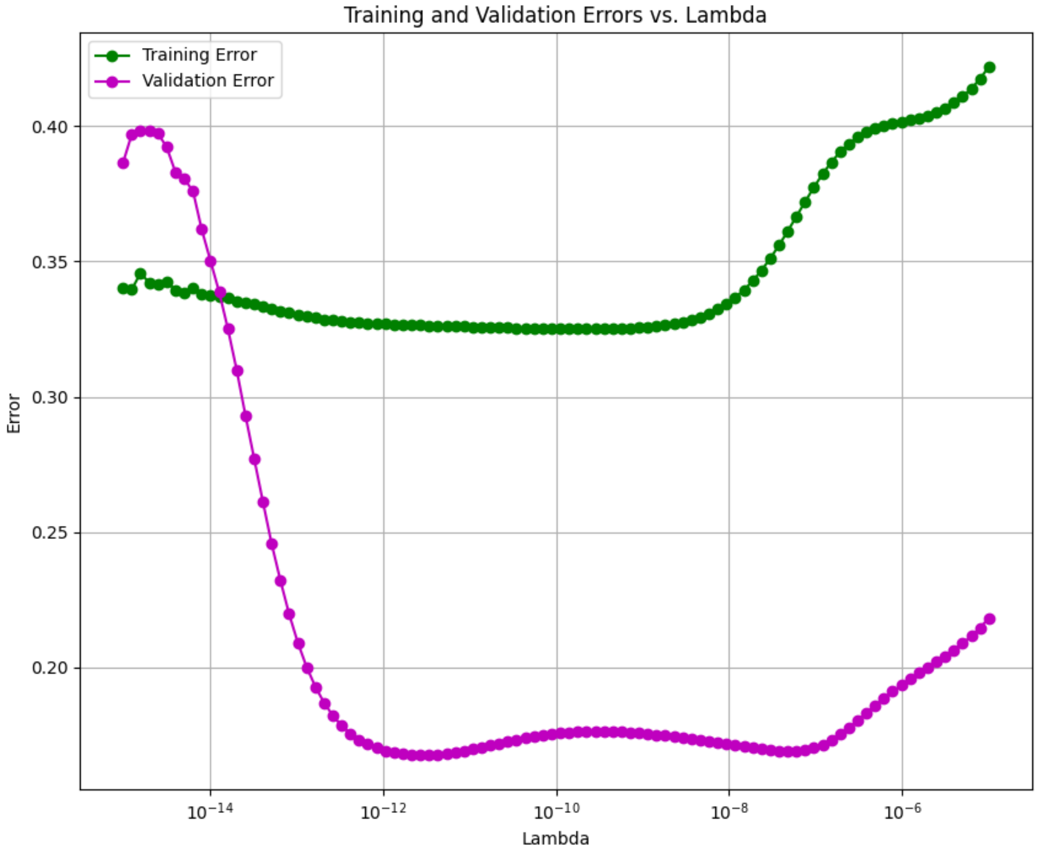
Nonetheless, it is crucial to acknowledge that escalating the model's capacity also introduces the risk of overfitting, which becomes particularly pronounced when the model's capacity reaches 10 (M=10). At this juncture, the model excessively molds itself to the training data, resulting in a curve that closely adheres to all the data points within the training set while maintaining minimal training error.

Part 2: model's capacity reaches stay at 11 adjusting lambda to control over fitting

For the model capacity M=9, we notice a large case of overfitting. This can be controlled using L2 Regularization.

$$\bar{\mathcal{C}}(\mathbf{w}) = \frac{1}{N}\sum_{i=1}^{N}\left[\sum_{k=0}^{D} w_k x_k^{(i)} - t^{(i)}\right]^2 + \lambda\sum_{k=1}^{D} w_k^2 .$$

First, finding the best lambda to control the over fitting:



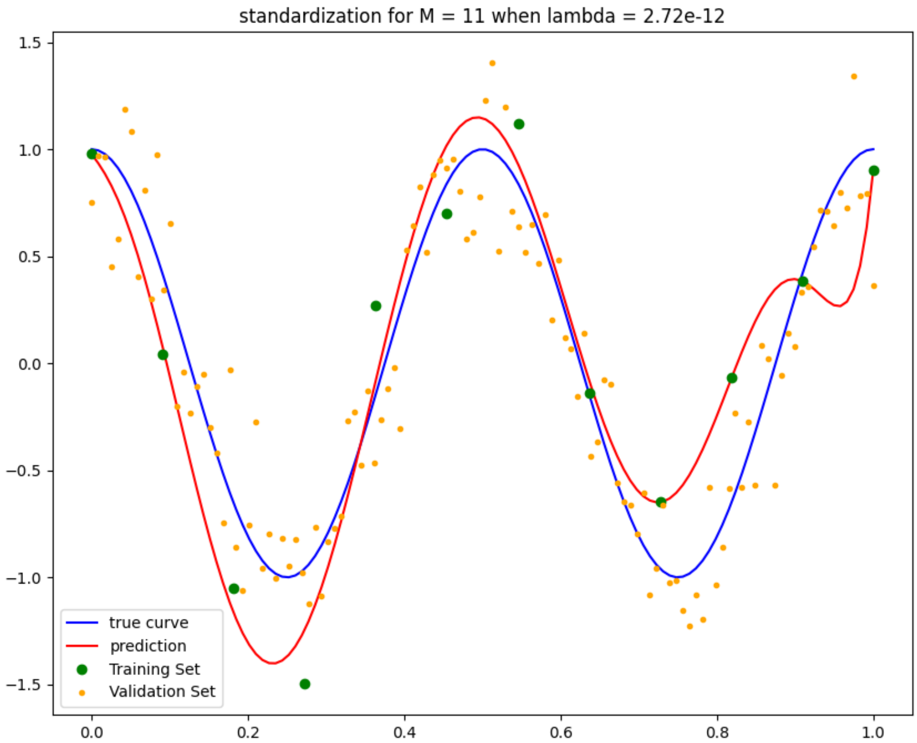
code generated:

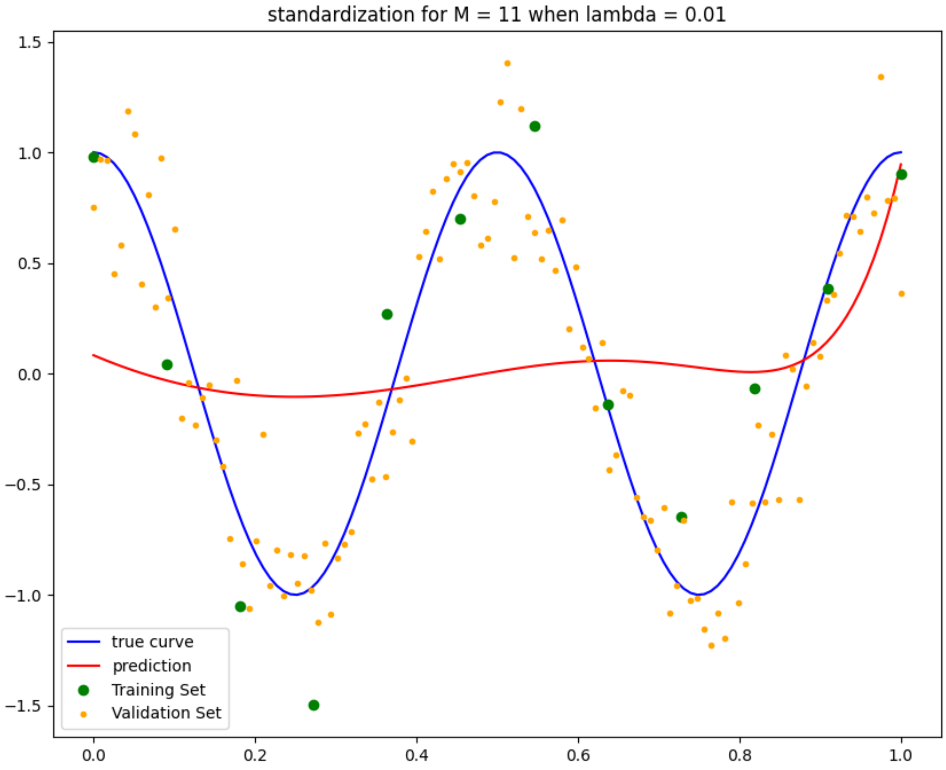
The lambda value with the smallest training error is 1.7886495290574352e-10 the training error is 0.3251050831021694

The lambda value with the smallest validation error is 2.7185882427329403e-12 the validation error is 0.1677275488364268

Discussion:

Above plot and reports show the training and validation error vs a range of lambda value ranging from e^-5 to e^-15. As the report mentioned, if we use lambda equals to 2.72e-12 as the penalties for larger values in the predictor; over fitting is reduced significantly. On the other hand, if we apply lambda equals to 0.01, the module will be under fitting. Plots show below:


standardization for M = 11 when lambda = 2.72e-12
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
0.0
0.2
0.4
0.6
0.8
1.0
true curve
prediction
Training Set
Validation Set


standardization for M = 11 when lambda = 0.01
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
0.0
0.2
0.4
0.6
0.8
1.0
true curve
prediction
Training Set
Validation Set

End