IITM-CS5691 : Pattern Recognition and Machine Learning — Release Date: Nov 1, 2023
Assignment 3 — Due Date: Nov 27, 2023, 23:59
(— Note: Assignment is doable by Nov 13 though!)

Roll No: — Name: SVM Vapnik
Collaborators (if any):
References/sources (if any):

- Use LaTeX to write up your solutions (in the solution blocks of the source LaTeX file of this assignment), submit the resulting rollno.asst2.answers.pdf file at Crowdmark by the due date, and properly drag that pdf's answer pages to the corresponding question in Crowdmark (do this properly, otherwise we won't be able to grade!). (Note: **No late submissions** will be allowed, other than one-day late submission with 10% penalty or four-day late submission with 30% penalty.)
- Please upload to Moodle a rollno.zip file containing three files: rollno.asst2.answers.pdf file mentioned above, and two code files for the programming question (rollno.ipynb file and rollno.py file). Do not forget to upload to Crowdmark your results/answers (including Jupyter notebook **with output**) for the programming question.
- Collaboration is encouraged, but all write-ups must be done individually and independently, and mention your collaborator(s), if any. The same rules apply for codes written for any programming assignments (i.e., write your own code; we will run plagiarism checks on codes).
- If you have referred to a book or any other online material or LLMs (Large Language Models like ChatGPT) for obtaining a solution, please cite the source. Again don't copy the source *as is* - you may use the source to understand the solution, but write up the solution in your own words (this also means that you cannot copy-paste the solution from LLMs!). Please be advised that *the lesser your reliance on online materials or LLMs* for answering the questions, *the more your understanding* of the concepts will be and *the more prepared you will be for the course exams*.
- Points will be awarded based on how clear, concise and rigorous your solutions are, and how correct your answer is. The weightage of this assignment is 12% towards the overall course grade.

1. (6 points) [A DIRECT/DISCRIMINANT APPROACH TO CLASSIFICATION] For the dataset below, we would like to learn a classifier, specifically a discriminant of the form: $\hat{y} = \text{sign}(wx)$ (assume $\text{sign}(u) = +1$ if $u \geqslant 0$, and -1 otherwise).

x	y
-1	-1
1	$+1$
20	$+1$

Let $z_i := z(x_i) := wx_i$. For a training dataset of size n, the parameter w of the classifier can be learnt by minimizing the

(L1) 0-1 loss function aka misclassification error $\sum_{i=1}^{n}(1-\text{sign}(y_i z_i))/2$,

(L2) squared loss function $\sum_{i=1}^{n}(y_i - z_i)^2$, or

(L3) logistic loss function $\sum_{i=1}^{n}\log(1+\exp(-y_i z_i))$.

(a) (2 points) The 0-1 loss function is the most intuitive choice to build a good classifier. What value of w will lead to such a good classifier for this dataset: $w = 0$ or $w = 1$?

Solution: (Solution taken from ma22m026)

Solution: Given data is

x	y
−1	−1
1	+1
20	+1

Let $z_i := z(x_i) := wx_i$. For a training dataset of size n, the parameter w of the classifier can be learnt by minimizing the

(L1) 0-1 loss function aka misclassification error $\sum_{i=1}^{n}(1-\text{sign}(y_i z_i))/2$

Case - 1: For $w = 0$,we get $y_i z_i = 0$ for i = 1,2,3 and so the 0-1 loss becomes

$$\sum_{i=1}^{3}(1-1)/2 = 0$$

as assume $\text{sign}(u) = +1$ if $u \geqslant 0$, and -1 otherwise.
Case - 2: For $w = 1$ we get the 0-1 loss as

$$\sum_{i=1}^{3}(1-\text{sign}(y_i z_i))/2 = (1-1)/2 + (1-1)/2 + (1-1)/2 = 0$$

As we get the same 0-1 loss value for w = 0 andw = 1, but for w = 0 we cannot classify as for each point x we get +1 as a prediction, so the value of w will lead to such a good classifier for this dataset under 0-1 loss is w = 1.

(b) (4 points) Between these values of w ($w = 0$ vs. $w = 1$), determine what value is preferred by the squared and logistic loss functions. Report the actual losses for these w values, and argue which loss function is better.
(Note: Optimizing squared loss is equivalent to applying linear regression methodology to solve this classification problem - did it work fine when there are outliers like $x = 20$ in the dataset?)

Solution: (Solution taken from ma21m025)

Solution: First, evaluate the squared and logistic loss function for $w = 0$ and $w = 1$

(a) Squared Loss Function

$$L = \sum_{i=1}^{n} (y_i - z_i)^2 = \sum_{i=1}^{n} (y_i - wx_i)^2$$

When $w = 0$,

$$\begin{aligned} L &= \sum_{i=1}^{n} (y_i)^2 \\ &= (-1)^2 + (1)^2 + (1)^2 \\ &= 3 \end{aligned}$$

When $w = 1$,

$$\begin{aligned} L &= \sum_{i=1}^{n} (y_i - x_i)^2 \\ &= (-1+1)^2 + (1-1)^2 + (1-20)^2 \\ &= 0 + 0 + (-19)^2 \\ &= 361 \end{aligned}$$

(b) Logistic Loss Function

$$L = \sum_{i=1}^{n} \log(1 + \exp(-y_i z_i)) = \sum_{i=1}^{n} \log(1 + \exp(-y_i w x_i))$$

When $w = 0$,

$$\begin{aligned} L &= \sum_{i=1}^{3} \log(1 + \exp(0)) \\ &= \sum_{i=1}^{3} \log(2) \\ &= 3 \log 2 \\ &= 0.903 \end{aligned}$$

When $w = 1$,

$$\begin{aligned} L &= \sum_{i=1}^{3} \log(1 + \exp(-y_i x_i)) \\ &= \log(1 + \exp(-1)) + \log(1 + \exp(-1)) + \log(1 + \exp(-20)) \\ &= 0.136 + 0.136 + 0 \\ &= 0.2720 \end{aligned}$$

Logistic Loss function performed well here. Since, Squared Loss function is very sensitive towards outliers, outliers can have a significant impact on the overall loss leading to a model that is heavily influenced by these outliers. Logistic loss function is better suited to this task since it is less affected by outliers as it focuses on log-likelihood of the predicted probabilities. The logistic loss function is better suited for the probabilistic interpretation of logistic regression and provides better robustness against outliers compared to mean square error.

2. (6 points) [THINKING LOGISTIC-ALLY...] Consider the scenario in which a user maintains a dataset consisting of songs that he has downloaded over a period of time. He also tracks the likes (-1)/dislikes(+1) for each song along with a set of features X_1, X_2. X_1 is a binary variable that takes value 1 if the song is sung by his favorite singer, and X_2 corresponds to song duration in minutes. This dataset with 10 data points is given below:

$$[X_1\ X_2] = \begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 5 & 10 & 13 & 2 & 3 & 5 & 2 & 10 & 10 & 3 \end{bmatrix}^T$$

$$y = \begin{bmatrix} -1 & -1 & -1 & -1 & -1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}^T$$

   (a) (3 points) Train a logistic regression model on this dataset by performing the gradient descent algorithm steps by setting the initial weights to 0. No bias (intercept term) is required and the step size $\eta = 1$. Report the updated weights at the end of two iterations.

Solution: (Solution taken from cs23m052)

You are currently grading Q2a

Solution: $[X_1\ X_2] = \begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 5 & 10 & 13 & 2 & 3 & 5 & 2 & 10 & 10 & 3 \end{bmatrix}^T$

$y = \begin{bmatrix} -1 & -1 & -1 & -1 & -1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}^T$

We know that the logistic loss can be written as,

logistic loss $= \sum_{i=1}^{n} \log(1 + \exp(-y_i z_i))$

And gradient wrt w is,

$\Delta R(w) = \sum_{i=1}^{n} \frac{\exp(-y_i z_i)}{1+\exp(-y_i z_i)}(-y_i x_i)$

Initial w = $[0\ 0]^T$.This makes $y_i z_i = 0$

Therefore,$\Delta R^I(w) = 1/2 * \sum_{i=1}^{n} -y_i x_i = 0.5 * (\begin{bmatrix} -1 \\ -5 \end{bmatrix} + \begin{bmatrix} 0 \\ -10 \end{bmatrix} + \begin{bmatrix} -1 \\ -13 \end{bmatrix} + \begin{bmatrix} 0 \\ -2 \end{bmatrix} + \begin{bmatrix} -1 \\ -3 \end{bmatrix} + \begin{bmatrix} 0 \\ 5 \end{bmatrix} + \begin{bmatrix} 0 \\ 2 \end{bmatrix} + \begin{bmatrix} 1 \\ 10 \end{bmatrix} + \begin{bmatrix} 0 \\ 10 \end{bmatrix} + \begin{bmatrix} 0 \\ 3 \end{bmatrix}) = \begin{bmatrix} 1 \\ 1.5 \end{bmatrix}$ at the first iteration.

This makes $w^I = w - \eta \Delta R^I(w) = \begin{bmatrix} 0 \\ 0 \end{bmatrix} - \begin{bmatrix} 1 \\ 1.5 \end{bmatrix} = \begin{bmatrix} -1 \\ -1.5 \end{bmatrix}$ Now, we have to again calculation the logistic gradient for second iteration with the new w^I.

$\Delta R^{II}(w) = \sum_{i=1}^{n} \frac{\exp(-y_i w^{II} * x_i)}{1+\exp(-y_i w^{II} * x_i)}(-y_i x_i) = \begin{bmatrix} -0.995726 \\ -29.76133 \end{bmatrix}$

Calculation is done using python. Link for the python code is given below.

```
import numpy as np
x = np.array([[1,5],[0,10],[1,13],[0,2],[1,5],[0,5],[0,2],[1,10],[0,10],[0,3]])
y = np.array([-1,-1,-1,-1,-1,1,1,1,1,1])
w = np.array([0,0])
R = np.array([0,0])
R1 =R
grad2 =R
grad1 = R
for i in range(10):
    R = (np.exp(-y[i]*w@x[i])/(1+ np.exp(-y[i]*w@x[i])))*-y[i]*x[i]
    grad1 = grad1 + R
print(grad1)
w1 = w-grad1
print(w1)
for i in range(10):
  R1 = (np.exp(-y[i]*w1@x[i])/(1+ np.exp(-y[i]*w1@x[i])))*-y[i]*x[i]
  grad2 = grad2 + R1
print(grad2)
w2 = w1-grad2
print(w2)
```

```
[1.   1.5]
[-1.   -1.5]
[ -0.99572852 -29.76133698]
[-4.27367848e-03  2.82613370e+01]
```

Now

$w^{II} = w^I - \eta \Delta R^{II}(w) = \begin{bmatrix} -1 \\ -1.5 \end{bmatrix} - \begin{bmatrix} -0.995726 \\ -29.76133 \end{bmatrix} = \begin{bmatrix} -0.004273 \\ 28.26133 \end{bmatrix}$

(b) (1 point) What will be the prediction for a new song with features [0, 20] using the trained logistic regression model?

Solution: (Solution taken from ma21m025)

Solution: Prediction with feature $[0, 20]$. Here $w_0 = -0.00427, w_1 = 28.26133$

$$\begin{aligned} y &= w_0 x_1 + w_2 x_2 \\ &= -0.00427x0 + 28.26133x20 \\ &= 565.22 \end{aligned}$$

Then sigmoid function = $\frac{1}{1+\exp(-565.22)} > 0.5$ Hence, prediction is 1.

(c) (2 points) Discuss if logistic regression is a good choice for addressing this specific problem. If not, what are other better options?

Solution: If we ignore the datapoints (in (x_1, x_2, y) format): $(0, 2, +1), (0, 2, -1)$, and $(0, 10, +1), (0, 10, -1)$ which have inherent noise to prevent them from being classified correctly by any model, the data is almost linearly separable and a linear model like logistic regression should do well. As more datapoints are gathered in the training dataset and if we observe a non-linear decision boundary is required, then other non-linear regression models like kernel SVM or ANN can be tried out.

3. (12 points) [SVM'S TO THE RESCUE] A Gaussian or Radial Basis Function (RBF) kernel with inverse width $k > 0$ is

$$K(u, v) = e^{-k||u-v||^2}.$$

The below figures show decision boundaries and margins for SVMs learned on the exact same dataset. The parameters used for the different runs are as follows:

(i) Linear Kernel with C = 1

(ii) Linear Kernel with C = 10

(iii) Linear Kernel with C = 0.1

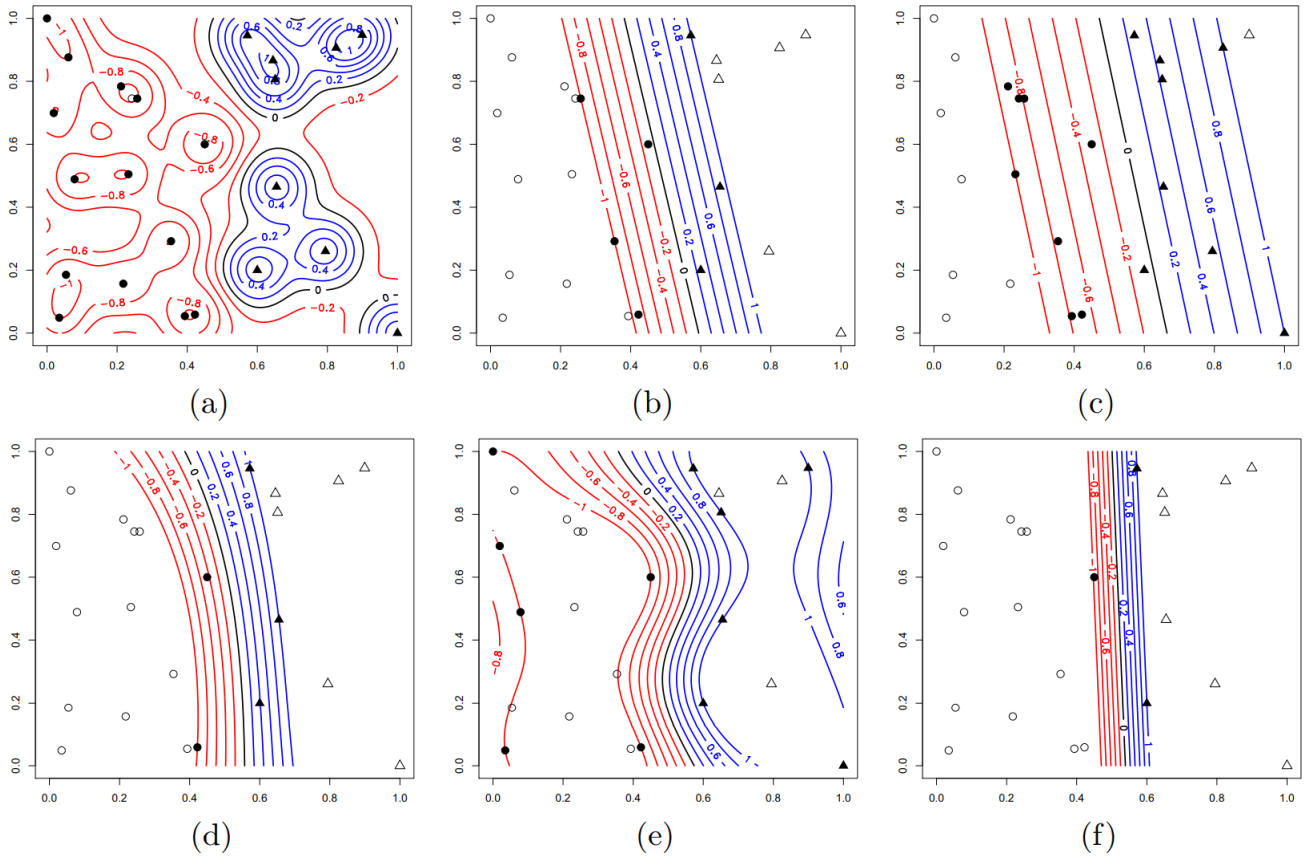
(iv) RBF Kernel with $k = 1$, C = 3

(v) RBF Kernel with $k = 0.1$, C = 15

(vi) RBF Kernel with $k = 10$, C = 1

Find out which figure plot would have resulted after each run mentioned above. Justify your answer.

In these plots, circles are Class 1, triangles are Class 2, and solid points are support vectors.



Solution: Solution by EE17B113 taken from the previous offering of PRML

Solution:
The factors which help in choosing the setting for each figure are :

- **Kernel** : Linear kernel will result in linear decision boundaries and RBF kernel will result in Gaussian-like curved boundaries.
- **C** : Higher C value corresponds to harder restrictions on the decision boundary margin, which leads to closely packed decision regions in the figure
- **k** : Higher k values corresponds to lower radii of decision boundaries in the case of RBF kernel.

Figures (a), (d), (e) have curved boundaries thus must be RBF kernel. Also based on the radius of the decision region, we can say that the k values are in the order (a)>(e)>(d). Figures (b), (c) and (f) have linear boundaries thus must be linear kernel. Based on the closeness of the decision boundaries, we can say that the order of C is (f)>(b)>(c). Using these informations, we can match the figures with the corresponding parameters.

a→ RBF Kernel, C=1, k=10	**b**→ Linear Kernel, C=1	**c**→ Linear Kernel, C=0.1
d→ RBF Kernel, C=15, k=0.1	**e**→ RBF Kernel, C=3, k=1	**f**→ Linear Kernel, C=10

4. (12 points) [GUESS-TIMATING BIAS AND VARIANCE] You are given a dataset consisting of 100 datapoints in this folder. You have to fit a polynomial ridge regression model to this data.

   As seen in class, a model's error can be decomposed into bias, variance, and noise. A "learning curve" provides an opportunity to determine the bias and variance of machine learning models, and to identify models that suffer from high bias (underfitting) or high variance (overfitting). The "learning curve" typically shows the training error and validation/testing error on the y-axis and the model complexity on the x-axis.

   (a) (2 points) Read the last Section (Section 4) on "Bias and Variance in practice" in this document, and summarize briefly how you will heuristically find whether your model suffers from (i) high bias, or (ii) high variance, using only the train and validation/test errors of the model.

   Solution: Solution given by MA21M025

Solution:

(i) High bias

Training error can be treated as amount of bias in the model. If the model is unable to fit the training data itself well, then it is likely that model has high bias i.e. model will not match the training data closely indicating UNDER-FITTING. High training data errors and high cross validation/test error will result in problem of high bias.

(ii) High variance

High variance means that the model is very sensitive to changes in the training data and can result in significant changes in the estimate of the target function when trained on different subsets of data from the same distribution. A low training error and high cross-validation/test error suggests high variance, indicating OVER-FITTING. In other words, the gap between the cross-validation error and training error can be treated as variance of the model. If the gap is large it indicates high variance.

(b) (2 points) Start with the code for polynomial regression from the tutorial (the code without in-built package functions in Tutorial #8) and add quadratic regularization functionality to the code. That is, your code should do polynomial regression with quadratic regularization that takes degree d and regularization parameter λ as input. Do not use any inbuilt functions from Python packages (except for plotting functions and functions to compute polynomial features for each data point).

Solution:

(c) (3 points) Run your code on the provided dataset for degree $d = 24$ and each λ in the set:

$$\{10^{-15}, 10^{-9}, 10^{-6}, 10^{-3}, 10^{-2}, 10^{-1}, 1, 10^{1}, 10^{2}, 10^{3}, 10^{6}, 10^{9}, 10^{15}\}$$

i. Perform 5-fold cross-validation on the 100 data points (20 data points in each fold). For each validation fold, compute both training (4-fold-based) and validation (1-fold-based) errors using the mean squared error measure.

ii. Calculate the average training and validation errors across the 5 folds.

Solution: Solution taken from cs23z075

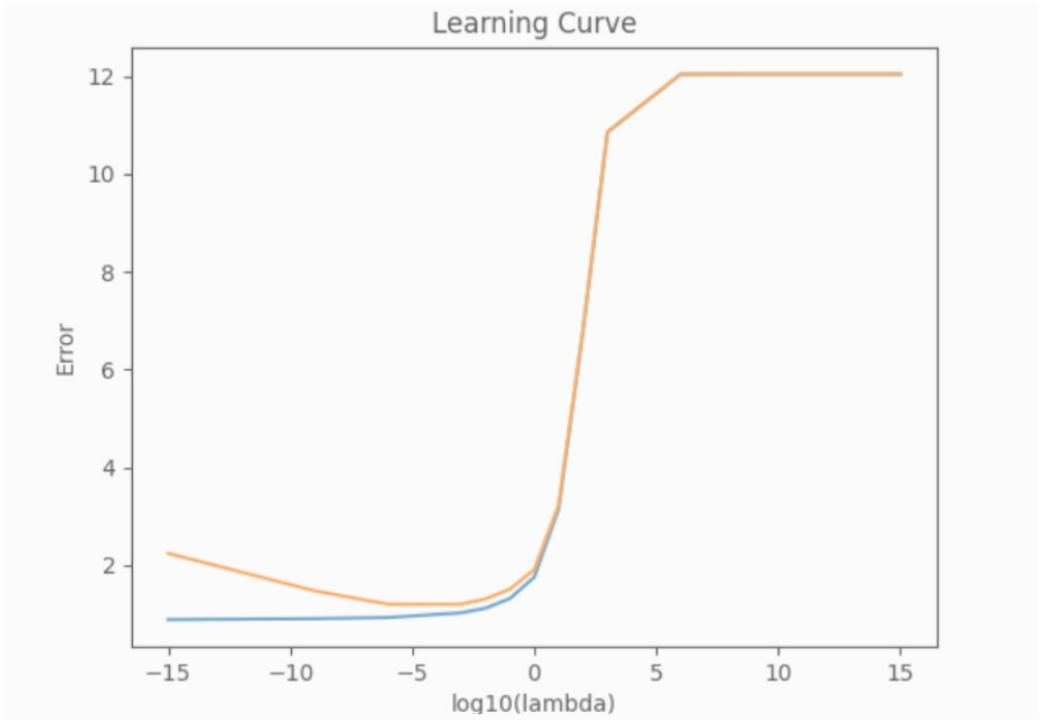
Solution: 4(c) Average training errors = [0.88631231, 0.90493905, 0.92890954, 1.02501644, 1.11827138, 1.3168566, 1.75276871, 3.15364915, 6.79510653, 10.85622836, 12.04445803, 12.04585416, 12.04585555]

Average validation errors= [2.24224477, 1.47637113, 1.20186052, 1.20244824, 1.3083495, 1.50771187, 1.90868861, 3.23656862, 6.83528197, 10.86580282, 12.04447009, 12.04585417, 12.04585555]

Difference between training and validation errors = [1.35593246, 0.571432075, 0.272950988, 0.177431799, 0.190078117, 0.190855267, 0.155919901, 0.0829194633, 0.0401754419, 0.00957445700, 0.000012, 0,0]

(d) (3 points) Construct a learning curve by plotting the average training and validation errors against the model complexity ($\log_{10} \lambda$). Based on this learning curve, identify the (i) model with the highest bias, (ii) model with the highest variance?, and (iii) the model that will work best on some unseen data.

Solution: Solution taken from cs23z075



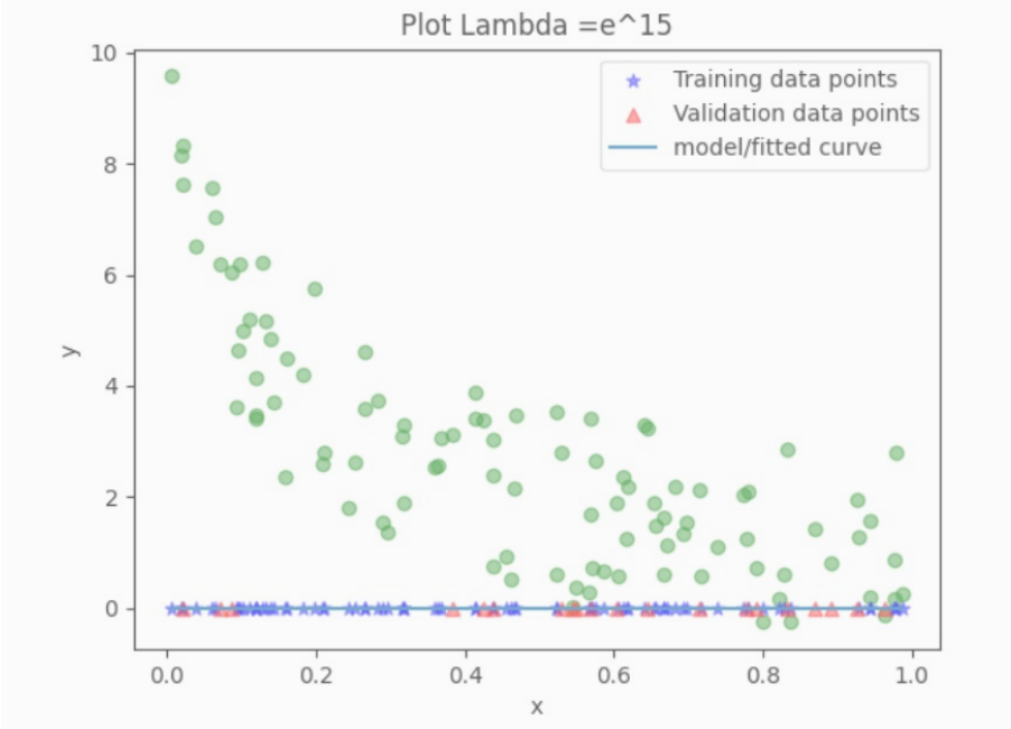
(i) Model with highest bias is model with high training error - Model with lambda = e^{15}

(ii) Model with highest variance is model with high difference between training and validation errors - Model with lambda = e^{-15}

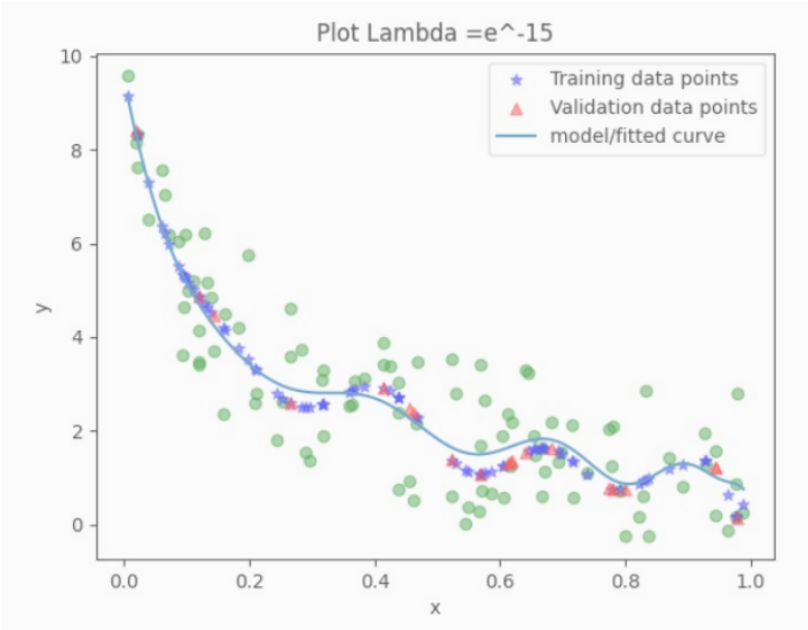
(iii) Model that will work best on some unseen data is Model with low validation error - Model with lambda = e^{-6}

(e) (2 points) Plot the fitted curve to the given data ($\hat{y}$ against x curve) for the three models reported in part (d) and superimposed with the training and validation data points for any one-fold.

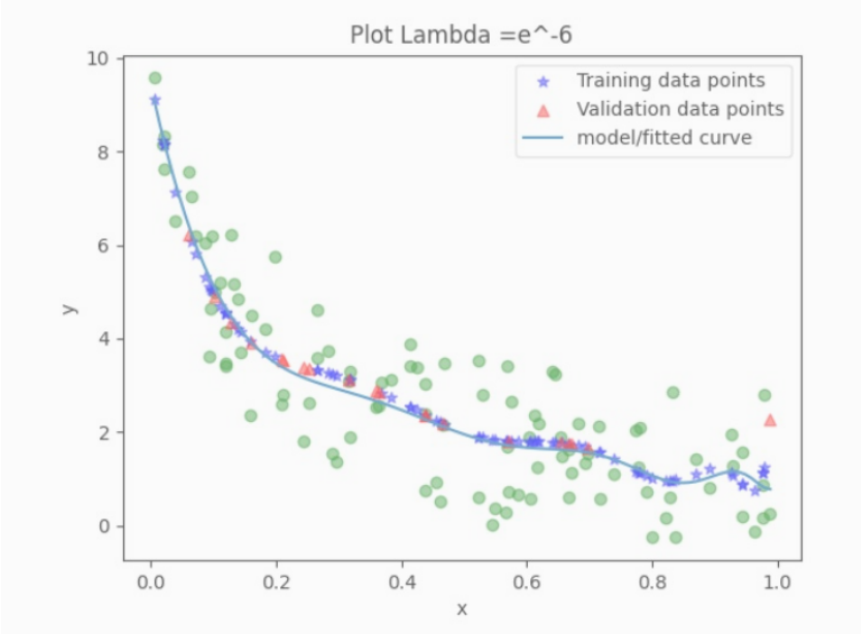
Solution: Solution taken from cs23z075
i) Model with the highest bias



ii) Model with the highest variance



iii) Model that works best on some unseen data



Please use the template.ipynb file in the same folder to prepare your solution. Provide your results/answers in the pdf file you upload to Crowdmark named rollno.asst3.answers.pdf, and submit your pdf and code separately also in this moodle link. The pdf+code submitted should be a rollno.zip file containing three files: rollno.asst3.answers.pdf, rollno.ipynb file (including your code as well as the exact same results/plots uploaded to Crowdmark) and the associated rollno.py file.