ME5775: Applied Machine Learning for MAE and Robotics
Homework 1: Linear regression, fitting a polynomial, model complexity and cross-validation

Due: Jan 25, 2022, Tuesday. Total points = 25.
Prof. Manoj Srinivasan

Guidelines As noted in syllabus, it is fine to submit the HW a couple of days later without asking for an extension. Look at the **Course Guidelines** document to see how exactly to submit the HW.

All HW submissions will have two parts: (1) a PDF file and (2) a zip file (named as per instructions) containing sub-folders called Q1, Q2 etc. containing all the data files needed, .m files corresponding to each question: Q1.m, etc. The submitted PDF will have screen-shots of the question, any written answers, MATLAB/python figures, etc. to produce a coherent narrative response to the HW. The PDF should not contain the code (at least for this HW).

I encourage working together. But to best learn coding, I suggest trying to solve the questions by yourselves first and then comparing notes and discussing with your classmates.

This HW expects you to use MATLAB as a default. We will introduce and use python in future HWs. But if you don't know MATLAB and are proficient in python, you are welcome to use python for this HW if you wish to do so, but you will need to translate the question appropriately for python.

Q1. Modeling linear thermal expansion of stainless steel

On page 52 (table A-1) of this DoE document on stress-strain data: `https://www.osti.gov/servlets/purl/6513543`, we find some raw data relating temperature T in Kelvin and percentage change in length (let's call this variable `percentChange` in MATLAB).

a) Copy-paste all the data into your program in `MATLAB` and plot the data as a scatter plot. Use the whole data set, ranging from T equals 233 K to 810 K.

b) Find a linear relation for the percentage change in length (say) in terms of temperature T (Kelvin), that is, a relation of the form $b_0 + b_1 T$. What are b_0 and b_1? Use two of the following and compare if they give the same answer: regress, fitlm, backslash, and one of the analytical formulas derived in lecture. (if using python, use)

c) What is the mean squared error for the fit?

Q2. Predicting miles per gallon of a car using a model derived from data

The following data repository has some automobile miles per gallon data from the 1970s:
`https://archive.ics.uci.edu/ml/datasets/Auto+MPG`

One of the motivations of this problem is that in real life, you'll get data in weird formats and with missing entries and you need to deal with that. So we expose you to and guide you through a little of that messiness here.

Note: Excel is available to all students via OSU's OCIO. You can also use a simple text editor or Word or another spreadsheet program instead of excel.

a) Click on the "Data Folder" link to download the following two files: `auto-mpg.data` and `auto-mpg.names`. Put these files in your HW subfolder called Q2. Open `auto-mpg.data` using excel. You will notice that the last column is text containing the model of the car. Delete that whole column and save the file again. Now,

scroll down the data-set. You will notice that the new data-set has 8 columns. What do these columns mean (see `auto-mpg.names` or the webpage itself)?

b) Now, in `MATLAB`, go to the correct folder, and say `load('auto-mpg.data')` in the command line. Note: type this out, copy-pasting code from this PDF may not work. You will see an error. Now, open the file in MATLAB again, but now by right clicking or double-clicking the file name. Scroll down. You will notice that some of the entries are missing and have a question mark instead of a number. Delete those rows with missing entries entirely (at least for the purposes of this HW). Now try `load('auto-mpg.data')` again and it should give no error and it will create a variable called `auto_mpg`, a matrix with 398 rows and 8 columns. Okay, now you are all set to use the data.

c) Fit a linear function with miles per gallon as output and all columns 2 to 6 together as input. To clarify, we are trying to construct a model of the form:

$$\text{mpg} = b_0 + b_1 \cdot \text{cylinders} + b_2 \cdot \text{displacement} + b_3 \cdot \text{horsepower} + \dots$$

Many inputs with one output. Report the coefficients. Feel free to use fitlm or the pseudoinverse or backslash or regress. If you want to plot something (not necessary), try plotting the left hand side versus the right hand side of the above equation, as this will visualize how good your prediction is.

d) Based on the linear function, say whether you think the mpg will increase or decrease if you increase the other variables (cylinders, displacement, horsepower, etc.).

Post-script: For both Q1 and Q2, because the different variables have different magnitudes and ranges, it may help numerically to normalize all the data being between 0 and 1 before fitting. This is especially important for fitting nonlinear models, less important for linear models. But we will examine this in a later HW.

Q3. Training error decreases with model complexity (for nested models).

Use `DatHW1Q2.mat` which has 30 data points: (x_i, y_i) where $i = 1 \dots 30$, where x_i and y_i are real numbers. Fit $y = f(x)$ each of the following models to the entire data and report the values of $b_0, b_1 \dots$. For doing this, do not use fitlm, but use the regress or the Moore-Penrose pseudo-inverse formula or the backslash operation to obtain $b_0, b_1 \dots$ and plot the fits in separate plots, as outlined in the lectures.

a) $f(x) = b_0 + b_1 x$.
b) $f(x) = b_0 + b_1 x + b_2 x^2$.
c) $f(x) = b_0 + b_1 x + b_2 x^2 + b_3 x^3$.
d) $f(x) = b_0 + b_1 x + b_2 x^2 + b_3 x^3 + b_4 x^4$.

e) For each of the models, determine the mean squared error $L = \sum_{i=1\dots N} \left[y_i - f(x)\right]^2 / N$. Plot MSE as a function of the degree of the polynomial, and note that they keep going down. For ordinary least squares, higher degree polynomials will always have equal or lower error than lower degree polynomials (as they contain the lower degree polynomials as special cases).

Q4. Inverse robot kinematics

A 2D robot arm shown in the figure below has servo motors at the joints A and B, and is able to directly control the joint angles θ_1 and θ_2. By changing joint angles θ_1 and θ_2, we can affect end-point position (x_B, y_B). You are give a data file `DatRobotManipulator.mat`, with four variables: `theta1Store`, `theta2Store`, `xBStore`, and `yBStore`, which has negligible measurement noise. We want to solve the so-called "inverse kinematics" problem approximately using least squares based polynomial regression. That is, we want to find functions $\theta_1 = f_1(x_B, y_B)$ and $\theta_2 = f_2(x_B, y_B)$ by minimizing $\sum(\theta_1 - f_1(x_B, y_B))^2$ and $\sum(\theta_2 - f_2(x_B, y_B))^2$ over all the data. Once we have such a function, we know what $\theta_{1,2}$ commands to give the servo motors to go to a particular (x_B, y_B). Use all provided data to:

a) Construct linear models for the angles given end-point positions and determine the mean squared error:

$$\theta_1 = a_0 + a_1 x_B + a_2 y_B \text{ and } \theta_2 = b_0 + b_1 x_B + b_2 y_B$$

b) Construct quadratic models for the angles given end-point positions and determine the mean squared error:

$$\theta_1 = a_0 + a_1 x_B + a_2 y_B + a_3 x_B^2 + a_4 y_B^2 + a_5 x_B y_B \text{ and } \theta_2 = b_0 + b_1 x_B + b_2 y_B + b_3 x_B^2 + b_4 y_B^2 + b_5 x_B y_B$$

That is, determine the best fit coefficients $a_{1,2,...}$ and $b_{1,2,...}$ in the above models.

Remark 1: For this simple robot, we can get these functions analytically using trigonometry. However, for more complex robots, it is very useful to derive such data-based models relating 'output' and 'inputs', so that we may use such models in the control of the robot.
Remark 2: Here, for simplicity, we are minimizing prediction error in θ_1 and θ_2. It may be better to solve a different problem that focuses on minimizing error in output (x_B, y_B).

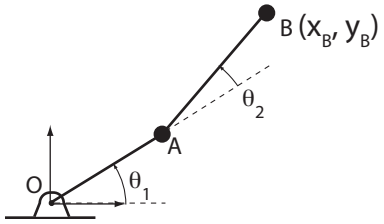

Figure 1: Two link manipulator. By changing joint angles θ_1 and θ_2, we can affect end-point position (x_B, y_B)

Q5. Which degree polynomial to use? k-fold Cross-validation

The file `DatHW1Q2.mat` has 30 data points.

a) Use 4-fold-cross-validation to determine the best polynomial degree that optimizes the mean validation or test error.

b) If it is known that y has noise with standard deviation equal to 3, what polynomial degree should you use?

Remark: if you're using the example code I provided for k-fold cross-validation, you may get an error that MATLAB doesn't have the function **crossvalind**. This is in the **bioinformatics** toolbox. To add the bioinformatics toolbox to your MATLAB install, you can go to apps-¿ get more apps-¿add-ons to just add this toolbox, without having to re-install all of MATLAB.