UTSA

CS 6243 Machine Learning

EE 6363: Advanced Machine Learning

Assignment: HW2

I. Assignment Instructions

Solve all given problems.

Submit your report in the dedicated folder on CANVAS.

Submit your report by the deadline. Delayed reports will not be graded.

Deadline: 10/04/2023, 11:59pm.

Reports must be typed and uploaded in PDF format. Handwritten reports will not be graded.

Any requested code must be presented at the end of your report in a dedicated appendix, titled “Code.” All codes must have adequate comments.

All display equations must be numbered. All figures must be numbered and captioned.

You can use any source (notes, books, online), but it must be cited in a References List at the end of your report.

Do not outsource this assignment (or parts of it) to another intelligent entity (whether human or AI). Except for what you explicitly cite, what you submit must be your own intellectual work.

Grade: This assignment corresponds to 6% of your final grade. You will be graded based on 1) correctness, 2) completeness, 3) clarity (equal weight).

Teams: Work in teams. Each team member must submit the exact same report on CANVAS (only students that submit a report will receive a grade). It is expected that all team members will put equal effort into this assignment.

The instructor may determine each student's grade on this assignment by individual examination during course office hours.

Terminology: Present = just show result. Derive = show all math steps. Implement in python = write python code and presented it in the dedicated appendix. Discuss/Compare = discuss/compare in words (no need for math/code).

II. Datasets

Dataset A:

This dataset contains input/output data generated from a noisy polynomial model: $y = f(x) + \epsilon$, where $x \in I = [-2, 2]$ and

$$f(x) = m(x, a) = p_K(x;\, a) := \sum_{m=0}^{K} x^m a_{m+1}\,.$$

Here, $K \geq 1$ denotes the polynomial order and $a \in R^{K+1}$ represents the parameter vector.

The dataset consists of two files: SYNDTR.csv, which contains 200 training data points, and SYNDTE.csv, which contains 1800 testing data points.

Each file has three columns: The first column represents the input (denoted as TRI for SYNDTR and TEI for SYNDTE). The second column signifies the output under low noise conditions (denoted as TROL for SYNDTR and TEOL for SYNDTE). The third column signifies the output under high noise conditions (denoted as TROH for SYNDTR and TEOH for SYNDTE).

Dataset B:

This dataset contains real-world data related to fish measurements. Specifically, the input x is 2-dimensional, where x_1 signifies fish width and x_2 signifies fish length. The output y is the fish weight.

The dataset is divided into two files: FISHDTR.csv, which contains the training data, and FISHDTE.csv, which contains the testing data.

Each file consists of three columns: "width," "length," and "weight." "Width" and "length" serve as the input features (denoted as TRI for FISHDTR and TEI for FISHDTE), while "weight" is the output (denoted as TRO for FISHDTR and TEO for FISHDTE).

III. Problems

Problem 1 (20%)

This problem focuses on Dataset A, initially considering the low-noise (LN) scenario.

Assume that you are aware that m is an M-order polynomial, denoted p_M, for some M. However, the exact order is unknown. Your objective is to train $\hat{a}$ such that $\hat{f}(x) = p_M(x, \hat{a})$ approximates the true $f(x)$, utilizing the available training examples for regression.

1. Implement in Python: Create a function "my_model_train" that takes TRI, TROL, and M as input parameters and minimizes train-MSE to return the trained parameters $\hat{a}$. Your function should employ SVD-based single-shot LS solution. Do not use any prebuilt poly-fit functions. Implement in Python: Create a function "my_model_test" that accepts $\hat{a}$, TEI, and TEOL as inputs and calculates and returns test-MSE.

2. Implement in Python: Conduct an experiment using the aforementioned functions to compute train-MSE and test-MSE for the optimized $\hat{a}$ with $M = 0, 1, \ldots, 10$. In Fig. 1 of your report plot train-MSE (red curve) and test-MSE (blue curve) vs. $M = 0, 1, \ldots, 10$. Discuss how train-MSE and test-MSE vary across M in Fig. 1.

3. Implement in Python: Repeat steps 2-6 for the high-noise (HN) scenario, using the data sets TRI, TEI, TROH, and TEOH. In Fig. 2 of your report, plot train-MSE (red curve) and test-MSE (blue curve) against M. Discuss how train-MSE and test-MSE vary across M in Fig. 2 and compare these observations with those in Fig. 1.

4. Implement in Python: Repeat steps 2-8, this time limiting the training set to the first 180 data points. Create corresponding Figs. 3 and 4 and discuss the results, comparing Figs. 2 and 3 with Figs. 1 and 2.

Problem 2 (20%) – Feature/Input/Output Correlations

This problem is based on Dataset B.

1. Implement in Python: Utilize TRI and TRO to compute matrix C. For each $(i, j) \in \{1,2\}^2$, $C(i, j)$ should contain the absolute value of the correlation coefficient between input features i and j. Additionally, $C(i, 3) = C(3, i)$ should contain the absolute value of the correlation coefficient between

input feature i and the output. Present matrix C. Discuss the observed correlations and identify which input feature appears to be most strongly correlated with the output.

2. Implement in Python: In Fig. 5 of your report, scatter-plot weight vs width for the training data. Then, in Fig. 6 of your report, scatter-plot weight vs length for the training data. Discuss your observations based on Figs. 5 and 6.

Problem 3 (20%) – Univariate Models

In this task, polynomial regression ($\hat{m} = p_M$ for some M) will be applied considering univariate input --i.e., using only one of the two available features. Initially, perform polynomial regression similar to Problem 1 for the single input feature "width" and for all values of $M = 0, 1, \ldots, 5$. Then, perform the same process for the single input feature "length."

1. Implement in Python: In Fig. 7 of your report, plot your estimated curve $\hat{f}(x) = p_M(x; \hat{a})$ for the input "width" and $M = 3$ against $x \in [1, 9]$ with a step size of 0.001. In the same figure, include a scatterplot of the training data for "weight" against "width." Discuss: Does your curve adequately capture the training data?

2. Implement in Python: In Fig. 8 of your report, plot your estimated curve $\hat{f}(x)$ for the input "length" and $M = 3$ against $x \in [1, 60]$, with a step size of 0.001. Include a scatter-plot of the training data for "weight" against "length" in the same figure. Discuss: Does your curve adequately represent the training data?

3. Implement in Python: In Fig. 9 of your report, plot test-MSE against $M = 1, 2, \ldots, 5$ for the input "width" (depicted by a blue curve) and for the input "length" (depicted by a red curve). Discuss: Which of the two features seems to be the most informative for the prediction task? How does this relate to the correlation coefficients computed in Problem 2? What trends are observed as M varies for the two models?

Problem 4 (20%) – Bivariate Linear

In this problem, we utilize both input features together to apply linear regression. We will use the bivariate model $\hat{m}(\boldsymbol{x}, \hat{\boldsymbol{a}}) = [1, \boldsymbol{x}^T] \ \hat{\boldsymbol{a}}$.

1. Present the objective function and the feasibility set for the train-MSE minimization problem. Derive the solution to the train-MSE minimization problem by enforcing the stationarity condition and solving it via SVD.

2. Implement in Python: Solve the train-MSE minimization problem using TRI/TRO data points from Dataset B, based on the method derived in the previous item. Subsequently, calculate and present the corresponding test-MSE. Compare/discuss the calculated test-MSE and to the test-MSEs obtained through the single-feature methods studied in Problem 3. Identify the factors contributing to any performance difference observed.

Problem 5 (20%) – Bivariate Non-Linear

In this problem, we extend the above multivariate linear model to multivariate polynomial model of order $M = 1,2,3$.

1. Present the polynomial model $\hat{m}$ for each order M (recall that we are working on multivariate input). Present the objective function for the train-MSE minimization problem.

2. Implement in Python: Solve the train-MSE minimization problem on the TRI/TRO data points from Dataset B, by enforcing stationarity condition and solving the resulting SLE via SVD. In Fig. 10 of your report plot train-MSE (red curve) and test-MSE (blue curve) vs $M = 1,2,3$. Compare/discuss the curves. Compare/discuss the test-MSE results with those attained by all previous uni- and bivariate models. Which model worked best?