Machine Learning
Università della Svizzera italiana

Assignment 1

Loika Arseni

May 1, 2024

!! !! Note that this file is just meant as a template for the report, in which we reported *part of* the assignment text for convenience. You must always refer to the text in the README.md file as the assignment requirements !! !!.

Tasks

This section should contain a detailed description of how you solved the assignment, including all required statistical analyses of the models' performance and a comparison between the linear regression and the model of your choice. Limit the assignment to 8-10 pages and do not include any code in the report.

Task 1

Use the family of models $f(\mathbf{x}, \boldsymbol{\theta}) = \theta_0 + \theta_1 \cdot x_1 + \theta_2 \cdot x_2 + \theta_3 \cdot \cos(x_1) + \theta_4 \cdot x_2 \cdot x_2 + \theta_5 \cdot \tanh(x_1)$ to fit the data.

a. Write in the report the formula of the model substituting parameters $\theta_0, \ldots, \theta_5$ with the estimates you've found:

   $$f(\mathbf{x}, \boldsymbol{\theta}) = 0.966 + 5.052 \cdot x_1 + (-4.005) \cdot x_2 + 7.021 \cdot \cos(x_1) + 1.998 \cdot x_2 \cdot x_2 + (-0.104) \cdot \tanh(x_1)$$

b. Evaluate the test performance of your model using the mean squared error as performance measure.
   - MSE (train) = 1.41
   - MSE (test) = 1.49

c. Implement Lasso Regression, what do you observe? What can you infer about the given family of models?
   - MSE (train) = 1.42
   - MSE (test) = 1.49

- Given that Lasso and linear regression yield very similar results, it could be concluded that the family of models is linear.

Task 2

Consider any family of non-linear models of your choice to address the above regression problem.

a. Evaluate the test performance of your model using the mean squared error as performance measure (same data as Task 1).
   - MSE (train) = 1.43
   - MSE (test) = 1.48

b. Compare your model with the linear regression of Task 1. Which one is statistically better?
   - By carrying out K-fold validation (with 10 folds), it can be observed that the best MSE for the feed-forward neural network is 1.34, while for linear regression it is 1.30, meaning the latter is statistically slightly better. However, after performing t-test on the regressors, the resultant T-score is -1.10, while the p-value is 0.272, meaning there is no significant statistical difference between the two models.

Task 3 (Bonus)

In the **GitHub repository of the course**, you will find a trained Torch learn model that we built using the same dataset you are given (**data_bonus**). This **baseline** model is able to achieve a MSE of **0.013**, when evaluated on the test set. You will get extra points if you provide a model of your choice whose test performance is **better** (i.e., the MSE is lower) than ours. Of course, you must also tell us why your model is performing better.

Questions

Q1. Training versus Validation

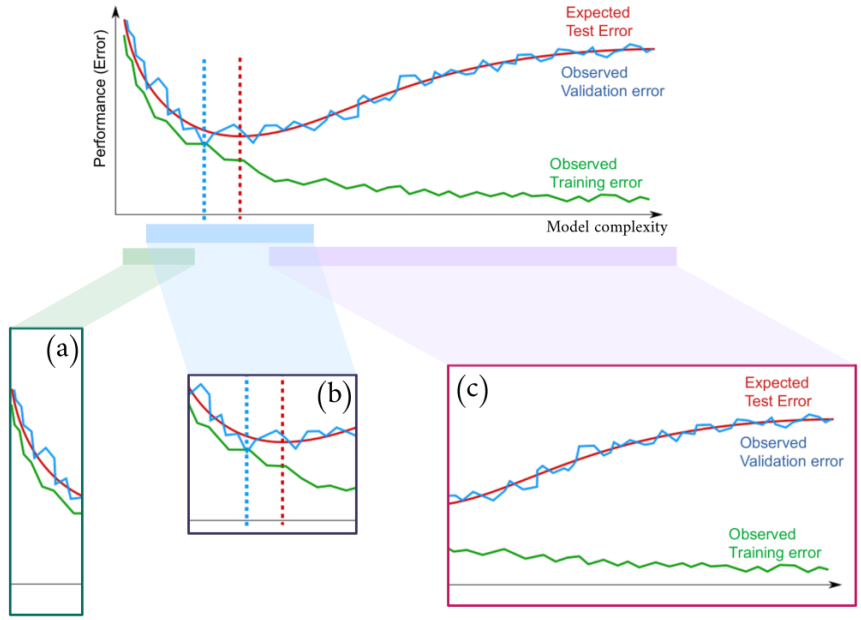
Q1.1 What is the whole figure about?

A1.1 The figure shows how the model performance, measured by the error rate, changes with increasing model complexity for training and test datasets.

Q1.2 Explain the behaviours of the curves in each of the three highlighted sections in the figure, namely (a), (b), and (c).

A1.2 (a) The model starts as underfitted. As its complexity increases (i.e. more parameters are considered), the model begins to fit the current data and improves at predicting unseen data, meaning both the test and training error rates decrease.

(b) When the model reaches a certain complexity, it performs by Occam's razor, where it considers a sufficient number of past data / parameters, and is able to effectively predict the unseen data. Its performance reaches its optimum level at this point.



(c) When model complexity increases further, it begins to consider noise as well as real patterns and so it becomes overfitted to the training data. This means its error rate for predicting unseen data increases, which is reflected by the increasing Expected Test Error.

Q1.2.a Can you identify any signs of overfitting or underfitting in the plot? If yes, explain which sections correspond to which concept.

A1.2.a Section (a) corresponds to underfitting. Section (c) corresponds to overfitting.

Q1.2.b How can you determine the optimal complexity of the model based on the given plot?

A1.2.b By finding the minimal point of the Observed Validation Error curve (marked by the blue dotted line).

Q1.3 Is there any evidence of high approximation risk? Why? If yes, in which of the below subfigures?

A1.3 While there is evidence of some approximation risk, marked by the area between the Observed Validation Error and the Expected Trial Error, it is not high. It is at its highest in figure (b).

Q1.4 Do you think that increasing the model complexity can bring the training error to zero? And the structural risk?

A1.4 Increasing the model complexity may bring the training error to 0 under the assumption that the training data set is infinite, not otherwise. The structural risk, however, cannot be 0 unless we pick a better family of models to begin with.

Q1.5 If the X axis represented the training iterations instead, would you think that the training procedure that generated the figure used early stopping? Explain why. (**NB:** ignore the subfigures and the dashed vertical lines)

A1.5 If X axis represented training iterations, then the training procedure generating the figure did not use early stopping. Early stopping would be used at the point when the performance on the validation dataset starts to degrade, which would be at the lowest point of the Observed Validation Error.

Q2. Linear Regression

Consider the following regression problem in which the task is to estimate the target variable $y = g(x) + \eta$, where $g(.)$ is unknown, $\eta \sim N(0, 1)$, and the input variable x is a bi-dimensional vector $x = [x_1, x_2]$. Suppose to have n training samples and to fit the data using a linear model family $f(x, \theta) = \theta_0 + \theta_1 * x_1 + \theta_2 * x_2$.

Now, we add another regressor (feature) x_3 (to obtain $f(x, \theta) = \theta_0 + \theta_1 * x_1 + \theta_2 * x_2 + \theta_3 * x_3$), and we fit a linear model on the same data again. Comment and compare how the (a.) training error, (b.) test error and (c.) coefficients would change in the following cases:

Q2.1 $x_3 = x_1 + 0.2 \cdot x_2$.

A2.1 Given that x_3 is a linear combination of the existing features, adding it to the existing model will provide more information to consider, increasing the model complexity. This will decrease the training error as the model will have more information to learn from. The test error, however, will likely increase, because adding a new parameter that is a linear combination of two of the existing parameters will raise the risk of the multicollinearity problem. This means it would make the coefficients more related, to one another. Additionally, in terms of the test/-training error graph, it would also move the state along the x-axis, likely causing overfitting.

Q2.2 $x_3 = x_1 * *2$ (in Python ** is the "power" operator, so 3 ** 2 = 3 * 3 = 9).

A2.2 In this case, x_3 is a non-linear combination of existing features. The introduction of non-linearity could improve the model's performance in estimating $g(.)$ and reduce the test error. Since adding a new parameter will increase the model complexity, it will also reduce the training error.

Q2.3 x_3 is a random variable independent from y.

A2.3 Given that x_3 is completely independent from y, it would mean that the parameter θ_3 would carry no useful information to the learning of the model, therefore the testing error would increase. The training error, on the other hand, could decrease, simply because the model complexity increases with the addition of a new parameter. The coefficients would remain in a linear combination.

Q2.3 How would your answers change if you were using Lasso Regression?

A2.3 Lasso regression would penalize parameters, thereby promoting sparsity within the model. It would help alleviate the effects of multicollinearity and overfitting. Therefore, if the terms x_1 and x_2 in Q2.1 were correlated to the extent of causing multicollinearity, the coefficient θ_3 would be shrank to 0, and make no negative effect on the test performance with Lasso regression. For Q2.1 and Q2.3, it would also prevent overfitting in the case that the added coefficients were reducing performance.

Q2.4 Explain the motivation behind Ridge and Lasso regression and their principal differences.

A2.4 The motivation behind Ridge and Lasso regression is to minimize the number of considered parameters by adding a shrinking penalty to the loss function, thereby removing the parameters that are less relevant. The key difference between them is that Ridge regression penalizes the square of the parameter, while Lasso regression penalizes the parameter itself.

Q3. Logistic Regression

Q3.1 What are the main differences between the logistic-regression and the perceptron?

A3.1 Perceptron is a classification algorithm for supervised learning. Its architecture consists of a single-layer neural network, which contains a set of nodes and the corresponding weights, which get evaluated by a heaviside function to provide a binary classification output. Logistic regression is a linear model that aims at training the network parameters so that the sigmoidal output is supported by a probabilistic framework. The principle difference then is that logistic regression is able to provide confidence of its output, while perceptron is unable to do so. Another difference is in the definition of their activation function - the perceptron uses a heaviside function, while logistic regression uses sigmoidal function.

Q3.2 Discuss the major limit they share and how neural networks can solve it.

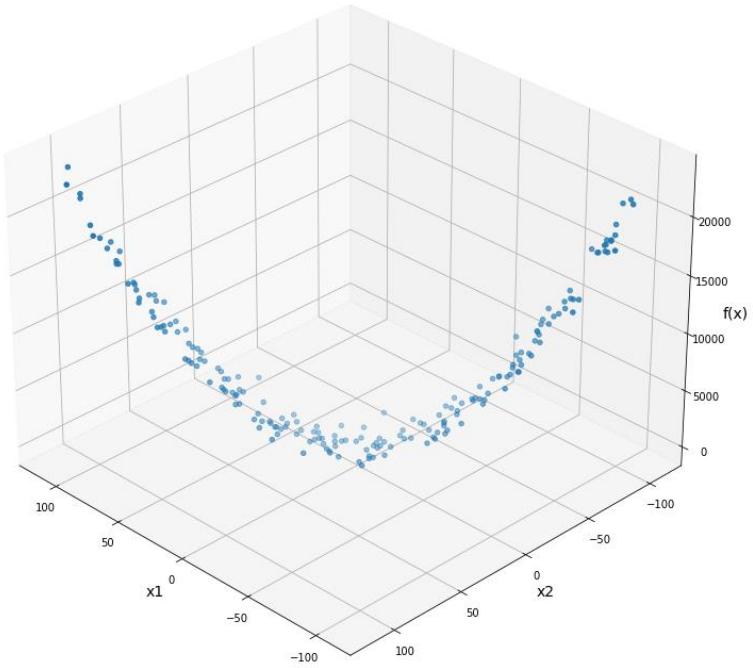
A3.2 The major limit of the perceptron and logistic regression is that they work best only on linearly-separable data, given that they are both binary classifiers. Feed-forward neural networks are able to overcome this limitation by introducing hidden layers, which are separated by non-linear activation functions, e.g. ReLU or tanh.

Q3.3 What is the role of activation functions in feed-forward neural networks.

A3.3 In FFNNs, the activation function determines if any node, with a given set of inputs and bias, should be considered in the subsequent learning.

Q4. Consider the regression problem shown in the picture below and answer each point.

Q4.1 Do you think a model of the family f(x, theta) = $\theta_0 + \theta_1 * x_1 + \theta_2 * x_2$ is a good choice for such task? Why?



A4.1 The provided model would not be a good choice for the given task since it is a linear model, and the problem presents non-linearly separable data (of quadratic form).

Q4.2 Do you think using a feed-forward neural network would improve the results?

A4.2 Using a FFNN would improve the results, since it has non-linear activation function between the hidden layers, allowing for better fits with non-linearly separable data.