Machine Learning
Università della Svizzera italiana

Assignment 1

Scannapieco Davide

May 8, 2022

The assignment is split into two parts: you are asked to solve a regression problem, and answer some questions. You can use all the books, material, and help you need. Bear in mind that the questions you are asked are similar to those you may find in the final exam, and are related to very important and fundamental machine learning concepts. As such, sooner or later you will need to learn them to pass the course. We will give you some feedback afterwards.

!! 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.

1 Regression problem

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 2500 words (formulas, tables, figures, etc., do not count as words) and do not include any code in the report.

1.1 Task 1

Use the family of models $f(\mathbf{x}, \boldsymbol{\theta}) = \theta_0 + \theta_1 \cdot x_1 + \theta_2 \cdot x_2 + \theta_3 \cdot \cos(x_2) + \theta_4 \cdot x_1^2$ to fit the data. Write in the report the formula of the model substituting parameters $\theta_0, \ldots, \theta_4$ with the estimates you've found:

$$f(\mathbf{x}, \boldsymbol{\theta}) = 1.73615855 - 0.17706444 \cdot x_1 - 0.62306431 \cdot x_2 + 0.00967743 \cdot \cos x_2 + 0.04090775 \cdot x_1^2$$

Evaluate the test performance of your model using the mean squared error as performance measure.

$$MSE = 1.286145222305325$$

To run the project you should install all the packages required listed in file *requirement.txt* just executing this command inside the terminal:

```
pip install -r requirements.txt
```

For this task I created a matrix 5x5 from the data and then I have a LinearRegression object with parameter fit_intercept = false since I added manually a column of ones. Then I just computed the MSE between the value that I expect and the value I get.

1.2 Task 2

Consider any family of non-linear models of your choice to address the above regression problem. Evaluate the test performance of your model using the mean squared error as performance measure. Compare your model with the linear regression of Task 1. Which one is **statistically** better?

For this task, I decided to use a Feed Forward Neural Network with different activation function and check which one was the best one depending on the Mean Squared Error evaluated on a validation set. The feed forward neural network I chose has five layers with hyperbolic tangent as activation function for the first four layers and the last layer has a linear activator. I divided the data into 3 parts distributed in the following way: 80% training set, 10% test set and 10% validation set.I chose Adam as optimizer with 500 epochs using early stopping if after 70 iterations (patience) the model would not improve. Then I decided to have the train the model with the same activator function but with different neurons - just for performance measurement - and the best one was with 15 neurons. The non linear model trained is statistically better than the linear model. This due thanks to checking with the T-Test equation where e is the mean and s the variance:

$$T = \frac{\bar{e}_{nn} - \bar{e}_{lr}}{\sqrt{\frac{s_{nn}^2}{l} + \frac{s_{lr}^2}{l}}}$$

Using this equation I get a T value that is outside the interval of confidence (-1.96,1.96) so this mean that we reject the null hypothesis and we check the variance. Since the variance of the non linear model is way better than the linear model's variance, the non linear model is statistically better.

1.3 Task 3 (Bonus)

In the **Github repository of the course**, you will find a trained Scikit-learn model that we built using the same dataset you are given. This baseline model is able to achieve a MSE of **0.0197**, when evaluated on the test set. You will get extra points if the test performance of your model is better (i.e., the MSE is lower) than ours. Of course, you also have to tell us why you think that your model is better.

For such task I decided to use the same model that I used for task 2 since it achieves a better MSE than the one provided by your model. I achieved a MSE value of 0.0146. I also checked the MSE on a TestSet that of course i generate and still get a value closer to 0.0146 which indicates that the model is not overfitting - early stopping helps also preventing this -. Even if the MSE is better of my model, just around the 50% of time I could prove it also with the T-test since the t-value obtained was outside the interval of confidence - I always obtain a better variance than the one provided by your model-. But doing a bit of reverse engineering using baseline_model.get_params() I notice that

the activation function is a RELU and it doesn't use early stopping which indicates that it could bring to overfitting if the number of epochs is huge. Also The size of the layer is quite big. The ratio between the size of the data and the number of parameter should be really less than one and your ratio is really close to 1. All these factors can explain why my model can perform better.

2 Questions

2.1 Q1. Training versus Validation

1.Q. Explain the behaviors of the curves in each of the three highlighted sections in the figure, namely (a), (b), and (c);

1.A.

- In the first highlighted section (a) we can see that all the parameters (the Expected Test Error, Observer Validation Error, Observed Training error) are all very high and close to each other (under-fitting).
- In the second one (b) we can clearly see that as the complexity of the model increases all the errors decreases and in specific the Expected Test Error and Observed Validation error reached their minimum.
- In the last figure we can see a raise of the Expected Test Error and Observed Validation error as the complexity of the model increases and as expected the observed training error keeps decrementing (over-fitting).

2.Q. Do you think the figure gives useful information to reduce the approximation risk? And to reduce the estimation risk? Explain why.

2.A.

- We can clearly see from this figure specifically in the second one (b) that the approximation risk is high since it depends only on the distance between the approximating model family and the process to generate the data. We can see that the complexity of the optimal model (minimum of the expected error) and the inherent risk (which we assume to be close to 0) is not that close. So this is not a very good model to generate data. So in order to reduce the approximation risk we can change the family of models for example if we assume a polynomial family has been used we can try with an exponential family.
- The estimation risk depends on the effectiveness of the learning procedure. We can see that the distance between the learning procedure and the process to generate the data is high. We can see that the complexity of the optimal model (minimum of the expected error) and the minimum of the validation error is not that close.

3.Q. Would you think that by further increasing the model complexity you would be able to bring the structural risk to zero? How would your answer change if your data was not affected by noise?

3.A.

- This is not possible in theory due to the fact that to get a 0 structural risk we should have a training data that covers all possible input. We can clearly see that it can not be possible looking at the formula of the structural risk:

$$V(\theta^0) = \int L(y, f(\theta, x))\, p_{x,y}\, dxy$$

and if we define y equal to:

$$y = g(x) + \eta$$

If we plug in the structural risk formula $g(x) = f(x, \theta^0)$ meaning that we learned completely the model, we would end up in this situation:

$$V(\theta^0) = \int \eta^2$$

Therefore this value won't go to 0 increasing the model complexity but the only way to reduce this value is to improve the problem at the start changing the family of model.

– As seen above the structural risk in case we have a inherent risk of 0 (no noise) would indeed go to 0 due to the fact that the inherent risk is 0 (η)

4.Q. 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)

4.A.

For this question we need to take into consideration the concept of patience. We assume that the number of iteration on the X axis is maximum 100. We can see that the validation error has been reached in the early stage of the iteration. So if we assume at the 20^{th} iteration we reached the minimum so the optimal error, and the patience is set to 20, we can say that the training procedure didn't use early stopping.

2.2 Q2. Linear Regression

Comment and compare how the (a.) training error, (b.) test error and (c.) coefficients would change in the following cases:

1.Q. $x_3 = x_1 + 3.0 \cdot x_2$.

1.A.

This is how our model would be modified if we add $x_3 \rightarrow$

$$f(x, \theta) = \theta_0 + \theta_1 \cdot x_1 + \theta_2 \cdot x_2 + \theta_3 \cdot x_3$$

$$= \theta_0 + \theta_1 \cdot x_1 + \theta_2 \cdot x_2 + \theta_3 \cdot (x_1 + 3x_2)$$

$$= \theta_0 + x_1 \cdot (\theta_1 + \theta_3) + x_2 \cdot (\theta_2 + 3\theta_3)$$

As we can see from this result there will be no change in terms of test and training error but coefficient will of course change. This is just because x_3 is a combination of x_1, x_2. Although the efficiency of the model will not improve and will be the same as the original one.

2.Q. $x_3 = x_1 \cdot x_2 \cdot x_2$

2.A.

This is how our model would be modified if we add $x_3 \rightarrow$

$$f(x, \theta) = \theta_0 + \theta_1 \cdot x_1 + \theta_2 \cdot x_2 + \theta_3 \cdot x_3$$

$$= \theta_0 + \theta_1 \cdot x_1 + \theta_2 \cdot x_2 + \theta_3 \cdot (x_1 \cdot x_2^2)$$

For this particular case, everything change since we do not have anymore a linear combination of the parameters but the model become quadratic so the training error, the test error and also the coefficients will change. In particular training error and test error will improve.

3.Q. Can we make any educated guess on what would be the value of θ_3 for each of the preceeding cases if we used Lasso Regression?

3.A.

For the first case an estimation of θ_3 value if we use Lasso regression would be 0 since it's already enough having θ_1, θ_2, there is no improvement of the training error and test error. For the second one since the training error and test error improved because the model of family has a higher complexity, it's not possible to make valid prior guess on θ_3 value.

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

4.A.

We use Lasso and Ridge regression with the aim to push as many parameters as possible towards zero by adding a shrinking penalty to the loss function simplifying so the model. We use this two type of regression to avoid multicollinearity, it means that it can happen that two or more predictor are correlated and so there is no more independent information in the regression model. For the Ridge regression we penalize the quadratic of the parameter while for the Lasso regression we penalize the parameter itself. For the ridge regression we can say that this penalization is not so fair since squaring a small number makes it even smaller but the original number could not be sufficiently small - we want to push them really really close to 0-. While for the lasso regression since we penalize just the parameter itself, we can say that this method is more fair but it increases the complexity of the problem since the loss function is not differentiable anymore.

2.3 Q3. Non-Linear Regression

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

1.A.

I don't think this is a good choice for such task since the family of model is able to describe a plane so it can be better if the family of model would be quadratic.

2.Q. Do you think using a feed forward neural network with activation function $h(x) = 1 - 2 \cdot x \cdot e^{-2}$ would improve the results?

2.A.

The feed forward neural network could improve it but in this case no due to the activation function. As we can see the activation function in x is not quadratic. The value e although has non linear value it's just a constant. So the output would just be a multiple linear transformations, which would lead just to a linear regression model.

3.Q. Do you think it would be possible to achieve good performance with a linear model? How?

3.A.

Yes, even if we have a linear model we can still achieve a good performance just setting the function as quadratic:

$$f(x, \theta) = \theta_0 + \theta_1 \cdot x_1^2 + \theta_2 \cdot x_2^2$$

4.Q. What is the purpose of the hidden activation function in a feed forward neural network?

4.A.

They can be divided in two types of activation:

- Linear Activation Function
- Non Linear Activation Function

The purpose of an hidden activation function in a ffnn is to apply a transformation function to the input neuron s.t we can generate the output of the neuron. This is very powerful when as activation function we have a non-linear function since we can approximate any continuous function. There are many non-linear functions such : sigmoid, hyperbolic tangent and Relu(rectified linear unit).