Methods in Machine Learning

Homework No. 2

Instructions:

- In your submission, you must provide a single Python file with a filename that follows the format: HW2_<LastName1_Lastname2>.py.
- Use the provided template code, named HW2.py, as a starting point for your implementation.
- Note that the template code is not guaranteed to be correct, and you are free to fix and modify or add pieces of code or functions as needed, but make sure that the print statements in the main function remain unchanged.
- **Avoid making any changes to the main part of the python implementation.**
- Please add your full name and ID in the first line of the python file.
- **Submission deadline: 23.4.2023**

You are given a dataset with a single input feature ($x \in \mathbb{R}^1$) and output (y).
Your task in this exercise is to fit the data to a regression quadratic model using gradient descent. The goal of this exercise is to help you better understand the concepts of empirical risk minimization, implement a gradient descend procedure, fit the model and calculate generalization error.
The python file attached includes boilerplate code that corresponds to the steps outlined in this exercise. Your task is to ensure the main section of the code runs correctly and the fitting processes result in a valid hypothesis that fits the training data and ideally generalizes well.
Basically, you only need to complete the code sections marked with '...'.
It is recommended that you read the entire exercise before start implementing the different parts.

Define the regression quadratic model

The regression quadratic model is defined as $y = w_0 + w_1 x + w_2 x^2$, where w_0, w_1, and w_2 are the model parameters to be learned. Complete the code of the `quadratic_model` function.

Define the empirical risk function

The empirical risk function measures how well the model fits the training data (it represent the loss function). It is defined as the mean squared error between the model's predictions and the true labels. Complete the missing parts in the corresponding function.

Define the gradient of the empirical risk function

The gradient of the empirical risk function tells us which direction to move the model parameters to minimize the risk. Your task here is to finalize the code that performs the derivation calculations.

Define the gradient descent algorithm

The gradient descent algorithm updates the model parameters using the gradient of the empirical risk function. The function will print the risk and model parameters at each iteration and return the final parameters that minimize the risk. Your objective here is to finalize the code so that it can conduct the fitting process by iterating a specified number of updates to the model parameters, moving them in the direction of the negative gradient.

Estimating the generalization error

The generalization error measures how well the model will perform on new, unseen data. To estimate this error, you can split the dataset into a training set and a validation set, and evaluate the model's performance on the validation set. You can then plot the training data and the estimated hypothesis using Matplotlib to visualize how well the model fits the data. Here's the code to do that:

Running the main function

The main function will load the dataset which is stored in the provided CSV file called `data.csv`, which has two columns: x and y.
Then the dataset will be split the data into train and validation set and run the gradient descend procedure to find the hypothesis that best fit into the training data.
Finally, it will print the validation risk and plot the training data and the estimated hypothesis.
The validation risk is an estimate of the **generalization error** of the model, and the plot shows how well the model fits the training data.

Debugging

Whereas we would expect that the empirical risk (loss) would decrease during the fitting process (the gradient descend), you may see that is increasing. What could be the problem? Recall what we discussed in class about the size of the learning rate. Try to decrease the learning rate and see what is the effect?

In addition, you may observe that the hypothesis (fitted model) is not close enough to the training data. What actions could you take to improve the fitting?