ECE241 PROJECT 3: The First Step in Machine Learning
Due: Dec 7, 2023, 11:59 pm on Gradescope

Introduction

In this project, you will implement your first machine learning algorithm to estimate the house pricing. You will use Linear Regression and Gradient Decent to build a model and estimate the price of given house.

This project consists of two parts: written part and programming part. In the written part, you have to finish the tasks by hand and answer some basic questions. *You need show all the steps necessary to get the solution.* You can only use the knowledge covered in lectures and discussions, those not covered in class won't get recognized with points regardless of correctness!

In the programming part, you have to write code to solve the problems and will be asked to show the results and answer some (short) questions related to your results. You may NOT use any machine learning specific libraries in your code, e.g., TensorFlow, PyTorch, or scikit-learn. You may use libraries like numpy, pandas, and matplotlib. Your code should be written in Python3.

Part I (Written)

In this part, we consider the dataset with a set of 5-feature housing price problem discussed in class. The data can be found in "part1.xls". You can download the file from Canvas. You can use Microsoft Excel to get solutions/draw figures when answer the problems below. In this case, report a screenshot of your Excel worksheet *and report the function/formula you used.*

We use feature vector $x^{(i)} = [x_1^{(i)}, x_2^{(i)}, \ldots, x_5^{(i)}]$ to represent the ith data point in the dataset and try to find a set of weights $W = [w_1, w_2, \ldots, w_5]^T$ (Note: X^T represents the transpose of matrix X) and a bias term b such that

$$prediction(x^{(i)}) = \sum_{i=1}^{5} x_i^{(i)} \times w_i + b$$

can be used to estimate the house price.

1. **[8 points]** Suppose a randomly initialized $W = [10, 1, 1, 1, 1]$ and $b = 10,000$ estimates the housing price in the following way:

$$prediction(x^{(i)}) = 10 \times x_1^{(i)} + x_2^{(i)} + x_3^{(i)} + x_4^{(i)} + x_5^{(i)} + 10000$$

   Calculate and report the estimated house price for all data points in the spread sheet.

2. **[6 points]** Evaluate how good the model by computing the Mean Squared Error (MSE) for the W and b above.

$$\text{MSE} = \frac{1}{|X|} \sum_i \left(y^{(i)} - prediction\left(x^{(i)}\right) \right)^2$$

where $|X|$ is the number of data points. $y^{(k)}$ is the true housing price for the kth house.

3. **[6 points]** Explain why *squared* is necessary to evaluate how good the model is. Mention one alternative for the square operation. (Note: you should propose something different than square, i.e., do NOT time the error with itself.)

4. **[6 points]** Take a look at the learning curve on the same dataset on three trainings shown in Figure 1. What do you think makes the different. Which one do you think is the best (report the training #) one and explain what happens to the other two.

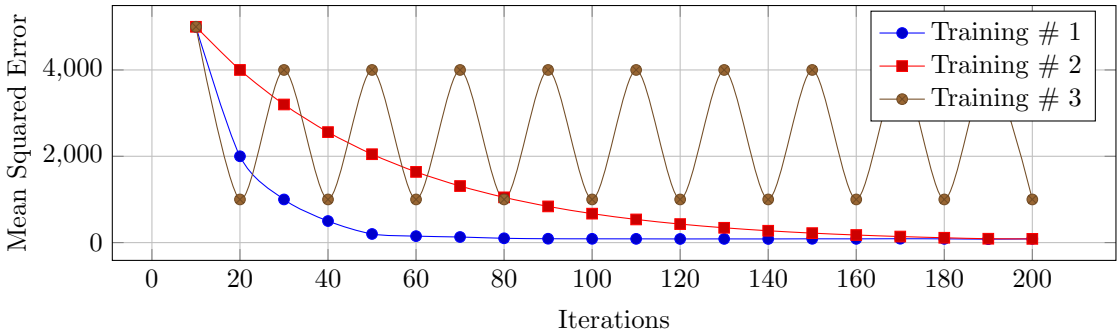

Figure 1: Learning Curve on the same dataset on three training process with different configuration.

5. **[7 points]** Suppose your model "perfectly" outputs the house price for the training set (i.e., with 0 *training* MSE), but it often times make extremely bad predictions when the model is deployed online. What might be the reason for the discrepancies in the training and deploying phase?

6. **[6 points]** In order to fix the problem in the previous question, what method discussed in class can be helpful? Write down the name of the method and describe how you can use it to fix the problem. Argue why this can help you with this problem.

Part II (Programming)

In this part, you will have access to the full housing price dataset and build a regression model to predict the house price. **You need to implement the algorithms from scratch.** The datasets to be used in this part are named "train.csv" and "test.csv". You can download the csv files from Canvas.

1. Read the training data from "train.csv".

2. **[7 points]** Before you start, always remember to take a look at the data you are going to deal with. Analyse the training set and report the following metrics:

   - How many records are there in the training set.
   - What is the mean value of the price.
   - What is the minimal and maximal price.
   - What is the standard derivation of the price.

3. **[7 points]** Show a histogram of the price.

4. **[7 points]** Some features are correlated with each other. Report a *pair-wise* scatter plot of the following features and report what you found. Describe what you could do to accelerate the training process without compromising too much accuracy. (You do not have to implement what you proposed here in the following questions.)

   - GrLivArea
   - BedroomAbvGr
   - TotalBsmtSF
   - FullBath

5. Implement function `pred` that calculates the predicted value of the price based on the current weights and feature values.

$$prediction(x^{(i)}) = \sum_{j=1}^{n} w_j \times x_j^{(i)}$$

6. Implement function `loss` that calculates the loss based on a set of predicted sale price and the correct sale price. In this task, you should implement the mean squared error as the loss function.

$$\text{MSE}(\hat{Y}, Y) = \frac{1}{|Y|} \sum_{k=1}^{|Y|} \left(\hat{y}^{(k)} - y^{(k)}\right)^2$$

   where $\hat{Y}$ is the matrix representation of the predicted housing price and Y is the matrix representation of the real housing price (i.e., $\hat{Y}[i] = prediction(x^{(i)})$, $Y[i] = y^{(i)}$).

7. Implement function `gradient` that calculates the gradient of loss function based on the predicted price and the correct price. For simplicity, we provide the gradient, you do not need to derive the expression for gradient in your code.

$$\nabla\text{MSE}(W) = \frac{2}{|Y|} \times X^T(\hat{Y} - Y)$$

   where $\nabla\text{MSE}(W)$ represent the gradient of the loss function with respect to weight W.

8. Implement function `update` that updates weights based on the gradient.

$$W_{t+1} = W_t - \alpha\nabla\text{MSE}(W_t)$$

9. Keep training your weights in your main function with Algorithm 1.

10. **[6 points]** First set α to be 0.2. Does your algorithm finds the minimal MSE? If so, report the number of iterations your algorithm converges. If not, what's happening and explain why that is the case.

11. **[12 points]** Now set $\alpha = 10^{-11}$ and $\alpha = 10^{-12}$. Run your algorithm for 500 iterations under both configurations and report a learning curve where the x-axis is the number of iterations and y-axis is the MSE. Your learning curve should have two lines in one plot and clearly identify the α value. An

Algorithm 1: Train you model.

Let W to be a randomly initialized weight matrix;
for *each iteration* **do**
    $\hat{Y} \leftarrow \text{pred}(X)$;
    $\text{MSE} \leftarrow \text{loss}(\hat{Y}, Y)$;
    $\nabla \leftarrow \text{gradient}(\hat{Y}, Y, X)$;
    $W \leftarrow W - \alpha\nabla$;

example learning curve is shown in Figure 2. Note, this is only an demonstration of what your plot should look like, this is not a learning curve of anything!

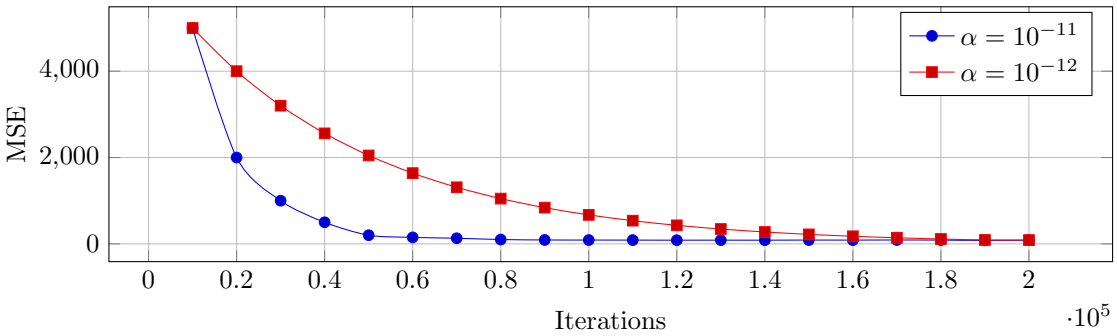

Figure 2: An example learning curve.

12. **[6 points]** For the two α, which one converges faster? Describe why it is this case.

13. **[6 points]** Predict the housing price for the test set (i.e., "test.csv"). Report the MSE for your model on the test set. In general, will your model achieves better MSE on the test set than the training set? Why?

Submission Instructions

- You should submit a report in PDF format answering all questions in the written part and show the required output in the programming part. You should also submit the code you wrote in the programming part.
- Your PDF report should have response to all tasks that has a blue indicator with corresponding points at the beginning.
- The report and the code should be submitted separately on Gradescope before the submission deadline. There will be no autograder setup for this project.
- There is no pre-submission checkpoint and no points for submitting early, but you are still encouraged to start early to avoid last-minute bugs.

- Your code for Part II should be in one file named "project3.py" or "project3.ipynb", any other file submitted to the autograder will be IGNORED! The code should be well documented and allow the grader to reproduce your result. Remember, code structure and readability consists of 10 points for this project! Failure to reproducing the results in your report will result in half of the credits for that task.