[Mini Project 2] **Project Code : SNRG**

sine-Curve Fitting using Standard Linear and Logistic Regression

Project Duration : 15-Feb-2021 ~~ 01-Mar-2021
Submission Information : (via) CSE-Moodle

Objective:
Rajesh boasted to his friends that he has been doing an online machine learning course and has now mastered regression techniques. He has not even started the course but wanted to show off in front of his friends. However he forgot the fact that the father of one of his friends was a data scientist at an eminent company. So his friend wanted to test him and thereby got his father to list out some questions which he could give Rajesh to solve. Rajesh is in a dilemma and now needs your help so that his truth does not come out.
Your task is to complete the tasks as mentioned below. It will involve writing code to perform basic curve fitting using standard regression techniques. *In particular, you have do the following:*

1. **Synthetic Data Generation and Curve Fitting**
   a. You will generate a synthetic dataset **[X,Y]** as follows. For the **X**, generate 10,000 samples randomly and uniformly in the range [0,1]. Now for **Y**, calculate **$Y = Y_0 + noise$**, where **$Y_0 = \sin(\pi x)$** and noise is to be generated randomly from a Gaussian distribution with mean = 0 and standard deviation = 0.2.
   NOTE : For all the following tasks in part 1, consider the first 500 samples (m = 500)
   b. Plot histogram of **X** to see how uniform the data is. **(FIGURE 1)**
   c. Split the dataset into Train (75%) and Test (25%) data
   NOTE : You can use standard libraries like numpy, matplotlib, sklearn for the above data preprocessing steps. For implementing the next curve fitting step, you cannot use any such library function and must write the code for h(x), RMSE loss and for running gradient descent .
   ## Numpy can still be used for array or matrix operations like matrix multiplication.
   d. Take, **n** = 3, where n is the highest power of x taken. Write a code to minimize root mean square error (**RMSE**) cost function over the training set using Gradient Descent by taking a model of the form **$h(x) = w_0 + w_1x + w_2x^2 + \text{-------} + w_nx^n$**. Use the final trained value of weights w to print RMSE on train data and test data respectively as follows
   Train RMSE error : [RMSE error value for train]
   Test RMSE error : [RMSE error value for test]
   e. Plot points of **Y vs X**, then on the same figure, plot **h(x)** so that it can be visualized how well **h(x)** fit points of **Y (FIGURE 2)**

2. **Variation of Loss/Error with m (sample size)**
   a. Take **M=500 samples.**
   b. Take **m** values from the list **= [10,20,50,100,200,500]**
   c. For each m, train the curve fitting model you used in part 1 using m samples only(no test data needed. Use all the m samples for training). Now after training, when you get the weights, use them to find the RMSE loss on the M samples(Remember you will train on m samples and calculate the RMSE loss on M samples)
   d. Print the RMSE values for each m in the following format

   Variation of RMSE with m
   10 [RMSE value at 10]

20 [RMSE value at 20]

… … … … … … … …

… … … … … … … …

e. Plot the points of RMSE values vs. m **(FIGURE 3)** and see how the error varies with m. **Justify why you think it does so.**

3. **Logistic Regression**

You are given the data for this part in the file named as **`log_reg_data.csv`**. It has two input/feature columns **X1** and **X2**, and a column for the output label **Y**

a. Perform Logistic Regression on these points taking **h(x) = sigmoid(Z)** where $\mathbf{Z = w_0 + w_1x_1 + w_2x_2 + w_3x_1^2 + w_4x_2^2}$ and perform Gradient Descent using an appropriate cost function (cost function used for linear regression won't work here). Use the final trained values of weights w to calculate accuracy on train and test data sets. Print them as follows

Train RMSE error : [RMSE error value for train]

Test RMSE error : [RMSE error value for test]

b. Plot **Y vs X** by plotting the points of **X1**, **X2** and use **Y** values to show points in different colours based on value of **Y**

Now on the same figure plot the points of curve **Z vs X** to show how well curve Z divides the points of X **(FIGURE 4)**

Note: *The program can be written in C / C++ / Java / Python programming language from scratch. No machine learning /data science /statistics package / library should be used for the implementation of curve fitting, loss and heuristic functions and weight updates in gradient descent except using numpy for standard matrix operations like matrix multiplication*

Submission Details: (to be submitted under the specified entry in CSE-Moodle)

1. ZIPPED Code Distribution in CSE-Moodle
2. A report including the justifications and plots/figures as asked in the given tasks

Submission Guidelines:

1. You may use one of the following languages: C/C++/Java/Python.
2. Your Programs should run on a Linux Environment.
3. You are **not** allowed to use any library apart from these (Also explore all these libraries if doing in Python, or equivalent of these):

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
import math
```

Your program should be standalone and should **not** use any *special purpose* library for Machine Learning. Numpy and Pandas may be used. And, you can use libraries for other purposes, such as generation and formatting of data.

4. You should submit the program file and README file and not the output/input file.
5. You should name your file as <GroupNo_ProjectCode.extension> (e.g., Group1_SNRG.pdf or Group1_SNRG.zip).
6. The submitted program file as well as the report file *should* have the following header comments:

Group Number
Roll Numbers : Names of members (listed line wise)
Project Number
Project Title

You should not use any code available on the Web. Submissions found to be plagiarised or having used ML libraries (except for parts where specifically allowed) will be awarded zero marks.

For any questions about the assignment, contact the following TA:
Ronit Samaddar (ronitsamaddar97@gmail.com)