**Homework Assignment 2**

Due: January 29, Friday, 11:59pm, 2021

The goal of this homework assignment is to master the programming of Linear Regression models. Sample codes are provided and you are required to *complete* missing lines, *evaluate* your codes and *report* your observations. Details instructions are as follows.

**Overview**. In this programing exercise, you will need to implement a linear regression algorithm for predicting student’s university GPAs from two features, i.e., Math SAT and Verb SAT. You will be instructed to program the GD method to solve the linear regression model from a set of training samples. Then, you will apply the model to predict the university GPAs of testing samples.

**Datasets**. The student’s data table is available in the attached ‘sat.csv file, which includes a data matrix. Each row represents one of the 105 students and includes five columns. We will use the columns: Math SAT, Verb SAT as input features, and the columns: University GPA as output labels. We split the whole dataset into two parts: the first 60 rows for training and the rest for testing.

**Part I: Implementing Linear Regression with Gradient Descent**

**Sample Codes**. The file “main\_ha2\_partI.py” provides the starting codes for 4 major steps: loading training and testing data, training the linear regression model, testing and evaluating the learned model.

The **first** step is to load student’s data and split them into two sets, one for training and the other for testing. The function download\_data() is used to load data from the ‘sat.csv’ file.

The **second** step will call the function gradientDescent() in GD.py, i.e., the implementation of the gradient descent method, to obtain the optimal parameters and costs over iteration. The cost function’s value over each iteration will plotted to obtain a convergence curve, as the following:



The **third** step will apply the learned model (i.e., the optimal parameters) to predict the GPA for testing students. The **last** step will return the average error and standard deviation (STD) of the evaluation results.

There are five PLACEHOLDERs in the provided scripts: three in ‘main\_ha2\_partI.py” and two in “GD.py”

*In “main\_ha2\_partI.py”,*

**PLACEHOLDER1**: you will need to change the two variables: alpha and MAX\_ITER and observe how the convergences curve and evaluation results change. Write down your observations in the report.

**PLACEHOLDER2**: You will also need to normalize the two input features and the output labels in the first step. There are a few different ways for normalizing these features, e.g., you can try scale every feature to be 0 and 1, or taking mean off every value, or others. The current lines use a function rescaleMatrix() in the dataNormalization.py, which provides other functions of normalization. Please describe in your report what the differences of these three functions. Please also test and report how the system performs differently over testing samples while using different normalization functions. Your analysis should include all the three functions.

PLACEHOLDER3: You will need to calculate the coefficient of determination (or R2) as a measure of testing performance.

*In “GD.py”,*

you will need to implement the gradient descent function gradientDescent (). Replace the temporary code lines in PLACEHOLDER with your own codes.

**PLACEHOLDER4:** write your codes to update theta, i.e., the parameters to estimate, using the gradient descent method.

**PLACEHOLDER5**: calculate the current value of the cost function with the updated model parameters, i.e. theta.

Please complete the above placeholders and study the script main\_ha2\_partI.py to accomplish the above analysis. In your report, include the convergence curves and quantitative results for each case of your studies.

**Part II: Empirical Study of Regression Models**

In addition to the ordinary linear regression model, in lecture2, we also introduced three variants of linear regressions, i.e., Ridge Regression, LASSO, and Elastic Net, which add various regularization terms to the original cost functions in order to address potential overfitting issues. The provided lecture codes (Lecture02-LinearRegression.ipynb) implement the above **four** regression models using Scikit-Learn, the machine learning Python module. In Part I of this assignment, you are instructed to implement the linear regression model from scratch.

This goal of this Part II assignment is to compare the above five regression algorithms under the same experimental setting. To proceed, you train all these five implementations using the exactly same training samples. Please use the R2 (coefficient of determination) over testing samples as the measure for success. For each algorithm, you might adjust the hyperparameters (e.g., learning rate, regularization coefficients) to get the optimal R2 over testing samples. Please include a table in your report to summarize the best R2 of each method over testing samples. Please also summarize your findings through the side-by-side comparisons of the above methods. Please clearly document your source codes used for PART II and submit them along with the .pdf report.

**Write-up**

**You need to prepare one and only one PDF file to include your writeup for both Part I and Part II.** In the report you will describe your algorithm and any decisions you made to write your algorithm a particular way. Then you will show and discuss the results of your codes following the above instructions. In the case of Part I, please show the convergence curves and quantitative results for each case of your implementations. For Part II, include the tables and any associated observations. Also, discuss anything extra you did. Feel free to add any other information you feel is relevant.

**What to Submit.**

* In your PDF report, please include all the requested results.
* **Part I**: All the files of the folder ‘part1’ and ensure the script main\_ha2\_ partI.py runs without errors. **Part II**: include all the codes you wrote for Part II in a single .py file, named ‘main\_ha2\_ partII.py’. Extra supporting files/scripts, if any, should be included as well.

**How to submit**

* Submit your source codes and report using the course site. The codes should be self-contained, and run without any error. Otherwise, penalty will be applied.