CS F320 Foundations of Data Science

Assignment-1

Submission Time & Date: 0900hrs on 19th October

Instructions:

- This assignment is a coding project and is expected to be done in groups. Each group can contain at most three members. Make sure that all members in the group are registered to this course.
- This assignment is expected to be done in Python using standard libraries like NumPy, Pandas and Matplotlib. You are not supposed to use libraries like scikit-learn for the regression models. Jupyter Notebook/Google Colab can be used. Refrain from directly copying codes/snippets from other groups or the internet as all codes will be put through a plagiarism check.
- All deliverable items (ex. .py files, .ipynb files, reports, images) should be put together in a single .zip file. Rename this file as A1_<id-of-first-member>_ <id-of-second- member>_ <id-of-third-member> before submission.

Submit the zip file using the google form link (to be announced) on or before the aforementioned deadline. Please note that this is a hard deadline and no extensions/exemptions will be given. The demos for this assignment will be held later which shall be conveyed to you. All group members are expected to be present during the demo.

Assignment 1 : Polynomial Regression and Regularization

Problem Statement

- The given dataset is a synthetic dataset consisting of 1,000 data points, each having one feature variable and one continuous target variable.
- Link to dataset : https://drive.google.com/file/d/1gggzuJKx0s-_tk23Ve4Lqm3eijznhHp1/view?usp=sharing
- This assignment extends the concepts of polynomial regression and introduces regularization and bias-variance trade-off analysis. The task will involve building regression models with varying complexity, applying regularization techniques, and analyzing the models' performance using bias and variance curves.

Task 1: Data Preprocessing

- Load the shared dataset into a pandas DataFrame.
- Normalize the feature variable by utilizing the formula: X' = (X - μ) / σ where μ represents the mean of feature value, and σ represents the standard deviation of feature values.

- Shuffle and split the dataset into training and testing sets (80% for training and 20% for testing).
- Visualize the distribution of X and Y to get an understanding of the data.

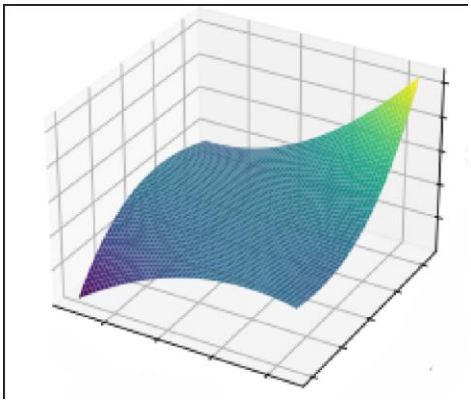
Task 2: Polynomial Regression and Regularization

- **Simple Polynomial Regression (degrees 1 to 9)**: Build polynomial regression models with degrees varying from 1 to 9. Train the models using batch gradient descent for 500 iterations.
- **Regularization**: Add L2 regularization (Ridge Regression) and build 9 new models, Experiment with different values of the regularization parameter λ between 0 and 1 (or log λ). Find the optimal λ using **Bias-Variance Decomposition i.e., a**nalyze the bias-variance trade-off for each polynomial degree of d with different values of λ. Specifically, calculate the bias, variance, and error for each λ, and plot the square of bias, variance, and square of bias + variance to find the optimal value of λ.
  You may refer to Appendix A to implement bias variance trade off.

Note: Learning rate can be varied and the best result should be documented

Task 3: Visualization of Results

- **Plot 1**: Train and test error vs. polynomial degree (1 to 9).
- **Plot 2**: Bias, variance, and total error vs. λ.to arrive the optimal λ. (Bias-Variance Tradeoff curve).
- **Plot 3**: Plot the best polynomial fit on the data points.
- **Plot 4**: A 3D plot where each x and y axis represent the hyperparameters learning rate, regularization strength and the z axis represents model performance (mean squared error) and provide an analysis from the plot about how different combinations of hyperparameters affect the model performance for all the models from degree 1 to 9.



Example of the 3d plot

Task 4: Comparative Analysis

- Perform a comparative analysis of the polynomial regression models with and without regularization.
- Discuss the behavior of the model as you increase the polynomial degree and how regularization helps mitigate overfitting.
- Analyze how the bias and variance change as you move from underfitting to overfitting polynomials.
- Report your observations on how the regularization parameter λ affects the model's complexity and performance.

What needs to be submitted ?

1. Code implementation in Python (Jupyter notebook or Colab).
2. A report that includes:
    - A detailed explanation of your approach.
    - Plots and visualizations as required by the tasks.
    - Observations on model performance, overfitting, underfitting, and regularization.

Tabulate the training and testing errors obtained using polynomial regression models of various degrees and your observations on overfitting.

In case of any queries you reach out to Teaching Assistants of the course - f20211404@hyderabad.bits-pilani.ac.in, f20202182@hyderabad.bits-pilani.ac.in

Appendix A

We can relate the Bias Variance decomposition to the commonly used terms overfitting and underfitting in the following informal way:

- ✓ ***Overfitting*** relates to having a High Variance model or estimator. To fight overfitting, we need to focus on reducing the Variance of the estimator, such as: increase regularization, obtain larger data set, decrease number of features, use a smaller model, etc.
- ✓ ***Underfitting*** relates to having a High Bias model or estimator. To fight underfitting, we need to focus on reducing the Bias in the estimator, such as: decrease regularization, use more features, use a larger model, etc.

The first step in improving generalization error is to characterize which component in the decomposition has the highest contribution, and go after that component. Unfortunately, there is no theoretically sound yet tractable way of calculating the breakdown. However, there are certain heuristics that are extremely useful. Loosely speaking:

- ✓ Training error can be treated as the amount of ***Bias in the model*** or estimator. If the model is unable to fit the training data itself well, then it is likely that the model has High Bias. This is the underfitting regime.
- ✓ Gap between cross-validation error and Training error can be treated as the ***Variance of the model*** or the estimator. If the Training error is low but the Cross-Validation error is high, it is very likely that model has High Variance. This is the overfitting regime.

We should always analyse the model performance by looking at the training error and cross-validation error simultaneously. This is the only tractable (albeit heuristic) way to obtain an estimate of the Bias and Variance components. Only then should we take steps that are targeted towards addressing either Bias or Variance purposefully.

Steps taken to fight overfitting (i.e. fight High Variance) generally do not necessarily help fight underfitting (i.e. High Bias). For example, it is futile to spend time and resources in obtaining more data (technique to fight High Variance) when the training error itself is high (symptom of High Bias).

Similarly, steps taken to fight underfitting (i.e. fight High Bias) generally do not necessarily help fight overfitting (i.e. High Variance). For example, it is futile to switch to a larger neural network (technique to fight High Bias) when the gap between cross-validation error and training error is high (symptom of High Variance).

Many times, steps taken to fight one (either High Bias or High Variance) can end up worsening the other. This is essentially how the Bias Variance trade-off is encountered in practice.