EAI6020: AI System Technologies

Prepared By:
Achilefu Maduabughichi
Okeke Onyedikachukwu

Assignment 1:
Regression Problem Using Neural Network and Bias-Variance Trade-off in PyTorch

Instructor:
Prof. Siddharth Rout

November 2024

Part 1: Report on Overfitting, Underfitting, and Bias-Variance Trade-off

---

Overfitting

Overfitting occurs when a machine learning model learns the details and noise in the training data to such an extent that it negatively impacts the performance of the model on new data. This is typically due to the model being too complex, with too many parameters relative to the amount and variability of training data. Indicators of overfitting include low training error but high validation error.

Underfitting

Underfitting happens when a machine learning model is too simple to capture the underlying patterns in the data, resulting in poor performance on both the training and validation datasets. It usually arises when the model lacks sufficient parameters, or the data is not well-represented by the model's assumptions. Underfitting can lead to high bias, meaning the model consistently makes errors.

Bias-Variance Trade-off

The bias-variance trade-off is a fundamental challenge in supervised learning.

- **Bias** refers to the error introduced by approximating a real-world problem with a simplified model. High bias can cause the model to miss relevant patterns, resulting in underfitting.
- **Variance** is the model's sensitivity to fluctuations in the training data. High variance can lead to overfitting, where the model fits to the noise in the data rather than the signal.

To achieve a balance, model complexity must be optimized. Adding layers or features reduces bias but increases variance, while simplifying the model increases bias but reduces variance. The goal is to find an ideal point where the model generalizes well to new data.

Part 2: Report on Regression Problem Using Neural Network and Bias-Variance Trade-off in PyTorch

---

1. Introduction

In this assignment, we modeled a regression problem using a dataset of loan originations from various educational institutions in the U.S. across multiple states. The main goal was to predict the amount of loans originated, based on various institutional and geographic features. We developed a neural network model using PyTorch, and we focused on understanding the bias-variance trade-off during the model training process.

2. Data Preprocessing

Before training the neural network, several preprocessing steps were performed on the dataset:

- **Data Cleaning**: Columns containing dollar values were cleaned by removing special characters such as $ and ,. Missing values were handled by replacing placeholders like - with 0 for consistency.

- **Encoding Categorical Variables**: Categorical columns such as 'School', 'State', and 'School Type' were transformed using one-hot encoding to ensure they could be utilized in the model.
- **Normalization**: Numerical columns, excluding identifiers such as 'OPE ID' and 'Zip Code', were standardized using StandardScaler to ensure that all features contributed equally to the model.
- **Feature Selection**: The target variable was the dollar amount of loans originated ($ of Loans Originated). The remaining columns, excluding identifier columns, were used as features.

The final feature matrix was converted into NumPy arrays, and missing values were filled appropriately.

3. Model Design

A feedforward neural network (also called a fully connected network) was implemented using PyTorch. The network consists of:

- **Input Layer**: The size of this layer corresponds to the number of features in the dataset.
- **Hidden Layers**:
    - The first hidden layer has 50 neurons, followed by the ReLU activation function.
    - The second hidden layer has 20 neurons, again followed by ReLU activation.
- **Output Layer**: This layer consists of a single neuron for predicting the continuous target variable (loan amounts).

This architecture balances model complexity to avoid overfitting (high variance) and underfitting (high bias), key considerations in the bias-variance trade-off.

4. Training and Validation Process

The dataset was split into training (80%) and validation (20%) sets to monitor the model's performance on unseen data.

- **Loss Function:** We used Mean Squared Error (MSE) to measure the difference between predicted and actual values, as it is suitable for regression tasks.
- **Optimizer**: The model was optimized using the Adam optimizer with a learning rate of 0.001. This optimizer was chosen for its ability to handle sparse gradients efficiently.

The training process was run for 1000 epochs. At each epoch, both training and validation losses were computed and tracked.

5. Performance Metrics

After training, the following performance metrics were calculated to evaluate the model:

- **Mean Squared Error (MSE)**: 0.19344144
- **Mean Absolute Error (MAE)**: 0.33423007
- **Root Mean Squared Error (RMSE)**: 0.19344144

6. Bias-Variance Tradeoff

During the training process, we observed the bias-variance trade-off by tracking the training and validation losses across 1000 epochs. Key observations include:

- **Training Loss:** The training loss decreased consistently, starting at 1.0687 in the first epoch and reaching 0.00005 by the 100th epoch. This demonstrates the model's ability to fit the training data well, suggesting low bias.
- **Validation Loss:** The validation loss also decreased, from 0.6916 to 0.2936 over the course of training. However, after a certain point, it stabilized, indicating that further training would not improve generalization. In fact, it suggests a point at which the model might begin to overfit if trained further, leading to high variance.
- **Training Loss**: The training loss consistently decreased from 1.1527 in the first epoch to near-zero values by epoch 1000, demonstrating that the model fit the training data well.
- **Validation Loss**: The validation loss decreased steadily from 0.7759 to 0.2099 over the 1000 epochs, indicating that the model was able to generalize to unseen data. However, the diminishing improvement in validation loss after a certain point suggested the model had reached its optimal level of generalization.

7. Results and Visualization

The training and validation losses were plotted across **1000 epochs** to visualize the bias-variance trade-off. Initially, both losses decreased, but as training progressed, the gap between them widened, demonstrating a typical trade-off between bias and variance:

Training Loss Summary:

- **Training Loss (Epoch 1)**: 1.0687
- **Training Loss (Epoch 1000)**: 0.00005
- **Validation Loss (Epoch 1)**: 0.6916
- **Validation Loss (Epoch 1000)**: 0.2936

These metrics indicate that the model was effective in minimizing the error between the predicted and actual loan amounts.

A graph was plotted to visualize the training and validation losses over the epochs. This visualization clearly showed that while the training loss kept improving, the validation loss reached a minimum and stabilized, indicating an optimal model fit with balanced bias and variance.

8. Conclusion

The neural network model successfully modeled the regression problem, achieving a good balance between bias and variance. This project highlights the importance of monitoring the bias-variance trade-off to ensure the model performs well on unseen data.

The training and validation losses demonstrated effective learning, with the validation loss stabilizing, indicating that the model was reaching its generalization limit. The calculated MSE, MAE, and RMSE values confirmed that the model performed well in predicting loan amounts with minimal error.

9. References

- Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press.
- Heaton, J. (2015). *Artificial Intelligence for Humans, Volume 3: Deep Learning and Neural Networks*. Heaton Research, Inc.
- Nielsen, M. (2015). *Neural networks and deep learning*. Determination Press.

Appendix

- **Loss Plot**:
    - A plot of training and validation losses across epochs shows the point of diminishing returns where validation loss no longer improves significantly.
    - The annotations on the graph point out where underfitting and overfitting occur, emphasizing the importance of stopping training early to avoid overfitting.