WOA7015 Advanced Machine Learning

Individual Assignment 2

Mohamed Aidiel Haikal Bin Mat Ziat

22082088

Introduction

This report compares the performance of a manually implemented logistic regression algorithm with a logistic regression model using PyTorch's built-in functions. The evaluation metric for comparison is prediction accuracy on a test dataset. The dataset, both training and testing, is a toy dataset consisting of two features (X_1, X_2) and a binary class label y.

Methodology

For the manual implementation, functions are derived based on the underlying mathematical concept of logistic regression, as outlined in the source code. These functions use only the NumPy library to handle the data and perform calculations. The key functions implemented are:

1. `CalcObj`: Calculates the objective function.
2. `CalcGrad`: Computes the gradient of the objective function.
3. `UpdateParams`: Use gradient descent to update weights.
4. `CheckConvg`: Checks for convergence to stop gradient descent.
5. `GradientDescent`: Combines functions 1-4 to execute the gradient descent algorithm.
6. `PredictLabels`: Predicts class labels for the test set.

For the PyTorch implementation, a logistic regression model was created using the following components:

1. Model: Defined using PyTorch's `nn.Linear` for linear classification and `torch.sigmoid()` for the logistic function.
2. Loss Function: Binary Cross-Entropy Loss.
3. Optimizer: Stochastic Gradient Descent (SGD) to minimize the loss function.

Note that for both methods, the decision threshold is assumed to be 0.5 i.e., if the predicted probability is greater or equal to 0.5, then the predicted class will be 1, and vice versa. Additionally, the PyTorch implementation includes an early stopping algorithm, which stops the training if the training loss does not decrease more than the tolerance set at 0.001 for at least 10 consecutive epochs (i.e., the patience level, set by user). This approach ensures that neither method trains longer than necessary.

Results

The results from both approaches are compared based on prediction accuracy.

Table 1

Prediction accuracy on toy test dataset for manual implementation and PyTorch function.

Evaluation metric	Manual implementation	PyTorch built-in function
Number of wrong predictions	1	1
Accuracy	0.95	0.95

As noted in Table 1, the prediction accuracy for both approaches is the same at 0.95, with only 1 out of 20 data points incorrectly predicted.

Discussion

Both approaches utilize logistic regression with gradient descent for the learning process and include early stopping. As such, the results are expected to be similar, as reflected in the consistent accuracy achieved by both the manual and PyTorch implementations.

Conclusion

In conclusion, this comparison highlights the consistency of logistic regression models implemented both manually and through PyTorch’s built-in functions. By mirroring the same core mechanics – gradient descent optimization with early stopping – both approaches achieve the same level of accuracy. This exercise also provides user a deeper understanding of how logistic regression works internally, which is crucial for effectively implementing and troubleshooting machine learning algorithms in real-world applications.