WEEK-12- Neural Network

1. Using the data synthesis R script provided by the instructor as part of the week 11 assignment instructions, produce datasets of the following sizes, and fit deep learning models with the configurations shown below. Associated with each model, record the following performance characteristics: training error, validation (i.e., holdout set) error, time of execution. Use an appropriate activation function.

Data Size	Configuration	Training error	Validation error	Time of execution
1000	1 hidden layer (4n)	0.0950	0.0600	19.26 seconds
10000	1 hidden layer (4n)	0.0030	0.0010	37.96 seconds
100000	1 hidden layer (4n)	0.0014	0.0012	207.25 seconds
1000	2 hidden layers (4n)	0.0450	0.0600	7.69 seconds
10000	2 hidden layers (4n)	0.0024	0.0030	25.04 seconds
100000	2 hidden layers (4n)	0.0017	0.0015	191.37 seconds

2. Based on the results, which model do you consider superior, among the deep learning models fit?

Among the results, the **1-hidden-layer model with 4 nodes** is considered the superior deep learning configuration. It achieved the **lowest overall validation error (0.0012)** on the largest dataset (100,000 rows), demonstrating stronger generalization capabilities. Its performance remained **consistently strong across all dataset sizes**, particularly excelling with larger data, where deep learning typically performs best. Although the **2-hidden-layer model trained faster** on smaller datasets (e.g., 7.69 seconds vs. 19.26 seconds on 1,000 rows), the **1-layer model matched or exceeded it in validation accuracy**, especially on medium and large datasets. Even at the 100,000-row scale, despite a slightly longer training time (207.25s vs. 191.37s), it maintained **lower training and validation errors**, proving more accurate overall. In conclusion, the **1-hidden-layer model** offers better scalability and accuracy, making it the most effective choice for high-volume deep learning tasks where precision outweighs training speed.

3. Next, report the results (for the particular number of observations) from applying xgboost (week 11 – provide the relevant results here in a table). Comparing the results from xgboost and deep learning models fit, which model would you say is superior to the others? What is the basis for your judgment?

The results demonstrate that training efficiency directly opposes predictive accuracy. Xgboost using Python with scikit-learn achieves the highest accuracy of 94.40% in under a second when working with 1,000-row datasets, while delivering superior results than both deep learning models. The deep learning models demonstrated slightly reduced accuracy while requiring extensive training durations when operating on this dataset size. The deep learning models achieve better accuracy than Xgboost when the dataset reaches 10,000 rows by reaching 97.54% accuracy. Deep learning models require more than 15 times longer training duration than Xgboost to achieve their improved accuracy. The performance improvement requires evaluation against the increased computational expenses, which must be considered for time-sensitive and resource-constrained applications. Deep learning reaches its peak performance with large datasets of 100,000 rows to achieve 98.67 % accuracy. The training duration becomes excessively long when the dataset reaches this size, since it exceeds 150 seconds. Xgboost maintains outstanding performance at this scale by reaching 98.65% accuracy within 6.35 seconds, which makes it more suitable for real-world applications. Overall, if maximum accuracy is the only priority and computational resources are not constrained, then deep learning models, particularly with one hidden layer, are a strong choice. However, when speed, resource efficiency, and ease of integration into production pipelines matter most, Xgboost with Python and scikit-learn stands out as the superior model. It balances high accuracy with rapid execution, making it ideal for most practical machine learning applications.