**Critical Thinking 3**

Sajith Joseph

Colorado State University Global

CSC580: Applying Machine Learning and Neural Networks - Capstone

Instructor: Dr. Pubali Banerjee

08/31/2025

**Linear Regression Using TensorFlow**

**Overview**

This assignment implements a simple linear regression model using TensorFlow. Synthetic linear data with noise is generated (y = 2x + 3 + uniform noise), then normalized to prevent numerical instability. A TensorFlow v1-style computation graph is built with placeholders for inputs (X, Y), trainable parameters (W, b), and an optimizer (SGD) to minimize MSE. The model is trained for 1000 epochs with learning\_rate=0.0001, and the final cost, weight, and bias are reported.

**Inputs and Outputs**

Inputs: 50 synthetic (x, y) data points formed from a linear trend (y = 2x + 3) with added uniform noise in range [-4, 4], then normalized using z-score normalization. Outputs: Final training cost, learned weight (W), and bias (b). Two plots are produced: a scatter plot of the normalized training data and a line plot showing the fitted regression line over the data.

**How to Run**

1. Ensure Python 3.8+, TensorFlow (2.x OK with tf.compat.v1), NumPy, and Matplotlib are installed.

2. Run `python linear\_regression\_tf1.py`.

3. The script prints training progress every 100 epochs and final results to console, and saves final cost, W, and b to `results.txt`.

4. The plots `training\_data.png` and `fitted\_line.png` are saved in the same directory.

**Technical Notes**

- Uses TensorFlow 1.x compatibility mode (tf.compat.v1) for placeholder/session semantics

- Fixed random seeds (numpy=101, tensorflow=101) for reproducible results

- Data normalization prevents numerical instability during training

- Learning rate reduced to 0.0001 for stable convergence

**Screenshot of training\_data.png:**

