AI Tools Assignment – Complete Report

Prepared by: Fredric Bobby

Part 1: Theoretical Understanding

Q1: TensorFlow provides static computation graphs and excellent production deployment tools, while PyTorch uses dynamic computation graphs ideal for research and experimentation. TensorFlow is often used for deployment, whereas PyTorch excels in flexibility during development.

Q2: (1) Used for quick model experimentation and visualization. (2) Supports reproducible data exploration and documentation in one notebook.

Q3: spaCy enhances NLP with pretrained pipelines for tokenization, POS tagging, and NER—going far beyond basic string operations by understanding linguistic context.

Comparative Analysis: Scikit-learn focuses on classical ML (regression, clustering) with easy APIs, while TensorFlow targets deep learning. Scikit-learn is simpler for beginners; TensorFlow has better deployment support.

Part 3: Ethics & Optimization (10%)

1. Ethical Considerations

Potential Biases:
In the MNIST model, bias may arise from class imbalance — for instance, certain digits appearing more frequently, leading to uneven accuracy. In Amazon Reviews, sentiment analysis can reflect cultural or linguistic bias since some expressions vary in tone across user groups.

Mitigation Tools:
- **TensorFlow Fairness Indicators** help visualize model performance across subgroups (e.g., gender, language). They highlight disparities and assist in retraining the model more fairly.
- **spaCy Rule-Based Systems** allow adding linguistic patterns that reduce false positives and adapt sentiment analysis to specific contexts (e.g., product-related slang). These ensure more equitable treatment of diverse user data.

2. Troubleshooting Challenge

A common TensorFlow bug involves mismatched dimensions between model output and labels or using an inappropriate loss function. For example, using categorical_crossentropy with integer labels instead of one-hot encoding.

Fix:
1. Ensure labels are correctly shaped (e.g., use sparse_categorical_crossentropy for integer labels).
2. Check final Dense layer units match class count (e.g., 10 neurons for MNIST).
3. Verify input dimensions in Conv2D match image shape: (28, 28, 1).
4. Always call model.summary() to debug layer outputs.

End of Report