MACHARIA JOSEPH

WEEK 3 Assignment

Short Answer Questions

Question 1

Primary difference

Feature	TensorFlow	PyTorch
1. Origin	Developed by Google Brain	Developed by Facebook's AI Research
2. Ecosystem	Part of larger Google ecosystem	Tightly integrated with python
3. API style	Multiple APIs (Keras)	More unified API
4. Graph Definition	Static computational graphs	Dynamic computational graphs
5. Debugging	More difficult	Easier
6. Deployment	Strong production deployment tools	Improving but historically weaker
7. Visualization	TensorBoard	TensorBoard + others
8. Community	Larger enterprise adoption	Strong in academia and research

When to Choose TensorFlow

1. **Production Deployment**: When you need to deploy models at scale (TF Serving, TF Lite, TF.js)
2. **Mobile/Edge Devices**: Better support for mobile and embedded systems
3. **Enterprise Environments**: When working with Google Cloud services or existing TF infrastructure
4. **Large-Scale Distributed Training**: Mature support for distributed training across multiple devices
5. **Pre-trained Models**: Access to many production-ready models via TF Hub

When to Choose PyTorch

1. **Research & Prototyping**: Faster experimentation with dynamic graphs
2. **Academic Projects**: Dominant in most recent ML research papers
3. **Python-Centric Workflows**: More intuitive for Python developers
4. **Custom Model Architectures**: Easier to implement novel architectures
5. **Debugging**: Immediate error feedback with eager execution
6. **Computer Vision**: Strong adoption in CV (TorchVision)

Question 2

Why Jupyter?

i. Allows iterative execution of code cells, making it easy to inspect data step-by-step.
ii. Supports inline visualizations (Matplotlib, Seaborn, Plotly) for quick insights.

Question 3

SpaCy improves upon simple string manipulation:

1. Linguistic understanding
2. Advanced text processing capabilities
3. Efficiency at scale
4. Pre-trained models

Basic String Operation work on:

1. Simple exact-match searches where linguistic understanding isn't needed
2. Extremely constrained environments where spaCy's model size is prohibitive

3. Preprocessing steps before NLP (e.g., removing special characters)

Comparative Analysis

Scikit-learn vs. TensorFlow

1. Target applications

Scikit-learn

- **Best for:**
  - Traditional ML algorithms (linear regression, SVM, random forests)
  - Tabular data (CSV, Excel, structured datasets)
  - Small to medium-sized datasets (CPU-friendly)
  - Feature engineering, model selection, and evaluation

TensorFlow

- **Best for:**
  - Deep learning (CNNs, RNNs, transformers)
  - Unstructured data (images, text, audio)
  - Large-scale datasets (GPU/TPU acceleration)
  - Custom neural architectures (flexible layer design)

2. Ease of Use for Beginners

Scikit-learn (Easier)

  - Simple, consistent API (fit(), predict(), score())
  - Minimal setup (no GPU required)
  - Great for learning ML fundamentals

TensorFlow (Harder)

    - High-level APIs (Keras) simplify deep learning
    - Pre-trained models available (TF Hub)

3. Community Support

Scikit-learn

- **Strong documentation** with clear examples
- **Widely taught** in academia and bootcamps
- **Stable API** (few breaking changes)

TensorFlow

- **Large but fragmented** (TF 1.x vs. 2.x, Keras vs. standalone)
- **More specialized** (Stack Overflow, GitHub, TF forums)
- **Industry adoption** (Google, Uber, Airbnb use TF in production)