**Part 1: Theoretical Understanding**

**1. Short Answer Questions**

**Q1: Difference between TensorFlow and PyTorch**

**TensorFlow**: Static graphs (optimized deployment), better for production, supports TensorFlow Lite/JS/Serving.

**PyTorch:** Dynamic graphs (define-by-run), easier for debugging and research.

Choose TensorFlow for **scaling** or PyTorch for **experimentation**.

**Q2: Use cases for Jupyter Notebooks**

* Interactive prototyping (visualizing models, inline output).
* Documentation + experiment tracking (with markdown and code together).

**Q3: spaCy vs basic string ops**

* spaCy uses **tokenization, POS tagging, NER**, and pre-trained models.
* Basic Python string ops can’t handle linguistic structures like “Apple” as a company vs. fruit.

### 2. ****Comparative Table: Scikit-learn vs TensorFlow****

| **Feature** | **Scikit-learn** | **TensorFlow** |
| --- | --- | --- |
| Application | Classical ML (SVM, Trees) | Deep learning (CNNs, RNNs, Transformers) |
| Ease of Use | Beginner-friendly | Steeper learning curve |
| Community | Strong for traditional ML | Huge community, production-ready |