THEORITICAL UNDERSTANDING

**1. Short Answer Questions**

**Q1: Differences between TensorFlow and PyTorch**

| **Criteria** | **TensorFlow** | **PyTorch** |
| --- | --- | --- |
| Computation | Static graph (eager available via tf.function) | Dynamic graph (eager by default) |
| Deployment | Easier deployment with TensorFlow Serving, TFLite, TensorFlow.js | Slightly more complex deployment |
| Debugging | Requires sessions (unless eager mode) | Pythonic, native debugging |
| When to Use | Production-grade apps, mobile, embedded, scalable cloud models | Prototyping, research, fast iteration |

**✅ Choose TensorFlow** for production systems.  
**✅ Choose PyTorch** for experimentation and rapid development.

**Q2: Two Jupyter Notebook Use Cases in AI**

1. **Interactive model prototyping:**  
   Test and tweak AI models in an interactive, incremental way.
2. **Visualizing AI results:**  
   Plot model accuracy, confusion matrices, or prediction outputs right inside notebook cells.

**Q3: How spaCy enhances NLP over basic string operations**

* **Advanced NLP tasks like NER, POS tagging, dependency parsing** without manual regex.
* Uses pretrained language models with contextual word understanding.
* Far more scalable and accurate for language tasks than simple .split(), .replace(), or re.sub() functions.

**2. Comparative Analysis: Scikit-learn vs. TensorFlow**

| **Criteria** | **Scikit-learn** | **TensorFlow** |
| --- | --- | --- |
| Target Application | Classical ML (SVM, Decision Trees, kNN) | Deep Learning (CNNs, RNNs, Transformers) |
| Ease for Beginners | Very beginner-friendly, simple API | Moderate learning curve, but improving |
| Community Support | Mature, large, stable community | Massive, especially in production AI |