1. Short Answer Questions

Q1: Explain the primary differences between TensorFlow and PyTorch. When would you choose one over the other?

TensorFlow excels in production deployment with robust tools like TF Serving and TensorFlow Lite, making it ideal for scalable applications, while PyTorch's dynamic computation graphs and Pythonic design make it better for research and rapid prototyping.

Q2: Describe two use cases for Jupyter Notebooks in AI development.

- It enables rapid prototyping and visualization
- It serve as shareable research notebooks

Q3: How does spaCy enhance NLP tasks compared to basic Python string operations?

- NLP enhances tasks by providing pre-trained models, linguistic annotations, and optimized tokenization.
- Python treats text as a sequence of characters without understanding meaning, grammar, or context.

2. Comparative Analysis

Compare Scikit-learn and TensorFlow in terms of:
Target applications (e.g., classical ML vs. deep learning).
Ease of use for beginners.
Community support.

- **Scikit-learn** is best suited for classical machine learning tasks such as regression, classification, and clustering, while **TensorFlow** is designed for deep learning applications like image recognition, natural language processing, and time series modeling.
- For beginners, Scikit-learn offers a more accessible and user-friendly interface with simple and consistent APIs, making it easier to learn and apply. In contrast, TensorFlow has a steeper learning curve, though its high-level Keras API helps simplify deep learning model development.
- Both libraries have strong community support, but TensorFlow benefits from a broader ecosystem and active backing from Google, making it more prominent in deep learning research and large-scale production environments.