PART ONE

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

TensorFlow employs a more community-oriented process that has very good support of deployment and production tools, whereas PyTorch is more flexible and easier to use, particularly on research and prototyping. TensorFlow initially ran on a static computation graph and PyTorch runs on dynamically typed computation graphs which can be executed in real-time and be debugged with ease.

You would use TensorFlow with research projects, fast prototyping and when flexibility is a requirement. TensorFlow should be used in cases where you have to scale and serve models, production environments, and mobile deployment.

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

1. **Model Prototyping:** Using Jupyter Notebooks, AI developers can write and test code in small chunks, easily building, debugging and experimenting with models interactively.
2. **Data Analysis and Visualization:** They work best to explore datasets, generate visualizations and share findings, and are thus ideal to present AI workflows and findings.

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

spaCy offers sophisticated natural language processing capabilities such as tokenization, part-of-speech tagging, named entity recognition and dependency parsing, that cannot be dealt with using regular Python string manipulation. The spaCy is in contrast to simple string approaches; however, the linguistic intuition is an easier way to analyze the text with increased and productive accuracy and meaning.

2. Comparative Analysis

- Compare Scikit-learn and TensorFlow in terms of:

  a. **Target applications (e.g., classical ML vs. deep learning).**

  Scikit-learn prevailingly supports classical machine learning such as decision trees, support vector machines, and linear regression. TensorFlow is also developed in deep learning, neural networks, e.g. CNN, RNN and large-scale AI models.

  b. Ease of use for beginners.

  Scikit-learn is less formidable with an unified, basic API and easy installation. The more complicated part of TensorFlow, particularly in deep learning applications, is easily overcome by the increased usability of TensorFlow in 2.x with Keras.

  c. Community support.

  Both are well-supported communities, but production and usage in industry tend towards TensorFlow. Scikit-learn is still used in the classroom and more generally by data scientists on conventional ML.