AI Tools and Applications – Part 1: Theoretical Understanding

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

TensorFlow and PyTorch are both open-source machine learning frameworks widely used in AI development, but they differ in how they operate and who they serve best.

TensorFlow:
- Uses static computation graphs (define-and-run)
- Steeper learning curve, more complex for beginners
- Strong deployment support (e.g., TensorFlow Lite, TensorFlow.js)
- Backed by Google, optimized for production

PyTorch:
- Uses dynamic computation graphs (define-by-run)
- Easier to use and debug (Pythonic syntax)
- Rapid prototyping and research-friendly
- Backed by Facebook, gaining popularity

When to choose which:
- Choose TensorFlow for production-ready systems, mobile/web deployment, or when using the TensorFlow ecosystem.
- Choose PyTorch for flexibility, experimentation, and research workflows.

2. Describe two use cases for Jupyter Notebooks in AI development.

1. Exploratory Data Analysis (EDA):
Jupyter makes it easy to load datasets, clean data, visualize trends, and test small code blocks interactively. This helps AI developers understand the data before building models.

2. Machine Learning Model Development:
Notebooks allow step-by-step testing of models with live feedback. Developers can run training, tuning, and evaluation all in one place, with the ability to include plots, markdown explanations, and experiment tracking.

3. How does spaCy enhance NLP tasks compared to basic Python string operations?

spaCy is a modern, fast NLP library designed for industrial use. It provides advanced natural language processing features that basic Python string tools lack.

Compared to basic Python:
- spaCy performs accurate tokenization and sentence segmentation
- Includes built-in Named Entity Recognition (NER)
- Supports Part-of-Speech (POS) tagging and dependency parsing
- Offers pre-trained pipelines for multiple languages
- Optimized for speed and efficiency in real-world NLP tasks

4. Comparative Analysis: Scikit-learn vs TensorFlow

Target Applications

- Scikit-learn is ideal for classical machine learning (e.g., decision trees, regression, clustering) with structured/tabular data.
- TensorFlow is best for deep learning tasks like image recognition, NLP, and time series forecasting involving unstructured data.

Ease of Use

- Scikit-learn has a simple, beginner-friendly API that is consistent across models.
- TensorFlow is more complex (especially in earlier versions), but TensorFlow 2.x with Keras has improved usability.

Community Support

- Scikit-learn has strong academic and research backing, with excellent documentation and tutorials.
- TensorFlow, backed by Google, has a vast global community and integrates with tools like TensorBoard, TensorFlow Lite, and TensorFlow Serving.

Conclusion

This part of the assignment highlights the differences between popular AI tools and their real-world use cases. TensorFlow and PyTorch are best suited for deep learning, while Scikit-learn remains the go-to for classical ML. Jupyter Notebooks offer an ideal interface for experimenting and documenting work, and spaCy brings powerful, production-ready NLP to Python workflows.