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

- **Programming Style**: PyTorch uses a dynamic computation graph which is more intuitive and Pythonic, while TensorFlow originally used static computation graphs
- **Debugging**: PyTorch integrates seamlessly with Python debuggers like `pdb`, making it easier to debug. TensorFlow's debugging was historically more complex.
- **Deployment**: TensorFlow has stronger production and deployment tools, such as TensorFlow Serving, TensorFlow Lite, and TensorFlow.js.
- **Community and Ecosystem**: PyTorch is preferred in research due to its flexibility and readability. TensorFlow is more common in industry applications due to its mature ecosystem.

When to choose:

- **PyTorch**: Best for research, rapid prototyping, and academic use.
- **TensorFlow**: Better for production deployment and when working with mobile/web integration or complex pipelines.

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

1. **Experimentation and Prototyping**: Jupyter allows researchers and developers to quickly test models, tweak hyperparameters, and visualize results inline, ideal for iterative development.
2. **Education and Documentation**: Notebooks support rich text, equations, and code together, making them perfect for tutorials, learning materials, or sharing reproducible experiments.

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

Answer:

spaCy provides **advanced NLP capabilities** such as:

- Tokenization, part-of-speech tagging, named entity recognition (NER), and dependency parsing — which are far beyond basic string manipulation.
- Pre-trained models for multiple languages and efficient performance on large texts.
- Better text understanding through linguistic features, unlike basic string operations which are limited to pattern matching, splitting, and searching.

Comparative Analysis

Compare Scikit-learn and TensorFlow in terms of:

Aspect	Scikit-learn	TensorFlow
Target Applications	Classical Machine Learning (e.g., SVM, Random Forest, Logistic Regression)	Deep Learning (e.g., Neural Networks, CNNs, RNNs, Transformers)
Ease of Use	Very beginner-friendly with a simple API and clear documentation	Steeper learning curve, especially for custom models and advanced use
Community Support	Large and active community in traditional ML	Large global community, particularly strong in deep learning and production tools

Part 2: Ethical Considerations; Potential Bias

1. MNIST Model

Bias Type: Dataset bias

> MNIST only includes grayscale digits written by predominantly American Census Bureau workers — it's **not representative** of global handwriting styles (e.g., different writing tools, left-handed writing)- May underperform on real-world digit images, especially from diverse age groups or non-Western populations.

2. Amazon Reviews NLP Model

Bias Type: Sentiment or language bias- The model may misinterpret reviews that contain:

  - **Sarcasm** ("Great, it broke in two days!")
  - **Cultural expressions** not in the training corpus
  - **Underrepresented product categories or brands**
- **Consequence**: Mislabelling sentiment can affect downstream decisions like product ranking or customer support triage.