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

TensorFlow vs PyTorch Differences

Primary Differences:

- **Computational Graph**: TensorFlow uses static graphs (define then run), while PyTorch uses dynamic graphs (define by run)
- **Debugging**: PyTorch offers easier debugging with standard Python debugging tools; TensorFlow requires specialized debugging tools
- **Deployment**: TensorFlow has superior production deployment options (TensorFlow Serving, TensorFlow Lite)
- **Learning Curve**: PyTorch is more intuitive for beginners; TensorFlow has steeper learning curve but more comprehensive ecosystem

When to Choose:

- **TensorFlow**: Production deployment, mobile/web applications, enterprise solutions, when you need TensorBoard integration
- **PyTorch**: Research, rapid prototyping, academic work, when debugging is crucial, dynamic neural networks

Describe two use cases for Jupyter Notebooks in AI development.

Jupyter Notebooks in AI Development

Use Case 1: Exploratory Data Analysis (EDA)

- Interactive data visualization and statistical analysis
- Quick iteration on data preprocessing steps
- Easy sharing of analysis with stakeholders through markdown cells

Use Case 2: Model Prototyping and Experimentation

- Rapid testing of different algorithms and hyperparameters
- Step-by-step model development with immediate feedback
- Documentation of experiments with markdown explanations

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

spaCy Enhancements:

- **Linguistic Intelligence**: Understands grammar, syntax, and semantics rather than just pattern matching
- **Pre-trained Models**: Comes with models trained on large corpora for accurate tokenization, POS tagging, and NER

- **Efficiency**: Optimized Cython implementation, much faster than regex-based string operations
- **Advanced Features**: Dependency parsing, word vectors, sentence segmentation that would require complex regex patterns

Q2. 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.

Aspect	Scikit-learn	TensorFlow
Target Applications	Classical ML (SVM, Random Forest, clustering, regression)	Deep learning, neural networks, large-scale ML
Ease of Use	Very beginner-friendly, consistent API, minimal code	Steeper learning curve, more verbose, complex setup
Community Support	Strong documentation, extensive tutorials, stable API	Huge community, frequent updates, extensive resources but fragmented