```
Start coding or generate with AI.
```

Q1: Differences Between TensorFlow and PyTorch

Feature	TensorFlow	PyTorch
Syntax	Static graph (TF 1.x), dynamic in TF 2	Dynamic computational graph
Debugging	More complex (better in TF 2.x)	Easier due to Pythonic structure
Deployment	TensorFlow Serving, TF Lite, TFJS	TorchServe (less mature)
Preferred for	Production-grade apps, mobile/edge	Research and prototyping

When to use:

- Use **TensorFlow** for production and deployment.
- Use **PyTorch** for fast experimentation and academic research.

Q2: Use Cases of Jupyter Notebooks

1. **Interactive AI Development**: You can run code cells, visualize outputs immediately, and experiment step-by-step with models.
2. **Reproducible Research**: Shareable with visualizations, markdown, code, and outputs in one file — ideal for collaboration and documentation.

Q3: spaCy vs Basic Python spaCy provides efficient NLP pipelines including tokenization, NER, POS tagging, dependency parsing — much faster and more accurate than using `str.split()` or regex.

For example:

- `spacy` identifies "iPhone 13" as a product.
- `Python string` just sees it as two words.

Feature	Scikit-learn	TensorFlow
Focus	Classical ML (SVM, decision trees)	Deep learning (CNNs, RNNs, DNNs)
Ease of use	Very beginner-friendly	Requires more setup, especially for DL
Community	Strong ML community	Larger due to industrial use, TFJS etc.

Double-click (or enter) to edit