1.TENSORFLOW VS PYTORCH

a.PRIMARY DIFFRENCES

1.Programming style and ease of use

Tensorflow

- Graph mode via tf.function- optimizes execution but increases complexity.
- Static graphs (default)- computation graphs are built in and then executed which leads to performance optimizations but requires more abstraction.
- Eager execution-introduced in tensorflow 2.x to match pytorch flexibility.

Pytorch

- Intuitive and pythonic- it feels like the python language which makes it easy to use for researchers and beginners.
- Dynamic computation graphs-easy to use for models that require flexibility such as conditional branches in networks and recurrent neural networks.
- Eager execution-operations are executed immediately like standard python code.

2.Ecosystem and libraries

Tensorflow

- Strong intergration with google cloud.
- Rich ecosystem eg TF hub,TF lite, Keras.
- Many pretrained models available.

Pytorch

- Simpler for custom research applications , computer vision and NLP.
- Very popular to academic and research facilities.
- Strong intergration with Pytorch lightning and huggingface transformers.

B, WHEN TO CHOOSE EACH FOR AI PROJECTS

Pytorch:

- When you want flexible debugging, faster prototyping or are working on research based AI models.
- When you plan to deploy with simpler pipelines or convert modules via ONNX.

Tensorflow:

- When you are in an enterprise environment or aligning with google cloud tools.
- When you want to achieve standardized tasks or large scale production systems with robust tooling.

- When your AI system needs industrial scale deployment.

2.USE CASES FOR JUPITER NOTEBOOKS IN AI DEVELOPMENT

1,AI model demonstration and documentation:

- Demonstrating AI concepts or models to students, collaborators or non-technical users alongside code.
- Example an instructor teaching about convulational neural networks can let students modify activation functions in real time to see how results change.
- Jupyter is perfect for explaining AI algorithms and creating narrated tutorials.

2,Experimentation and prototyping:

- Jupyter actively contributes to rapid development and testing of AI models especially in its initial stages.
- One can visualize data and model performance immediately with libraries like matplotlib, seaborn or tensorboard.
- Example when building a neural network to classify handwritten digits using MNIST dataset one can load and preprocess data in one cell.

3, HOW TO SPACY ENHANCES NLP TASKS

- Spacy understands linguistic structure in terms of tokens, sentences ,named entities ,parts of speech and enables deep text understanding like parsing and semantic analysis while basic python uses simple string methods like find.
- Spacy identifies entities like persons, locations, dates and organizations through named entity recognition while python depends on regex or manual rule creation.
- Spacy assigns grammatical roles to each word through part-of-speech tagging and dependency parsing while python lacks any built in capability for syntactic analysis.

COMPARATIVE ANALYSIS:scikit-learnvs tensorflow

TARGET APPLICATIONS

ASPECT	SCIKIT-LEARN	TENSORFLOW
Ease of use	Beginner friendly and easy to use	Complex and used in large scale models.
Primary use case	Classical machine learning	Deep learning and neural networks.
Data types	Structured data	Unstructured data ,audio,images and text.

Community support	Large, mature, academic and ML users.	Very large, global includes researchers, developers and production engineers.