Theoretical Part

1. Short Answer Questions

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

Primary Differences:

TensorFlow	PyTorch
Static Computation Graph (Define-and-run)	**Dynamic Computation Graph** (Define-by-run)
More verbose syntax, steeper learning curve	Pythonic syntax, easier to learn and debug
Better production deployment with TensorFlow Serving	Originally research-focused, now improved production support
Strong visualization with TensorBoard	Good visualization but less integrated
Keras API integration for high-level abstraction	torch.nn for neural network operations

When to choose TensorFlow:

- Production deployment and scalability requirements
- Mobile and embedded systems development
- When using TensorFlow Extended (TFX) for ML pipelines
- Enterprise environments with existing TensorFlow infrastructure

When to choose PyTorch:

- Research and rapid prototyping
- When dynamic computation graphs are needed
- Academic and research environments
- When easier debugging and Pythonic code is preferred

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

1. Exploratory Data Analysis and Model Prototyping

- Interactive data visualization and statistical analysis
- Rapid experimentation with different algorithms and parameters
- Step-by-step data preprocessing and feature engineering
- Immediate feedback on model performance with inline plots

2. Educational Demonstrations and Documentation

- Creating interactive tutorials with code, visualizations, and explanations
- Sharing reproducible research with combined code, results, and narrative
- Collaborative development with mixed code and markdown documentation
- Model interpretation and result presentation to stakeholders

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

spaCy Advantages:

Basic String Operations	spaCy NLP
Pattern matching with regex	**Linguistic understanding** with trained models
Simple keyword searching	**Named Entity Recognition** (people, organizations, locations)
Manual text splitting	**Dependency parsing** for grammatical structure
No semantic understanding	**Word vectors** for semantic similarity
Limited to exact matches	**Lemmaization** and morphological analysis

Key spaCy Enhancements:

- **Pre-trained models** for multiple languages
- **Efficient processing** with compiled Cython code
- **Linguistic features** like part-of-speech tagging, dependency parsing
- **Entity recognition** for extracting structured information
- **Custom pipeline components** for domain-specific tasks

- **Integration** with machine learning workflows

3. Comparative Analysis: Scikit-learn vs TensorFlow

Aspect	Scikit-learn	TensorFlow
Target Applications	**Classical Machine Learning**: Classification, regression, clustering, dimensionality reduction	**Deep Learning**: Neural networks, computer vision, NLP, reinforcement learning
Algorithm Types	Traditional ML algorithms (SVM, Random Forest, K-means)	Neural networks, custom architectures, transfer learning
Data Scale	Medium-sized datasets that fit in memory	Large-scale datasets with batch processing
Hardware Usage	CPU-focused, limited GPU support	GPU/TPU acceleration for training and inference
Model Types	Shallow models, ensemble methods	Deep neural networks, custom architectures

Ease of Use for Beginners:

Scikit-learn Advantages:

- Consistent API design (`fit()`, `predict()`, `transform()`)
- Minimal code required for standard algorithms
- Extensive documentation with practical examples
- No deep learning knowledge required
- Easy hyperparameter tuning with GridSearchCV

TensorFlow Challenges:

- Steeper learning curve with computational graphs
- More verbose code for simple models
- Requires understanding of neural network concepts
- Complex debugging with graph execution

TensorFlow Advantages with Keras:

- High-level Keras API simplifies model building
- Pre-built layers and models available
- Good for learning deep learning fundamentals

Community Support:

Scikit-learn Community:

- **Mature ecosystem** with stable, well-tested algorithms
- **Strong academic backing** and extensive research papers
- **Comprehensive documentation** with tutorials and examples
- **Active maintenance** with regular updates and bug fixes
- **Large user base** across industry and academia

TensorFlow Community:

- **Massive ecosystem** with TensorFlow Hub, TensorBoard, TFX
- **Google backing** with strong corporate support
- **Extensive online courses** and certifications
- **Research community** adoption for cutting-edge models
- **Production focus** with enterprise deployment tools