Part 1

Question 1: TensorFlow vs PyTorch

TensorFlow and PyTorch are both deep-learning frameworks, but they differ in how they build and run neural networks. TensorFlow, developed by Google, relies on computation graphs that can be optimized before running. It integrates easily with deployment tools such as TensorFlow Lite and TensorFlow Serving, which makes it strong for production use. PyTorch, created by Meta, uses a more flexible "define-by-run" system. It lets a developer build and modify networks on the fly, making it ideal for research and fast experimentation.

Choice Summary

Use PyTorch when you want quick experimentation, transparency, and easier debugging.

Use TensorFlow when your main goal is large-scale training, cross-platform deployment, or mobile integration.

Question 2: Two Uses of Jupyter Notebooks in AI Development

1. Exploratory Data Analysis (EDA) – Analysts and data scientists use notebooks to inspect and visualize data interactively. They can run one cell at a time to check missing values, plot graphs, or summarize statistics.
2. Model Prototyping and Documentation – Developers can train and test AI models inside the same notebook that holds explanations, figures, and results. This makes research reports reproducible and easy to share.

Question 3: How spaCy Enhances NLP Compared to Basic String Operations

spaCy provides a full natural-language pipeline with tokenization, part-of-speech tagging, named-entity recognition, and dependency parsing. These tasks go far beyond what basic string methods like .split() or .find() can do. Instead of just matching patterns, spaCy understands sentence structure and context. It also uses pretrained statistical models, which improves both speed and accuracy in text analysis.

Comparative Analysis: Scikit-learn vs TensorFlow

Feature Scikit-learn TensorFlow

Main Purpose Traditional machine-learning algorithms such as linear regression, SVMs, and decision trees Deep-learning models such as CNNs, RNNs, and transformers Ease for Beginners Very easy to learn; simple fit/predict API Steeper learning curve, especially for neural-network design Community and Support Mature academic and research community Very active industry-driven community focused on large-scale AI

Part 3

1. Ethical Considerations

Machine-learning models can develop bias when their training data does not represent the full diversity of real-world inputs.

For MNIST: The dataset mainly contains clean, centered digits written by specific groups of people. A model trained on it may misclassify digits written in unusual handwriting styles or from under-represented age groups.

For Amazon Reviews: Sentiment analysis may reflect social or cultural bias if reviews from some regions or languages dominate the data. Words that are neutral in one culture could be interpreted as negative in another.

Reducing Bias

TensorFlow Fairness Indicators can compare metrics such as accuracy or false-positive rate across user groups to expose hidden bias.

spaCy Rule-based Systems allow developers to create transparent text-processing rules that make the model's behavior easier to audit and adjust.

2. Troubleshooting Common TensorFlow Errors

When debugging TensorFlow scripts:

Dimension Mismatch: Confirm that the shape of training data matches the model's expected input, for example (28, 28, 1) for grayscale images.

Incorrect Loss Function: Use sparse_categorical_crossentropy when labels are integers, and categorical_crossentropy when they are one-hot encoded.

Overfitting: Add dropout layers, reduce the number of epochs, or use early stopping to keep the model from memorizing training data.