

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

TensorFlow	PyTorch
⑩ Static computation graphs (TF 2.x allows eager execution)	Dynamic computation graphs (easier to debug)
⑩ Preferred for deployment, production, and mobile (TensorFlow Lite)	Preferred for research and experimentation
⑩ Developed by Google	Developed by Facebook

- ⑩ I will choose PyTorch for fast prototyping and flexibility and TensorFlow for deployment-ready pipelines and robust tooling.

Describe two use cases for Jupyter Notebooks in AI development.

1. **Prototyping Models:** Write and run code in chunks, test models interactively.
2. **Data Analysis & Visualization:** Use Pandas, Matplotlib or Seaborn for quick insights and visual exploration.

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

1. Through Efficient and accurate tokenization, POS tagging, and NER.,
2. Offering pre-trained pipelines unlike basic Python strings which can't understand linguistic structure.

Compare Scikit-learn and TensorFlow in terms of:

1. Target applications (e.g., classical ML vs. deep learning).

Scikit-learn:

- ⑩ Primarily designed for **classical machine learning algorithms** such as:

- Support Vector Machines (SVM)
- Decision Trees
- K-Nearest Neighbors (KNN)
- Logistic and Linear Regression

It is best suited for **structured/tabular data** and quick experimentation in data science projects.

TensorFlow:

Built for deep learning and large-scale machine learning, it supports:

- ⑩ Convolutional Neural Networks (CNNs)
- ⑩ Recurrent Neural Networks (RNNs)
- ⑩ Transformer

It is ideal for image, speech, NLP, and time series applications, and production deployment.

2. Ease of Use for Beginners

⑩ Scikit-learn:

- Highly beginner-friendly, with:
 - A consistent and simple API
 - Easy-to-follow documentation
- Minimal setup to train and evaluate models

⑩ TensorFlow:

Has a steeper learning curve, especially when using low-level APIs.

However:

- The Keras API within TensorFlow improves usability
- Offers more flexibility and power at the cost of added complexity

3. Community Support

⑩ Scikit-learn:

- Strong community for **academic, research, and industry use** in traditional ML
- Widely used in **data science education** and MOOCs
- Regular updates and stable ecosystem

⑩ TensorFlow:

- Backed by **Google**, with a vast global developer community
- Extensive ecosystem: TensorBoard, TensorFlow Lite, TensorFlow Serving
- Abundant tutorials, forums, GitHub repos, and support for cutting-edge research

Task 1: Classical ML with Scikit-learn

Classification Report:				
	precision	recall	f1-score	support
0	1.00	1.00	1.00	10
1	1.00	1.00	1.00	9
2	1.00	1.00	1.00	11
accuracy			1.00	30
macro avg	1.00	1.00	1.00	30
weighted avg	1.00	1.00	1.00	30
Accuracy: 1.00				
Precision (macro): 1.00				
Recall (macro): 1.00				

Task 2: Deep Learning with TensorFlow/PyTorch

sing Sequential models, prefer using an Input(shape) object as the first layer in the model instead.
super().__init__(activity_regularizer=activity_regularizer, **kwargs)
2025-07-16 13:48:19.325188: E external/local_xla/xla/stream_executor/cuda/cuda_platform.cc:51] failed call to cuInit: INTERNAL: CUDA error: Fail
2025-07-16 13:48:19.431905: W external/local_xla/xla/tsl/framework/cpu_allocator_impl.cc:83] Allocation of 169344000 exceeds 10% of free system
1688/1688 15s 8ms/step - accuracy: 0.8967 - loss: 0.3316 - val_accuracy: 0.9857 - val_loss: 0.0523
Epoch 2/5
1688/1688 15s 9ms/step - accuracy: 0.9837 - loss: 0.0508 - val_accuracy: 0.9873 - val_loss: 0.0440
Epoch 3/5
1688/1688 17s 10ms/step - accuracy: 0.9894 - loss: 0.0320 - val_accuracy: 0.9910 - val_loss: 0.0385
Epoch 4/5
1688/1688 16s 9ms/step - accuracy: 0.9928 - loss: 0.0223 - val_accuracy: 0.9883 - val_loss: 0.0410
Epoch 5/5
1688/1688 18s 11ms/step - accuracy: 0.9945 - loss: 0.0158 - val_accuracy: 0.9902 - val_loss: 0.0372
313/313 1s 4ms/step - accuracy: 0.9876 - loss: 0.0385
Test accuracy: 0.9900
313/313 1s 3ms/step

Classification Report:

	precision	recall	f1-score	support
0	0.99	0.99	0.99	980
1	1.00	0.99	0.99	1135
2	0.99	0.99	0.99	1032
3	0.99	0.99	0.99	1010
4	0.99	1.00	0.99	982
5	0.98	0.99	0.98	892
6	1.00	0.98	0.99	958
7	0.98	1.00	0.99	1028
8	0.99	0.99	0.99	974
9	1.00	0.98	0.99	1009
accuracy			0.99	10000
macro avg	0.99	0.99	0.99	10000
weighted avg	0.99	0.99	0.99	10000

1/1 0s 32ms/step
1/1 0s 36ms/step
1/1 0s 34ms/step
1/1 0s 30ms/step
1/1 0s 30ms/step

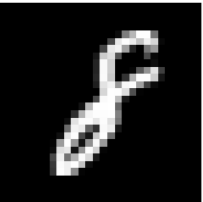
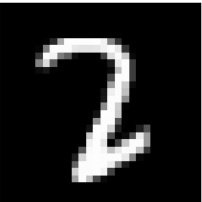
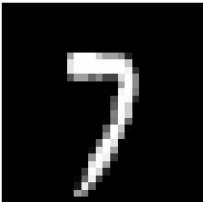
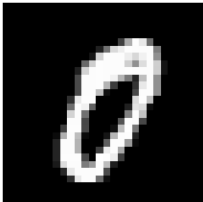
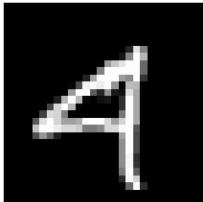
Pred: 4

Pred: 0

Pred: 7

Pred: 2

Pred: 8



Task 3: NLP with spaCy

```
Review: I love the sound quality of the Sony headphones! Highly recommendable.  
Named Entities: [('Sony', 'ORG')]  
Sentiment: Positive  
  
Review: The Apple iPhone 15 has an amazing camera but it is expensive.  
Named Entities: None found  
Sentiment: Positive  
  
Review: This Samsung Galaxy watch is overpriced and not user-friendly.  
Named Entities: None found  
Sentiment: Negative  
  
Review: The HP laptop performs well for daily tasks and has a sleek design.  
Named Entities: [('HP', 'ORG'), ('daily', 'DATE')]  
Sentiment: Positive  
  
Review: I had a terrible experience with the Dell monitor.  
Named Entities: [('Dell', 'PERSON')]  
Sentiment: Negative
```

Ethical considerations:

- ⑩ Brand/product bias: The model may associate certain brands with good/bad sentiment based on frequency, not quality.
- ⑩ **Language/grammar bias:** Reviews written by non-native speakers may be misclassified due to grammatical structure.
- ⑩ Sentiment bias: Positive words may be overrepresented in certain products, skewing predictions.

How to Mitigate Biases

TensorFlow Fairness Indicators

- ⑩ Helps track model performance across user-defined slices (e.g., gender, product type).
- ⑩ Can display metrics like precision, recall, and false positive rate by group, identifying if a model is unfairly biased.

spaCy Rule-Based Systems

- ⑩ For NER tasks, spaCy allows creation of custom patterns instead of relying only on trained models.
- ⑩ Ensures important entities (like local brands or minority names) aren't ignored.
- ⑩ Helps override biased predictions made by statistical models using handcrafted, inclusive logic.

Troubleshooting Challenge: Fixing a Buggy TensorFlow Script

Buggy version

```
model = tf.keras.Sequential([
    tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dense(10)
])
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
model.fit(x_train, y_train, epochs=5)
```

Fixed version.

```
import tensorflow as tf

from tensorflow.keras.datasets import mnist
from tensorflow.keras.utils import to_categorical

# Load data
(x_train, y_train), (x_test, y_test) = mnist.load_data()

# Normalize inputs
x_train = x_train.reshape(-1, 28*28).astype('float32') / 255.0
```

```
x_test = x_test.reshape(-1, 28*28).astype('float32') / 255.0
```

```
# Convert labels to one-hot (or use sparse_categorical_crossentropy)
```

```
y_train_cat = to_categorical(y_train, 10)
```

```
y_test_cat = to_categorical(y_test, 10)
```

```
# Define model
```

```
model = tf.keras.Sequential([  
    tf.keras.layers.Input(shape=(784,)),  
    tf.keras.layers.Dense(64, activation='relu'),  
    tf.keras.layers.Dense(10, activation='softmax')  
])
```

```
# Compile and train
```

```
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
```

```
model.fit(x_train, y_train_cat, epochs=5, validation_data=(x_test, y_test_cat))
```