1. Ethical Considerations

Potential Biases in the MNIST Model:

- The **MNIST dataset** (handwritten digits) is relatively neutral, but it can still contain **representation bias** for example, most digits were written by American students, meaning handwriting styles from other regions or languages (like Arabic numerals or African scripts) might be underrepresented.
- This could make the model **less accurate** when used on digits written by people with different handwriting patterns.
- Another concern is **data imbalance**, where some digits (like "1" and "0") might appear more frequently than others, leading the model to perform better on those.

How Tools Could Help:

- **TensorFlow Fairness Indicators** could help by showing disparities in accuracy across different subgroups (e.g., handwriting styles, brightness levels, or sources of data). This allows us to identify where the model performs poorly and retrain it with more balanced samples.
- **spaCy's rule-based systems** could help if the data were extended to handwritten *text* rather than digits rules could catch inconsistencies or language-based biases (like gendered terms or cultural bias in text interpretation).

Potential Biases in the Amazon Reviews Model:

- The **Amazon Reviews sentiment analysis model** might learn **linguistic or cultural bias** for instance, associating certain phrases or dialects with negative sentiment.
- It could also show **gender or product-type bias**: for example, reviews of tech products (often written by men) could be rated more positively due to data imbalance, while beauty product reviews (often written by women) might be rated differently due to biased word usage in training data.
- **Imbalanced datasets** (more positive than negative reviews) could cause the model to overpredict "positive" sentiments.

How Tools Could Help:

- **TensorFlow Fairness Indicators** can measure the model's performance across review categories (gendered language, product type, sentiment class) to identify unfair patterns.
- **spaCy's rule-based systems** could be used before training to filter or flag sensitive or biased words (e.g., gendered adjectives like "bossy" or "strong"), helping to reduce the propagation of these biases in training.

2. Troubleshooting Challenge (Buggy TensorFlow Script)

Let's assume the provided TensorFlow code for MNIST had the following common errors:

- Mismatch between **input shape** and model architecture
- Incorrect **loss function** for classification
- Missing **activation** in the final layer

Here's an example of **buggy code** and a **corrected version**

❌ **Buggy Code**

```
model = tf.keras.Sequential([
  tf.keras.layers.Dense(128, input_shape=(28,28)),
  tf.keras.layers.Dense(10)
])
model.compile(optimizer='adam',  loss='mse', metrics=['accuracy'])
```

Issues:

1. The input shape should be **(784,)** because each image is flattened from 28×28 pixels.
2. The final layer should have a **softmax activation** for multiclass classification.
3. The loss function should be **sparse_categorical_crossentropy** (for integer labels).

✅ **Fixed Code**

```
model = tf.keras.Sequential([
  tf.keras.layers.Flatten(input_shape=(28, 28)),
  tf.keras.layers.Dense(128, activation='relu'),
  tf.keras.layers.Dense(10, activation='softmax')
])


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

✅ **Explanation of Fixes:**

- **Flatten Layer:** Converts 2D image input to 1D vector for dense layers.
- **ReLU Activation:** Helps model learn nonlinear patterns.
- **Softmax:** Converts logits to probabilities for 10 output classes.
- **Correct Loss:** sparse_categorical_crossentropy properly handles integer-labeled targets.
