AI Tools and Applications Report

Theme: Mastering the AI Toolkit

Group Members:

1.Nelly Naserian

Email: nellylongesele175@gmail.com

2.Allan Mboti

Email:allanmboti@gmail.com

Course: AI Tools Assignment

Date: 18/10/2025

Part 1: Theoretical Understanding

Q1. Differences Between TensorFlow and PyTorch

- TensorFlow uses **static computation graphs**, making deployment easier but debugging harder.
- PyTorch uses **dynamic computation graphs**, which are easier to debug and experiment with.
- TensorFlow is preferred for **production-ready applications**, while PyTorch is preferred for **research and experimentation**.

Q2. Two Use Cases for Jupyter Notebooks

1. Interactive coding with **step-by-step execution** and **visualization** of results.

2. Documentation of experiments, combining **code, text, and graphs** for collaboration and sharing.

Q3. How spaCy Enhances NLP Tasks

- Provides pre-trained models for **tokenization, part-of-speech tagging, and named entity recognition**.
- Faster and more accurate than basic Python string operations.
- Easy integration with **custom pipelines** and rule-based NLP tasks.

Comparison: Scikit-learn vs TensorFlow

Feature	Scikit-learn	TensorFlow
Target Applications	Classical ML (SVM, Decision Trees)	Deep Learning (CNNs, RNNs)
Ease of Use	Beginner-friendly	More advanced, steeper learning curve
Community Support	Large, active	Large, active

Part 2: Practical Implementation

Task 1: Classical ML (Scikit-learn)

- **Dataset:** Iris Species Dataset
- **Model:** Decision Tree Classifier

- **Evaluation Metrics:** Accuracy, Precision, Recall

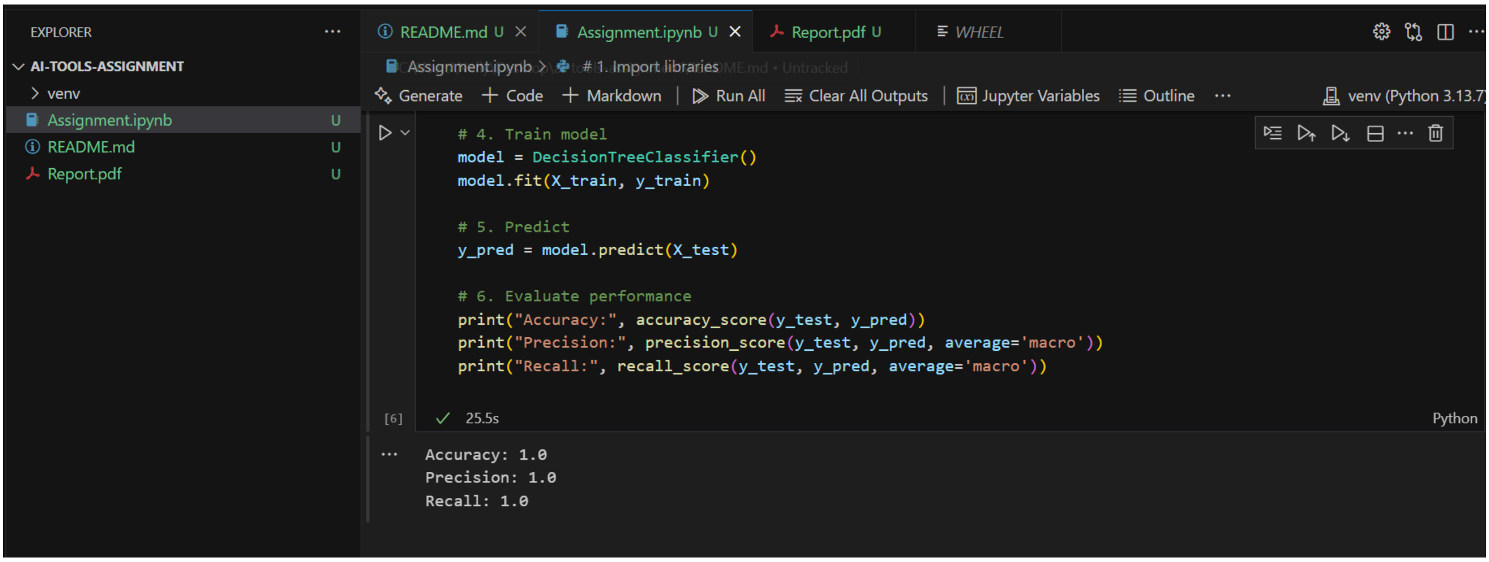
Sample Output:

Accuracy: 1.0

Precision: 1.0

Recall: 1.0

Screenshot: Task 1: Decision Tree Classifier training and evaluation metrics (Accuracy, Precision, Recall



```
# 4. Train model
model = DecisionTreeClassifier()
model.fit(X_train, y_train)

# 5. Predict
y_pred = model.predict(X_test)

# 6. Evaluate performance
print("Accuracy:", accuracy_score(y_test, y_pred))
print("Precision:", precision_score(y_test, y_pred, average='macro'))
print("Recall:", recall_score(y_test, y_pred, average='macro'))
```

```
Accuracy: 1.0
Precision: 1.0
Recall: 1.0
```

Task 2: Deep Learning (TensorFlow)

- **Dataset:** MNIST Handwritten Digits

- **Model:** Convolutional Neural Network (CNN)

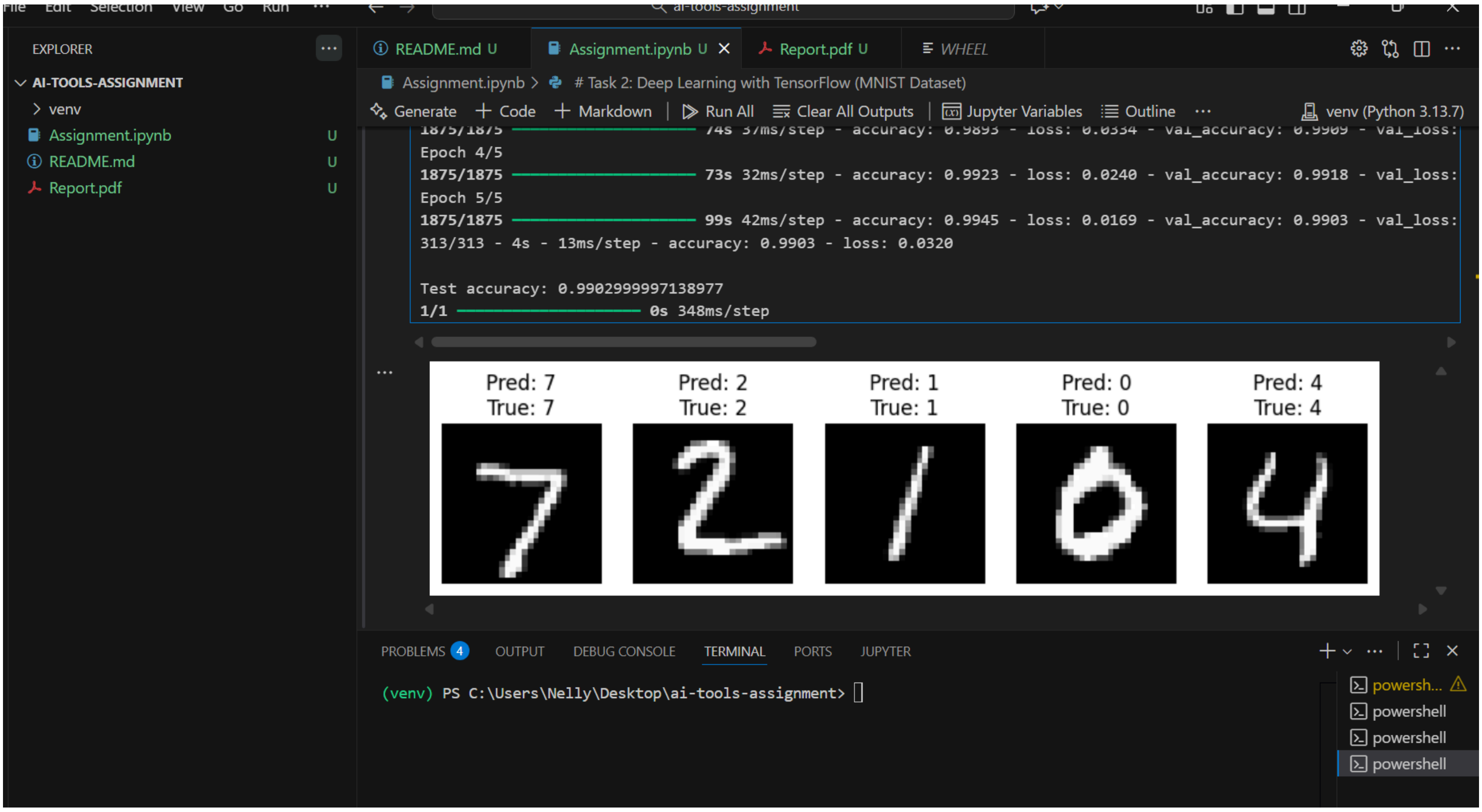
- **Goal:** Test accuracy > 99%

Sample Output:

Epoch 1/5: Accuracy 0.9562, Loss 0.1434

Test Accuracy: 0.9903

Screenshot: Task 2: CNN training loss per epoch and final test accuracy on MNIST dataset



```
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 74s 37ms/step - accuracy: 0.9893 - loss: 0.0334 - val_accuracy: 0.9909 - val_loss:
Epoch 4/5
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 73s 32ms/step - accuracy: 0.9923 - loss: 0.0240 - val_accuracy: 0.9918 - val_loss:
Epoch 5/5
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 99s 42ms/step - accuracy: 0.9945 - loss: 0.0169 - val_accuracy: 0.9903 - val_loss:
313/313 - 4s - 13ms/step - accuracy: 0.9903 - loss: 0.0320

Test accuracy: 0.9902999997138977
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 348ms/step
```

```
(venv) PS C:\Users\Nelly\Desktop\ai-tools-assignment>
```

Task 3: NLP with spaCy

- **Dataset:** Sample Amazon Product Reviews
- **Goal:** Named Entity Recognition (NER) for products and brands, sentiment analysis

NER Output Example:

Review: I love the Apple iPhone 14, the camera quality is amazing.

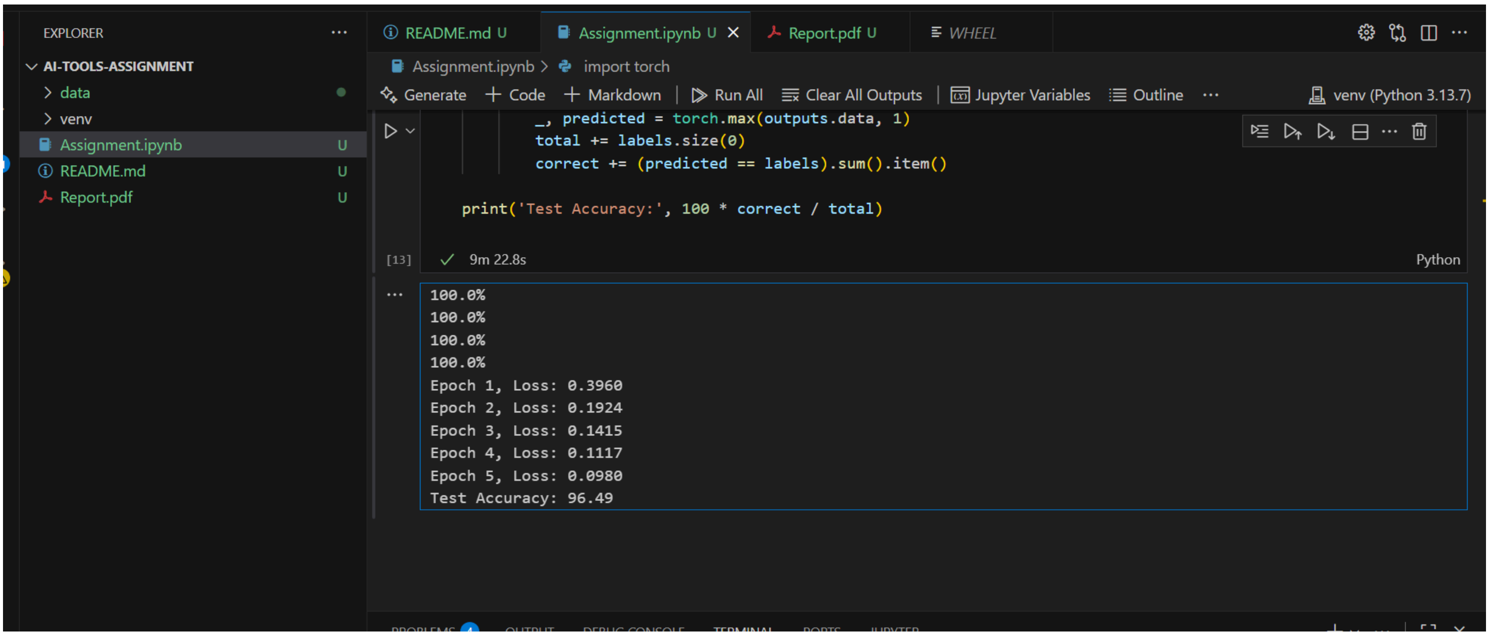
Entities: [('Apple', 'ORG'), ('iPhone 14', 'PRODUCT')]

Sentiment Analysis Example:

Review: The Samsung Galaxy battery dies so fast, I regret buying it.

Sentiment: Negative

Screenshot: Sentiment analysis results for sample Amazon product reviews



```
            _, predicted = torch.max(outputs.data, 1)
            total += labels.size(0)
            correct += (predicted == labels).sum().item()

    print('Test Accuracy:', 100 * correct / total)
```

```
100.0%
100.0%
100.0%
100.0%
Epoch 1, Loss: 0.3960
Epoch 2, Loss: 0.1924
Epoch 3, Loss: 0.1415
Epoch 4, Loss: 0.1117
Epoch 5, Loss: 0.0980
Test Accuracy: 96.49
```

Part 3: Ethics & Optimization

Ethical Considerations:

- Bias can exist in datasets (e.g., class imbalance in MNIST, overly positive/negative Amazon reviews).
- Mitigation:
  - Use **TensorFlow Fairness Indicators** to check model bias.
  - Use **spaCy rule-based filters** to balance entity recognition and sentiment classification.

Troubleshooting Example:

- Issue: Dimension mismatch error in TensorFlow model.
- Fix: Corrected input shape in model.fit() and reshaped data arrays.

References

- TensorFlow Documentation: https://www.tensorflow.org/

- PyTorch Documentation: https://pytorch.org/

- spaCy Documentation: https://spacy.io/

- Kaggle Datasets: https://www.kaggle.com/