PLP WEEK 3 AI TOOLS ASSIGNMENT

Part 1: Theoretical Understanding

1. Short Answer Questions

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

- **TensorFlow** is a symbolic computation framework developed by Google. It uses static computation graphs (in earlier versions), allowing for optimized deployment across platforms.
- **PyTorch**, developed by Facebook, uses dynamic computation graphs, meaning operations are computed on the fly, making debugging and experimentation easier.

Choose **TensorFlow** for production-ready applications, model deployment (e.g., with TensorFlow Serving), and mobile support. Choose **PyTorch** for research, experimentation, and rapid prototyping due to its Pythonic style and ease of debugging.

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

- **Interactive Prototyping**: Jupyter allows iterative development and testing of machine learning models with real-time output display.
- **Data Visualization & Exploration**: It supports inline plots (e.g., with Matplotlib or Seaborn), making it easy to explore datasets and evaluate model performance.

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

spaCy provides pre-trained language models and optimized pipelines that handle tokenization, POS tagging, dependency parsing, and named entity recognition. Unlike basic Python string functions, spaCy is context-aware, language-specific, and efficient, supporting large-scale NLP tasks in real-time.

2. Comparative Analysis

Compare Scikit-learn and TensorFlow in terms of:

- Target applications (e.g., classical ML vs. deep learning).
- Ease of use for beginners.
- Community support.

Feature	Scikit-learn	TensorFlow
Target Applications	Classical ML (SVMs, trees, regression)	Deep Learning (CNNs, RNNs, DNNs
Ease of Use	Beginner-friendly, minimal setup	Steeper learning curve, more code required
Community Support	Mature community, wide documentation	Rapidly growing, strong industry backing

Part 2: Practical Implementation

Part 3: Ethics & Optimization

1. **Ethical Considerations**

Bias in MNIST Digit Classifier

MNIST seems objective, but still has potential biases:

- **Input quality bias**: MNIST digits are centered and grayscale. Real-world input may vary (off-center, noisy, colored).
- **Model overfitting**: The model may overfit to "clean" data and misclassify stylized or handwritten digits from diverse populations (e.g., left-handed writing styles).
- **Accessibility**: Model may fail for users with motor impairments or alternative drawing styles.

Bias in Amazon Reviews Classifier

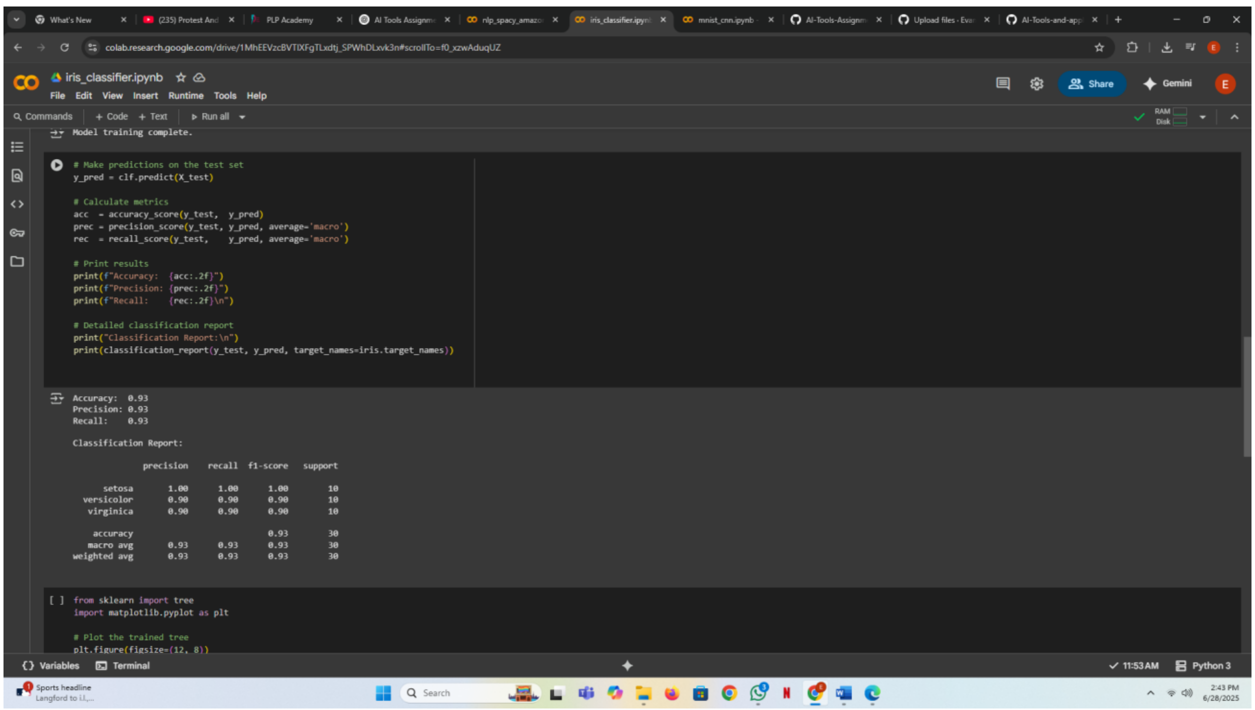
When using text data like Amazon reviews:

 • **Sentiment bias:** Models might associate certain product categories, demographics, or dialects with negative or positive sentiment.

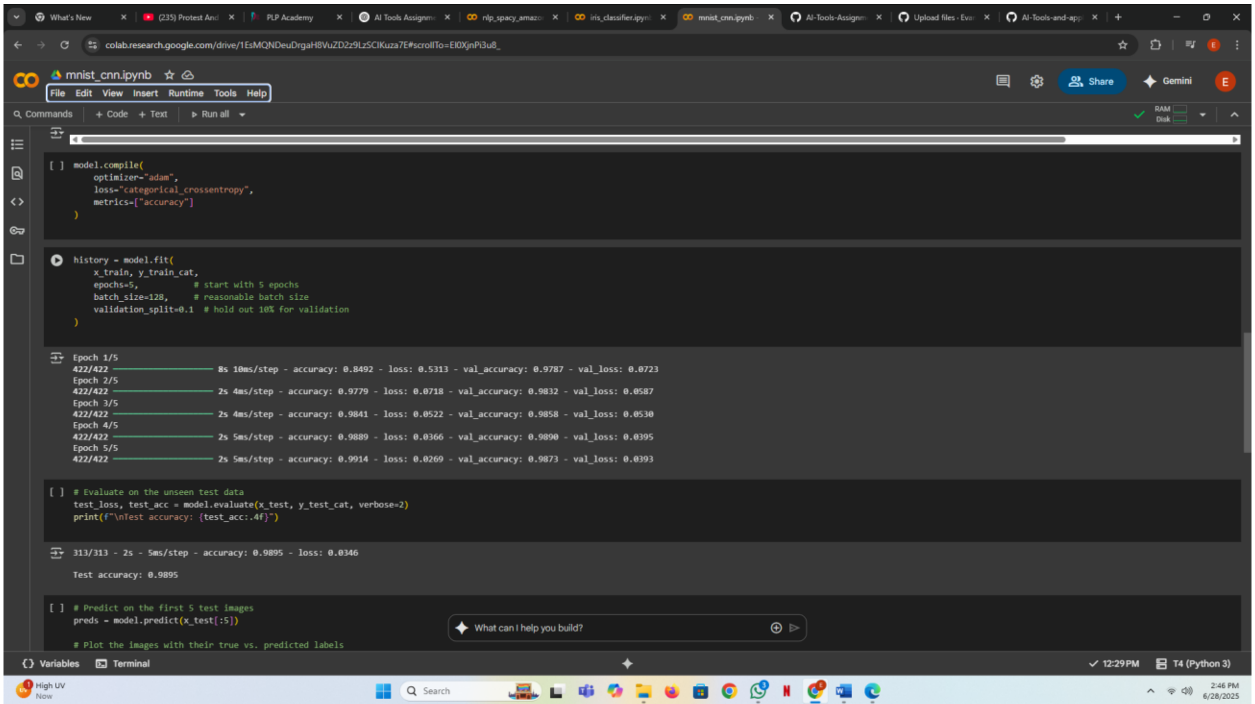
• **Representation bias:** If most training reviews come from one region or language style, the model may underperform elsewhere.

• **Toxicity or gender bias:** Words associated with certain groups may receive skewed sentiment scores.

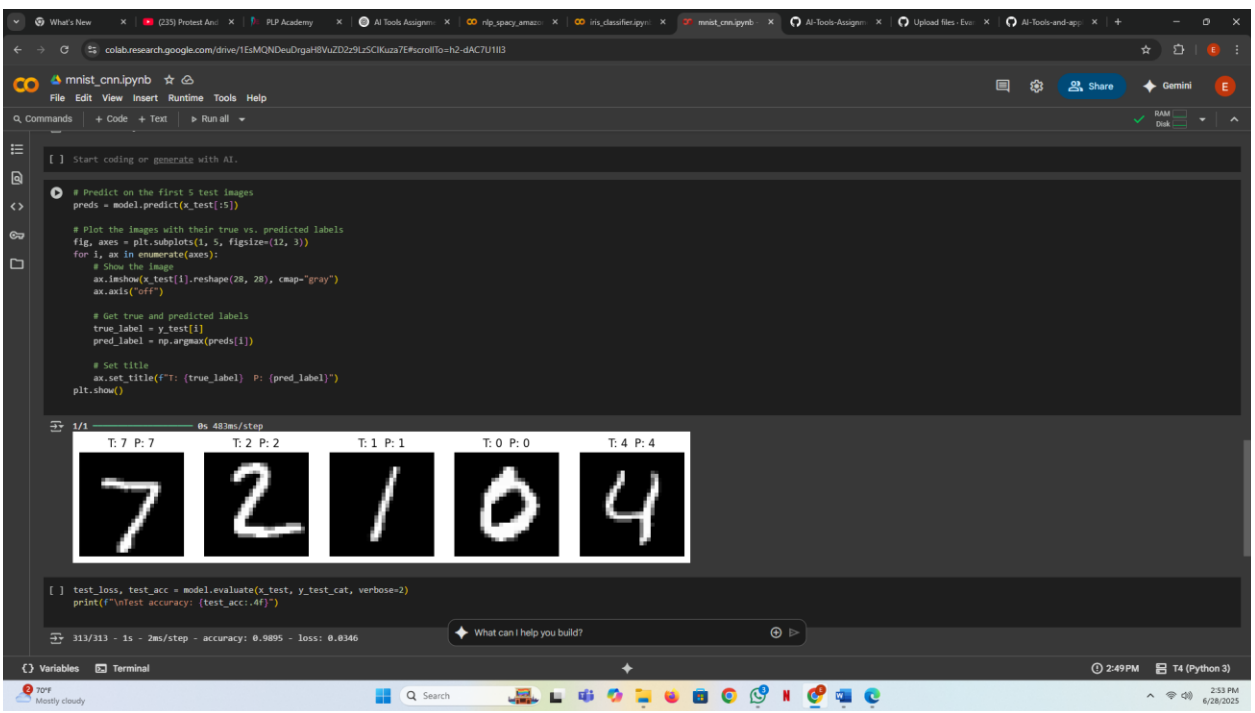
Accuracy scores for Decision tree classifier for iris species



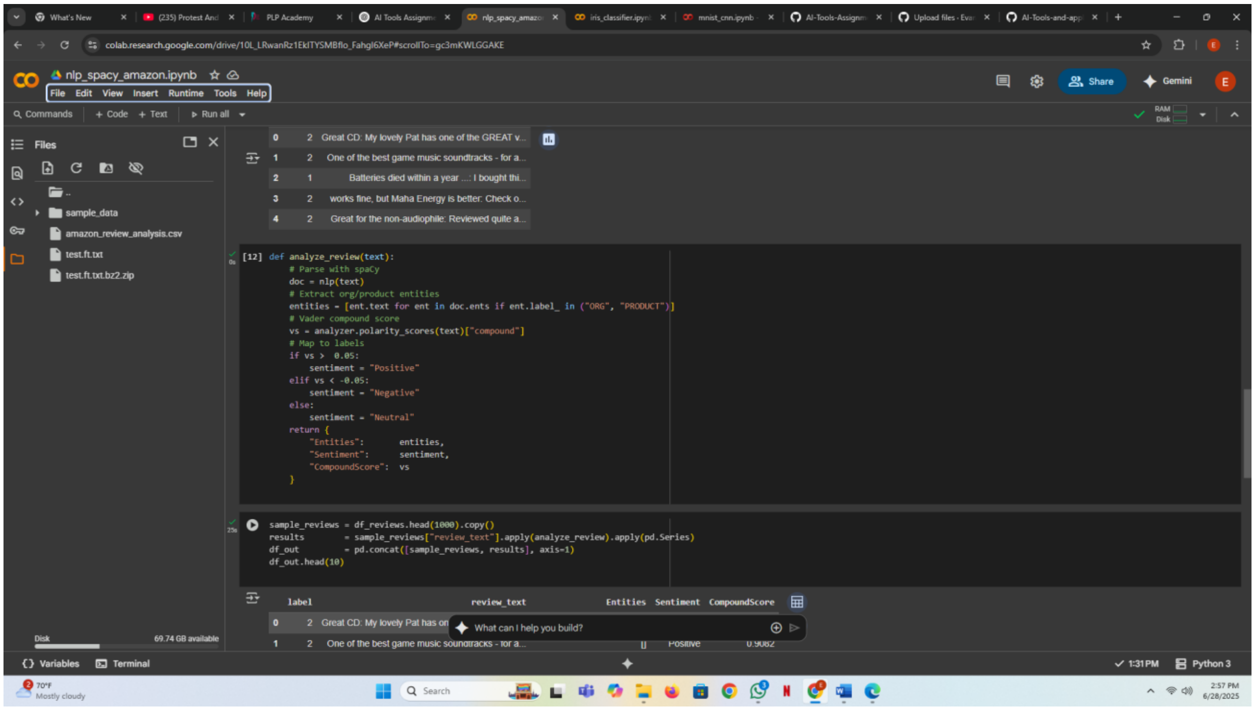
Epoch values for CNN model to classify handwritings



Handwritten digits and predictions



Classical ML with spaCy



nlp_spacy_amazon.ipynb
File Edit View Insert Runtime Tools Help
Commands + Code + Text Run all
Files
sample_data
amazon_review_analysis.csv
test.ft.txt
test.ft.txt.bz2.zip
0 2 Great CD: My lovely Pat has one of the GREAT v...
1 2 One of the best game music soundtracks - for a...
2 1 Batteries died within a year ...: I bought thi...
3 2 works fine, but Maha Energy is better: Check o...
4 2 Great for the non-audiophile: Reviewed quite a...
[12] def analyze_review(text):
# Parse with spaCy
doc = nlp(text)
# Extract org/product entities
entities = [ent.text for ent in doc.ents if ent.label_ in ("ORG", "PRODUCT")]
# Vader compound score
vs = analyzer.polarity_scores(text)["compound"]
# Map to labels
if vs > 0.05:
sentiment = "Positive"
elif vs < -0.05:
sentiment = "Negative"
else:
sentiment = "Neutral"
return {
"Entities": entities,
"Sentiment": sentiment,
"CompoundScore": vs
}
sample_reviews = df_reviews.head(1000).copy()
results = sample_reviews["review_text"].apply(analyze_review).apply(pd.Series)
df_out = pd.concat([sample_reviews, results], axis=1)
df_out.head(10)
label review_text Entities Sentiment CompoundScore
0 2 Great CD: My lovely Pat has on
What can I help you build?
1 2 One of the best game music soundtracks - for a... [] Positive 0.9082
Disk 69.74 GB available
Variables Terminal
1:31 PM Python 3
70°F Mostly cloudy
Search
2:57 PM 6/28/2025