Unit 1 Hands-on: Generative AI & NLP Fundamentals

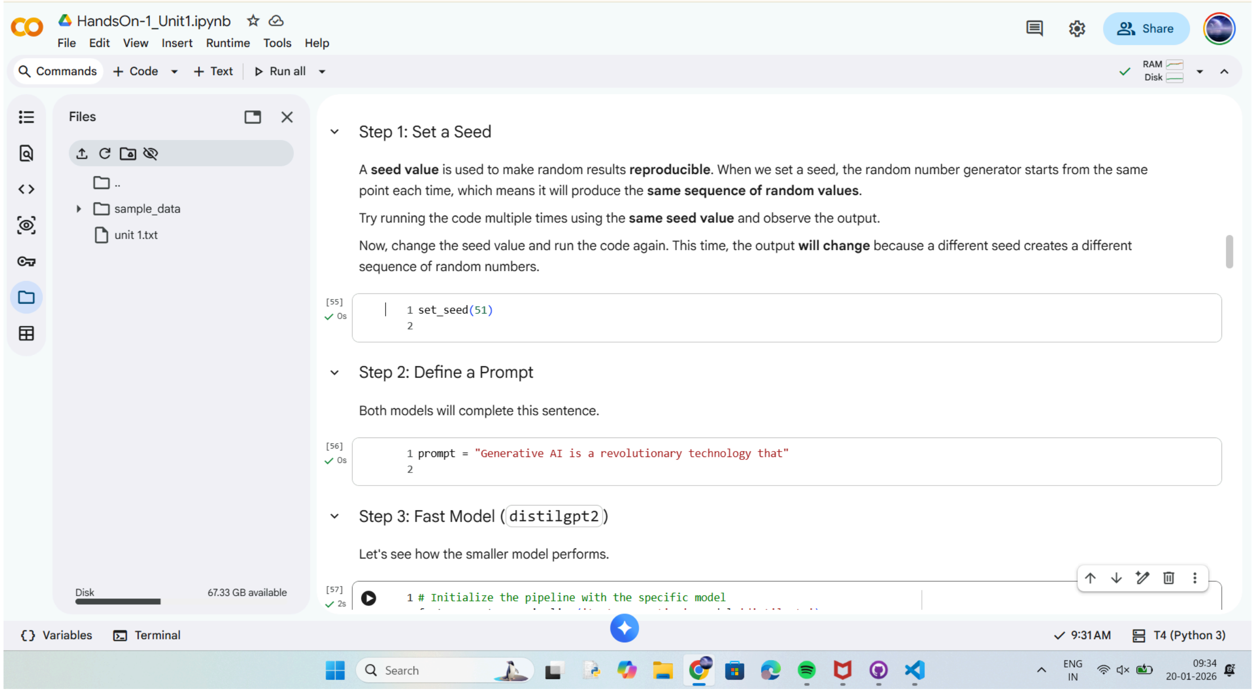
A R KEERTHANA
PES2UG23CS001

Observations from the HandsOn-1_Unit1.ipynb:

Section 1: Introduction & Setup

- Hugging Face acts as a centralized platform for accessing pretrained AI models, datasets, and demos, reducing the need to train models from scratch.
- The transformers library provides a simple interface to load and use state-of-the-art NLP models across different frameworks.
- The pipeline() function abstracts preprocessing, model inference, and post-processing into a single high-level call.
- External libraries like nltk and os support traditional NLP tasks and file handling.
- Loading external text files allows the notebook to treat them as a knowledge base for downstream NLP tasks.

Section 2: Generative AI – Dumb vs Smart Models



Step 1: Set a Seed

A **seed value** is used to make random results **reproducible**. When we set a seed, the random number generator starts from the same point each time, which means it will produce the **same sequence of random values**.

Try running the code multiple times using the **same seed value** and observe the output.

Now, change the seed value and run the code again. This time, the output **will change** because a different seed creates a different sequence of random numbers.

[55] ✓ 0s

```
set_seed(51)

```

Step 2: Define a Prompt

Both models will complete this sentence.

[56] ✓ 0s

```
prompt = "Generative AI is a revolutionary technology that"

```

Step 3: Fast Model (`distilgpt2`)

Let's see how the smaller model performs.

[57] ✓ 2s

```
# Initialize the pipeline with the specific model
```

- Setting a random seed ensures reproducible text generation outputs.
- Changing the seed value results in different generated text, highlighting the stochastic nature of language models.
- Smaller distilled models like distilgpt2 generate text faster but may lose coherence.
- Larger models like gpt2 generally produce more context-aware and meaningful text.
- Model size and training depth directly affect output quality and relevance.

Section 3: NLP Fundamentals – Under the Hood

Tokenization

- Text must be converted into tokens before being processed by a model.
- Each token is mapped to a unique numerical ID understood by the model.
- Tokenization can split words into subword units rather than whole words.

Part-of-Speech (POS) Tagging

- POS tagging assigns grammatical labels such as noun, verb, or adjective to words.
- It helps models understand sentence structure and grammatical roles.
- The same word can receive different tags depending on context.
- Example of POS Tagging

  Consider the sentence: "The quick brown fox jumps over the lazy dog."

  After performing POS Tagging, we get:

  "The" is tagged as determiner (DT)

  "quick" is tagged as adjective (JJ)

  "brown" is tagged as adjective (JJ)

  "fox" is tagged as noun (NN)

  "jumps" is tagged as verb (VBZ)

  "over" is tagged as preposition (IN)

  "the" is tagged as determiner (DT)

  "lazy" is tagged as adjective (JJ)

  "dog" is tagged as noun (NN)

Named Entity Recognition (NER)

- NER identifies and classifies entities like names, organizations, and locations.
- Pretrained BERT-based models can accurately extract structured information from raw text.
- Filtering by confidence score improves the reliability of extracted entities.

Section 4: Advanced Applications

Summarization

- Different summarization models balance speed and output quality.
- Distilled models generate faster summaries with less detail.
- Larger models produce more coherent and informative summaries.
- Model choice depends on performance requirements and resource constraints.

Question Answering

- Question answering models extract answers directly from a given context.
- The quality of answers depends on how clearly the information appears in the context.
- This task demonstrates practical information retrieval from unstructured text.

Masked Language Modeling

- Masked language modeling predicts missing words based on surrounding context.
- It is the core training objective behind BERT-style models.
- The model assigns probabilities to multiple possible token predictions, showing contextual understanding.