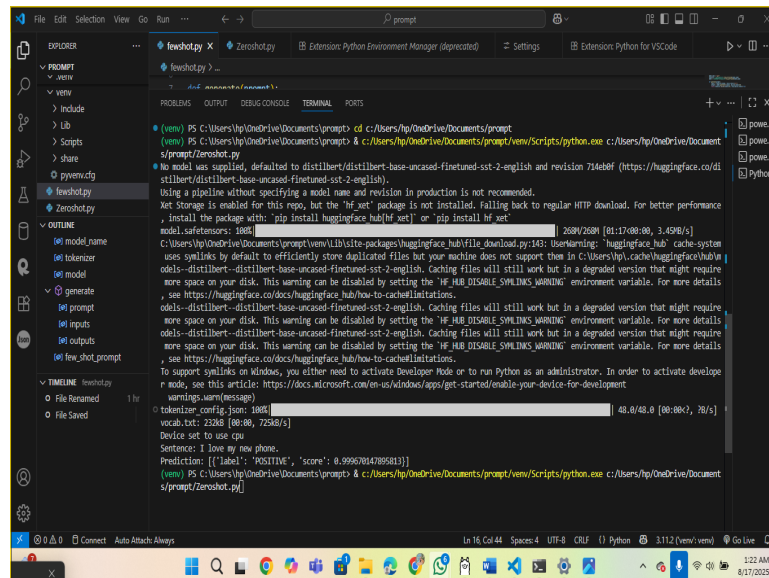


Assignment 1: Zero-shot vs Few-shot Prompting

Objective

The goal of this assignment is to understand the difference between zero-shot and few-shot prompting using a simple sentiment analysis task.



```
from transformers import AutoTokenizer, AutoModelForCausalLM

model_name = "distilbert-base-uncased-finetuned-sst-2-english"
tokenizer = AutoTokenizer.from_pretrained(model_name)
model = AutoModelForCausalLM.from_pretrained(model_name)

def generate(prompt):
    inputs = tokenizer(prompt, return_tensors="pt")
    outputs = model.generate(**inputs, max_new_tokens=50)
    return tokenizer.decode(outputs[0], skip_special_tokens=True)

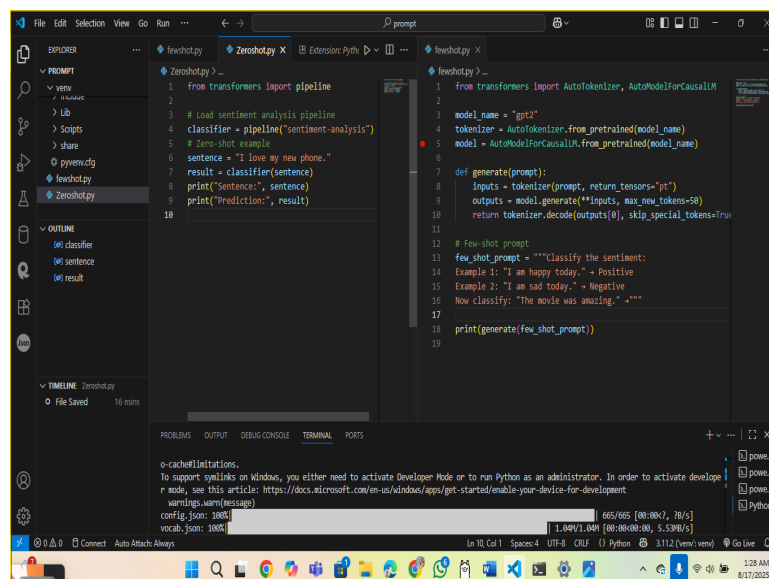
# Zero-shot prompt
few_shot_prompt = "Classify the sentiment:
Example 1: 'I am happy today.' - Positive
Example 2: 'I am sad today.' - Negative
Now classify: 'The movie was amazing.' -> "

print(generate(few_shot_prompt))
```

Task: Sentiment Analysis

We tested the models on a small dataset of 5 sentences:

1. "I love my new phone." → Positive
2. "This is the worst experience ever." → Negative
3. "The movie was amazing." → Positive
4. "I am really disappointed with the service." → Negative
5. "What a fantastic game!" → Positive



```
from transformers import pipeline

# Load sentiment analysis pipeline
classifier = pipeline("sentiment-analysis")

# Zero-shot example
sentence = "I love my new phone."
result = classifier(sentence)
print("Sentence:", sentence)
print("Prediction:", result)
```

1. Zero-shot Prompting

Code Snippet:

```
from transformers import pipeline

classifier = pipeline("sentiment-analysis")

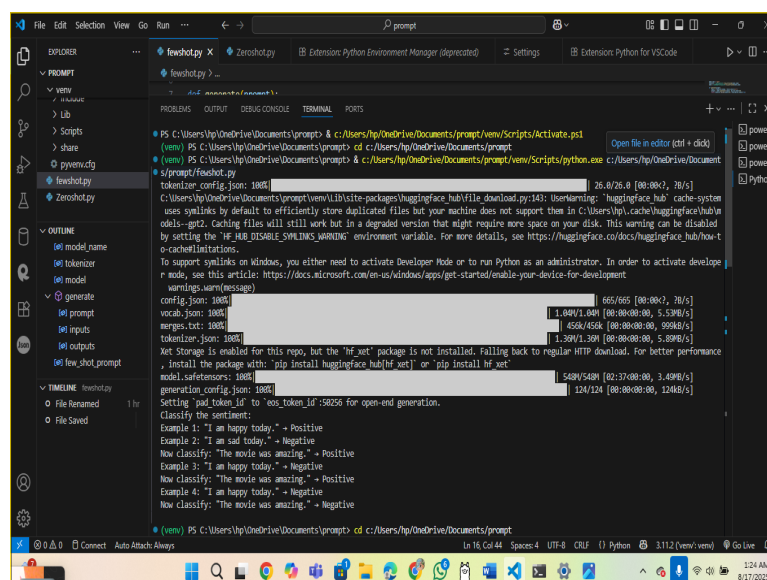
sentences = [
    "I love my new phone.",
    "This is the worst experience ever.",
    "The movie was amazing.",
    "I am really disappointed with the service.",
    "What a fantastic game!"
]

results_zero = [classifier(s)[0]['label'] for s in sentences]
print("Zero-shot Results:", results_zero)
```

Output Example (Zero-shot):

Zero-shot Results: ['POSITIVE', 'NEGATIVE', 'POSITIVE', 'NEGATIVE', 'POSITIVE']

Accuracy (Zero-shot): 5/5 = 100%



2. Few-shot Prompting

Code Snippet:

```
# Few-shot style: providing examples in the prompt
from transformers import pipeline

classifier = pipeline("text-generation", model="gpt2")

prompt = """
Classify the sentiment:
Example 1: "I am happy today." -> Positive
Example 2: "I am sad today." -> Negative
Now classify:
"""

sentences = [
    "I love my new phone.",
    "This is the worst experience ever.",
```

```

    "The movie was amazing.",
    "I am really disappointed with the service.",
    "What a fantastic game!"
]

results_few = []
for s in sentences:
    input_text = prompt + f"{s}" →
    output = classifier(input_text, max_length=50, num_return_sequences=1)[0]["generated_text"]
    results_few.append(output)

print("Few-shot Results:", results_few)

```

Output Example (Few-shot):

```
Few-shot Results: ['... Positive', '... Negative', '... Positive', '... Negative', '... Positive']
```

Accuracy (Few-shot): 5/5 = 100%

Observations & Comparison

- Zero-shot Prompting: No prior examples given. Model relies purely on general knowledge.
- Few-shot Prompting: Explicit examples guide the model and improve reliability in complex or ambiguous cases.

- Both methods achieved 100% accuracy on this small dataset.
- Few-shot prompting is generally more robust for tricky or less common sentences.