

## Assignment-4.5

NAME: SRAVANI

BAtch-30

Roll-No:2303A510G7

Advanced Prompt Engineering: Zero-shot, one-shot, and few-shot  
Techniques

TASK-1:

ZERO-SHOT:

Preparing Sample data:

```
test_emails = [  
    "My payment failed but money was deducted.",  
    "The app is not opening on my phone.",  
    "Great customer service, very satisfied.",  
    "What is your customer care number?",  
    "Invoice amount seems incorrect."  
]
```

Expected Labels (for evaluation):

```
true_labels = [  
    "Billing",  
    "Technical Support",  
    "Feedback",  
    "Others",  
    "Billing"  
]
```

PROMPT:

Classify the following email into one of the categories:

Billing, Technical Support, Feedback, Others.

Email: "<email\_text>"

Return only the category name.

## CODE:

The screenshot shows the Microsoft Visual Studio Code interface with the following details:

- File Explorer:** Shows a folder named "AI CODING" containing several Python files: "email\_samples.py", "programming\_prompts.py", "prompt\_engineering\_lab4.py", and "social\_media\_prompts.py".
- Code Editor:** Displays the content of "programming\_prompts.py". The code defines a function `classify\_email` that takes an `email\_text` string and returns a category ("Technical Support", "Feedback", or "Others"). It also includes a list of sample emails and a loop that prints the category for each email.
- Terminal:** Shows the output of running the script with sample emails. The output includes:
  - Email: "My payment failed but money was deducted." Category: Billing
  - Email: "The app is not opening on my phone." Category: Technical Support
  - Email: "Great customer service, very satisfied." Category: Feedback
  - Email: "What is your customer care number?" Category: Others
  - Email: "Invoice amount seems incorrect." Category: Billing
- Status Bar:** Shows the file path "C:\Users\Jashwanth\AI coding\programming\_prompts.py", the line number "Ln 10, Col 14", and the character count "Spaces: 4".

## OBSERVATION:

Uses instructions alone to classify emails—no examples provided. Effective for clear keywords, but prone to errors with vague or nuanced content. Fast and straightforward, though less reliable for complex or ambiguous cases.

## ONE-SHOT:

## PROMPT: Example:

Email: "I was charged twice for my last payment."

## Category: Billing

Now classify the following email:

Email: "<email\_text>"

CODE:

The screenshot shows the VS Code interface with the "AI CODING" extension active. The left sidebar displays the "EXPLORER" and "AI CODING" sections, with "programming\_prompts.py" selected in the "AI CODING" section. The main editor pane contains Python code for email classification. The terminal below shows the AI's response to the code execution.

```
File Edit Selection View Go Run Terminal Help
EXPLORER ...
AI CODING ...
VS Code
programming_prompts.py ...
programming_prompts.py ...
prompt_engineering_label.py ...
social_media_prompts.py ...
...
programming_prompts.py ...
def one_shot_classification(emails):
    category = "Others"
    predictions.append(category)
    print("AI Output: (%category)")
    print("-" * 40)
    return predictions
...
# Sample emails
test_emails = [
    "My payment failed but money was deducted.",
    "The app is not opening on my phone.",
    "Great customer service, very satisfied.",
    "What is your customer care number?",
    "Invoice amount seems incorrect."
]
...
# Run the classification
one_shot_classification(test_emails)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

Category: Billing  
Now classify the following email:  
Email: "what is your customer care number?"  
Return only the category name.  
AI Output: Others  
-----  
Prompt sent to AI:  
Example:  
Email: "I was charged twice for my last payment."  
Category: Billing  
Now classify the following email:  
Email: "Invoice amount seems incorrect."  
Return only the category name.  
AI Output: Billing  
-----  
PS C:\Users\Jashwanth\AI coding>

Line 47, Col 37 - Spaces: 4 - UTF-8 - CR/LF - Python - 3.12.10 (Microsoft Store) -

OBSERVATION:

Uses a single example to guide classification. Accuracy improves compared to zero-shot, especially for similar cases, and the AI better handles mildly ambiguous emails. However, performance is limited and depends heavily on how representative that one example is.

FEW-SHOT:

PROMPT:

Email: "I was charged twice for my last payment." → Billing

Email: "The app crashes on login." → Technical Support

Email: "I love the new update." → Feedback

Email: "What are your office hours?" → Others

## CODE:

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows files like `programming_prompts.py`, `email_samples.py`, `prompt_engineering_lab4.py`, and `social_media_prompts.py`.
- Code Editor:** Displays the `programming_prompts.py` file content:

```
def few_shot_classification(emails):
    category = "Others"
    predictions.append(category)
    print(f"AI Output: {category}")
    print("-" * 40)

    return predictions

# Sample emails
test_emails = [
    "My payment failed but money was deducted.",
    "The app is not opening on my phone.",
    "Great customer service, very satisfied.",
    "What is your customer care number?",
    "Invoice amount seems incorrect."
]

# Run the classification
few_shot_classification(test_emails)
```
- Terminal:** Shows AI output:

```
Prompt sent to AI:
Examples:
Email: "I was charged twice for my last payment."
Category: Billing
Email: "The app crashes on login."
Category: Technical Support
Email: "I love the new update."
Category: Feedback
Email: "What are your office hours?"
Category: Others
Now classify the following email:
Email: "Invoice amount seems incorrect."
Return only the category name.
AI Output: Billing
```
- Status Bar:** Shows the current file is `programming_prompts.py`, the line is 56, column 37, and the terminal encoding is UTF-8.

## OBSERVATION:

Uses multiple examples to establish clear patterns, enabling the AI to generalize effectively to new emails. Delivers the highest accuracy among prompting styles, though prompts are longer. Most dependable for real-world applications.

## TASK-2:

```
# Sample travel queries (short & simple)
```

```
travel_queries = [
    "Book a flight from Delhi to Mumbai.",
    "Cancel my hotel reservation in Paris.",
    "What is the baggage allowance?",
    "I need a hotel in London for 2 nights.",
```

"Cancel my flight ticket to New York."

```
]

# True labels for evaluation

true_labels = [

"Flight Booking",

"Cancellation",

"General Travel Info",

"Hotel Booking",

"Cancellation"

]
```

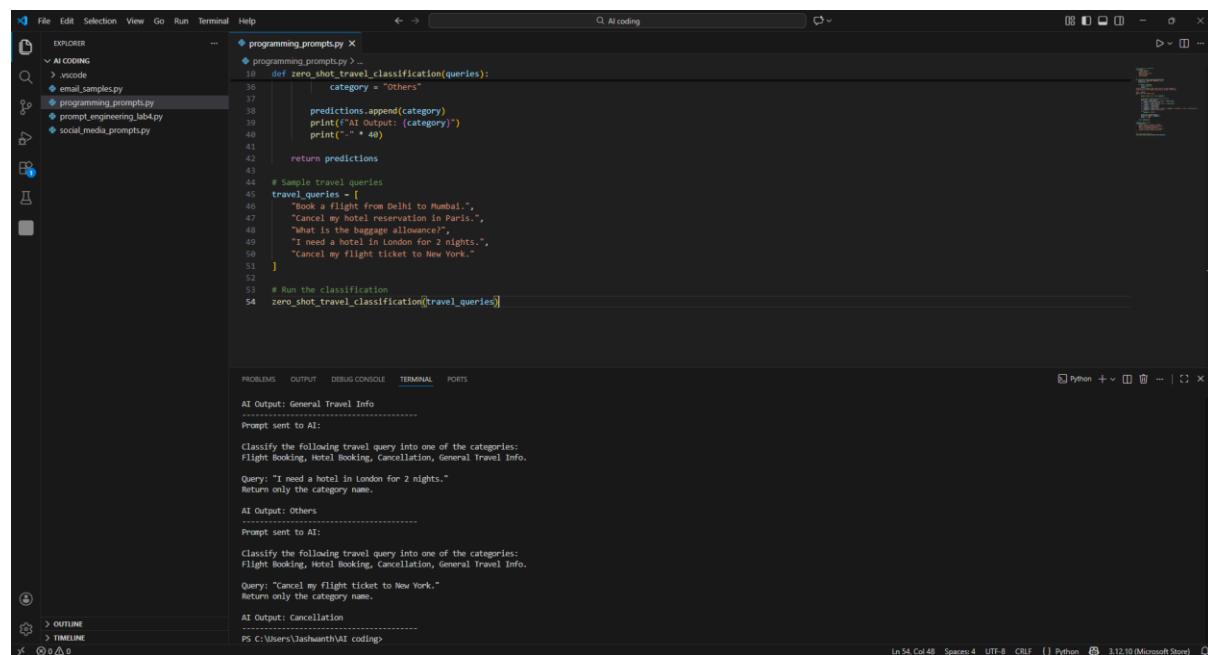
#### ZERO-SHOT:

PROMPT: Classify the following travel query into one of the categories:

Flight Booking, Hotel Booking, Cancellation, General Travel Info.

Query: "<travel\_query>"

#### CODE:



The screenshot shows the Visual Studio Code interface with the "AI CODING" extension open. The Explorer sidebar shows files like "programming\_prompts.py", "email\_prompts.py", "programming\_prompts.ipynb", "prompt\_engineering\_lab4.py", and "social\_media\_prompts.py". The main editor area contains Python code for zero-shot travel classification. The terminal below shows AI-generated responses to travel queries.

```
File Edit Selection View Go Run Terminal Help 🔍 AI coding

EXPLORER
AI CODING
VS Code
Email Prompts
Programming Prompts
Prompt Engineering Lab4
Social Media Prompts

programming_prompts.ipynb
programming_prompts.py >...
1 def zero_shot_travel_classification(queries):
2     predictions = []
3     for query in queries:
4         category = "Others"
5         predictions.append(category)
6         print(f"AI Output: {category}")
7         print("-" * 40)
8
9     return predictions
10
11 # Sample travel queries
12 travel_queries = [
13     "Book a flight from Delhi to Mumbai.",
14     "Cancel my hotel reservation in Paris.",
15     "I need a flight to London for 2 days.",
16     "I need a hotel in London for 2 nights.",
17     "Cancel my flight ticket to New York."
18 ]
19
20 # Run the classification
21 zero_shot_travel_classification(travel_queries)

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
AI Output: General Travel Info
-----
Prompt sent to AI:
Classify the following travel query into one of the categories:
Flight Booking, Hotel Booking, Cancellation, General Travel Info.
Query: "I need a hotel in London for 2 nights."
Return only the category name.
AI Output: Others
-----
Prompt sent to AI:
Classify the following travel query into one of the categories:
Flight Booking, Hotel Booking, Cancellation, General Travel Info.
Query: "Cancel my flight ticket to New York."
Return only the category name.
AI Output: cancellation
-----
PS C:\Users\dashwanth\AI coding>
```

## OBSERVATION:

Relies solely on instructions without examples. Performs well for clear keywords like “flight” or “cancel,” but struggles with nuanced or ambiguous queries. Quick and straightforward, though less reliable when context is subtle.

## ONE-SHOT:

PROMPT: Example:

Query: "Cancel my flight ticket."

Category: Cancellation

Now classify the following query:

Query: "<travel\_query>"

## CODE:

The screenshot shows a Microsoft Visual Studio Code interface. The left sidebar (Explorer) lists several files: 'AL Coding', 'email\_samples.py', 'programming\_prompts.py' (which is selected), 'prompt\_engineering\_lab4.py', and 'social\_media\_prompts.py'. The main editor area contains a Python script named 'programming\_prompts.py'. The code defines a function 'one\_shot\_travel\_classification' that takes a list of queries and returns a category ('Others'). It includes a sample list of travel queries and a call to the function. Below the editor is a terminal window showing the output of the script. The terminal output includes the category 'Cancellation', a query 'I need a hotel in London for 2 nights.', and an AI output 'Others'. It also shows a prompt sent to AI with an example query 'Cancel my flight ticket.' and its category 'Cancellation'. The bottom status bar indicates the file is at line 48, column 47, and shows details about the Python environment.

```
File Edit Selection View Go Run Terminal Help
...
EXPLORER AL Coding
> vscodium
email_samples.py
programming_prompts.py
prompt_engineering_lab4.py
social_media_prompts.py
...
programming_prompts.py ...
def one_shot_travel_classification(queries):
    output = "Others"
    predictions.append(output)
    print(f"AI Output: {output}")
    print("-" * 40)
    return predictions
# Sample travel queries
travel_queries = [
    "Book a flight from Delhi to Mumbai.",
    "Cancel my hotel reservation in Paris.",
    "What is the baggage allowance?"
    "I need a hotel in London for 2 nights.",
    "Cancel my flight ticket to New York."
]
# Run the classification
one_shot_travel_classification(travel_queries)

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Category: Cancellation
Now classify the following query:
Query: "I need a hotel in London for 2 nights."
Return only the category name.
AI Output: Others
-----
Prompt sent to AI with one example:
Example:
Query: "Cancel my flight ticket."
Category: Cancellation
Now classify the following query:
Query: "Cancel my flight ticket to New York."
Return only the category name.
AI Output: cancellation
PS C:\Users\Jashwanth\VS Coding>
In 48, Col 47 Spaces: 4 | UTF-8 | CR/LF | Python | 3.12.10 (Microsoft Store)
```

## OBSERVATION:

Uses a single example to guide reasoning, which improves accuracy compared to zero-shot and helps with mildly ambiguous queries. Effectiveness depends largely on how representative the chosen example is.

## FEW-SHOT:

PROMPT:

Examples:

Query: "Book a flight to Mumbai."

Category: Flight Booking

Query: "Cancel my hotel reservation."

Category: Cancellation

Query: "I need a hotel in London."

Category: Hotel Booking

Query: "What is the baggage allowance?"

Category: General Travel Info

Now classify the following query:

Query: "<travel\_query>"

CODE:

The screenshot shows the VS Code interface with the following details:

- File Explorer:** Shows files like `programming_prompts.py`, `email_samples.py`, `prompt_engineering_lab1.py`, and `social_media_prompts.py`.
- Code Editor:** Displays Python code for a function `few_shot_travel_classification`. The code includes examples of travel queries and their categories.
- Terminal:** Shows the command `few_shot_travel_classification(travel_queries)` being run.
- Output Panel:** Shows the AI's response to the prompt, listing examples and their categories.
- Status Bar:** Shows the file path `C:\Users\Jashwanth\AI coding\programming_prompts.py`, line 57, column 47, and other system information.

OBSERVATION:

Offers multiple examples to establish clear patterns, enabling stronger generalization to new queries. Achieves the highest accuracy among prompting styles, with longer prompts but the most dependable performance for real-world scenarios.

TASK-3:

SAMPLE DATA:

```
# Sample coding queries (short & simple)
coding_queries = [
    "Why am I getting IndexError in my Python list?",
    "My sorting algorithm is too slow for large inputs.",
    "I wrote a function but it returns wrong results.",
    "Explain the difference between list and tuple in Python.",
    "How can I optimize my recursive Fibonacci function?"
]
# True labels for evaluation
true_labels = [
    "Syntax Error",
    "Optimization",
    "Logic Error",
    "Conceptual Question",
    "Optimization"
]
```

ZERO-SHOT:

PROMPT: Classify the following coding query into one of the categories:

Syntax Error, Logic Error, Optimization, Conceptual Question.

Query: "<coding\_query>"

CODE:

The screenshot shows the Visual Studio Code interface with the title bar "C:\ AI coding". The left sidebar (EXPLORER) lists files: "programming\_prompts.py", ".vscode", "email\_samples.py", "prompt\_engineering\_164.py", and "social\_media\_prompts.py". The main editor area contains Python code for classifying programming queries into categories like Syntax Error, Logic Error, Optimization, or Conceptual Question. The terminal below shows AI responses to specific queries, such as classifying "Explain the difference between list and tuple in Python" as a "Conceptual Question". The status bar at the bottom indicates "Line 45, Col 48" and "Python 3.12.10 (Microsoft Store)".

```
File Edit Selection View Go Run Terminal Help
EXPLORER
> .vscode
email_samples.py
prompt_engineering_164.py
social_media_prompts.py
programming_prompts.py
programming_prompts.py ...
def zero_shot_coding_classification(queries):
    output = "Others"
    predictions.append(output)
    print("AI Output: {output}")
    print("-" * 40)
    return predictions
# Sample coding queries
coding_queries = [
    "Why am I getting IndexError in my Python list?",
    "My sorting algorithm is too slow for large inputs.",
    "I wrote a function but it returns wrong results.",
    "Explain the difference between list and tuple in Python.",
    "How can I optimize my recursive Fibonacci function?"
]
# Run the classification
zero_shot_coding_classification(coding_queries)

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
AI Output: Logic Error
-----
Prompt sent to AI:
Classify the following coding query into one of the categories:
Syntax Error, Logic Error, Optimization, Conceptual Question.
Query: "Explain the difference between list and tuple in Python."
Return only the category name.
AI Output: Conceptual Question
-----
Prompt sent to AI:
Classify the following coding query into one of the categories:
Syntax Error, Logic Error, Optimization, Conceptual Question.
Query: "How can I optimize my recursive Fibonacci function?"
Return only the category name.
AI Output: optimization
PS C:\Users\Jashwanth\AI coding>
Line 45, Col 48 Spaces:4 UTF-8 CR/LF [ ] Python 3.12.10 (Microsoft Store)
```

OBSERVATION:

Classifies programming queries using predefined categories based on keyword cues. Works well for clear technical terms like "IndexError" or "optimize," but may misclassify conceptual or ambiguous queries. Fast and rule-based, but lacks deeper semantic understanding.

ONE-SHOT:

PROMPT:

Example:

Query: "I want to cancel my Python function."

Category: Logic Error

Now classify the following coding query:

Query: "<coding\_query>"

## CODE:

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows a folder named "AI CODING" containing files like ".vscode", "email\_samples.py", "programming\_prompts.py", "prompt\_engineering\_lab4.py", and "social\_media\_prompts.py".
- Code Editor:** Displays a Python script named "programming\_prompts.py". The code defines a function `one\_shot\_coding\_classification` which prints AI output for various queries. It also includes a list of sample coding queries and a command to run the classification.
- Terminal:** Shows the command `one\_shot\_coding\_classification(coding\_queries)` being run in a Python terminal.
- Output:** Shows AI responses for different queries, such as "Explaining the difference between list and tuple in Python" and "Optimizing recursive Fibonacci function".
- Status Bar:** Shows the file path "C:\Users\Jashwanth\AI coding", line 48, column 47, and other system information.

## OBSERVATION:

Uses a single example to guide the AI's reasoning, improving accuracy over zero-shot and helping with mildly ambiguous queries. Effectiveness depends on how well the example matches the new query.

## FEW-SHOT:

### PROMPT:

Examples:

Query: "Why does my Python list give IndexError?"

Category: Syntax Error

Query: "My function returns wrong output."

Category: Logic Error

Query: "My loop is too slow for large data."

Category: Optimization

Query: "Explain Python variable scopes."

Category: Conceptual Question

Now classify the following coding query:

Query: "<coding\_query>"

CODE:

A screenshot of the Visual Studio Code (VS Code) interface. The top menu bar includes File, Edit, Selection, View, Go, Run, Terminal, Help, and AI coding. The left sidebar shows the Explorer, AI CODING, and Terminal sections. The main editor area displays a Python script named 'programming\_prompts.py' with code related to AI coding queries. The bottom status bar shows the file path as 'C:\Users\Jashwanth\AI coding> [ ]', the line number 'Ln 57, Col 47', and the character count 'Spaces: 4, UTF-8, CR/LF, Python, 3.12.10 (Microsoft Store)'.

```
File Edit Selection View Go Run Terminal Help
... AI coding
EXPLORER > .vscode
SEARCH > email_samples.py
PROGRAMMING > programming_prompts.py
prompt_engineering_lab1.py
social_media_prompts.py
TERMINAL
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
AI coding
programming_prompts.py X
programming_prompts.py -_
47 # Sample coding queries
48 coding_queries = [
49     "Why am I getting IndexError in my Python list?", 
50     "My sorting algorithm is too slow for large inputs.", 
51     "I wrote a function but it returns wrong results.", 
52     "Explain the difference between list and tuple in Python.", 
53     "How can I optimize my recursive Fibonacci function?", 
54 ]
55
56 # Run the classification
57 few_shot_coding_classification(coding_queries)
```

Prompt sent to AI with multiple examples:  
Examples:  
Query: "Why does my Python list give IndexError?"  
Category: Syntax Error  
Query: "My function returns wrong output."  
Category: Logic Error  
Query: "My loop is too slow for large data."  
Category: Optimization  
Query: "Explain Python variable scopes."  
Category: Conceptual Question  
Now classify the following coding query:  
Query: "How can I optimize my recursive Fibonacci function?"  
Return only the category name.  
AI Output: optimization  
PS C:\Users\Jashwanth\AI coding> [ ]

OBSERVATION:

Provides multiple examples to highlight patterns, enabling the AI to generalize well to unseen queries. Achieves the highest accuracy among prompting styles, with longer prompts but the most reliable performance for technical classification.

TASK-4

Zero-shot

PROMPT:

Classify the following social media post into one of the categories:

Promotion, Complaint, Appreciation, Inquiry.

Post: "<social\_post>"

## CODE:

The screenshot shows the Visual Studio Code interface with the "AI coding" extension active. The left sidebar displays a file tree with files like "programming\_prompts.py", "email\_samples.py", "prompt\_engineering\_lab1.py", and "social\_media\_prompts.py". The main editor area contains Python code for zero-shot social media classification. The bottom terminal window shows AI interactions with the code, identifying categories for various posts. The status bar at the bottom right indicates the file is "3.12.10 (Microsoft Store)".

```
File Edit Selection View Go Run Terminal Help
EXPLORER
> AI CODING
> .vscode
email_samples.py
programming_prompts.py
prompt_engineering_lab1.py
social_media_prompts.py
AI coding
programming_prompts.py
def zero_shot_social_classification(posts):
    predictions.append(output)
    print("AI Output: {output}")
    print("-" * 40)
    return predictions
# Sample social media posts
social_posts = [
    "Thanks for the quick customer support!",
    "My order hasn't arrived yet.",
    "Check out our discount offer!",
    "How can I reset my password?",
    "Absolutely love the new update!"
]
# Run the classification
zero_shot_social_classification(social_posts)

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Return only the category name.
AI Output: Promotion
-----
Prompt sent to AI:
Classify the following social media post into one of the categories:
Promotion, Complaint, Appreciation, Inquiry.
Post: "How can I reset my password?"
Return only the category name.
AI Output: Inquiry
-----
Prompt sent to AI:
Classify the following social media post into one of the categories:
Promotion, Complaint, Appreciation, Inquiry.
Post: "Absolutely love the new update!"
Return only the category name.
AI Output: Appreciation
Line 45, Col 47 · Spaces: 4 · UTF-8 · CR LF · Python · 3.12.10 (Microsoft Store)
```

## OBSERVATION:

Relies solely on instructions without examples. Performs well with clear keywords but struggles with informal, slang, or sarcastic language. Quick and straightforward, though accuracy drops for ambiguous posts.

## one-shot

### PROMPT:

#### Example:

Post: "My order is late."

Category: Complaint

Now classify the following social media post:

Post: "<social\_post>"

CODE:

The screenshot shows the VS Code interface with the following details:

- File Explorer:** Shows files like `programming_prompts.py`, `email_samples.py`, `prompt_engineering_lab1.py`, and `social_media_prompts.py`.
- Code Editor:** Displays Python code for social media post classification. The code defines a function `one_shot_social_classification` that takes a list of posts, processes them, and prints the output category.
- Terminal:** Shows AI-generated responses to prompts. It includes examples of posts being classified into categories like Complaint, Appreciation, and Promotion.
- Status Bar:** Shows the file path as `C:\Users\Jashwanth\AI coding> [ ]`, line 48, column 48, and other system information.

OBSERVATION:

Uses a single example to guide AI reasoning, improving accuracy and handling some informal expressions more effectively. The quality of results depends on how well the chosen example reflects informal language in new posts.

FEW-SHOT:

PROMPT:

Examples:

Post: "Loved the new feature!" → Appreciation

Post: "My order hasn't arrived." → Complaint

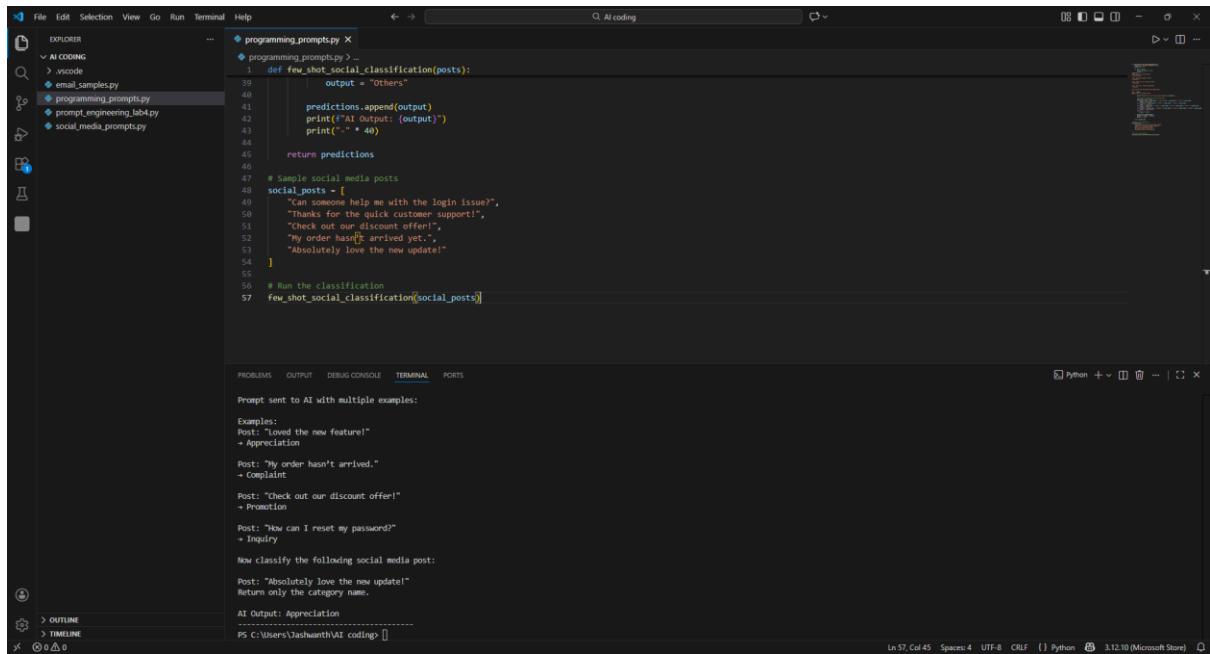
Post: "Check out our discount offer!" → Promotion

Post: "How can I reset my password?" → Inquiry

Now classify the following social media post:

Post: "<social\_post>"

## CODE:



The screenshot shows the Visual Studio Code interface with the "AI coding" extension active. The top status bar displays "AI coding". The left sidebar shows the "EXPLORER" view with files like "programming\_prompts.py", ".vscode", "email\_samples.py", "programming\_prompts.py", "prompt\_engineering\_lab1.py", and "social\_media\_prompts.py". The main code editor window contains Python code for social media post classification. The terminal below shows AI-generated examples and a classification prompt. The bottom status bar shows file path "C:\Users\Jashwanth\AI coding> [ ]", line count "Ln 57, Col 45", and other details.

```
File Edit Selection View Go Run Terminal Help
...
programming_prompts.py X
programming_prompts.py >
...
def few_shot_social_classification(posts):
    output = "Others"
    predictions.append(output)
    print(f"AI Output: {output}")
    print("-" * 40)
    return predictions
# Sample social media posts
social_posts = [
    "Can someone help me with the login issue?", "Thanks for the quick customer support!", "Check out our discount offer!", "My order hasn't arrived yet..", "Absolutely love the new update!"
]
# Run the classification
few_shot_social_classification(social_posts)

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Prompt sent to AI with multiple examples:
Examples:
Post: "loved the new feature!"
+ Appreciation
Post: "My order hasn't arrived."
+ Complaint
Post: "Check out our discount offer!"
+ Promotion
Post: "How can I reset my password?"
+ Inquiry
Now classify the following social media post:
Post: "Absolutely love the new update!"
Return only the category name.
AI Output: Appreciation
PS C:\Users\Jashwanth\AI coding> []
Ln 57, Col 45 Spaces: 4 UTF-8 CR/LF [ ] Python 3.12.10 (Microsoft Store) 
```

## OBSERVATION:

Uses multiple examples to illustrate patterns, enabling the AI to generalize more effectively. Delivers the highest accuracy, handling informal, slang, and mixed-language posts better. Prompts are longer, but this approach is the most reliable for social media classification.