

Lab Assignment – 4.5

Hall Ticket No.: 2303A510D9

Bt– 29

Suppose that you work for a company that receives hundreds of customer emails daily. Management wants to automatically classify emails into categories like "Billing", "Technical Support", "Feedback", and "Others" before assigning them to appropriate departments. Instead of training a new model, your task is to use prompt engineering techniques with an existing LLM to handle the classification.

Tasks to be completed are as below a.

Prepare Sample Data:

- Create or collect 10 short email samples, each belonging to one of the 4 categories.

b. Zero-shot Prompting:

- Design a prompt that asks the LLM to classify a single email without providing any examples.

- Example prompt:

“Classify the following email into one of the following categories:

Billing, Technical Support, Feedback, Others. Email: ‘I have not

received my invoice for last month.’” c. One-shot Prompting:

- Add one labeled example before asking the model to classify a new email.

d. Few-shot Prompting:

- Use 3–5 labeled examples in your prompt before asking the model to classify a new email. e.

Evaluation:

- Run all three techniques on the same set of 5 test emails.

- Compare and document the accuracy and clarity of responses.

```

1 # Zero-shot, One-shot, Few-shot
2
3 def llm_simulator(prompt):
4     """
5     This function simulates an LLM response.
6     In real applications, this is where API calls go.
7     """
8     print("\n--- PROMPT SENT TO MODEL ---")
9     print(prompt)
10    print("-----")
11
12    # Dummy output for assignment
13    return "Predicted Category"
14
15
16
17 # 1. EMAIL CLASSIFICATION
18
19 def email_zero_shot(email):
20     prompt = f"""
21     Classify the email into:
22     Billing, Technical Support, Feedback, Others
23
24     Email: {email}
25     Category:
26     """
27     return llm_simulator(prompt)
28
29
30 def email_one_shot(email):
31     prompt = f"""
32     Example:
33     Email: My payment was deducted twice.
34     Category: Billing
35
36     Now classify:
37     Email: {email}
38     Category:
39     """
40     return llm_simulator(prompt)
41
42
43 def email_few_shot(email):
44     prompt = f"""
45     Example 1:
46     Email: I was charged extra.
47     Category: Billing
48
49     Example 2:
50     Email: App crashes on login.
51     Category: Technical Support
52
53     Example 3:
54     Email: Great service!
55     Category: Feedback
56
57     Now classify:
58     Email: {email}
59     Category:
60     """
61     return llm_simulator(prompt)
62
63
64
65 # 2. TRAVEL QUERY CLASSIFICATION
66
67 def travel_few_shot(query):

```

```

68     """
69     Example 1:
70     Email: I was charged extra.
71     Category: Billing
72
73     Example 2:
74     Email: App crashes on login.
75     Category: Technical Support
76
77     Example 3:
78     Email: Great service!
79     Category: Feedback
80
81     Now classify:
82     Email: {email}
83     Category:
84     """
85     return llm_simulator(prompt)
86
87
88
89 # 2. TRAVEL QUERY CLASSIFICATION
90
91 def travel_few_shot(query):

```

2. Travel Query Classification Scenario:

A travel assistant must classify queries into Flight Booking, Hotel Booking, Cancellation, or General Travel Info.

Tasks:

- Prepare labeled travel queries.
- Apply Zero-shot prompting.

- c. Apply One-shot prompting.
- d. Apply Few-shot prompting.
- e. Compare response consistency.

```

41 def email_few_shot(email):
42     """
43     """
44     return llm_simulator(prompt)
45
46 # 2. TRAVEL QUERY CLASSIFICATION
47
48 def travel_few_shot(query):
49     prompt = f"""
50     Example 1:
51     Query: Book a flight to Delhi
52     Category: Flight Booking
53
54     Example 2:
55     Query: Cancel my hotel booking
56     Category: Cancellation
57
58     Example 3:
59     Query: Best time to visit Goa?
60     Category: General Travel Info
61
62     Now classify:
63     Query: {query}
64     Category:
65     """
66     return llm_simulator(prompt)
67
68 # 3. PROGRAMMING QUESTION CLASSIFICATION
69
70 def programming_few_shot(question):
71     prompt = f"""
72     Example 1:
73     Question: Missing semicolon error
74
75     """
76     return llm_simulator(prompt)

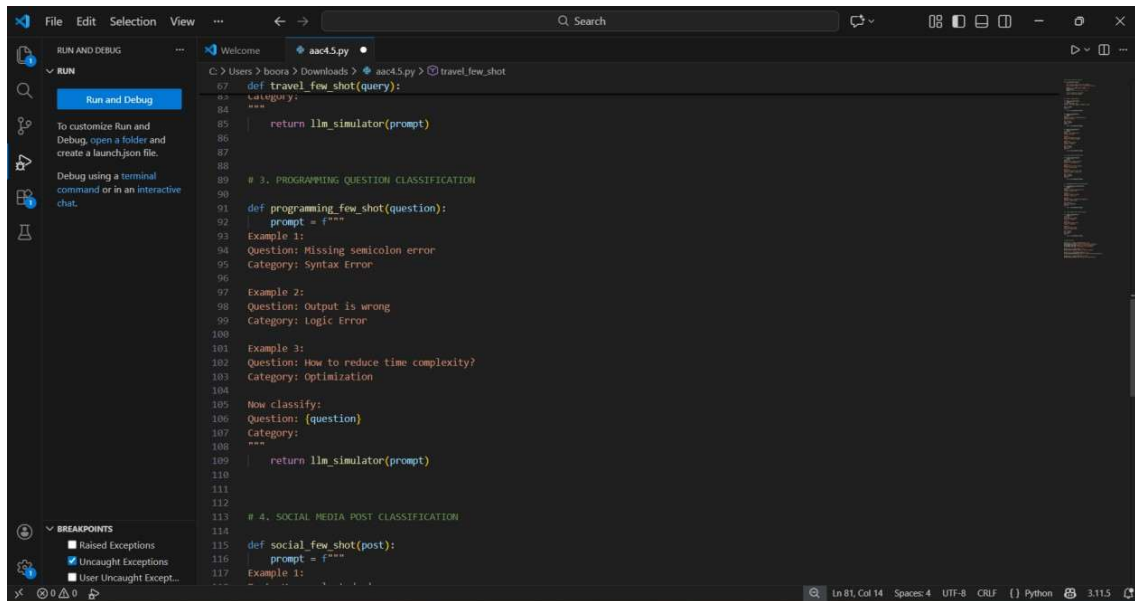
```

3. Programming Question Type Identification Scenario:

A coding help chatbot must classify queries into Syntax Error, Logic Error, Optimization, or Conceptual Question.

Tasks:

- a. Prepare coding-related user queries.
- b. Perform Zero-shot classification.
- c. Perform One-shot classification.
- d. Perform Few-shot classification.
- e. Analyze improvements in technical accuracy.

A screenshot of a Python IDE window. The main editor area shows a Python script with several functions. The first function, `travel_few_shot(query)`, uses an `llm_simulator` to generate a response. The second function, `programming_few_shot(question)`, uses a similar `llm_simulator` to generate a response. The third function, `social_few_shot(post)`, also uses an `llm_simulator`. The script includes comments for each function and example prompts. The left sidebar shows the 'RUN AND DEBUG' panel with a 'Run and Debug' button. The bottom status bar indicates the file is 'aac4.5.py' and the editor is in 'Python' mode.

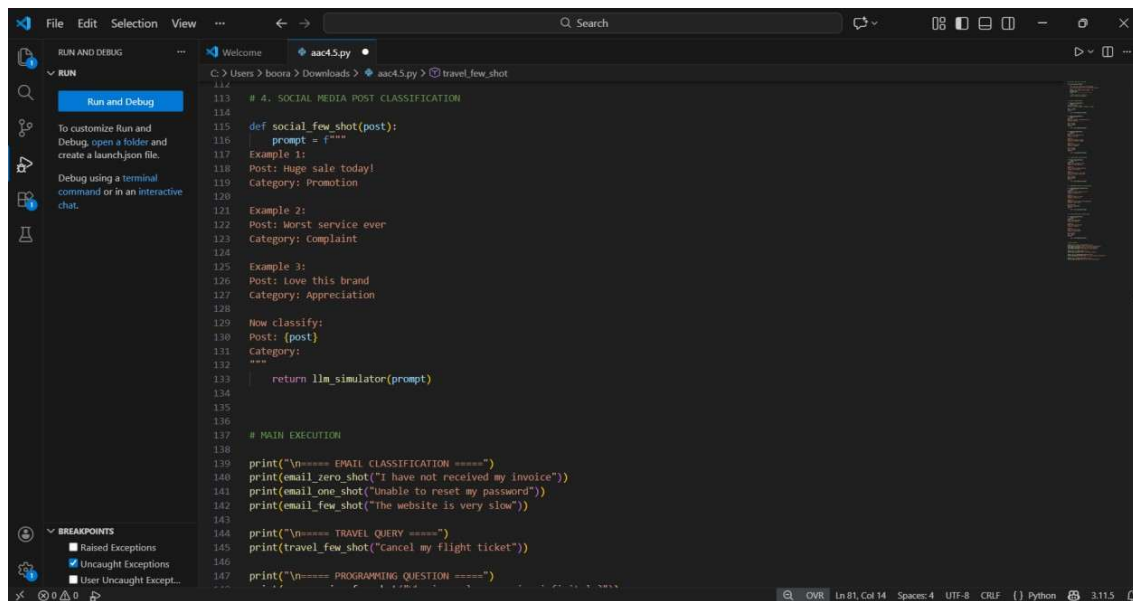
```
67 def travel_few_shot(query):
68     prompt = f"""
69     """
70
71     return llm_simulator(prompt)
72
73 # 3. PROGRAMMING QUESTION CLASSIFICATION
74
75 def programming_few_shot(question):
76     prompt = f"""
77     Example 1:
78     Question: Missing semicolon error
79     Category: Syntax Error
80
81     Example 2:
82     Question: Output is wrong
83     Category: Logic Error
84
85     Example 3:
86     Question: How to reduce time complexity?
87     Category: Optimization
88
89     Now classify:
90     Question: {question}
91     Category:
92     """
93
94     return llm_simulator(prompt)
95
96 # 4. SOCIAL MEDIA POST CLASSIFICATION
97
98 def social_few_shot(post):
99     prompt = f"""
100     Example 1:
101     """
102
103     return llm_simulator(prompt)
```

4. Social Media Post Categorization Scenario:

A social media analytics tool must classify posts into Promotion, Complaint, Appreciation, or Inquiry.

Tasks:

1. Prepare sample social media posts.
2. Use Zero-shot prompting.
3. Use One-shot prompting.
4. Use Few-shot prompting.
5. Analyze informal language handling.



The image shows a Visual Studio Code editor window with a Python file named `aac4.5.py` open. The file path is `C:\Users\boora\Downloads> aac4.5.py > travel_few_shot`. The code is as follows:

```
113 # 4. SOCIAL MEDIA POST CLASSIFICATION
114
115 def social_few_shot(post):
116     prompt = f"""
117     Example 1:
118     Post: Huge sale today!
119     Category: Promotion
120
121     Example 2:
122     Post: Worst service ever
123     Category: Complaint
124
125     Example 3:
126     Post: Love this brand
127     Category: Appreciation
128
129     Now classify:
130     Post: {post}
131     Category:
132     """
133     return llm_simulator(prompt)
134
135
136
137 # MAIN EXECUTION
138
139 print("\n===== EMAIL CLASSIFICATION =====")
140 print(email_zero_shot("I have not received my invoice"))
141 print(email_one_shot("Unable to reset my password"))
142 print(email_few_shot("The website is very slow"))
143
144 print("\n===== TRAVEL QUERY =====")
145 print(travel_few_shot("Cancel my flight ticket"))
146
147 print("\n===== PROGRAMMING QUESTION =====")
148
```

The left sidebar shows the 'RUN AND DEBUG' panel with a 'Run and Debug' button. Below it, there are instructions: 'To customize Run and Debug, open a folder and create a launch.json file.' and 'Debug using a terminal command or in an interactive chat.' At the bottom of the sidebar, there is a 'BREAKPOINTS' section with checkboxes for 'Raised Exceptions', 'Uncaught Exceptions', and 'User Uncaught Except...'. The status bar at the bottom indicates 'OVR', 'Ln 81, Col 14', 'Spaces: 4', 'UTF-8', 'CRLF', 'Python', and '3.11.5'.

OUTPUT:

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\boora\Downloads> & 'c:\Users\boora\AppData\Local\Programs\Python\Python31
5.18.0-win32-x64\bundled\libs\debugpy\launcher' '50529' '--' 'C:\Users\boora\Downloads'
```

===== EMAIL CLASSIFICATION =====

--- PROMPT SENT TO MODEL ---

Classify the email into:
Billing, Technical Support, Feedback, Others

Email: I have not received my invoice
Category:

Predicted Category

--- PROMPT SENT TO MODEL ---

Example:
Email: My payment was deducted twice.
Category: Billing

Now classify:
Email: Unable to reset my password
Category:

Predicted Category

--- PROMPT SENT TO MODEL ---

Example 1:
Email: I was charged extra.
Category: Billing

Example 2:
Email: App crashes on login.
Category: Technical Support

Example 3:

Example 1:
Question: Missing semicolon error
Category: Syntax Error

Example 2:
Question: Output is wrong
Category: Logic Error

Example 3:
Question: How to reduce time complexity?
Category: Optimization

Now classify:
Question: Why is my loop running infinitely?
Category:

Predicted Category

===== SOCIAL MEDIA POST =====

--- PROMPT SENT TO MODEL ---

Example 1:
Post: Huge sale today!
Category: Promotion

Example 2:
Post: Worst service ever
Category: Complaint

Example 3:
Post: Love this brand

Example 2:
Post: Worst service ever
Category: Complaint

Example 3:
Post: Love this brand
Category: Complaint

Example 3:
Post: Love this brand

Example 3:
Post: Love this brand