



School of Computer Science and Artificial Intelligence

Lab Assignment-4.1

Course Title : AI Assistant Coding

Name of Student : Shaik Naved Ahmed

Enrollment No. : 2303A54053

Batch No. : 47-B

Problem Statement 1: Customer Email Classification

A company receives a large number of customer emails every day and wants to automatically classify them into the following categories:

- Billing
- Technical Support
- Feedback
- Others

Instead of training a new machine learning model, the company decides to use prompt engineering techniques with an existing large language model.

Tasks

1. Prepare five short sample emails, each belonging to one of the above categories.
2. Write a zero-shot prompt to classify a given email into one of the categories without providing any examples.
3. Write a one-shot prompt by including one labeled email example and ask the model to classify a new email.
4. Write a few-shot prompt by including two or three labeled email examples and ask the model to classify a new email.
5. Compare the outputs obtained using zero-shot, one-shot, and few-shot prompting techniques and briefly comment on their effectiveness

1. Sample Customer Emails (5)

1. Billing:

“I was charged twice for my subscription this month. Please resolve this issue.”

2. Technical Support:

“The mobile app crashes every time I try to log in.”

3. Feedback:

“I really like the new dashboard design. It’s much easier to use.”

4. Others:

“I want to know your office working hours during holidays.”

5. Billing:

“Can you send me a copy of last month’s invoice?”

2. Zero-Shot Prompt

Classify the following customer email into one of these categories: Billing, Technical Support, Feedback, Others.

Return only the category name.

SCREENSHOT OF GENERATED CODE:

The screenshot shows the AIAC IDE interface. The left sidebar displays a file tree under 'EXPLORER' with files like 'Lab-01.pages', 'Lab-01.pdf', 'Lab-02.docx', 'Lab-02.pages', 'Lab-02.pdf', 'Lab-31.pages', 'Lab-31.pdf', 'Lab-41.py', 'Lab-41.pages', 'Lab-43.pdf', 'Lab-43.py', and 'lab-4.3.py.code-work...'. The main editor window titled 'Welcome' contains the following Python code:

```
#problem statement-1
1 ""Billing:"I was charged twice for my subscription this month. Please resolve this issue."
2 Technical Support:"The mobile app crashes every time I try to log in."
3 Feedback:"I really like the new dashboard design. Its much easier to use."
4 Others:"I want to know your office working hours during holidays."
5 Billing:"Can you send me a copy of last months invoice?"
6
7 Classify the following customer email into one of these categories: Billing, Technical Support, Feedback, Others.
8 Return only the category name.'"
9
10 def classify_email(email):
11     billing_keywords = ["charged", "subscription", "invoice", "payment", "billing"]
12     technical_support_keywords = ["crashes", "log in", "error", "bug", "technical issue"]
13     feedback_keywords = ["like", "dislike", "suggestion", "feedback", "improvement"]
14
15     email_lower = email.lower()
16
17     if any(keyword in email_lower for keyword in billing_keywords):
18         return "Billing"
19     elif any(keyword in email_lower for keyword in technical_support_keywords):
20         return "Technical Support"
21     elif any(keyword in email_lower for keyword in feedback_keywords):
22         return "Feedback"
23     else:
24         return "Others"
25
26 email = "I was charged twice for my subscription this month. Please resolve this issue."
27 category = classify_email(email)
28 print(category) # Output: Billing
```

The right side of the interface includes a 'CHAT' panel with a message 'Build with Agent' and a status bar at the bottom indicating 'Ln 28, Col 35 Spaces: 4 UTT-8 LF {} Python 3.13.5'.

INPUT & OUTPUT:

Input Email	Output	Reason
"I was charged twice for my subscription this month."	Billing	Contains keywords charged and subscription.

EXPLANATION:

The function classifies an email by checking for specific keywords related to billing, technical support, and feedback. It converts the email to lowercase to avoid case-sensitivity issues and then matches keywords using conditional checks. If no keywords are found, the email is categorized as Others.

3. One-Shot Prompt

Classify the customer email into one of these categories:
Billing, Technical Support, Feedback, Others.

Example:

Email: *I was charged twice for my subscription.*

Category: Billing

SCREENSHOT OF GENERATED CODE:

The screenshot shows a code editor interface with a dark theme. On the left is the Explorer sidebar containing files like Lab-01.pages, Lab-01.pdf, Lab-02.docx, Lab-02.pages, Lab-02.pdf, Lab-31.pages, Lab-31.pdf, Lab-31.py, Lab-41.pages, Lab-41.py, Lab-43.pages, Lab-43.pdf, Lab-43.py, and lab-4.3.py.code-wor... In the center, the main editor window displays the following Python code:

```
30     '''Classify the customer email into one of these categories:
31     Billing, Technical Support, Feedback, Others.
32
33     Example:
34     Email: I was charged twice for my subscription.
35     Category: Billing'''
36
37     def classify_email(email):
38         billing_keywords = ["charged", "subscription", "invoice", "payment", "billing"]
39         technical_support_keywords = ["crashes", "log in", "error", "bug", "technical issue"]
40         feedback_keywords = ["like", "dislike", "suggestion", "feedback", "improvement"]
41
42         email_lower = email.lower()
43
44         if any(keyword in email_lower for keyword in billing_keywords):
45             return "Billing"
46         elif any(keyword in email_lower for keyword in technical_support_keywords):
47             return "Technical Support"
48         elif any(keyword in email_lower for keyword in feedback_keywords):
49             return "Feedback"
50         else:
51             return "Others"
52
53     # Example usage
54     email = "The mobile app crashes every time I try to log in."
55     category = classify_email(email)
56     print(category) # Output: Technical Support
```

On the right side of the editor, there is a "Build with Agent" panel with a message: "Build with Agent. AI responses may be inaccurate. Generate Agent Instructions to onboard AI onto your codebase." Below this is a small input field with placeholder text "Describe what to build next". At the bottom of the editor, status bar information includes "Ln 54, Col 45", "Spaces: 4", "UTF-8", "LF", "Python", and "3.13.5".

INPUT & OUTPUT:

Input Email	Output
“The mobile app crashes every time I try to log in.”	Technical Support

EXPLANATION:

The function classifies an email by searching for predefined keywords related to Billing, Technical Support, and Feedback. It converts the email text to lowercase to ensure accurate matching and checks each keyword list in order. If a matching keyword is found, the corresponding category is returned; otherwise, the email is classified as Others.

4. Few-Shot Prompt

Classify the customer email into one of these categories:
Billing, Technical Support, Feedback, Others.

Examples:

Email: *My card was charged extra this month.*

Category: Billing

Email: *The app crashes whenever I open it.*

Category: Technical Support

Email: *I really like the new app design.*

Category: Feedback

SCREENSHOT OF GENERATED CODE:

```

57  '''Classify the customer email into one of these categories: Billing, Technical Support, Feedback, Others.
58  Examples: Email: My card was charged extra this month. Category: Billing
59  Email: The app crashes whenever I open it. Category: Technical Support
60  Email: I really like the new app design. Category: Feedback'''
61
62  def classify_email(email):
63      billing_keywords = ["charged", "subscription", "invoice", "payment", "billing"]
64      technical_support_keywords = ["crashes", "log in", "error", "bug", "technical issue"]
65      feedback_keywords = ["like", "dislike", "suggestion", "feedback", "improvement"]
66
67      email_lower = email.lower()
68
69      if any(keyword in email_lower for keyword in billing_keywords):
70          return "Billing"
71      elif any(keyword in email_lower for keyword in technical_support_keywords):
72          return "Technical Support"
73      elif any(keyword in email_lower for keyword in feedback_keywords):
74          return "Feedback"
75      else:
76          return "Others"
77
78  # Example usage
79  email = "I really like the new dashboard design. Its much easier to use."
80  category = classify_email(email)
81  print(category) # Output: Feedback

```

The screenshot shows a code editor interface with a dark theme. The left sidebar lists files in the 'AIAc' folder, including 'Lab-01.pages', 'Lab-01.pdf', 'Lab-02.docx', 'Lab-02.pages', 'Lab-02.pdf', 'Lab-03.pages', 'Lab-03.pdf', 'Lab-3.1.pdf', 'Lab-4.1.pages', 'Lab-4.1.py', 'Lab-4.3.pages', 'Lab-4.3.pdf', and 'Lab-4.3.py'. The main area displays the Python code for 'lab-4.1.py'. Below the code editor is a terminal window showing the execution of the script and its output: 'Billing', 'Technical Support', 'Feedback'. The bottom status bar indicates the file is in Python mode, with line 79, column 36, spaces: 4, UTF-8, LF, and Python version 3.13.5.

INPUT & OUTPUT:

Input Email	Output
“I really like the new dashboard design. Its much easier to use.”	Feedback

EXPLANATION:

The function converts the email text to lowercase and checks for predefined keywords related to billing, technical support, and feedback. Since the email contains the word “like”, which is a feedback keyword, it is classified as Feedback.

5. Comparison & Effectiveness

Prompting Technique	Accuracy	Clarity	Reliability	Remarks
Zero-Shot Prompting	Medium	Depends on email clarity	Low	May misclassify vague or mixed emails
One-Shot Prompting	High	Clear	Medium	Single example improves understanding
Few-Shot Prompting	Very High	Very clear	High	Multiple examples give best classification

Problem Statement 2: Intent Classification for Chatbot Queries

A company wants to deploy a chatbot to handle customer queries. Each query must be classified into one of the following intents:

Account Issue, Order Status, Product Inquiry, or General Question using prompt engineering techniques.

Tasks to be Completed

1. Prepare Sample Data Create 6 short chatbot user queries, each mapped to one of the four intents.

2. Zero-shot Prompting

Design a prompt that asks the LLM to classify a user query into the given intent categories without examples.

3. One-shot Prompting

Provide one labeled query in the prompt before classifying a new query.

4. Few-shot Prompting

Include 3–5 labeled intent examples to guide the LLM before classifying a new query.

5. Evaluation

Apply all three techniques to the same set of test queries and document differences in performance.

1. Sample Chatbot User Queries

Query No.	User Query	Intent
1	“I can’t log into my account even after resetting my password.”	Account Issue
2	“Where is my order? It was supposed to arrive yesterday.”	Order Status
3	“Does this phone support 5G and wireless charging?”	Product Inquiry
4	“What are your customer support working hours?”	General Question
5	“My account was locked after multiple failed login attempts.”	Account Issue
6	“Can I track my order using my mobile number?”	Order Status

2. Zero-Shot Prompt

Classify the following user query into one of these intents: Account Issue, Order Status, Product Inquiry, General Question. Return only the intent name.

INPUT & OUTPUT:

Input Query	Output	Reason
I can’t log into my account even after resetting my password.	Account Issue	Contains keywords “log into my account” and “password”.

SCREENSHOT OF GENERATED CODE:

The screenshot shows a code editor interface with the following details:

- File Explorer:** Shows files in the "AIAC" folder, including Lab-01.pdf, Lab-02.docx, Lab-02.pages, Lab-03.pdf, Lab-31.pdf, Lab-31.py, Lab-41.pages, Lab-41.py, Lab-43.pages, Lab-43.pdf, and lab-43.py.code-workspace.
- Code Editor:** The main pane displays Python code for intent classification. The code defines a function `classify_query(query)` that iterates through a list of keywords for account issues, order status, and product inquiries. If a keyword is found, it returns the corresponding intent category (Account Issue, Order Status, Product Inquiry, or General Question). If no keywords match, it returns "General Question". An example usage is shown at the bottom.
- Terminal:** The terminal shows the command `/usr/local/bin/python3 /Users/navedahmedshaik/Documents/3-2/AIAC/lab-4.1.py` being run twice, with the output "Account Issue" and "Order Status" respectively.
- Bottom Status Bar:** Shows the line number (Ln 120), column (Col 39), spaces (Spaces: 4), encoding (UTF-8), line separator (LF), file type (Python), and version (3.13.5).
- Right Sidebar:** Features a "Build with Agent" section with a message: "AI responses may be inaccurate. Generate Agent Instructions to onboard AI onto your codebase." It also includes a "Describe what to build next" input field and dropdowns for "Agent" and "Auto".

```

88     #problem statement-2
89     '''| Query No. | User Query
90     | 1   | "I can't log into my account even after resetting my password." | Intent
91     | 2   | "Where is my order? It was supposed to arrive yesterday." | Account Issue
92     | 3   | "Where is my order?" | Order Status
93     | 4   | "Does this phone support 5G and wireless charging?" | Product Inquiry
94     | 5   | "What are your customer support working hours?" | General Question
95     | 6   | "My account was locked after multiple failed login attempts." | Account Issue
96     | 7   | "Can I track my order using my mobile number?" | Order Status
97     ...
98
99     '''Classify the following user query into one of these intents:
100    Account Issue, Order Status, Product Inquiry, General Question.
101    Return only the intent name.'''
102    def classify_query(query):
103        account_issue_keywords = ["log into my account", "password", "account locked", "failed login"]
104        order_status_keywords = ["where is my order", "track my order", "order status", "arrive"]
105        product_inquiry_keywords = ["support 5g", "wireless charging", "product inquiry", "features"]
106
107        query_lower = query.lower()
108
109        if any(keyword in query_lower for keyword in account_issue_keywords):
110            return "Account Issue"
111        elif any(keyword in query_lower for keyword in order_status_keywords):
112            return "Order Status"
113        elif any(keyword in query_lower for keyword in product_inquiry_keywords):
114            return "Product Inquiry"
115        else:
116            return "General Question"
117
118    # Example usage
119    query = "I can't log into my account even after resetting my password."
120    intent = classify_query(query)
121    print(intent) # Output: Account Issue

```

EXPLANATION

The function converts the query to lowercase and checks it against predefined keyword lists for different categories. If any keyword related to account issues, order status, or product inquiries is found, it returns the corresponding category. If no keywords match, it classifies the query as a General Question

3. One-Shot Prompt

Classify the following user query into one of these intents:

Account Issue, Order Status, Product Inquiry, General Question.

Example:

User Query: *I forgot my password and cannot access my account.*

Intent: Account Issue

INPUT & OUTPUT:

Input Query	Output	Reason
Where is my order? It was supposed to arrive yesterday.	Order Status	Contains keywords "where is my order" and "arrive".

SCREENSHOT OF GENERATED CODE:

The screenshot shows a dark-themed code editor interface. On the left is the Explorer sidebar with files like 'Lab-01.pages', 'Lab-01.pdf', 'Lab-02.docx', etc. The main area shows a file named 'lab-4.1.py' with the following code:

```
123  '''Classify the following user query into one of these intents: Account Issue, Order Status, Product Inquiry
124  Example: User Query: I forgot my password and cannot access my account. Intent: Account Issue'''
125  def classify_query(query):
126      account_issue_keywords = ["log into my account", "password", "account locked", "failed login"]
127      order_status_keywords = ["Where is my order", "track my order", "order status", "arrive"]
128      product_inquiry_keywords = ["support 5g", "wireless charging", "product inquiry", "features"]
129
130      query_lower = query.lower()
131
132      if any(keyword in query_lower for keyword in account_issue_keywords):
133          return "Account Issue"
134      elif any(keyword in query_lower for keyword in order_status_keywords):
135          return "Order Status"
136      elif any(keyword in query_lower for keyword in product_inquiry_keywords):
137          return "Product Inquiry"
138      else:
139          return "General Question"
140
141  # Example usage
142  query = "Where is my order? It was supposed to arrive yesterday."
143  intent = classify_query(query)
144  print(intent) # Output: Order Status
```

Below the code editor is a terminal window showing command-line interactions:

```
(base) navedahmedshaik@Naveds-MacBook-Air AIAC % /usr/local/bin/python3 /Users/navedahmedshaik/Documents/3-2/AIAC/lab-4.1.py
Billing
Technical Support
Feedback
(base) navedahmedshaik@Naveds-MacBook-Air AIAC % /usr/local/bin/python3 /Users/navedahmedshaik/Documents/3-2/AIAC/lab-4.1.py
Billing
Technical Support
Feedback
Account Issue
(base) navedahmedshaik@Naveds-MacBook-Air AIAC % /usr/local/bin/python3 /Users/navedahmedshaik/Documents/3-2/AIAC/lab-4.1.py
Billing
Technical Support
Feedback
Account Issue
Order Status
(base) navedahmedshaik@Naveds-MacBook-Air AIAC %
```

On the right side of the interface, there is a 'CHAT' bar with a message bubble icon and a 'Build with Agent' section containing the text: 'Build with Agent', 'AI responses may be inaccurate.', 'Generate Agent Instructions to onboard', and 'AI onto your codebase.'.

EXPLANATION:

The function converts the query to lowercase and checks it against predefined keyword lists. Since the query contains order-related keywords, it is classified as Order Status.

4. Few-Shot Prompt

Classify the following user query into one of these intents:

Account Issue, Order Status, Product Inquiry, General Question.

Examples:

User Query: *My account got locked after too many login attempts.*

Intent: Account Issue

User Query: *When will my order be delivered?*

Intent: Order Status

User Query: *Does this laptop come with a warranty?*

Intent: Product Inquiry

User Query: *What payment methods do you accept?*

Intent: General Question

SCREENSHOT OF GENERATED CODE:

The screenshot shows a code editor interface with a dark theme. On the left is the Explorer sidebar containing files like 'Lab-01.pages', 'Lab-01.pdf', 'Lab-02.docx', 'Lab-02.pages', 'Lab-02.pdf', 'Lab-31.pages', 'Lab-31.pdf', 'Lab-31.py', 'Lab-41.pages', 'Lab-41.py', 'Lab-43.pages', 'Lab-43.pdf', 'Lab-43.py', and 'Lab-43.py.code-workspace'. The main editor window displays a Python script named 'lab-4.1.py' with the following code:

```

150 Account Issue, Order Status, Product Inquiry, General Question.
151
152 Examples:
153 User Query: My account got locked after too many login attempts.
154 Intent: Account Issue
155
156 User Query: When will my order be delivered?
157 Intent: Order Status
158
159 User Query: Does this laptop come with a warranty?
160 Intent: Product Inquiry
161
162 User Query: What payment methods do you accept?
163 Intent: General Question
164
165 def classify_query(query):
166     account_issue_keywords = ["log into my account", "password", "account locked", "failed login"]
167     order_status_keywords = ["where is my order", "track my order", "order status", "arrive"]
168     product_inquiry_keywords = ["support 5g", "wireless charging", "product inquiry", "features"]
169
170     query_lower = query.lower()
171
172     if any(keyword in query_lower for keyword in account_issue_keywords):
173         return "Account Issue"
174     elif any(keyword in query_lower for keyword in order_status_keywords):
175         return "Order Status"
176     elif any(keyword in query_lower for keyword in product_inquiry_keywords):
177         return "Product Inquiry"
178     else:
179         return "General Question"
180
181 # Example usage
182 query = "Does this phone support 5G and wireless charging?"
183 intent = classify_query(query)
184 print(intent) # Output: Product Inquiry

```

The bottom right corner of the editor shows the status bar with 'Ln 182, Col 41 Spaces: 4 UTF-8 LF () Python 3.13.5'.

INPUT & OUTPUT:

Input Query	Output	Reason
Where is my order? It was supposed to arrive yesterday.	Order Status	Contains keywords “where is my order” and “arrive”.

EXPLANATION:

The function converts the query to lowercase and checks it against predefined keyword lists. Since the query contains order-related keywords, it is classified as Order Status.

5. Evaluation of Prompting Techniques

Technique	Accuracy	Intent Understanding	Consistency	Remarks
Zero-Shot	Medium	Basic	Low	Works only for very clear queries
One-Shot	High	Better	Medium	Example improves intent recognition
Few-Shot	Very High	Strong	High	Best performance for ambiguous queries