

Assignment-4.1

Name: Arjun Manoj

H. No:2303A52134

Batch:44

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.

Dataset (used for testing)

- Billing: “My card was charged twice for the same invoice.”
- Technical Support: “The application freezes after the latest update.”
- Feedback: “Your customer service response time has improved a lot.”
- Others: “Do you have offices in Hyderabad?”

2. Write a zero-shot prompt to classify a given email into one of the categories without providing any examples.

Prompt:

Determine the most suitable category for the email below.

Possible categories:

Billing

Technical Support

Feedback

Others

Email content:

"I am unable to connect to the server and keep getting an error message."

Respond with only the category name.

Code:

```

# =====
# TASK 1: Sample Emails
# =====

sample_emails = {
    "Billing": "I was charged twice for my subscription this month.",
    "Technical Support": "The app crashes whenever I try to log in.",
    "Feedback": "I love the new interface, it looks great!",
    "Others": "What are your office working hours?"
}

print("Sample Emails:")
for category, email in sample_emails.items():
    print(f"{category}: {email}")

```

```

Sample Emails:
Billing: I was charged twice for my subscription this month.
Technical Support: The app crashes whenever I try to log in.
Feedback: I love the new interface, it looks great!
Others: What are your office working hours?

```

Observation:

- The model depends entirely on task understanding.
- Suitable for simple and clearly worded emails.
- Less reliable for emails with mixed intent.

3. Write a one-shot prompt by including one labeled email example and ask the model to classify a new email.

Prompt:

An email is classified into predefined categories.

Example:

Email: "I was billed extra for my subscription."

Category: Billing

Now analyze the email below and assign the correct category.

Email:

"I cannot log into my account after changing my password."

Choose from:

Billing

Technical Support

Feedback

Others

Answer with only one category.

Code:

```
one_shot_example = {  
    "email": "I was billed extra for my subscription.",  
    "category": "Billing"  
}  
  
test_email = "I cannot log into my account after changing my password."  
  
print("One-Shot Example:")  
print(one_shot_example)  
print("\nTest Email:")  
print(test_email)  
  
One-Shot Example:  
{'email': 'I was billed extra for my subscription.', 'category': 'Billing'}  
  
Test Email:  
I cannot log into my account after changing my password.
```

Observation:

- One example guides the model's reasoning.
- More accurate than zero-shot prompting.
- Still limited due to minimal context.

4. Write a few-shot prompt by including two or three labeled email examples and ask the model to classify a new email.

Prompt:

Emails are grouped into categories based on their purpose.

Examples:

Email: "My payment was deducted twice."

Category: Billing

Email: "The app crashes when I open it."

Category: Technical Support

Email: "The new design looks very clean."

Category: Feedback

Now classify the following email.

Email:

"Are there any offers available for students?"

Select only one category:

Billing

Technical Support

Feedback

Others

Code:

```

few_shot_samples = [
    ("My payment was deducted twice.", "Billing"),
    ("The app crashes when I open it.", "Technical Support"),
    ("The new design looks very clean.", "Feedback")
]

new_email = "Are there any offers available for students?"

print("Few-Shot Samples:")
for email, category in few_shot_samples:
    print(email, "->", category)

print("\nEmail to Classify:")
print(new_email)

```

```

Few-Shot Samples:
My payment was deducted twice. -> Billing
The app crashes when I open it. -> Technical Support
The new design looks very clean. -> Feedback

```

```

Email to Classify:
Are there any offers available for students?

```

Observation:

Multiple examples improve pattern recognition.

Produces the most consistent results.

Best approach when model training is not possible.

5. Compare the outputs obtained using zero-shot, one-shot, and few-shot prompting techniques and briefly comment on their effectiveness

Technique	Effectiveness	Reason
Zero-Shot	Moderate	No contextual learning
One-Shot	Good	Single example helps

Few-Shot	High	Learns from patterns
----------	------	----------------------

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:

1. Prepare Sample Data

Create 6 short chatbot user queries, each mapped to one of the four intents

Sample Chatbot Queries

Query	Intent
“I am unable to log into my account.”	Account Issue
“My password reset link is not working.”	Account Issue
“Where is my order right now?”	Order Status
“Has my order been shipped?”	Order Status
“Does this laptop support fast charging?”	Product Inquiry
“What are your customer support hours?”	General Question

2. Zero-shot Prompting

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

Prompt:

Classify the following chatbot query into one of these intents:

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

User Query:

"I have not received my order yet."

Respond with only the intent name.

Code:

```
zero_shot_task = "Intent classification without examples"

query = "I have not received my order yet."

intents = [
    "Account Issue",
    "Order Status",
    "Product Inquiry",
    "General Question"
]

print("Task:", zero_shot_task)
print("Query:", query)
print("Possible Intents:", intents)

Task: Intent classification without examples
Query: I have not received my order yet.
Possible Intents: ['Account Issue', 'Order Status', 'Product Inquiry', 'General Question']
```

Observation:

The model depends entirely on instruction clarity.

Works well for straightforward queries.

Can misclassify ambiguous or overlapping intents.

3. One-shot Prompting

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

Prompt:

Identify the intent of a chatbot query.

Example:

Query: "I cannot access my account after changing my password."

Intent: Account Issue

Now classify the following query.

Query:

"Can you tell me when my order will arrive?"

Choose one intent:

Account Issue, Order Status, Product Inquiry, General Question

Answer with only the intent name.

Code:

```
one_shot_example = {  
    "query": "I cannot access my account after changing my password.",  
    "intent": "Account Issue"  
}  
  
test_query = "Can you tell me when my order will arrive?"  
  
print("One-Shot Example:", one_shot_example)  
print("Query to Classify:", test_query)  
  
One-Shot Example: {'query': 'I cannot access my account after changing my password.', 'intent': 'Account Issue'}  
Query to Classify: Can you tell me when my order will arrive?
```

Observation:

- Single example helps the model understand intent mapping.
- More accurate than zero-shot.
- Still limited due to only one reference.

4. Few-shot Prompting

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

Prompt:

Determine the intent of chatbot queries.

Examples:

Query: "My account is locked."

Intent: Account Issue

Query: "Is my order delivered?"

Intent: Order Status

Query: "Does this phone support 5G?"

Intent: Product Inquiry

Query: "What payment methods do you accept?"

Intent: General Question

Now classify the following query.

Query:

"My order has been delayed for two days."

Choose one intent:

Account Issue, Order Status, Product Inquiry, General Question

Code:

```
few_shot_examples = [
    ("My account is locked.", "Account Issue"),
    ("Is my order delivered?", "Order Status"),
    ("Does this phone support 5G?", "Product Inquiry"),
    ("What payment methods do you accept?", "General Question")
]

new_query = "My order has been delayed for two days."

print("Few-Shot Examples:")
for q, i in few_shot_examples:
    print(q, "->", i)

print("\nQuery to Classify:", new_query)

Few-Shot Examples:
My account is locked. -> Account Issue
Is my order delivered? -> Order Status
Does this phone support 5G? -> Product Inquiry
What payment methods do you accept? -> General Question

Query to Classify: My order has been delayed for two days.
```

Observation:

- Multiple examples provide strong contextual understanding.
- Most consistent and accurate results.
- Best approach when model training is not possible.

5. Evaluation

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

Technique	Predicted Intent	Performance
Zero-Shot	Order Status	Moderate
One-Shot	Order Status	Good
Few-Shot	Order Status	High