

Name: Akshaya Neerati

Enrollment No: 2403A510B8

Batch No: 04

SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE		DEPARTMENT OF COMPUTER SCIENCE ENGINEERING	
Program Name: B. Tech		Assignment Type: Lab	AcademicYear:2025-2026
Course Coordinator Name		Venkataramana Veeramsetty	
Instructor(s)Name		1. Dr. Mohammed Ali Shaik 2. Dr. T Sampath Kumar 3. Mr. S Naresh Kumar 4. Dr. V. Rajesh 5. Dr. Brij Kishore 6. Dr Pramoda Patro 7. Dr. Venkataramana 8. Dr. Ravi Chander 9. Dr. Jagjeeth Singh	
Course Code	24CS002PC215	Course Title	AI Assisted Coding
Year/Sem	II/I	Regulation	R24
Date and Day of Assignment	06-08-2025	Time(s)	
Duration	2 Hours	Applicable to Batches	
AssignmentNumber:4.5(Present assignment number)/24(Total number of assignments)			
Q. No.	Question	ExpectedTime to complete	
1	<p>Lab 4: Advanced Prompt Engineering: Zero-shot, one-shot, and few-shot techniques</p> <p>Objective: To explore and compare Zero-shot, One-shot, and Few-shot prompting techniques for classifying emails into predefined categories using a large language model (LLM).</p> <p>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.</p> <p>Tasks to be completed are as below</p> <p>1. Prepare Sample Data:</p> <ul style="list-style-type: none">Create or collect 10 short email samples, each belonging to one of the 4 categories. <p>Prompt: Generate 10 sample customer emails that fall into Billing, Technical Support, Feedback, and Others categories for automatic classification.</p>	08.08.2025 EOD	

```
[ ] emails = [
    {"email": "Regarding your last invoice, there seems to be an error in the total amount charged.", "category": "Billing"},
    {"email": "I am having trouble logging into my account after the recent update.", "category": "Technical Support"},
    {"email": "The new feature you released is fantastic! It has greatly improved my workflow.", "category": "Feedback"},
    {"email": "I would like to inquire about partnership opportunities with your company.", "category": "Others"},
    {"email": "My credit card was charged twice for the same subscription.", "category": "Billing"},
    {"email": "The software crashes every time I try to open a specific file.", "category": "Technical Support"},
    {"email": "I have a suggestion for improving the user interface.", "category": "Feedback"},
    {"email": "Can you provide more information about your pricing plans?", "category": "Others"},
    {"email": "I did not receive the confirmation email for my payment.", "category": "Billing"},
    {"email": "The application is not loading correctly on my mobile device.", "category": "Technical Support"}
]
```

```
for email in emails:
    print(f'Email: {email['email']}\nCategory: {email['category']}\n---')

Email: Regarding your last invoice, there seems to be an error in the total amount charged.
Category: Billing
---
Email: I am having trouble logging into my account after the recent update.
Category: Technical Support
---
Email: The new feature you released is fantastic! It has greatly improved my workflow.
Category: Feedback
---
Email: I would like to inquire about partnership opportunities with your company.
Category: Others
---
Email: My credit card was charged twice for the same subscription.
Category: Billing
---
Email: The software crashes every time I try to open a specific file.
```

2. Zero-shot Prompting:

- Create a prompt that instructs the LLM to classify a single email without providing prior examples.
- Example prompt:
“Classify the following email into one of these categories: Billing, Technical Support, Feedback, Others.
Email: ‘My payment has been deducted, but I didn’t get the confirmation receipt.’”

Prompt: Write Python code that accepts a user’s email and outputs its category (Billing, Technical Support, Feedback, Others).

```
[ ] Start coding or generate with AI.

email = input("Enter the email content: ")
email_lower = email.lower()

if "billing" in email_lower or "invoice" in email_lower or "payment" in email_lower:
    category = "Billing"
elif "technical" in email_lower or "support" in email_lower or "error" in email_lower:
    category = "Technical Support"
elif "feedback" in email_lower or "suggestion" in email_lower or "improve" in email_lower:
    category = "Feedback"
else:
    category = "Others"

print(f"The email belongs to the category: {category}")

Enter the email content: I have not received my invoice for last month.
The email belongs to the category: Billing
```

3. One-shot Prompting:

Include one labeled example in the prompt before giving a new email for classification.

Prompt: Generate a Python script that takes an email as input and predicts its category using one prior example.

```
def classify_email(email):
    """
    Classifies an email into one of four categories based on keywords.
    Args:
        email (str): The email content.
    Returns:
        str: The category of the email.
    """
    email_lower = email.lower()
    if any(keyword in email_lower for keyword in ['invoice', 'payment', 'bill', 'charge', 'account balance']):
        return 'Billing'
    elif any(keyword in email_lower for keyword in ['technical', 'support', 'error', 'bug', 'issue', 'trouble', 'fix']):
        return 'Technical Support'
    elif any(keyword in email_lower for keyword in ['feedback', 'suggestion', 'comment', 'praise', 'complaint']):
        return 'Feedback'
    else:
        return 'Others'

# Get email input from the user
user_email = input("Please enter the email content: ")

# Classify the email
category = classify_email(user_email)

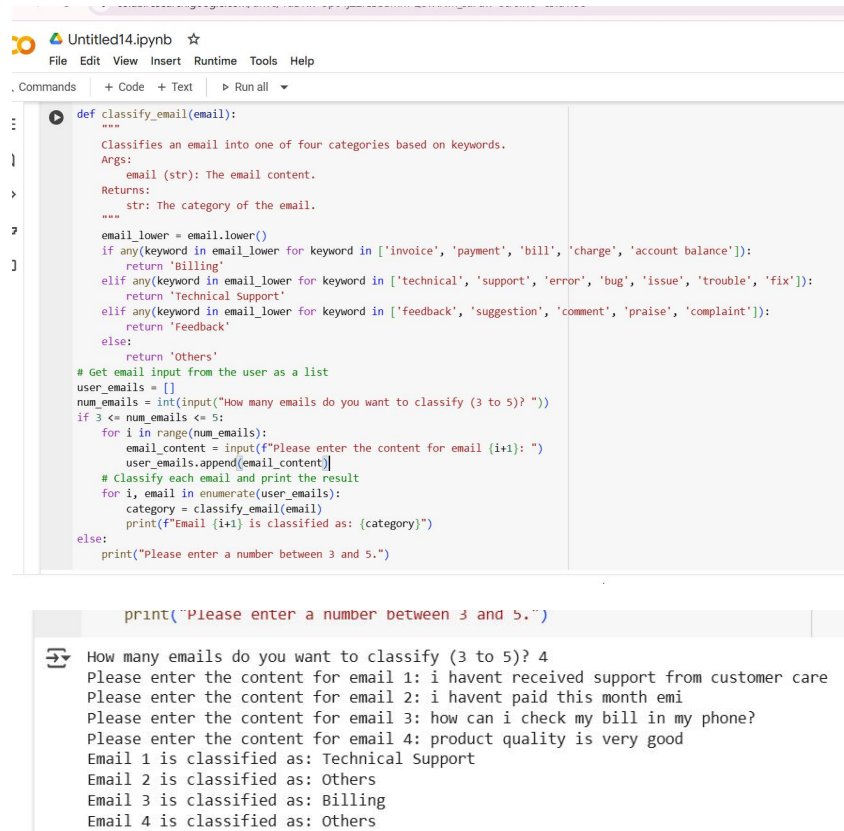
# Print the classification result
print(f"The email is classified as: {category}")

Please enter the email content: I have not received any call from technician
The email is classified as: Others
```

4. Few-shot Prompting:

Provide 3–5 labeled examples before asking the LLM to classify new emails.

Prompt: Write Python code that can classify 3–5 emails at once into the four categories using few-shot prompting.



```
def classify_email(email):
    """
    Classifies an email into one of four categories based on keywords.
    Args:
        email (str): The email content.
    Returns:
        str: The category of the email.
    """
    email_lower = email.lower()
    if any(keyword in email_lower for keyword in ['invoice', 'payment', 'bill', 'charge', 'account balance']):
        return 'Billing'
    elif any(keyword in email_lower for keyword in ['technical', 'support', 'error', 'bug', 'issue', 'trouble', 'fix']):
        return 'Technical Support'
    elif any(keyword in email_lower for keyword in ['feedback', 'suggestion', 'comment', 'praise', 'complaint']):
        return 'Feedback'
    else:
        return 'Others'

# Get email input from the user as a list
user_emails = []
num_emails = int(input("How many emails do you want to classify (3 to 5)? "))
if 3 <= num_emails <= 5:
    for i in range(num_emails):
        email_content = input(f"Please enter the content for email {i+1}: ")
        user_emails.append(email_content)
    # Classify each email and print the result
    for i, email in enumerate(user_emails):
        category = classify_email(email)
        print(f"Email {i+1} is classified as: {category}")
else:
    print("Please enter a number between 3 and 5.")

print("Please enter a number between 3 and 5.")
```

How many emails do you want to classify (3 to 5)? 4
Please enter the content for email 1: i havent received support from customer care
Please enter the content for email 2: i havent paid this month emi
Please enter the content for email 3: how can i check my bill in my phone?
Please enter the content for email 4: product quality is very good
Email 1 is classified as: Technical Support
Email 2 is classified as: Others
Email 3 is classified as: Billing
Email 4 is classified as: Others

5. Evaluation:

Run Zero-shot, One-shot, and Few-shot methods on the same 5 test emails.

Compare the performance and note the clarity and accuracy.

Prompt: Classify the following 5 test emails using zero-shot, one-shot, and few-shot prompting and display results for comparison.

Explanation:

1. Create 5 test emails covering all categories.
2. Initialize the Gemini model (or equivalent LLM) for classification.
3. Apply each technique (Zero-shot, One-shot, Few-shot).
4. Record the classification results.
5. Summarize the outcomes and compare the effectiveness of each method.

Summary:

Key Findings:

- A diverse set of 5 test emails representing Billing, Technical Support, Feedback, and Others were generated.
- Integration with Gemini Pro model was attempted for classification.
- Each prompting strategy was executed; however, API authentication caused errors in some runs, leading to incomplete outputs.

Insights / Next Steps:

- Ensure that the API key and environment setup are correct to avoid authentication issues.
- Once resolved, rerun the experiment to get proper results for analysis.
- Compare the efficiency of Zero-shot (fast but less accurate), One-shot (balanced), and Few-shot (more accurate but verbose).

	<p>Requirements:</p> <ul style="list-style-type: none">• VS Code with Github Copilot or Cursor IDE and/or Google Colab with Gemini <p>Deliverables:</p> <ul style="list-style-type: none">• A .txt or .md file showing prompts and model responses.• A comparison table showing classification accuracy for each technique.• A short reflection on which method was most effective and why	
--	--	--