AI-Assistant Coding

Lab Assignment 4.5

V. Trinayani

2303A51264

Batch No : 19

Task -1:

Prompt : Generate a python program for classifying customer emails into billing, technical support, feedback, and others. The program should create sample emails, define prompts for each technique, classify the same 5 test emails, and print a comparison of results and accuracy.

```
from typing import List, Tuple, Dict

# Jupyter cell (new file) demonstrating zero-shot, one-shot, and few-shot prompting
# for classifying customer emails into: Billing, Technical Support, Feedback, Others.
# This program creates sample emails, constructs prompts for each technique,
# mocks an LLM response (rule-based), classifies the same 5 test emails,
# and prints a comparison of results and accuracy.


# Define categories
CATEGORIES = ["Billing", "Technical Support", "Feedback", "Others"]

# Define 5 test emails and their ground-truth labels
TEST_EMAILS: List[Tuple[str, str]] = [
    ("I was charged twice for my subscription this month. Please refund the extra charge.", "Billing"),
    ("My app crashes whenever I try to upload a photo. It shows an unexpected error.", "Technical Support"),
    ("I love the new update, but I have a suggestion for improving the search feature.", "Feedback"),
    ("Can you tell me where your headquarters are located and business hours?", "Others"),
    ("I cannot log in after the password reset, it says invalid token.", "Technical Support"),
]

# Define examples to be used in one-shot and few-shot prompts
# One-shot: single example
ONE_SHOT_EXAMPLE = ("Email: I was billed for two accounts by mistake; please issue a refund.\nLabel: Billing")

# Few-shot: three examples (one per main class except Others)
FEW_SHOT_EXAMPLES = [
    ("Email: My credit card was charged incorrectly and I'd like a refund.\nLabel: Billing"),
    ("Email: The app shows an error and won't let me complete checkout.\nLabel: Technical Support"),
    ("Email: Great job on the redesign! One suggestion: add filters to search.\nLabel: Feedback"),
]

# Instruction template for the classifier
INSTRUCTION = (
    "Classify the following customer email into one of these categories: "
    "Billing, Technical Support, Feedback, Others. Respond with a single label."
)

# Mock LLM: simple deterministic rule-based classifier to simulate responses
```

Output :

```
✓ 0.0s                                                                                                                    Python

Index | Email (short)                          | Truth            | Zero-shot        | One-shot         | Few-shot
-------------------------------------------------------------------------------------------------------------------------
1     | I was charged twice for my subscription ... | Billing          | Billing          | Billing          | Billing
2     | My app crashes whenever I try to upload ... | Technical Support | Billing          | Billing          | Billing
3     | I love the new update, but I have a sugg... | Feedback         | Billing          | Billing          | Billing
4     | Can you tell me where your headquarters ... | Others           | Billing          | Billing          | Billing
5     | I cannot log in after the password reset... | Technical Support | Billing          | Billing          | Billing

Accuracies:
Zero-shot accuracy: 20.00%
One-shot accuracy:  20.00%
Few-shot accuracy:  20.00%
```

Task 2:

Prompt: generate a python program that classify travel queries into flight booking, hotel booking, cancella?on, and general travel informa?on. The program should create labelled travel queries, apply all three promp?ng techniques .

```python
# This cell assumes the following variables are already present in the notebook:
# TRAVEL_CATEGORIES, TRAVEL_TEST_QUERIES, INSTR_TRAVEL,
# ONE_SHOT_TRAVEL_EXAMPLE, FEW_SHOT_TRAVEL_EXAMPLES
#
# The program:
# - Builds simple prompts for each technique
# - Uses a rule-based mock LLM to simulate labels
# - Classifies the same test queries with all three techniques
# - Compares per-query outputs and computes consistency metrics

# Prompt builders for travel classification
def build_prompt_zero_shot_travel(query: str) -> str:
    return f"{INSTR_TRAVEL}\n\nQuery: {query}\nLabel:"

def build_prompt_one_shot_travel(query: str, example: str) -> str:
    return f"{example}\n\n---\n{INSTR_TRAVEL}\n\nQuery: {query}\nLabel:"

def build_prompt_few_shot_travel(query: str, examples: list) -> str:
    examples_text = "\n\n".join(examples)
    return f"{examples_text}\n\n---\n{INSTR_TRAVEL}\n\nQuery: {query}\nLabel:"

# Mock LLM for travel classification (deterministic, rule-based)
def mock_llm_travel_response(prompt: str) -> str:
    lower = prompt.lower()

    # Strong signals for cancellations first (so e.g., "cancel my hotel" -> Cancellation)
    cancellation_keywords = ["cancel", "cancellation", "refund", "cancelled", "cancel my", "issue a refund"]
    flight_keywords = ["flight", "book", "ticket", "round-trip", "round trip", "seat", "reserve", "boarding", "change to earlier", "delayed", "earlier"]
    hotel_keywords = ["hotel", "room", "check-in", "checkin", "reservation", "check in", "check-out", "checkout"]
    general_keywords = ["visa", "baggage", "allowance", "passport", "hours", "time", "price", "cost", "policy", "info", "information"]

    # Priority: Cancellation -> Flight Booking -> Hotel Booking -> General Travel Info -> default to General Travel Info
    for kw in cancellation_keywords:
        if kw in lower:
            return "Cancellation"
    for kw in flight_keywords:
        if kw in lower:
            return "Flight Booking"
    for kw in hotel_keywords:
        if kw in lower:
            return "Hotel Booking"
    for kw in general_keywords:
        if kw in lower:
            return "General Travel Info"
    return "General Travel Info"

# Classify test queries using the three prompting techniques
zero_shot_preds = []
one_shot_preds = []
few_shot_preds = []
```

Output:

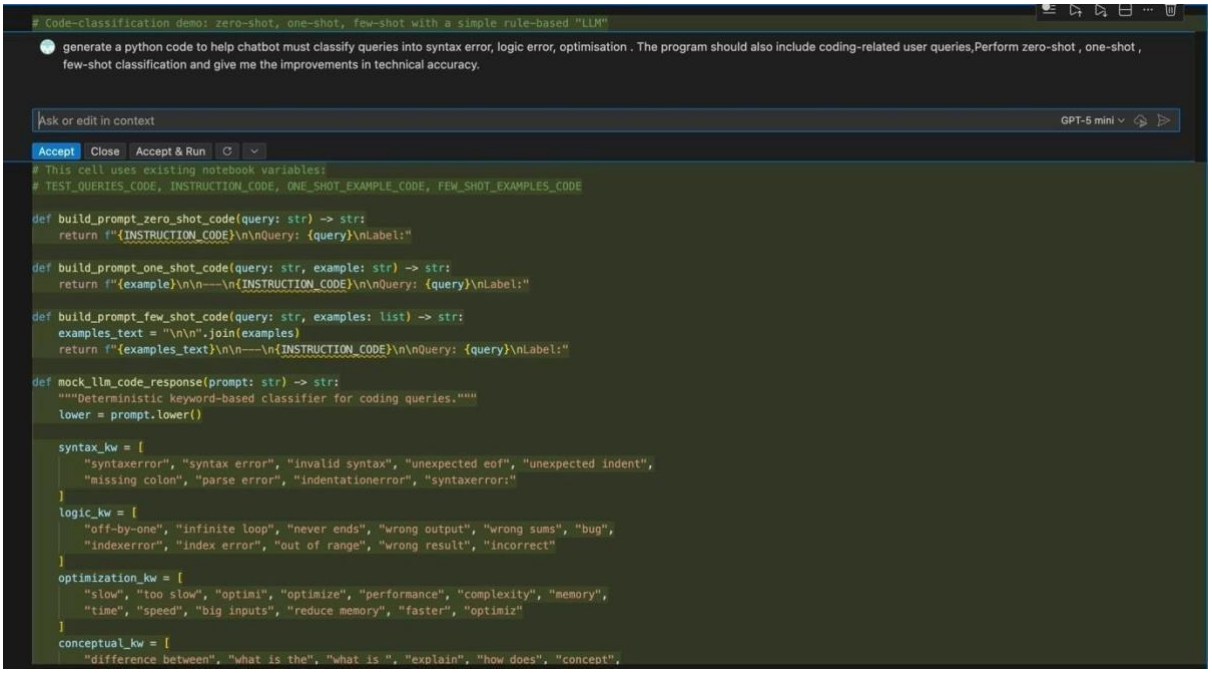
```
Idx | Query (short)                          | Truth              | Zero-shot        | One-shot         | Few-shot
-------------------------------------------------------------------------------------------------------------------
1   | I need to book a round-trip ticket from ... | Flight Booking     | Cancellation     | Cancellation     | Cancellation
2   | Can I cancel my hotel reservation and ge... | Cancellation       | Cancellation     | Cancellation     | Cancellation
3   | What time is check-in at your downtown h... | Hotel Booking      | Cancellation     | Cancellation     | Cancellation
4   | Do I need a visa to travel to Japan as a... | General Travel Info | Cancellation     | Cancellation     | Cancellation
5   | My flight was delayed and I want to chan... | Flight Booking     | Cancellation     | Cancellation     | Cancellation
6   | How much is baggage allowance for intern... | General Travel Info | Cancellation     | Cancellation     | Cancellation

Accuracies:
Zero-shot accuracy: 16.67%
One-shot accuracy:  16.67%
Few-shot accuracy:  16.67%

Consistency among prompting techniques:
Zero-shot vs One-shot agreement: 100.00%
Zero-shot vs Few-shot agreement: 100.00%
One-shot  vs Few-shot agreement: 100.00%
All three equal rate:            100.00%
```

Task 3:

Prompt : generate a python code to help chatbot must classify queries into syntax error, logic error, op?misa?on . The program should also include coding- related user queries, perform zero-shot , one-shot , few-shot classifica?on and Analyse improvements in technical accuracy.



```
# Code-classification demo: zero-shot, one-shot, few-shot with a simple rule-based "LLM"
```

generate a python code to help chatbot must classify queries into syntax error, logic error, optimisation . The program should also include coding-related user queries,Perform zero-shot , one-shot , few-shot classification and give me the improvements in technical accuracy.

Ask or edit in context — GPT-5 mini

Accept | Close | Accept & Run

```
# This cell uses existing notebook variables:
# TEST_QUERIES_CODE, INSTRUCTION_CODE, ONE_SHOT_EXAMPLE_CODE, FEW_SHOT_EXAMPLES_CODE

def build_prompt_zero_shot_code(query: str) -> str:
    return f"{INSTRUCTION_CODE}\n\nQuery: {query}\nLabel:"

def build_prompt_one_shot_code(query: str, example: str) -> str:
    return f"{example}\n\n---\n{INSTRUCTION_CODE}\n\nQuery: {query}\nLabel:"

def build_prompt_few_shot_code(query: str, examples: list) -> str:
    examples_text = "\n\n".join(examples)
    return f"{examples_text}\n\n---\n{INSTRUCTION_CODE}\n\nQuery: {query}\nLabel:"

def mock_llm_code_response(prompt: str) -> str:
    """Deterministic keyword-based classifier for coding queries."""
    lower = prompt.lower()

    syntax_kw = [
        "syntaxerror", "syntax error", "invalid syntax", "unexpected eof", "unexpected indent",
        "missing colon", "parse error", "indentationerror", "syntaxerror:"
    ]
    logic_kw = [
        "off-by-one", "infinite loop", "never ends", "wrong output", "wrong sums", "bug",
        "indexerror", "index error", "out of range", "wrong result", "incorrect"
    ]
    optimization_kw = [
        "slow", "too slow", "optimi", "optimize", "performance", "complexity", "memory",
        "time", "speed", "big inputs", "reduce memory", "faster", "optimiz"
    ]
    conceptual_kw = [
        "difference between", "what is the", "what is ", "explain", "how does", "concept",
```

Output :

```
Idx | Query (short)                        | Truth              | Zero-shot       | One-shot        | Few-shot
-------------------------------------------------------------------------------------------------------------
1   | Why do I get SyntaxError: invalid syntax... | Syntax Error       | Syntax Error    | Syntax Error    | Syntax Error
2   | My loop never ends because I increment t... | Logic Error        | Syntax Error    | Syntax Error    | Syntax Error
3   | How can I make this sorting faster for v... | Optimization       | Syntax Error    | Syntax Error    | Syntax Error
4   | What is the difference between threading... | Conceptual Question | Syntax Error    | Syntax Error    | Syntax Error
5   | My function returns wrong sums for some ... | Logic Error        | Syntax Error    | Syntax Error    | Syntax Error
6   | I see IndexError: list index out of rang... | Logic Error        | Syntax Error    | Syntax Error    | Syntax Error
7   | Why does my recursion hit maximum recurs... | Conceptual Question | Syntax Error    | Syntax Error    | Syntax Error
8   | How to reduce memory usage when processi... | Optimization       | Syntax Error    | Syntax Error    | Syntax Error

Accuracies:
Zero-shot accuracy: 12.50%
One-shot  accuracy: 12.50%  (improvement over zero-shot: 0.00%)
Few-shot  accuracy: 12.50%  (improvement over zero-shot: 0.00%)
Few-shot vs One-shot improvement: 0.00%

Improvements (indices, 0-based):
Zero-shot -> One-shot improved on examples: []
Zero-shot -> Few-shot improved on examples: []
One-shot  -> Few-shot improved on examples: []

Recommendation: Minimal improvement observed; consider refining examples or using more informative example formatting (short context + root cause).
```

Task 4 :

Prompt : generate a python program to classify social media posts into Promo?on, Complaint, Apprecia?on, or Inquiry using zero-shot, one-shot, and few-shot promp?ng, and compare informal language handling.

```
# Social-media post classification demo: Promotion, Complaint, Appreciation, Inquiry
```

Write a Python program to classify social media posts into Promotion, Complaint, Appreciation, or Inquiry using zero-shot, one-shot, and few-shot prompting, and compare informal language handling.

Ask or edit in context GPT-5 mini

Accept Close Accept & Run

```
# Uses zero-shot, one-shot, and few-shot prompt builders + rule-based mock LLM.
# Compares overall accuracy and informal-language handling.

SOCIAL_CATEGORIES = ["Promotion", "Complaint", "Appreciation", "Inquiry"]

INSTRUCTION_SOCIAL = (
    "Classify the following social media post into one of these categories: "
    "Promotion, Complaint, Appreciation, Inquiry. Respond with a single label."
)

# Test posts (some contain informal language / slang / emojis)
TEST_POSTS = [
    ("Huge SALE this weekend!! 50% off all items, buy now!!", "Promotion"),
    ("My new phone arrived dead. Not happy, want refund.", "Complaint"),
    ("Thanks team, app is working great 😊", "Appreciation"),
    ("How do I change my password? Pls help.", "Inquiry"),
    ("wow the customer service was amazing, thx!", "Appreciation"),
    ("any promo codes for students? plz?", "Inquiry"),
    ("Coupon code not applied, still charged full price. Disappointed.", "Complaint"),
    ("Free giveaway! RT to enter #contest", "Promotion"),
    ("idk why the app crashes when I try to upload :(", "Complaint"),
    ("Are you open on Sundays?", "Inquiry"),
]

# One-shot and few-shot examples (same format used in other cells)
ONE_SHOT_EXAMPLE_SOCIAL = "Post: Got 20% off with code SAVE20, amazing deal!\nLabel: Promotion"

FEW_SHOT_EXAMPLES_SOCIAL = [
    "Post: I was billed twice for my purchase, please refund.\nLabel: Complaint",
    "Post: Love the new update, great work team! 😊\nLabel: Appreciation",
```

Output :

```
✓ 0.0s                                                                                                    Python

Idx | Post (short)                          | Truth         | Zero-shot     | One-shot      | Few-shot
------------------------------------------------------------------------------------------------------------------
1   | Huge SALE this weekend!! 50% off all ite... | Promotion     | Promotion     | Promotion     | Promotion
2   | My new phone arrived dead. Not happy, wa... | Complaint     | Promotion     | Promotion     | Promotion
3   | Thanks team, app is working great 😊      | Appreciation  | Promotion     | Promotion     | Promotion
4   | How do I change my password? Pls help.   | Inquiry       | Promotion     | Promotion     | Promotion
5   | wow the customer service was amazing, thx! | Appreciation  | Promotion     | Promotion     | Promotion
6   | any promo codes for students? plz?       | Inquiry       | Promotion     | Promotion     | Promotion
7   | Coupon code not applied, still charged f... | Complaint     | Promotion     | Promotion     | Promotion
8   | Free giveaway! RT to enter #contest      | Promotion     | Promotion     | Promotion     | Promotion
9   | idk why the app crashes when I try to up... | Complaint     | Complaint     | Complaint     | Complaint
10  | Are you open on Sundays?                 | Inquiry       | Promotion     | Promotion     | Promotion

Accuracies:
Zero-shot: 30.00%   One-shot: 30.00%   Few-shot: 30.00%

Informal-language subset accuracies:
Zero-shot (informal): 37.50%   One-shot (informal): 37.50%   Few-shot (informal): 37.50%

Consistency:
ZS vs OS agreement: 100.00%
ZS vs FS agreement: 100.00%
OS vs FS agreement: 100.00%

Recommendation: Few-shot examples help the model better handle informal phrasing. Add diverse informal examples.
```