Assignment– 4.5

Batch-29

2303A51655

Task1:Supposethatyouworkforacompanythatreceiveshundreds of customer emails daily. Management wants to automatically classifyemails 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 promptengineeringtechniqueswithanexistingLLMtohandlethe classification.

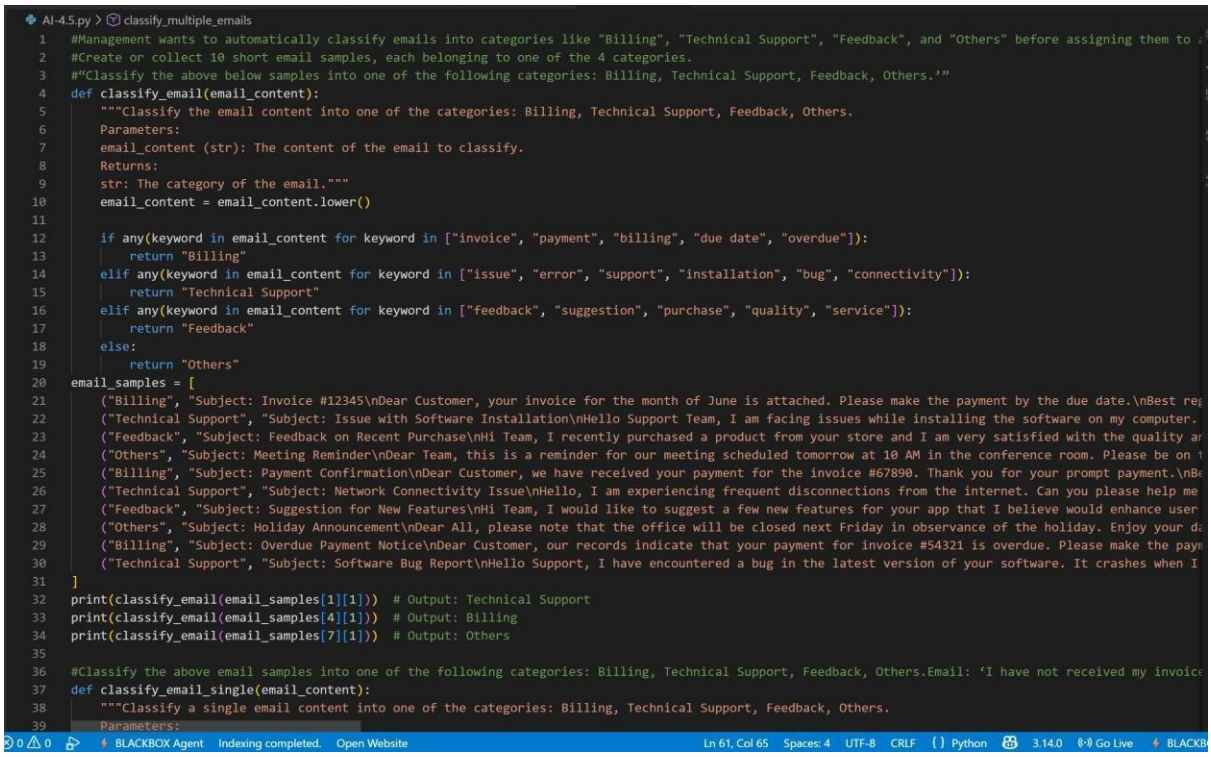
PROMPT:

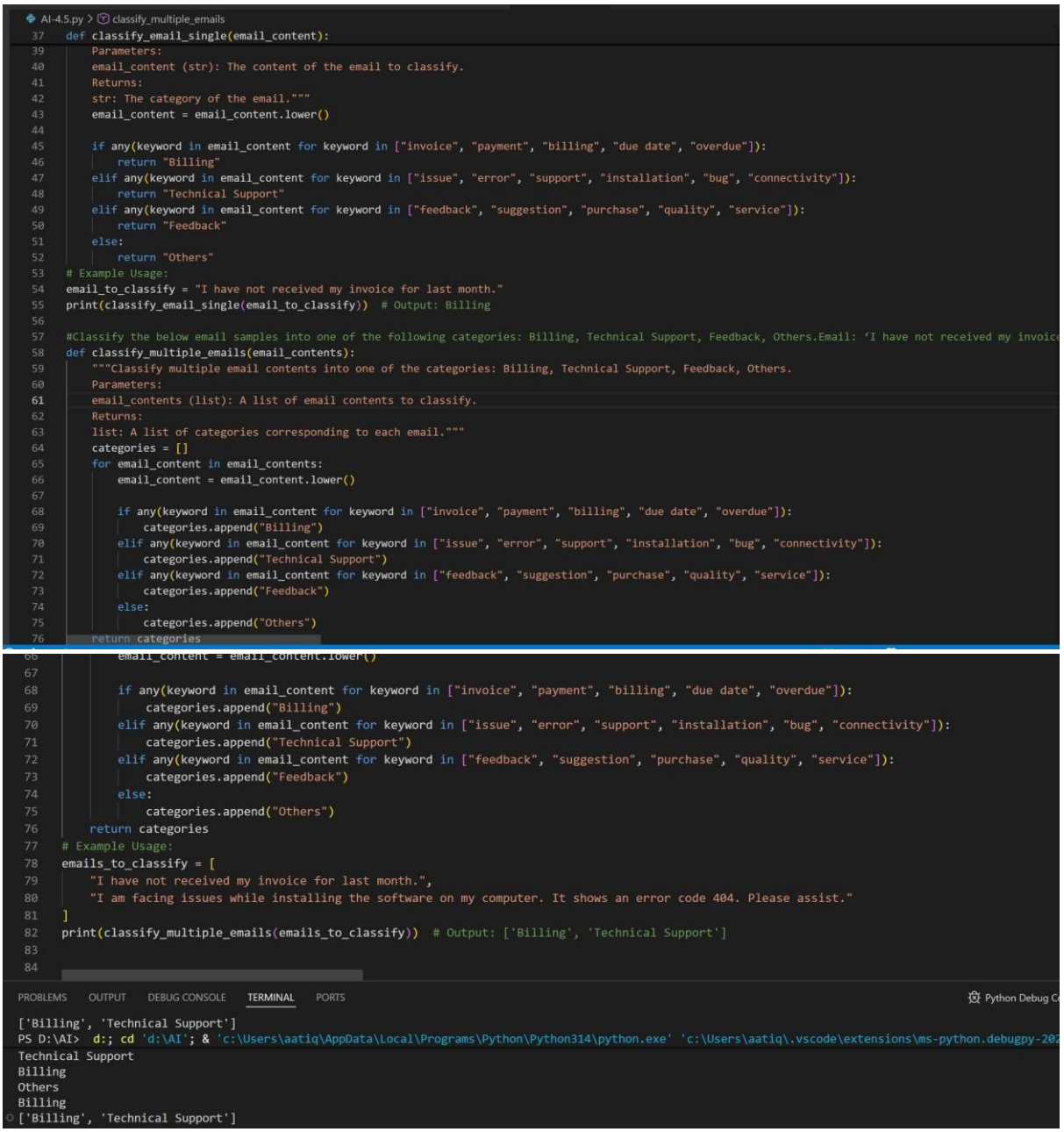
Classifytheabovebelowsamplesintooneofthefollowing categories: Billing, Technical Support, Feedback, Others.

Classifytheaboveemailsamplesintooneofthefollowingcategories: Billing,TechnicalSupport,Feedback,Others.Email:‘Ihavenot received my invoice for last month.

Classifythebelowemailsamplesintooneofthefollowingcategories: Billing,Technical Support,Feedback, Others.Email:‘Ihavenot receivedmyinvoiceforlastmonth.Billing,"Subject:Invoice#12345 Dear Customer, your invoice for the month of June is attached. Pleasemakethepaymentbytheduedate.Bestregards,Billing Team","TechnicalSupport","Subject:IssuewithSoftwareInstallation Hello Support Team, I am facing issues while installing the software on my computer. It shows an error code 404. Please assist. Thanks, User".

CODEandOUTPUT :


AI-4.5.py > classify_multiple_emails
#Management wants to automatically classify emails into categories like "Billing", "Technical Support", "Feedback", and "Others" before assigning them to
#Create or collect 10 short email samples, each belonging to one of the 4 categories.
#"Classify the above below samples into one of the following categories: Billing, Technical Support, Feedback, Others.'"
def classify_email(email_content):
"""Classify the email content into one of the categories: Billing, Technical Support, Feedback, Others.
Parameters:
email_content (str): The content of the email to classify.
Returns:
str: The category of the email."""
email_content = email_content.lower()
if any(keyword in email_content for keyword in ["invoice", "payment", "billing", "due date", "overdue"]):
return "Billing"
elif any(keyword in email_content for keyword in ["issue", "error", "support", "installation", "bug", "connectivity"]):
return "Technical Support"
elif any(keyword in email_content for keyword in ["feedback", "suggestion", "purchase", "quality", "service"]):
return "Feedback"
else:
return "Others"
email_samples = [
("Billing", "Subject: Invoice #12345\nDear Customer, your invoice for the month of June is attached. Please make the payment by the due date.\nBest re
("Technical Support", "Subject: Issue with Software Installation\nHello Support Team, I am facing issues while installing the software on my computer.
("Feedback", "Subject: Feedback on Recent Purchase\nHi Team, I recently purchased a product from your store and I am very satisfied with the quality a
("Others", "Subject: Meeting Reminder\nDear Team, this is a reminder for our meeting scheduled tomorrow at 10 AM in the conference room. Please be on t
("Billing", "Subject: Payment Confirmation\nDear Customer, we have received your payment for the invoice #67890. Thank you for your prompt payment.\nBe
("Technical Support", "Subject: Network Connectivity Issue\nHello, I am experiencing frequent disconnections from the internet. Can you please help me
("Feedback", "Subject: Suggestion for New Features\nHi Team, I would like to suggest a few new features for your app that I believe would enhance user
("Others", "Subject: Holiday Announcement\nDear All, please note that the office will be closed next Friday in observance of the holiday. Enjoy your d
("Billing", "Subject: Overdue Payment Notice\nDear Customer, our records indicate that your payment for invoice #54321 is overdue. Please make the paym
("Technical Support", "Subject: Software Bug Report\nHello Support, I have encountered a bug in the latest version of your software. It crashes when I
]
print(classify_email(email_samples[1][1])) # Output: Technical Support
print(classify_email(email_samples[4][1])) # Output: Billing
print(classify_email(email_samples[7][1])) # Output: Others
#Classify the above email samples into one of the following categories: Billing, Technical Support, Feedback, Others.Email: 'I have not received my invoic
def classify_email_single(email_content):
"""Classify a single email content into one of the categories: Billing, Technical Support, Feedback, Others.
Parameters:
0 0
BLACKBOX Agent
Indexing completed.
Open Website
Ln 61, Col 65
Spaces: 4
UTF-8
CRLF
Python
3.14.0
Go Live
BLACKB



```
AI-4.5.py > classify_multiple_emails
def classify_email_single(email_content):
    Parameters:
    email_content (str): The content of the email to classify.
    Returns:
    str: The category of the email."""
    email_content = email_content.lower()

    if any(keyword in email_content for keyword in ["invoice", "payment", "billing", "due date", "overdue"]):
        return "Billing"
    elif any(keyword in email_content for keyword in ["issue", "error", "support", "installation", "bug", "connectivity"]):
        return "Technical Support"
    elif any(keyword in email_content for keyword in ["feedback", "suggestion", "purchase", "quality", "service"]):
        return "Feedback"
    else:
        return "Others"
# Example Usage:
email_to_classify = "I have not received my invoice for last month."
print(classify_email_single(email_to_classify))  # Output: Billing

#Classify the below email samples into one of the following categories: Billing, Technical Support, Feedback, Others.Email: 'I have not received my invoic
def classify_multiple_emails(email_contents):
    """Classify multiple email contents into one of the categories: Billing, Technical Support, Feedback, Others.
    Parameters:
    email_contents (list): A list of email contents to classify.
    Returns:
    list: A list of categories corresponding to each email."""
    categories = []
    for email_content in email_contents:
        email_content = email_content.lower()

        if any(keyword in email_content for keyword in ["invoice", "payment", "billing", "due date", "overdue"]):
            categories.append("Billing")
        elif any(keyword in email_content for keyword in ["issue", "error", "support", "installation", "bug", "connectivity"]):
            categories.append("Technical Support")
        elif any(keyword in email_content for keyword in ["feedback", "suggestion", "purchase", "quality", "service"]):
            categories.append("Feedback")
        else:
            categories.append("Others")
    return categories
        email_content = email_content.lower()

        if any(keyword in email_content for keyword in ["invoice", "payment", "billing", "due date", "overdue"]):
            categories.append("Billing")
        elif any(keyword in email_content for keyword in ["issue", "error", "support", "installation", "bug", "connectivity"]):
            categories.append("Technical Support")
        elif any(keyword in email_content for keyword in ["feedback", "suggestion", "purchase", "quality", "service"]):
            categories.append("Feedback")
        else:
            categories.append("Others")
    return categories
# Example Usage:
emails_to_classify = [
    "I have not received my invoice for last month.",
    "I am facing issues while installing the software on my computer. It shows an error code 404. Please assist."
]
print(classify_multiple_emails(emails_to_classify))  # Output: ['Billing', 'Technical Support']


PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS                    Python Debug C
['Billing', 'Technical Support']
PS D:\AI>  d:; cd 'd:\AI'; & 'c:\Users\aatiq\AppData\Local\Programs\Python\Python314\python.exe' 'c:\Users\aatiq\.vscode\extensions\ms-python.debugpy-202
Technical Support
Billing
Others
Billing
['Billing', 'Technical Support']
```

Justification:

Thistaskshowshowpromptengineeringcanbeusedtosortemailsautomatically.Zero-shot prompting works for simple emails. But it may give wrong results when emails are unclear. One-shotandfew-shotpromptinghelpthemodelunderstandbetterbygivingexamples.This reduces manual work and saves time for the company.

PROMPT:

AtravelassistantmustclassifyqueriesintoFlightBooking,HotelBooking,Cancellation,or General Travel Info. Tasks: a. Prepare labeled travel queries.

Classifythe travel queryintooneof thecategories: FlightBooking,HotelBooking, Cancellation,GeneralTravelInfo."IneedtobookaflighttoNewYorknextweek.",

Task2:TravelQueryClassification

AtravelassistantmustclassifyqueriesintoFlightBooking,HotelBooking,Cancellation,or General Travel Info.

Tasks:

a. Preparelabeledtravelqueries.

b. ApplyZero-shotprompting.

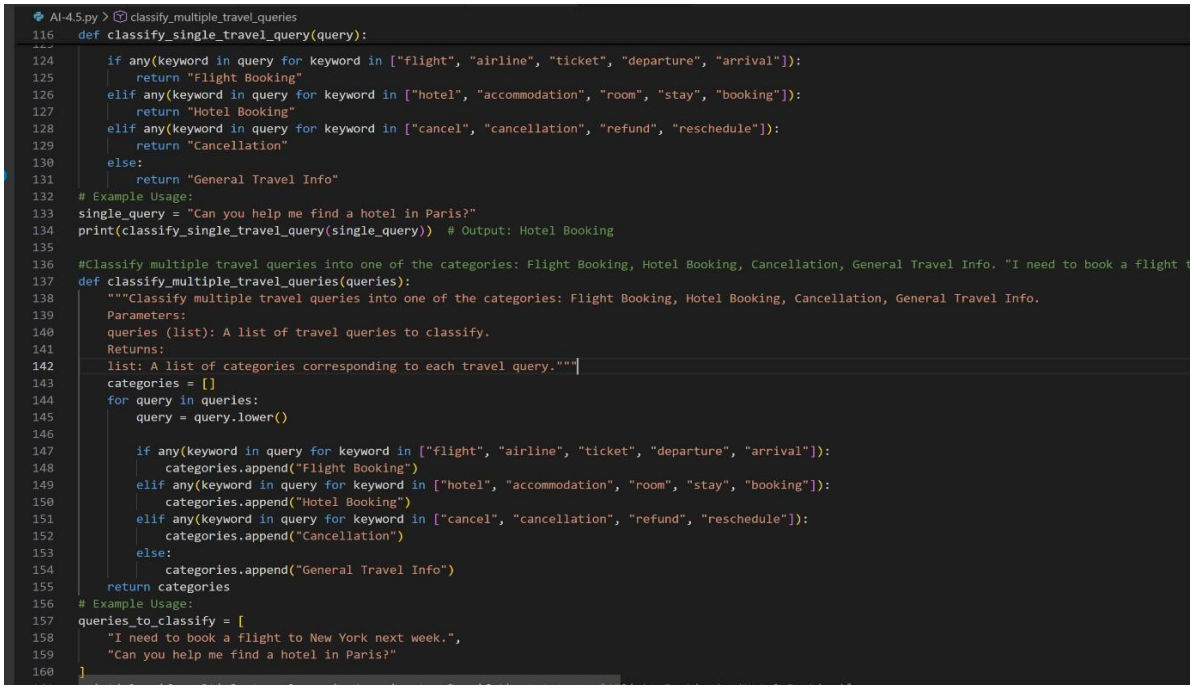
c. ApplyOne-shot prompting.

d. ApplyFew-shotprompting.

e. Compareresponseconsistency.

Classify multiple travel queries into one of the categories: Flight Booking, Hotel Booking, Cancellation,GeneralTravelInfo."IneedtobookaflighttoNewYorknextweek.","Canyou help me find a hotel in Paris?"

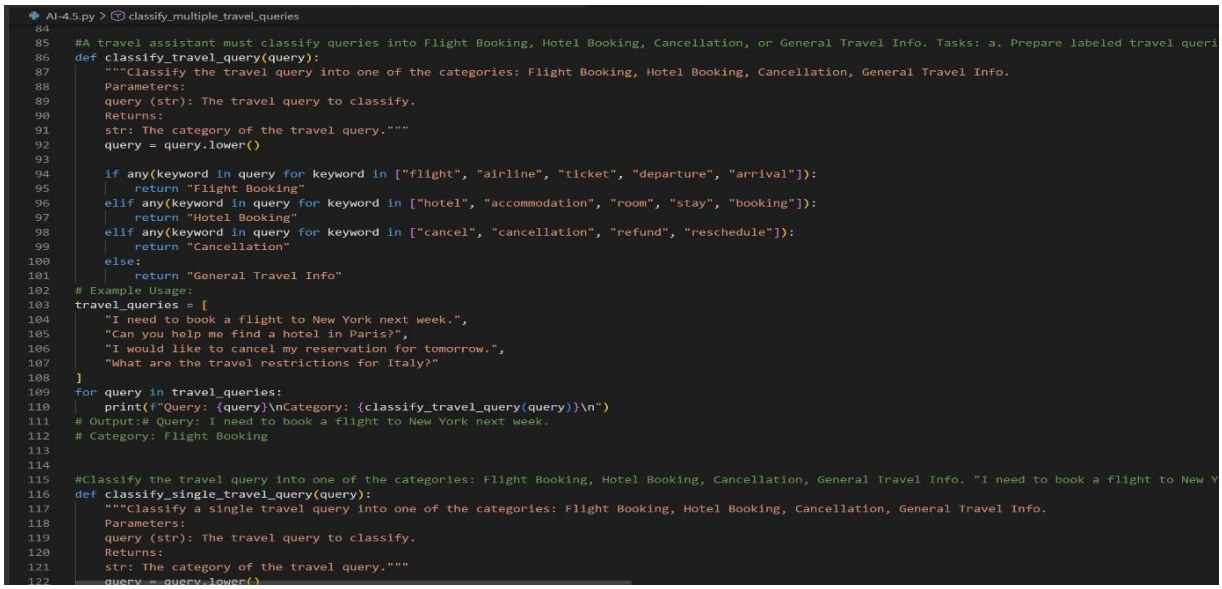
Classifymultipletravelqueriesintooneof thecategories:FlightBooking,HotelBooking, Cancellation,GeneralTravelInfo."CanyouhelpmefindahotelinParis?""Iwouldliketo cancelmyreservationfortomorrow.","WhatarethetravelrestrictionsforItaly?""**CODE and OUTPUT :**



```python
AI-4.5.py > classify_multiple_travel_queries
def classify_single_travel_query(query):
    if any(keyword in query for keyword in ["flight", "airline", "ticket", "departure", "arrival"]):
        return "Flight Booking"
    elif any(keyword in query for keyword in ["hotel", "accommodation", "room", "stay", "booking"]):
        return "Hotel Booking"
    elif any(keyword in query for keyword in ["cancel", "cancellation", "refund", "reschedule"]):
        return "Cancellation"
    else:
        return "General Travel Info"
# Example Usage:
single_query = "Can you help me find a hotel in Paris?"
print(classify_single_travel_query(single_query))  # Output: Hotel Booking

#Classify multiple travel queries into one of the categories: Flight Booking, Hotel Booking, Cancellation, General Travel Info. "I need to book a flight t
def classify_multiple_travel_queries(queries):
    """Classify multiple travel queries into one of the categories: Flight Booking, Hotel Booking, Cancellation, General Travel Info.
    Parameters:
    queries (list): A list of travel queries to classify.
    Returns:
    list: A list of categories corresponding to each travel query."""
    categories = []
    for query in queries:
        query = query.lower()

        if any(keyword in query for keyword in ["flight", "airline", "ticket", "departure", "arrival"]):
            categories.append("Flight Booking")
        elif any(keyword in query for keyword in ["hotel", "accommodation", "room", "stay", "booking"]):
            categories.append("Hotel Booking")
        elif any(keyword in query for keyword in ["cancel", "cancellation", "refund", "reschedule"]):
            categories.append("Cancellation")
        else:
            categories.append("General Travel Info")
    return categories
# Example Usage:
queries_to_classify = [
    "I need to book a flight to New York next week.",
    "Can you help me find a hotel in Paris?"
]
```



```
AI-4.5.py > classify_multiple_travel_queries
84
85  #A travel assistant must classify queries into Flight Booking, Hotel Booking, Cancellation, or General Travel Info. Tasks: a. Prepare labeled travel queri
86  def classify_travel_query(query):
87      """Classify the travel query into one of the categories: Flight Booking, Hotel Booking, Cancellation, General Travel Info.
88      Parameters:
89      query (str): The travel query to classify.
90      Returns:
91      str: The category of the travel query."""
92      query = query.lower()
93
94      if any(keyword in query for keyword in ["flight", "airline", "ticket", "departure", "arrival"]):
95          return "Flight Booking"
96      elif any(keyword in query for keyword in ["hotel", "accommodation", "room", "stay", "booking"]):
97          return "Hotel Booking"
98      elif any(keyword in query for keyword in ["cancel", "cancellation", "refund", "reschedule"]):
99          return "Cancellation"
100     else:
101         return "General Travel Info"
102 # Example Usage:
103 travel_queries = [
104     "I need to book a flight to New York next week.",
105     "Can you help me find a hotel in Paris?",
106     "I would like to cancel my reservation for tomorrow.",
107     "What are the travel restrictions for Italy?"
108 ]
109 for query in travel_queries:
110     print(f"Query: {query}\nCategory: {classify_travel_query(query)}\n")
111 # Output:# Query: I need to book a flight to New York next week.
112 # Category: Flight Booking
113
114
115 #Classify the travel query into one of the categories: Flight Booking, Hotel Booking, Cancellation, General Travel Info. "I need to book a flight to New Y
116 def classify_single_travel_query(query):
117     """Classify a single travel query into one of the categories: Flight Booking, Hotel Booking, Cancellation, General Travel Info.
118     Parameters:
119     query (str): The travel query to classify.
120     Returns:
121     str: The category of the travel query."""
122     query = query.lower()
```

```
#Classify multiple travel queries into one of the categories: Flight Booking, Hotel Booking, Cancellation, General Travel Info. "Can you help me find a hot
def classify_more_travel_queries(queries):
    """Classify multiple travel queries into one of the categories: Flight Booking, Hotel Booking, Cancellation, General Travel Info.
    Parameters:
    queries (list): A list of travel queries to classify.
    Returns:
    list: A list of categories corresponding to each travel query."""
    categories = []
    for query in queries:
        query = query.lower()

        if any(keyword in query for keyword in ["flight", "airline", "ticket", "departure", "arrival"]):
            categories.append("Flight Booking")
        elif any(keyword in query for keyword in ["hotel", "accommodation", "room", "stay", "booking"]):
            categories.append("Hotel Booking")
        elif any(keyword in query for keyword in ["cancel", "cancellation", "refund", "reschedule"]):
            categories.append("Cancellation")
        else:
            categories.append("General Travel Info")
    return categories
# Example Usage:
more_queries_to_classify = [
    "Can you help me find a hotel in Paris?",
    "I would like to cancel my reservation for tomorrow.",
    "What are the travel restrictions for Italy?"
]
print(classify_more_travel_queries(more_queries_to_classify))  # Output: ['Hotel Booking', 'Cancellation', 'General Travel Info']
```

```
PS D:\AI>  & 'c:\Users\aatiq\AppData\Local\Programs\Python\Python314\python.exe' 'c:\Users\aatiq\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\
er' '55871' '--' 'D:\AI\AI-4.5.py'
Query: I need to book a flight to New York next week.
Category: Flight Booking

Query: Can you help me find a hotel in Paris?
Category: Hotel Booking

Query: I would like to cancel my reservation for tomorrow.
Category: Cancellation

Query: What are the travel restrictions for Italy?
Category: General Travel Info

Hotel Booking
['Flight Booking', 'Hotel Booking']
['Hotel Booking', 'Cancellation', 'General Travel Info']
PS D:\AI>
```

Justification:

Travel questions often look similar but have different meanings. Zero-shot prompting may confusetheintent.One-shotpromptinggivesasmallexample,sothemodelperformsbetter. Few-shotpromptinggivesmoreexamplesandimprovesaccuracy.Thishelpstravelassistants give correct responses.

PROMPT:

Task3.ProgrammingQuestionTypeIdentification

Scenario:

AcodinghelpchatbotmustclassifyqueriesintoSyntaxError,LogicError,Optimization,or Conceptual Question.

Tasks:

a. Preparecoding-relateduserqueries.

b. PerformZero-shotclassification.

c. PerformOne-shotclassification.

d. PerformFew-shotclassification.

Analyzeimprovementsintechnicalaccuracy

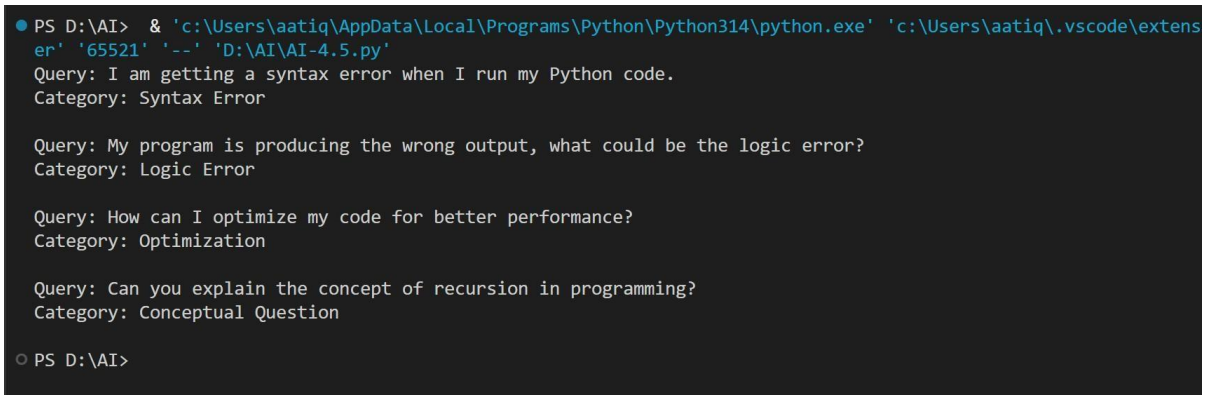
AcodinghelpchatbotmustclassifyqueriesintoSyntaxError,LogicError,Optimization,or Conceptual Question. Tasks:a. Prepare coding-related user queries.b. Perform Zero-shot classification.c. Perform One-shot classification.d. Perform Few-shot classification.

CODEandOUTPUT:

```python
#A coding help chatbot must classify queries into Syntax Error, Logic Error, Optimization, or Conceptual Question. Tasks:a. Prepare coding-related user qu
def classify_coding_query(query):
    """Classify the coding query into one of the categories: Syntax Error, Logic Error, Optimization, Conceptual Question.
    Parameters:
    query (str): The coding query to classify.
    Returns:
    str: The category of the coding query."""
    query = query.lower()

    if any(keyword in query for keyword in ["syntax error", "unexpected indent", "missing parenthesis", "invalid syntax"]):
        return "Syntax Error"
    elif any(keyword in query for keyword in ["logic error", "wrong output", "incorrect result", "bug"]):
        return "Logic Error"
    elif any(keyword in query for keyword in ["optimize", "performance", "efficiency", "speed up"]):
        return "Optimization"
    elif any(keyword in query for keyword in ["how to", "what is", "explain", "concept"]):
        return "Conceptual Question"
    else:
        return "Others"
# Example Usage:
coding_queries = [
    "I am getting a syntax error when I run my Python code.",
    "My program is producing the wrong output, what could be the logic error?",
    "How can I optimize my code for better performance?",
    "Can you explain the concept of recursion in programming?"
]
for query in coding_queries:
    print(f"Query: {query}\nCategory: {classify_coding_query(query)}\n")
# Output:# Query: I am getting a syntax error when I run my Python code.
# Category: Syntax Error
# Query: My program is producing the wrong output, what could be the logic error?
# Category: Logic Error
# Query: How can I optimize my code for better performance?
# Category: Optimization
# Query: Can you explain the concept of recursion in programming?
# Category: Conceptual Question

```



Justification:

Programmingquestionsneedcorrecttechnicalunderstanding.Zero- shot prompting may mix up syntax and logic errors. One-shot prompting gives guidance with an example. Few-shot prompting improvesaccuracybyshowingdifferentquestiontypes.Thishelps the chatbot give better coding help.

Task4.SocialMediaPostCategorizationScenario:

AsocialmediaanalyticstoolmustclassifypostsintoPromotion, Complaint, Appreciation, or Inquiry.

Tasks:

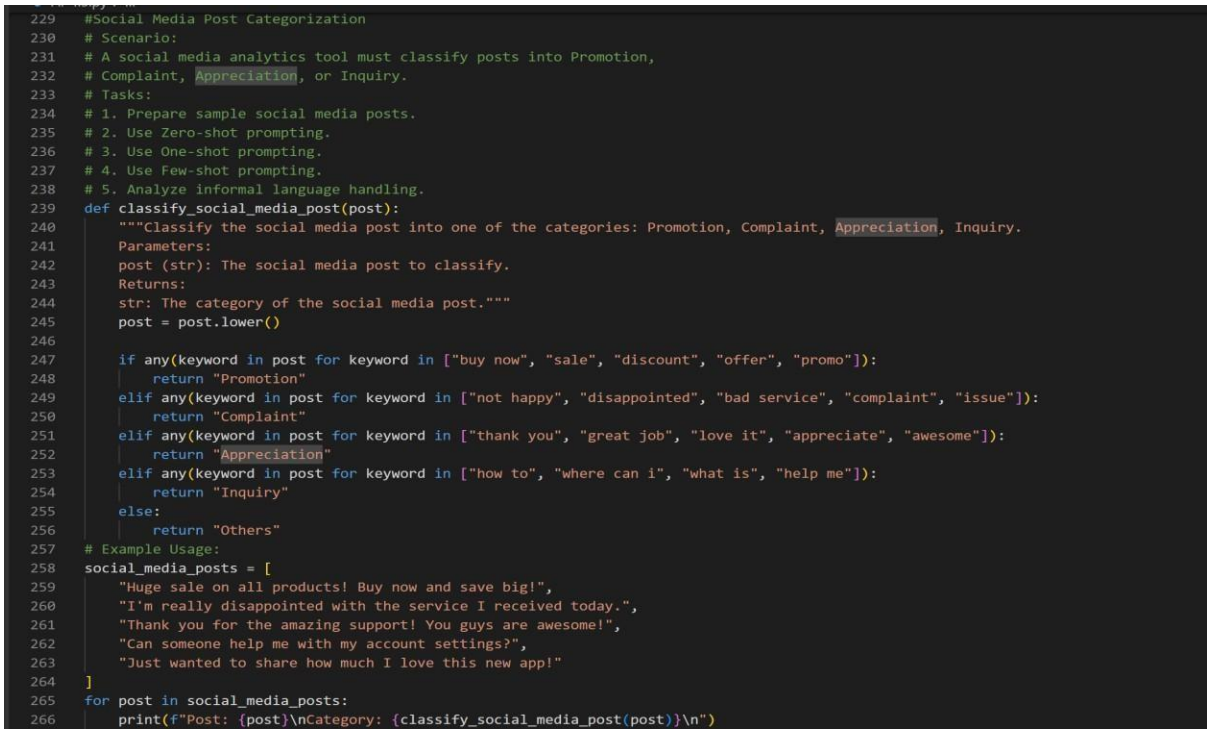
1. Preparesamplesocialmediaposts.
2. UseZero-shotprompting.
3. UseOne-shotprompting.
4. UseFew-shotprompting.
5. Analyzeinformallanguagehandling.

PROMPT:

SocialMediaPostCategorization,Asocialmediaanalyticstoolmust classify posts into Promotion,Complaint,Appreciation,orInquiry.1.Preparesample social media posts. 2. Use Zero-shot prompting. 3. Use One-shot

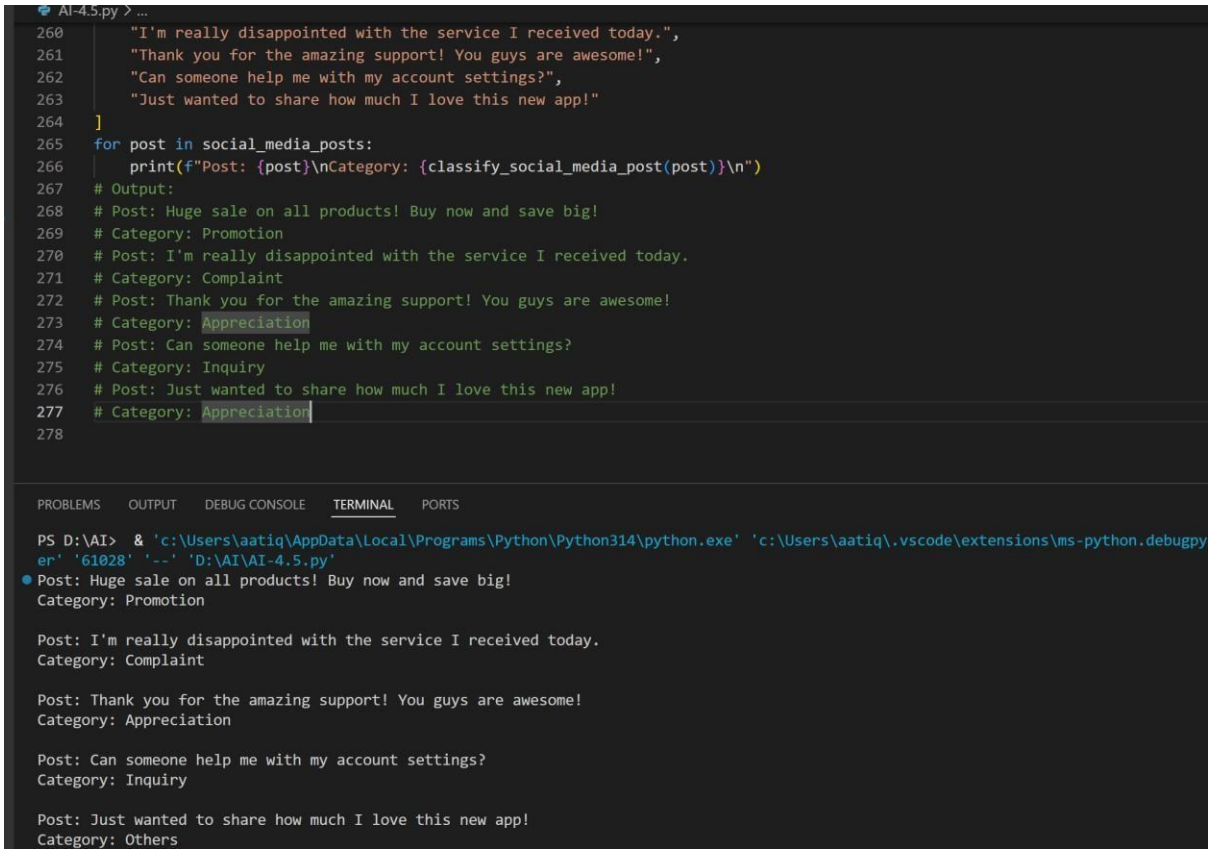
prompting.4.UseFew-shotprompting.5.Analyzeinformallanguage handling.

CODEandOUTPUT :



```python
#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.
def classify_social_media_post(post):
    """Classify the social media post into one of the categories: Promotion, Complaint, Appreciation, Inquiry.
    Parameters:
    post (str): The social media post to classify.
    Returns:
    str: The category of the social media post."""
    post = post.lower()

    if any(keyword in post for keyword in ["buy now", "sale", "discount", "offer", "promo"]):
        return "Promotion"
    elif any(keyword in post for keyword in ["not happy", "disappointed", "bad service", "complaint", "issue"]):
        return "Complaint"
    elif any(keyword in post for keyword in ["thank you", "great job", "love it", "appreciate", "awesome"]):
        return "Appreciation"
    elif any(keyword in post for keyword in ["how to", "where can i", "what is", "help me"]):
        return "Inquiry"
    else:
        return "Others"
# Example Usage:
social_media_posts = [
    "Huge sale on all products! Buy now and save big!",
    "I'm really disappointed with the service I received today.",
    "Thank you for the amazing support! You guys are awesome!",
    "Can someone help me with my account settings?",
    "Just wanted to share how much I love this new app!"
]
for post in social_media_posts:
    print(f"Post: {post}\nCategory: {classify_social_media_post(post)}\n")
```



```
AI-4.5.py > ...
    "I'm really disappointed with the service I received today.",
    "Thank you for the amazing support! You guys are awesome!",
    "Can someone help me with my account settings?",
    "Just wanted to share how much I love this new app!"
]
for post in social_media_posts:
    print(f"Post: {post}\nCategory: {classify_social_media_post(post)}\n")
# Output:
# Post: Huge sale on all products! Buy now and save big!
# Category: Promotion
# Post: I'm really disappointed with the service I received today.
# Category: Complaint
# Post: Thank you for the amazing support! You guys are awesome!
# Category: Appreciation
# Post: Can someone help me with my account settings?
# Category: Inquiry
# Post: Just wanted to share how much I love this new app!
# Category: Appreciation

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS D:\AI>  & 'c:\Users\aatiq\AppData\Local\Programs\Python\Python314\python.exe' 'c:\Users\aatiq\.vscode\extensions\ms-python.debugpy
er' '61028' '--' 'D:\AI\AI-4.5.py'
Post: Huge sale on all products! Buy now and save big!
Category: Promotion

Post: I'm really disappointed with the service I received today.
Category: Complaint

Post: Thank you for the amazing support! You guys are awesome!
Category: Appreciation

Post: Can someone help me with my account settings?
Category: Inquiry

Post: Just wanted to share how much I love this new app!
Category: Others
```

Justification:

Social media posts use informal language and emojis. Zero-shot prompting may not understandthetonecorrectly.One-shotpromptinghelpsalittlebyshowinganexample. Few-shotpromptingworksbetterbyusingmultipleexamples.Thisimprovesunderstanding of user posts.