

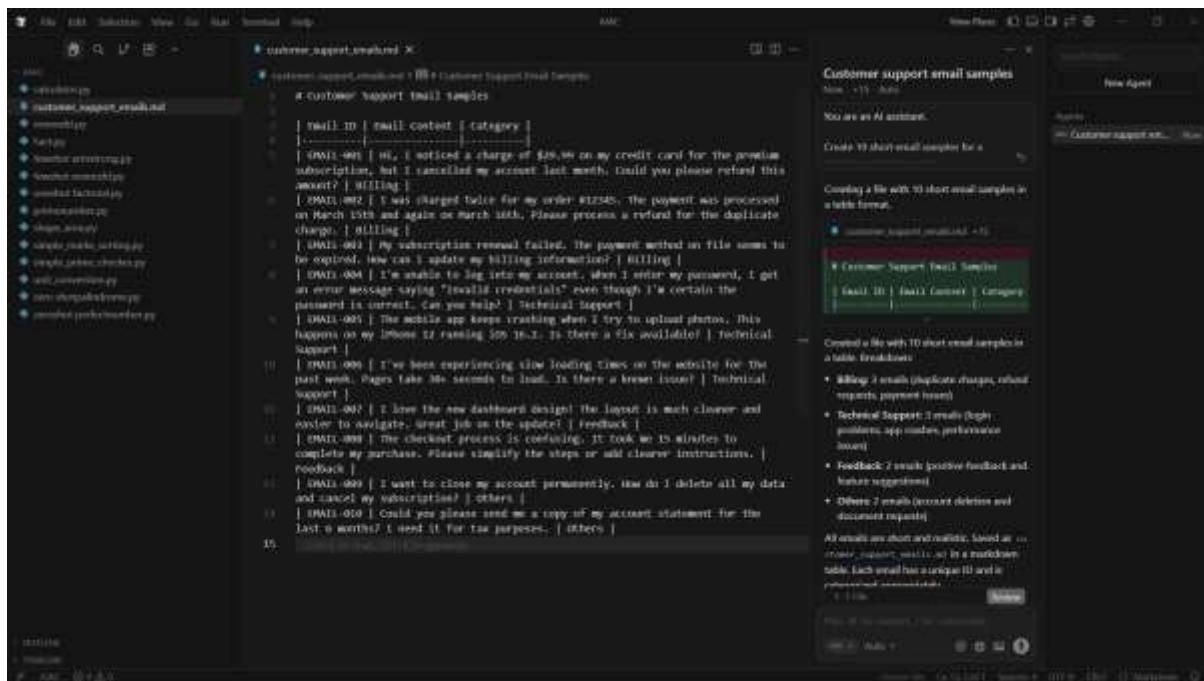
# School of Computer Science and Artificial Intelligence

## Lab Assignment -4.5

<b>Program</b>	<b>:B. Tech (CSE)</b>
<b>Specialization</b>	<b>:AIML</b>
<b>Course Title</b>	<b>: AI ASSISTED CODING</b>
<b>Course Code</b>	<b>: 23CS002PC304</b>
<b>Semester</b>	<b>: VI</b>
<b>Academic Session</b>	<b>: 2025-2026</b>
<b>Name of Student</b>	<b>: K. Shiva Charan</b>
<b>Enrollment No.</b>	<b>: 2303A52160</b>
<b>Batch No.</b>	<b>: 34</b>

### a. Prepare Sample Data:

- Create or collect 10 short email samples, each belonging to one of the 4 categories.



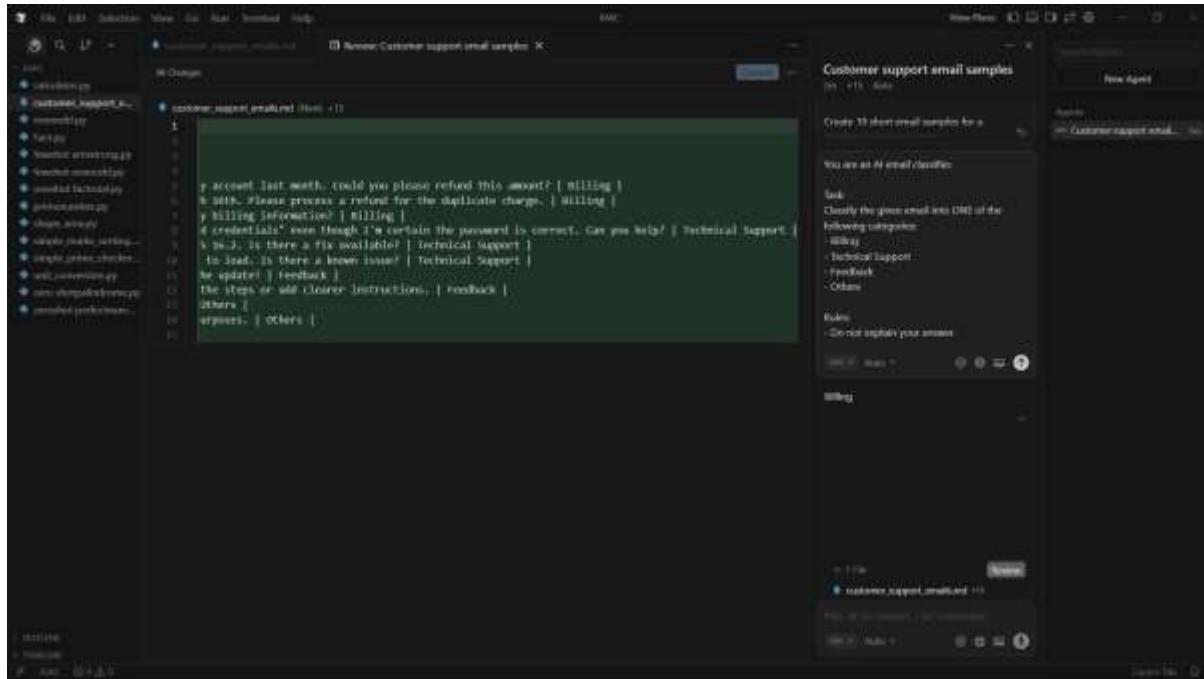
### b. Zero-shot Prompting:

- Design a prompt that asks the LLM to classify a single email without providing any examples.

- Example prompt:

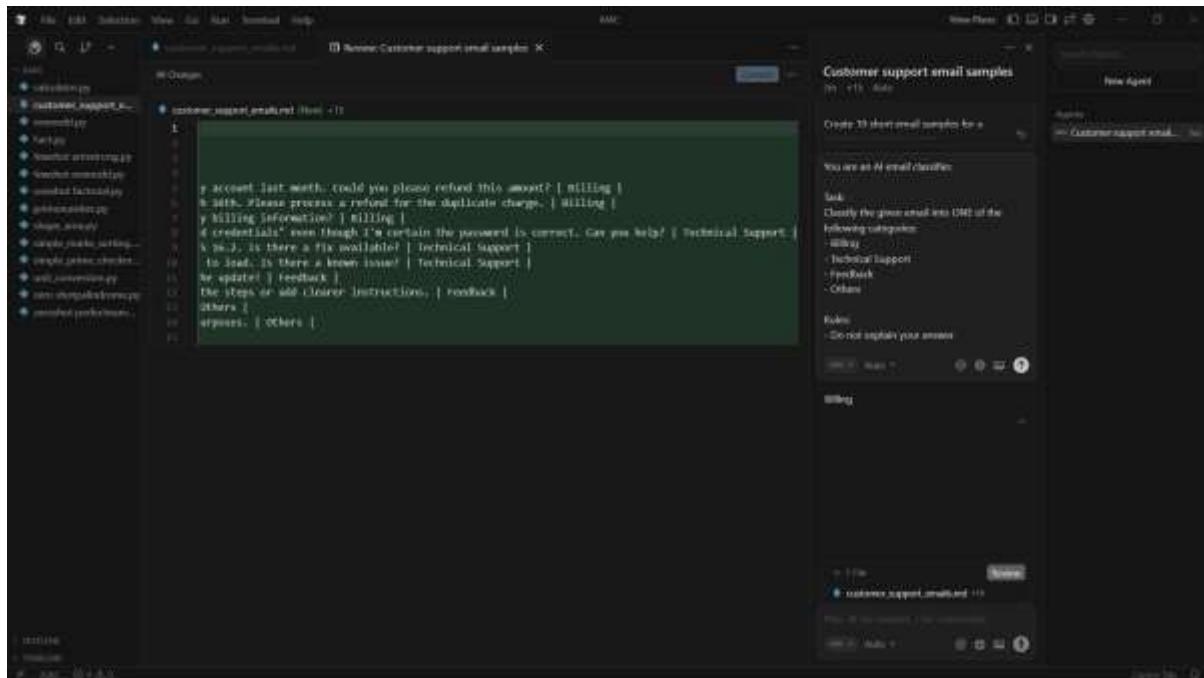
"Classify the following email into one of the following categories:

Billing, Technical Support, Feedback, Others. Email: 'I have not received my invoice for last month.'



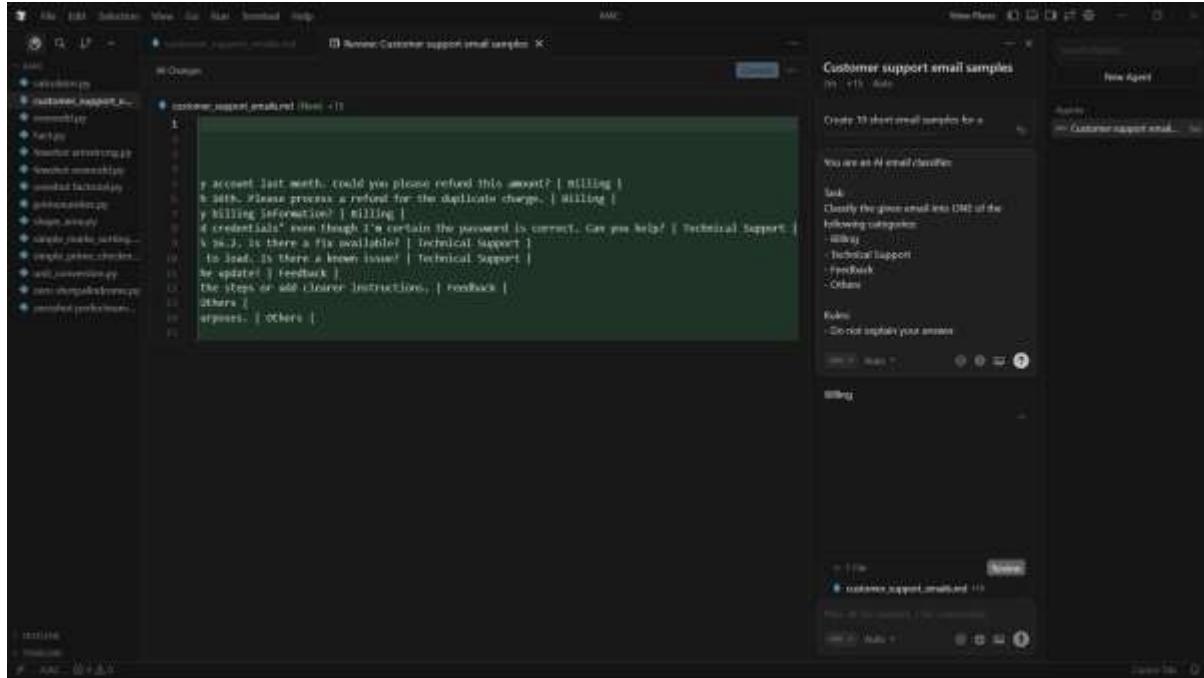
### c. One-shot Prompting:

- Add one labeled example before asking the model to classify a new email.



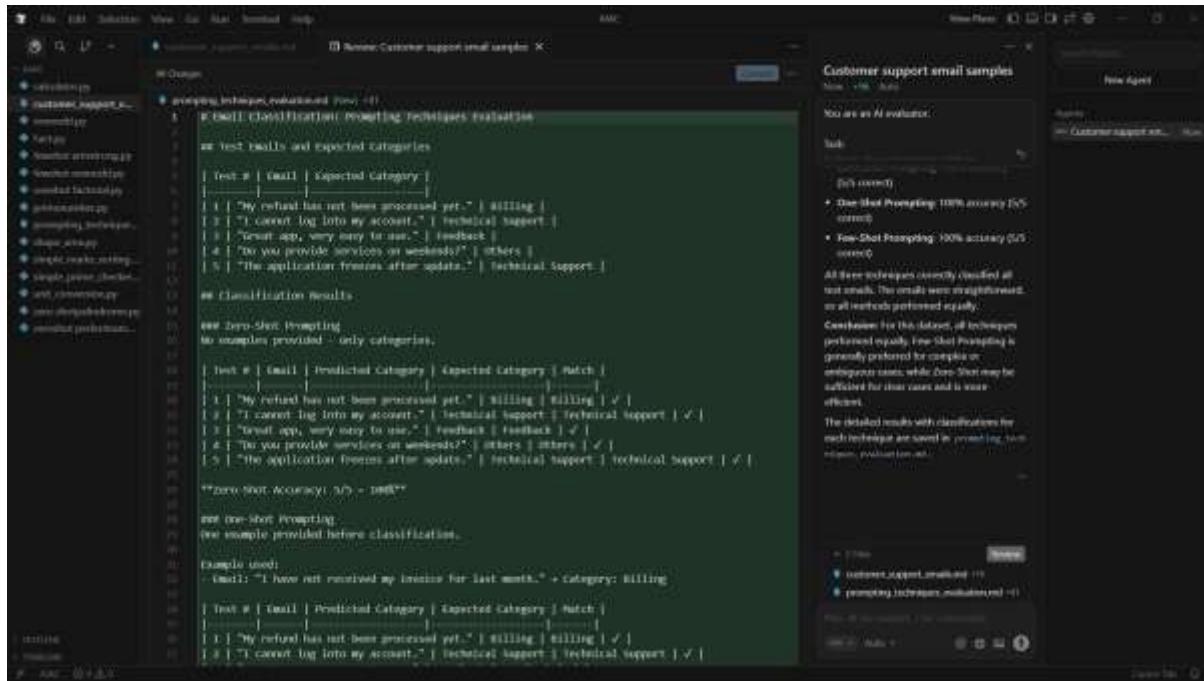
#### d. Few-shot Prompting:

- Use 3–5 labeled examples in your prompt before asking the model to classify a new email.



#### e. Evaluation:

- Run all three techniques on the same set of 5 test emails.
- Compare and document the accuracy and clarity of responses.



```

# Examples used:
# Email: "I have not received my invoice for last month." -> Category: Billing
# Email: "The application crashes when I open it." -> Category: Technical Support
# Email: "I really like the new update of your app." -> Category: Feedback
# Email: "What are your office working hours?" -> Category: Others

| test # | email | predicted category | expected category | match |
| --- | --- | --- | --- | --- |
| 1 | "My refund has not been processed yet." | Billing | Billing | ✓ |
| 2 | "I cannot log into my account." | Technical Support | Technical Support | ✓ |
| 3 | "Great app, very easy to use." | Feedback | Feedback | ✓ |
| 4 | "Do you provide services on weekends?" | Others | Others | ✓ |
| 5 | "The application freezes after update." | Technical Support | Technical Support | ✓ |

**Zero-Shot Accuracy: 5/5 = 100%**  

**Summary Table:**


| Technique           | Correct Predictions | Total Emails | Accuracy |
|---------------------|---------------------|--------------|----------|
| Zero-Shot Prompting | 5                   | 5            | 100%     |
| Few-Shot Prompting  | 5                   | 5            | 100%     |
| One-Shot Prompting  | 5                   | 5            | 100%     |


**Conclusion:**
All three prompting techniques achieved perfect accuracy (100%) on this test set. The test emails can be categorized correctly by all methods. Zero-Shot is the most efficient, followed by Few-Shot and One-Shot. All methods perform equally well on this dataset. However, few-shot prompting may be more efficient for complex or ambiguous cases, while zero-shot may be sufficient for clear cases and is more efficient.

```

You are an AI evaluator.  
Task:  
 Is it correct?  
• One-Shot Prompting: 100% accuracy (5/5 correct)  
• Few-Shot Prompting: 100% accuracy (5/5 correct)  
All three techniques correctly classified all test emails. The results were straightforward, so all methods performed equally.  
Conclusion: For this dataset, all techniques performed equally. Few-Shot Prompting is generally preferred for complex or ambiguous cases, while Zero-Shot may be sufficient for clear cases and is more efficient.  
The detailed results with classifications for each technique are saved in [prompting\\_small\\_evaluation.csv](#).

## 2. Travel Query Classification

Scenario:

A travel assistant must classify queries into Flight Booking, Hotel Booking, Cancellation, or General Travel Info.

Tasks:

- a. Prepare labeled travel queries.
- b. Apply Zero-shot prompting.
- c. Apply One-shot prompting.
- d. Apply Few-shot prompting.
- e. Compare response consistency.

Travel assistant query labeling

Query	Category
query: "I want to book a flight from Delhi to Mumbai."	Flight Booking
query: "Show me available flights for tomorrow."	Flight booking
query: "I need a hotel near the airport in Bangalore."	Hotel booking
query: "Book a 3-star hotel in Chennai for two nights."	Hotel booking
query: "Cancelled my flight scheduled for next Monday."	Cancellation
query: "I want to cancel my hotel reservation."	Cancellation
query: "What documents are required for international travel?"	General travel info
query: "What is the baggage allowance for domestic flights?"	General travel info

Travel Queries

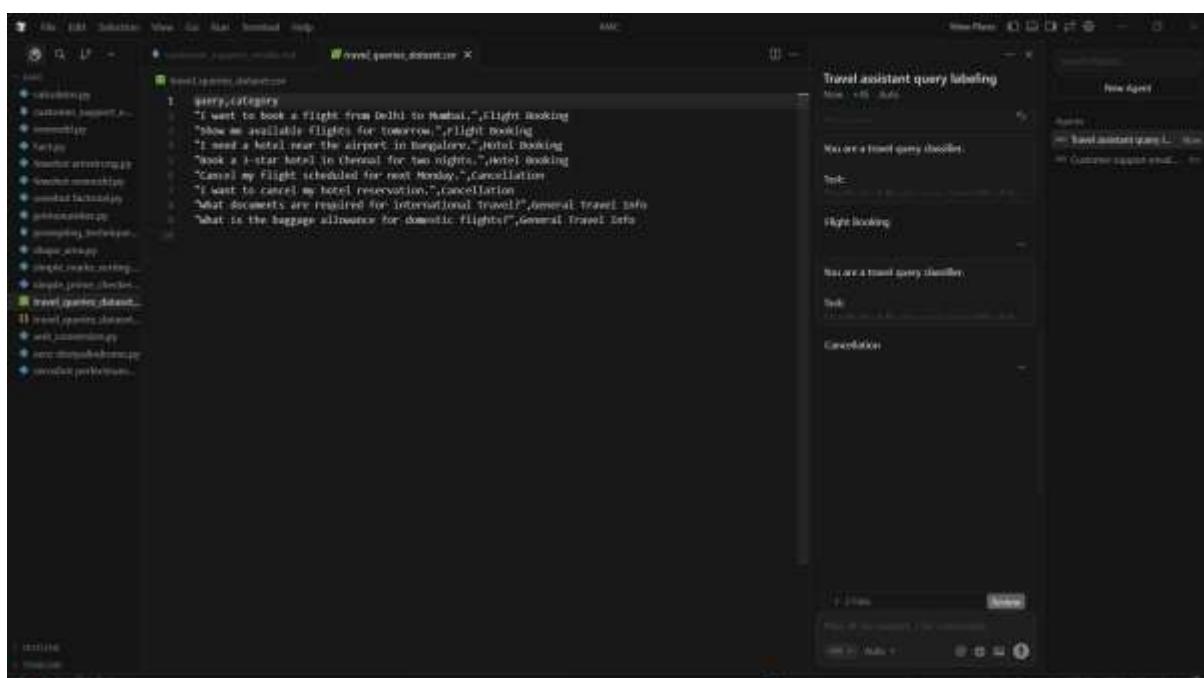
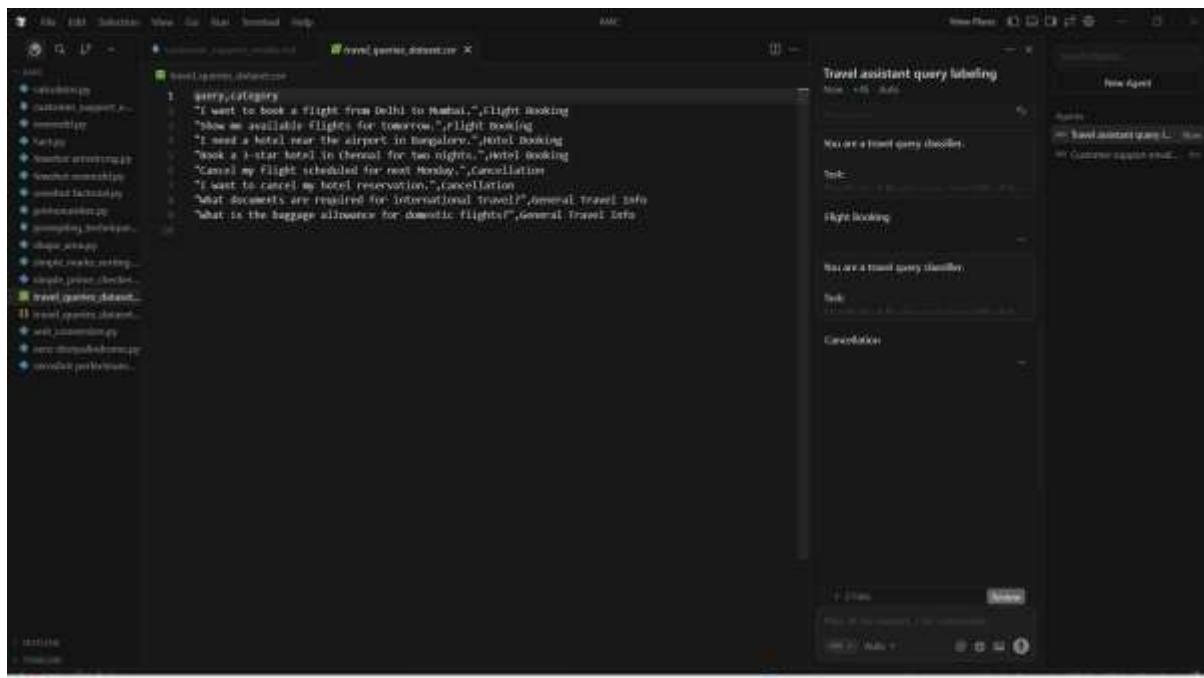
- Flight booking
- Hotel booking
- Flight cancellation
- Hotel cancellation
- General travel info
- Domestic flight info

Travel Assistant

- Create labeled travel queries for a travel assistant system.
- View travel\_query\_annotation.csv
- Customer support ticket

The screenshot shows the Oracle Database SQL Developer interface. The left sidebar displays a tree view of database objects, including schema, tables, and views. A table named 'travel\_query\_classifier' is selected. The main area contains a SQL query:`SELECT query_id, query_text, category FROM travel_query_classifier`

The results of this query are displayed in the bottom pane, showing several rows of travel-related queries categorized into 'Flight booking' and 'Hotel booking'. On the right side of the screen, there is a 'Travel assistant query labeling' panel. This panel includes a search bar, a list of labeled queries, and a 'Create labeled travel queries for a travel assistant system' section. Below this, there are sections for 'dataset', 'label', and 'Help'.



The screenshot shows a Jupyter Notebook interface with two tabs: "Travel Query Classifier Evaluation" and "Untitled". The "Travel Query Classifier Evaluation" tab contains Python code for classifying travel queries into categories like flight booking, hotel booking, cancellation, and general travel info. It includes functions for generating test queries, defining expected categories, and zero-shot classification. A sidebar on the right displays a "Travel assistant query labeling" interface with a text input field, a progress bar, and a "Run" button.

```
class TravelQueryClassifierEvaluation:
    def __init__(self):
        self.test_queries = [
            "Book a flight from Ahmedabad to New Delhi",
            "Find a budget hotel in New Delhi",
            "Cancel my hotel reservation",
            "What documents are required for international travel?",
            "I want to cancel my flight."
        ]
        self.expected_categories = [
            "Flight Booking",
            "Hotel Booking",
            "Cancellation",
            "General Travel Info",
            "General"
        ]
        self.categories = [
            "Flight Booking",
            "Hotel Booking",
            "Cancellation",
            "General Travel Info"
        ]

    def zero_shot_classify(query):
        ...
        zero_shot_prompts_only_instructions_no_examples
        prompt = f'''You are a travel query classifier.
        TASK: Classify the following query into one of the categories:
        {query}'''

    def eval_results():
        for result in [zero_shot_results, one_shot_results, few_shot_results]:
            for i, result in enumerate(result['results']):
                print()
                print(">" * 100)
                print(result['category'])
                print(">" * 100)

                accuracy = result['accuracy']
                zero_shot_results['technique'], zero_shot_results['accuracy'],
                one_shot_results['technique'], one_shot_results['accuracy'],
                few_shot_results['technique'], few_shot_results['accuracy']

            best_technique = max(accuracy, inv_accuracy.get())
            best_accuracy = accuracy[best_technique]

            print(f'Default classification technique: {best_technique}')
            print(f'Accuracy: {best_accuracy:.4f}%')
            print()

        if few_shot_results['accuracy'] > one_shot_results['accuracy'] > zero_shot_results['accuracy']:
            print('Observation: Few-shot prompting demonstrates the highest consistency, followed by One-shot, and then Zero-shot. Providing examples helps the model better understand the classification.')
        elif one_shot_results['accuracy'] > zero_shot_results['accuracy']:
            print('Observation: One-shot prompting shows better performance than zero-shot, indicating the prompt even with a single example can improve classification consistency.')
        else:
            print('Observation: Results vary across techniques. Additional examples may help improve consistency.')

        print(">" * 100)
        print(results)
```

This screenshot shows the continuation of the Jupyter Notebook code. The "Travel Query Classifier Evaluation" tab now includes a comparison of different prompting techniques (Zero-shot, One-shot, Few-shot) based on accuracy. The code prints out the results for each technique and observes that few-shot prompting consistently yields the highest accuracy. A sidebar on the right shows a "Travel assistant query labeling" interface with a text input field, a progress bar, and a "Run" button.

```
def eval_results():
    for result in [zero_shot_results, one_shot_results, few_shot_results]:
        for i, result in enumerate(result['results']):
            print()
            print(">" * 100)
            print(result['category'])
            print(">" * 100)

            accuracy = result['accuracy']
            zero_shot_results['technique'], zero_shot_results['accuracy'],
            one_shot_results['technique'], one_shot_results['accuracy'],
            few_shot_results['technique'], few_shot_results['accuracy']

        best_technique = max(accuracy, inv_accuracy.get())
        best_accuracy = accuracy[best_technique]

        print(f'Default classification technique: {best_technique}')
        print(f'Accuracy: {best_accuracy:.4f}%')
        print()

    if few_shot_results['accuracy'] > one_shot_results['accuracy'] > zero_shot_results['accuracy']:
        print('Observation: Few-shot prompting demonstrates the highest consistency, followed by One-shot, and then Zero-shot. Providing examples helps the model better understand the classification.')
    elif one_shot_results['accuracy'] > zero_shot_results['accuracy']:
        print('Observation: One-shot prompting shows better performance than zero-shot, indicating the prompt even with a single example can improve classification consistency.')
    else:
        print('Observation: Results vary across techniques. Additional examples may help improve consistency.')

    print(">" * 100)
    print(results)
```

### 3. Programming Question Type Identification

Scenario:

A coding help chatbot must classify queries into Syntax Error, Logic

Error, 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.
- e. Analyze improvements in technical accuracy.

The screenshot displays a desktop application interface for 'coding.guru AI Assistant'. The main window title is 'coding.guru AI Assistant - X'. On the left side, there's a sidebar with a tree view containing categories like 'coding', 'chatbot', 'coding\_query\_status', 'customer\_support', 'modality', 'intent', 'context', 'conceptual', 'error', 'usage', 'new\_query\_status', 'new\_query\_status', 'new\_query\_classifier', 'new\_query\_classifier', 'new\_query\_classifier', 'new\_query\_classifier', 'new\_query\_classifier', and 'context\_preferences'. The main content area is divided into two panes:

- Programming help chatbot queries**: This pane is titled 'You are an AI assistant.' and contains a list of 28 numbered items, each with a small icon and some text. The items are:
  1. Why do I get 'SyntaxError: invalid syntax' when using if-else in Python? - Syntax Error
  2. How do I fix 'caught TypeError: cannot read property of undefined' in javascript? - Syntax Error
  3. My SQL query says 'You have an error in your SQL syntax' - what's wrong? - Syntax Error
  4. What's wrong with my Python list comprehension syntax? - Syntax Error
  5. My loop runs forever - how do I fix an infinite loop? - Logic Error
  6. Why does my function return None instead of the expected value? - Logic Error
  7. My array indexing gives wrong results - off by one error help? - Logic Error
  8. Why does my comparison operator (< vs ==) not work as expected? - Logic Error
  9. How can I make my database query run faster? - Optimization
  10. What's the best way to optimize nested loops in python? - Optimization
  11. My function takes too long - how to improve time complexity? - Optimization
  12. How do I reduce memory usage in my Java application? - Optimization
  13. What's the difference between a list and a tuple in python? - Conceptual Question
  14. When should I use recursion vs iteration? - Conceptual Question
  15. How does garbage collection work in Java? - Conceptual Question
  16. What's the difference between REST API and GraphQL? - Conceptual Question
- New Agent**: This pane is titled 'You are a programming question classifier.' and contains a list of 16 numbered items under the heading 'task':
  1. My loop runs forever - how do I fix an infinite loop?
  2. Why does my function return None instead of the expected value?
  3. My array indexing gives wrong results - off by one error help?
  4. Why does my comparison operator (< vs ==) not work as expected?
  5. How can I make my database query run faster?
  6. What's the best way to optimize nested loops in python?
  7. My function takes too long - how to improve time complexity?
  8. How do I reduce memory usage in my Java application?
  9. What's the difference between a list and a tuple in python?
  10. When should I use recursion vs iteration?
  11. How does garbage collection work in Java?
  12. What's the difference between REST API and GraphQL?
  13. How do I fix an infinite loop?
  14. Why does my function return None instead of the expected value?
  15. My array indexing gives wrong results - off by one error help?
  16. Why does my comparison operator (< vs ==) not work as expected?

At the bottom of the application window, there are several status indicators and icons, including 'Status: Online', 'Last Seen: 10 min ago', 'Last Chat: 10 min ago', and 'Last Sync: 10 min ago'.

Programming help chatbot queries

New Agent

You are a programming question classifier.

Task

Topic

Logic Error

Optimization

1. Why do I get 'SyntaxError: invalid syntax' when using if-else in Python? , Syntax Error  
2. How do I fix 'Uncaught TypeError: Cannot read property of undefined' in javascript? , Syntax Error  
3. My SQL query says 'You have an error in your SQL syntax - what's wrong?' , Syntax Error  
4. What's wrong with my Python list comprehension syntax? , Syntax Error  
5. My loop runs forever - how do I fix an infinite loop? , Logic Error  
6. My array filtering gives wrong results - off by one error? , Logic Error  
7. My function doesn't work as expected? , Logic Error  
8. What's the best way to optimize nested loops in python? , Optimization  
9. My function takes too long - how to improve time complexity? , Optimization  
10. What's the difference between a list and a tuple in python? , Conceptual Question  
11. When should I use recursion vs iteration? , Conceptual Question  
12. How does garbage collection work in Java? , Conceptual Question  
13. What's the difference between REST API and GraphQL? , Conceptual Question

Topic

Logic Error

Optimization

1. Why do I get 'SyntaxError: invalid syntax' when using if-else in Python? , Syntax Error  
2. How do I fix 'Uncaught TypeError: Cannot read property of undefined' in javascript? , Syntax Error  
3. My SQL query says 'You have an error in your SQL syntax - what's wrong?' , Syntax Error  
4. What's wrong with my Python list comprehension syntax? , Syntax Error  
5. My loop runs forever - how do I fix an infinite loop? , Logic Error  
6. My array filtering gives wrong results - off by one error? , Logic Error  
7. My function doesn't work as expected? , Logic Error  
8. What's the best way to optimize nested loops in python? , Optimization  
9. My function takes too long - how to improve time complexity? , Optimization  
10. What's the difference between a list and a tuple in python? , Conceptual Question  
11. When should I use recursion vs iteration? , Conceptual Question  
12. How does garbage collection work in Java? , Conceptual Question  
13. What's the difference between REST API and GraphQL? , Conceptual Question

The screenshot shows a Jupyter Notebook cell containing Python code for evaluating zero-shot, one-shot, and few-shot accuracy across three prompting techniques. The code uses a dictionary to map technique names to their respective zero-shot, one-shot, and few-shot accuracy values. It then prints the highest accuracy for each technique and the overall best accuracy.

```
# prompting_technique_evaluation.py

def main():
    for t, (query, expected) in enumerate(TEST_QUERIES):
        print(f"\nQuery: {query} (Expected: {expected})")
        print(f"Zero-Shot: {zero_shot_results[t][0]}")
        print(f"One-Shot: {one_shot_results[t][0]}")
        print(f"Few-Shot: {few_shot_results[t][0]}")

    print("\nZero-Shot Accuracy Results:")
    print(f"Zero-Shot Accuracy: {zero_shot_accuracy}")
    print(f"One-Shot Accuracy: {one_shot_accuracy}")
    print(f"Few-Shot Accuracy: {few_shot_accuracy}")

    print("\nOverall Accuracy Results:")
    print(f"Overall Accuracy: {best_accuracy}")
    print(f"Technique: {best_technique}")

    print(f"\nHighest Accuracy: {max(best_accuracy)} ({best_technique})")

if __name__ == "__main__":
    main()
```

On the right side of the screen, there is a "Programming help chatbot queries" interface. It shows a message from "New Agent" asking about zero-shot evaluation. Below it, there are sections for "Real-world applications" (mentioning zero-shot for entity extraction), "Recommendations" (mentioning zero-shot for efficiency), and "The complete evaluation script and detailed results are saved in `prompting_techniques.ipynb`".

## 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.

i want prompt for vsc code in python

The screenshot shows a Jupyter Notebook interface with several open cells. The left sidebar contains a tree view of files and notebooks, with the current notebook expanded.

**Social Media Post Classifier Evaluation**

```
social_media_post_classifier_evaluation.ipynb
```

**Code Cells:**

```
social_media_post_classifier_evaluation.py
```

Social media Post Classifier - Prompting Techniques Evaluation  
Compares zero-shot, one-shot, and few-shot prompting methods

```
import csv
```

A prompt template for data preparation:  
prompt\_prep = """  
You are an AI assistant.  
Task:  
Create sample social media posts for analysis.  
  
Categories:  
1. Promotion  
2. Complaint  
3. Appreciation  
4. Inquiry  
  
Requirements:  
- Create at least 2 posts per category  
- Posts should include informal and casual language  
- Output the result in a table format with columns:  
Post ID | Post Content | Category  
...  
  
# Load posts and expand categories  
test\_posts = [

Sample social media posts for analysis

```
new_york_restaurant_promotion.csv
```

prompt\_prep = """  
You are an AI assistant.  
Task:  
Create sample social media posts for analysis.  
  
Categories:  
1. Promotion  
2. Complaint  
3. Appreciation  
4. Inquiry  
  
Requirements:  
- Create at least 2 posts per category  
- Posts should include informal and casual language  
- Output the result in a table format with columns:  
Post ID | Post Content | Category  
...  
  
# Load posts and expand categories  
test\_posts = [

Checking for coding errors and running the code

```
! python social_media_post_classifier_evaluation.py
```

```
# File Edit Selection View Cell Help
social_media_posts_classifier_main.py X
social_media_posts_classifier_main.py 1 of 10 1 of 2 files > Details Load file
social_media_posts_classifier_main.py
  - output only the category name
  - do not explain your answer

Post:
  "Why is this app crashing again? So annoying!"

prompt_low_shot = """
You are a social media post classifier.

task:
Classify the following social media post into one of the categories:
- Promotional
- Complaint
- Appreciation
- Inquiry

Example:
Post: "Love this new update! Works perfectly!"
Category: Appreciation

Now classify the following post...

Notes:
- Output only the category name
- do not explain your answer

Post:
  "When will the next sale start?"

prompt_low_shot = """
You are a social media post classifier.

task:
Classify the following social media post into one of the categories:
- Promotional
- Complaint
- Appreciation
- Inquiry

Example:
Post: "Check out our latest discounts, grab them now!"
Category: Promotional

Post: "This service is terrible, totally disappointed."
Category: Complaint

Post: "Amazing support team, thanks a lot!"
Category: Appreciation

Post: "Can someone tell me how to reset my password?"
Category: Inquiry

Now classify the following post...

Notes:
- Output only the category name
- do not explain your answer

Post:
  "Not my order today, super happy with the quality!"
```

Sample social media posts for analysis  
None → Add file

prompt\_low\_shot = """  
You are a social media post classifier.

The output now includes:  
• prompt\_low\_shot — template for generating sample posts  
• prompt\_low\_shot — one-shot classification example  
• prompt\_low\_shot — one-shot classification example

The example post "When will the next sale start?" will be classified as Inquiry because it contains "when will".

All changes are complete and there are no pending errors. The script is ready to use.

Load All Keep All social\_media\_posts.dat social\_media\_posts\_low\_shot.dat

```
# File Edit Selection View Cell Help
social_media_posts_classifier_main.py X
social_media_posts_classifier_main.py 1 of 17 1 of 2 files > Details Load file
social_media_posts_classifier_main.py
  - output only the category name
  - do not explain your answer

Post:
  "When will the next sale start?"

prompt_low_shot = """
You are a social media post classifier.

task:
Classify the following social media post into one of the categories:
- Promotional
- Complaint
- Appreciation
- Inquiry

Example:
Post: "Check out our latest discounts, grab them now!"
Category: Promotional

Post: "This service is terrible, totally disappointed."
Category: Complaint

Post: "Amazing support team, thanks a lot!"
Category: Appreciation

Post: "Can someone tell me how to reset my password?"
Category: Inquiry

Now classify the following post...

Notes:
- Output only the category name
- do not explain your answer

Post:
  "Not my order today, super happy with the quality!"
```

Sample social media posts for analysis  
None → Add file

prompt\_low\_shot = """  
You are a social media post classifier.

The output now includes:  
• prompt\_low\_shot — template for generating sample posts  
• prompt\_low\_shot — one-shot classification example  
• prompt\_low\_shot — one-shot classification example  
• prompt\_low\_shot — one-shot classification example

The example post "Not my order today, super happy with the quality!" will be classified as Appreciation because it contains "happy" and "quality".

All changes are complete with no pending errors. The script is ready to use.

Load All Keep All social\_media\_posts.dat social\_media\_posts\_low\_shot.dat

