TASK 1 – Rating Prediction via Prompting

1. Objective

The goal of this task is to classify Yelp reviews into star ratings (1–5) using different prompting strategies and evaluate: Accuracy (Predicted vs Actual Stars), JSON Validity ,Reliability (Does the model produce consistent results across repeated runs?), Consistency (Variance of predictions across runs). We implemented three prompting strategies: Zero-Shot, Few-Shot, and Rubric-Based. All prompts were tested on a sample of 200 Yelp reviews.

2. Dataset Overview

Used: yelp.csv
Selected Fields: 'text', 'stars'
Sampled: 200 reviews randomly
Cleaned: Removed missing values

3. API Setup (OpenRouter)

I used the OpenRouter API with model: meta-llama/llama-3.1-8b-instruct
Temperature = 0.0 for deterministic output
JSON extracted using regex parsing for robustness

4. Prompt Designs (Three Approaches)

4.1 Zero-Shot Prompt

Design Idea: Model gets no examples; must infer rating from raw text.

Prompt:
You are a helpful assistant. Read the review below and assign a rating 1–5.
Return ONLY:
{
  "predicted_stars": <number>
}
Review:
"{review_text}"

Why this version? Tests model's base behavior, no examples or guidance, fastest execution.

4.2 Few-Shot Prompt

Design Idea: We give two labeled examples before classification.

Prompt:
You classify reviews into 1–5 stars.
Return ONLY JSON:
{
  "predicted_stars": <number>,
  "explanation": "short reason"
}
Examples:
Review: "Amazing food!"
Response: {"predicted_stars": 5, "explanation": "very positive"}
Review: "Food was cold and tasteless."
Response: {"predicted_stars": 2, "explanation": "negative food quality"}
Now classify: "{review_text}"

Why this version? Helps model understand positive/negative patterns, encourages stable formatting, improves reasoning.

4.3 Rubric-Based Prompt

Design Idea: We give a 5-level sentiment rubric.

Prompt:
Rate the review using this rubric:
1: very negative
2: negative
3: mixed
4: positive
5: very positive
Return ONLY JSON:
{
  "predicted_stars": <number>,
  "explanation": "short reason"
}
Review:
"{review_text}"

Why this version? Ensures uniform criteria, reduces ambiguity, helps prevent extreme biases.

5. Evaluation Metrics

For each prompt style, I computed:
- Accuracy: (true stars == predicted stars)
- JSON Validity: (valid JSON returned?)
- Reliability / Consistency: repeated predictions on 10 reviews, checked for stability

6. Final Results

6.1 Accuracy & JSON Validity

Prompt Type	Accuracy	JSON Validity
Zero-Shot	0.660	1.0
Few-Shot	0.610	1.0
Rubric	0.605	1.0

6.2 Reliability & Consistency

Prompt Type	Consistency Score
Zero-Shot	1.0
Few-Shot	1.0
Rubric	1.0

7. Summary & Discussion

Zero-Shot Prompt = Most Accurate, Fastest, Highest classification accuracy
Few-Shot: Helps with reasoning, Slightly biased accuracy, Useful for explanations
Rubric Prompt: Best for structured reasoning, Lower accuracy, Highly interpretable, Consistent guidelines

8. Final Conclusion

Prompt engineering significantly impacts model accuracy. Small changes in phrasing affect prediction outcomes.
Zero-shot prompting performed best for this dataset. All prompt types produced 100% valid JSON.
Reliability testing shows stable and deterministic behavior.
Project requirements fulfilled: 3 prompting methods, explanations, 200 review evaluation, JSON validity, reliability & consistency analysis, final comparison table & discussion.