International Journal of Learning, Teaching and Educational Research
Vol. 25, No. 8, pp. 124-155, August 2026
https://doi.org/10.26803/ijlter.25.8.5
Received May 19, 2026; Revised Jul 13, 2026; Accepted Jul 15, 2026

Evaluating Command-Line Interface-Based Agentic Large Language Model Coding Tools for Non-English Thematic Analysis: A 3 x 3 x 3 Factorial Study of Models, Prompts and Stochastic Variability

Chawalit Chanintonsongkhla and Thananya Chongcharoenkit*
School of Dentistry, University of Phayao
Phayao, Thailand

Abstract. Command-line interface (CLI)-based agentic coding tools enable large language models (LLMs) to autonomously read local data files, write and execute analysis scripts, and generate outputs, but whether they can reliably perform thematic analysis of non-English educational data remains understudied. This 3 × 3 × 3 factorial experiment compared three frontier LLMs (Claude Opus 4.6, GPT-5.3-Codex, and Gemini 3 Pro Preview) across three prompt tiers (detailed, structured, and exploratory) with three repetitions per condition, yielding 27 runs. Each run processed 89 anonymized student evaluations (Likert ratings and open-ended Thai comments) from a Year 3 preventive dentistry course over three academic years. Outputs were evaluated against 10 required items covering statistical computation, thematic analysis, and visualization, with the generated scripts inspected to classify categorization as direct reading or keyword matching. Statistical computation and overview charts each passed 93% (25/27), and detail charts passed in all 27 runs, with failures confined to exploratory prompts. Thematic reading was the primary difference. Opus and Gemini read Thai directly while GPT usually generated keyword-matching scripts. Structured prompts achieved the highest thematic reading pass rate at 78% (7/9), followed by detailed at 67% (6/9) and exploratory at 33% (3/9). Identical configurations produced different outputs across repetitions, with theme counts ranging from 4 to 31. CLI-based agentic coding tools can perform thematic analyses of Thai-language educational feedback, but natural language processing output reliability depends on model selection, prompt specificity, and stochastic variability. Multiple independent runs are recommended to assess consistency.

Keywords: Agentic coding tools; thematic analysis; large language models; prompt engineering; natural language processing

Citation: Chanintonsongkhla, C., & Chongcharoenkit, T. (2026). Evaluating Command-Line Interface-Based Agentic Large Language Model Coding Tools for Non-English Thematic Analysis: A 3 x 3 x 3 Factorial Study of Models, Prompts and Stochastic Variability. *International Journal of Learning, Teaching and Educational Research*, *25*(8), 124–155. https://doi.org/10.26803/ijlter.25.8.5

*Corresponding author: Thananya Chongcharoenkit; *thananya.ch@up.ac.th*

1. Introduction

Health professions curricula have increasingly adopted vertical integration, an approach that blends basic science instruction with clinical practice from the earliest training years and gradually increases clinical exposure as students advance (Wijnen-Meijer et al., 2020). The Year 3 preventive dentistry course examined in this study combines lectures, laboratory exercises, and preclinical practice in the year preceding clinical rotations. Students provide overall satisfaction ratings on Likert scales alongside open-ended Thai-language comments freely describing what they valued and what they found challenging.

The quantitative ratings are straightforward to aggregate, but analyzing the qualitative free-text responses requires more effort. Braun and Clarke's thematic analysis framework (Braun & Clarke, 2006; Braun & Clarke, 2019) describes this process as requiring iterative familiarization, coding, and theme development. For a course that integrates diverse learning activities and collects evaluations across multiple cohorts, this iterative process can be time-consuming.

1.1 Literature review and conceptual framework

Reviews of AI applications in dental and medical education have identified automated assessment and feedback generation as areas where large language models (LLMs) show promise (Lucas et al., 2024; El-Hakim et al., 2025). Kasneci et al. (2023) reviewed these opportunities more broadly, including personalized learning and adaptive feedback. Empirical studies have tested these capabilities, specifically for qualitative analysis. LLMs such as GPT-4 and GPT-3.5 can perform multi-label classification of 2,500 course comments with Jaccard similarity approaching that of human coders (Parker et al., 2025).

LLM-assisted deductive coding can also reach stable response rates across repeated iterations (Tai et al., 2024), and multi-agent LLM frameworks have processed thousands of medical education feedback responses for sentiment and topic analysis (Cai et al., 2025). Human-LLM collaborative thematic coding has achieved substantial inter-rater agreement (Cohen's kappa 0.61-0.65) in deductive categorization tasks (Wen et al., 2026). Thematic analysis has been applied in educational research to evaluate assessment tools in higher education (Jita & Thaanyane, 2025), and to examine the role of ChatGPT in the student learning experience (Almulla & Ali, 2024).

Large language models can perform new tasks without task-specific training data, a capacity referred to as zero-shot learning. Brown et al. (2020) demonstrated that scaling a language model to 175 billion parameters (GPT-3) produced strong zero-shot and few-shot performance across diverse natural language processing (NLP) tasks including translation, question-answering, and text classification. Recent studies have extended this paradigm to information extraction, showing that LLMs can extract structured information from unstructured text in both English and Chinese without labeled training examples (Wei et al., 2023). In this study, LLM-based reading describes this zero-shot text interpretation process, in which a model reads and categorizes free-text responses through general language understanding.

Prior studies, however, have largely relied on hosted web or application programming interface (API) workflows where data are sent to an external service that cannot directly access local file systems, installed libraries, or computing environments. A newer category, command-line interface (CLI)-based agentic coding tools, (also termed coding agents) (Denny et al., 2024; Jimenez et al., 2024), operates differently. These tools combine two capabilities. As a code executor, the tool invokes installed programs and libraries to read data files, compute statistics, and generate figures, producing auditable intermediate outputs at each step. As a language model, CLI-based tools can read free-text comments directly to identify thematic categories, a task that conventionally requires either word-matching algorithms or manual human coding.

Benchmarks such as SWE-bench have evaluated frontier LLMs on real-world software engineering tasks requiring autonomous code-execution capability (Jimenez et al., 2024). LLM-assisted code generation has grown rapidly as a research area (Jiang et al., 2026), and these tools are now increasingly used in both educational and professional contexts (Denny et al., 2024). Prior studies have evaluated LLMs for educational feedback analysis through hosted web or API workflows (Parker et al., 2025) or multi-agent pipelines (Cai et al., 2025), but whether CLI-based agentic coding tools can perform end-to-end educational data analysis locally, including qualitative analysis of non-English text, remains unexplored, a gap that is both empirical and methodological.

Current large language models generate text by sampling from a probability distribution over the vocabulary. This process is inherently stochastic; identical inputs can yield different outputs, and this variation compounds over long generations. This variability has been documented in sentiment classification (Herrera-Poyatos et al., 2025) and in deductive qualitative coding, where response rates required repeated iterations to stabilize (Tai et al., 2024). Languages such as Thai are written continuously without spaces between words, requiring specialized tokenization for word boundary identification (Arreerard et al., 2022; Phatthiyaphaibun et al., 2023).

Without proper segmentation, naive substring or regex matching produces false positives from matching within longer words and false negatives from missing semantically relevant phrases. Multilingual LLMs exhibit reduced fidelity on low-resource languages, partly attributed to tokenizer bias and English-centric pretraining (Qin et al., 2025), factors that can compound the variability already present in stochastic generation.

1.2 Objectives, significance and research questions

This study design combines reflexive thematic analysis, in which qualitative coding is interpretive and several valid readings of the same data are possible (Braun & Clarke, 2019), with zero-shot language understanding, in which a model categorizes free text through general language capability without task-specific training (Brown et al., 2020). The primary objective is to evaluate whether CLI-based agentic coding tools can reliably perform end-to-end analysis of non-

English course-evaluation data, and to characterize how model choice, prompt specificity, and repetition affect that reliability.

Evidence on how such tools handle qualitative analysis of non-English educational feedback is lacking, and this study provides that evidence under controlled, repeated conditions as a preliminary evaluation of CLI-based agentic coding tools through a factorial experiment comparing three frontier LLMs, three levels of prompt specificity, and three independent repetitions, using Thai-language dental course evaluation data. The contribution of this study is to demonstrate the capability of CLI-based agentic coding tools to assist with thematic analysis of non-English educational feedback, and to characterize per-run variation in outputs across model choice, prompt specificity, and repetition. The study addressed three research questions:

1. Do CLI-based agentic coding tools reliably perform end-to-end thematic analysis of Thai-language student evaluations from a preclinical dental course?
2. How do model architecture and prompt specificity affect task compliance?
3. What variability does arise across repeated runs under identical configurations?

2. Methodology

2.1 Data collection and analysis

This study employed a fully crossed 3 (Model) × 3 (Prompt Specificity) × 3 (Repetition) factorial design, yielding 27 total runs. The independent variables were the model, the prompt specificity tier, and the repetition. Each run was scored against a predefined list of required tasks, and its generated scripts and outputs were inspected to determine how it categorized the open-ended feedback. All the runs used the same input data and were executed on a single date. Repetition was treated as an analytical dimension rather than simple replication, allowing assessment of within-condition variability. The study protocol was approved by the Human Research Ethics Committee (HREC-UP-HSST 1.1/038/69).

2.2 Input data

The input data consisted of 89 anonymized student evaluations from a Year 3 preventive dentistry course at a Thai dental school. The course covered the final preclinical year before students entered Year 4 clinical practice, combining lectures with hands-on procedures such as paired oral examinations, recording dental indices, applying fluoride and dental sealants, and assessing and managing dental caries risk. Evaluations were collected across three academic years, 2023 (n = 29), 2024 (n = 30), and 2025 (n = 30).

Course evaluation forms recorded academic years as Thai Buddhist Era years 2566, 2567, and 2568. These correspond to 2023, 2024, and 2025, used throughout the tables and figures for international readability. The three academic years represent separate student cohorts rather than repeated measurements of the same students, and were included to test whether the tools preserved year-by-year aggregation and trend reporting. All 27 analysis runs were executed on a

single date (Section 2.3). The academic years denote only when the original evaluations were collected.

Each evaluation contained an anonymized student identifier (S001 through S089), the academic year, and three Likert-scale items rated on a 1 to 5 scale measuring satisfaction with the teaching process, satisfaction with learning, and dissatisfaction where 1 indicates no dissatisfaction. A separate open-ended Thai-language text field asked students to describe what they liked, disliked, and would suggest about the course. This institutional evaluation form provided authentic, routinely collected feedback, and its open-ended field allowed students to raise concerns not anticipated by the fixed-response items. Of the 89 evaluations, 17 had null text fields, 7 contained only a dash or the word ไม่มี (none), and the remaining 65 contained meaningful text feedback.

2.3 Models and prompts

Three frontier-class LLMs were selected from major commercial providers available as of February 2026 as Claude Opus 4.6 (claude-opus-4-6, Claude CLI v2.1.34), GPT-5.3-Codex (gpt-5.3-codex, Codex CLI v0.98.0), and Gemini 3 Pro Preview (gemini-3-pro-preview, Gemini CLI v0.27.3). All the experiments were executed on a single date (February 7, 2026) to minimize the risk of mid-experiment model updates. No dated snapshot identifiers were available for any model. Each run used default generation settings without modifying sampling parameters. The study was not sponsored by any model provider.

Each tool had access to Python libraries for data analysis and visualization (pandas, openpyxl, matplotlib, numpy) and could autonomously read input files, write and execute scripts, and refine outputs through successive iterations within a single session. Figure 1 shows a representative CLI interface using Claude Code, with Codex and Gemini sharing the same terminal-based interaction model. The panel illustrated the tool (Claude Code) responding to a brief example prompt that asked it to read the satisfaction dataset and report mean scores per item by academic year.

Three prompt tiers were designed at different levels of specificity (Appendix 3). Prompt 1 (detailed procedural, 372 lines) prescribed every analysis step and output format. Its four-step thematic procedure had the model read every non-null feedback entry, identify recurring themes, name each category, and assign students to categories. The model was explicitly instructed to discover categories from the text rather than use pre-defined ones. Prompt 2 (structured-concise, 101 lines) specified required outputs and content expectations without procedural instructions, stating that thematic categories should be discovered by reading all feedback rather than using pre-defined categories. Prompt 3 (exploratory, 21 lines) listed four analysis tasks (statistical analysis, thematic categorization, summary, and visualizations) with the instruction to "discover categories from the data" but without specifying the analytical procedure or prohibiting pre-defined categories.

All three prompts directed the model to save Python scripts, produce markdown reports, and generate chart figures. Prompts 1 and 2 specifically required visualizations of mean satisfaction scores broken down by academic year, with per-item Likert rating frequency distributions for each year.

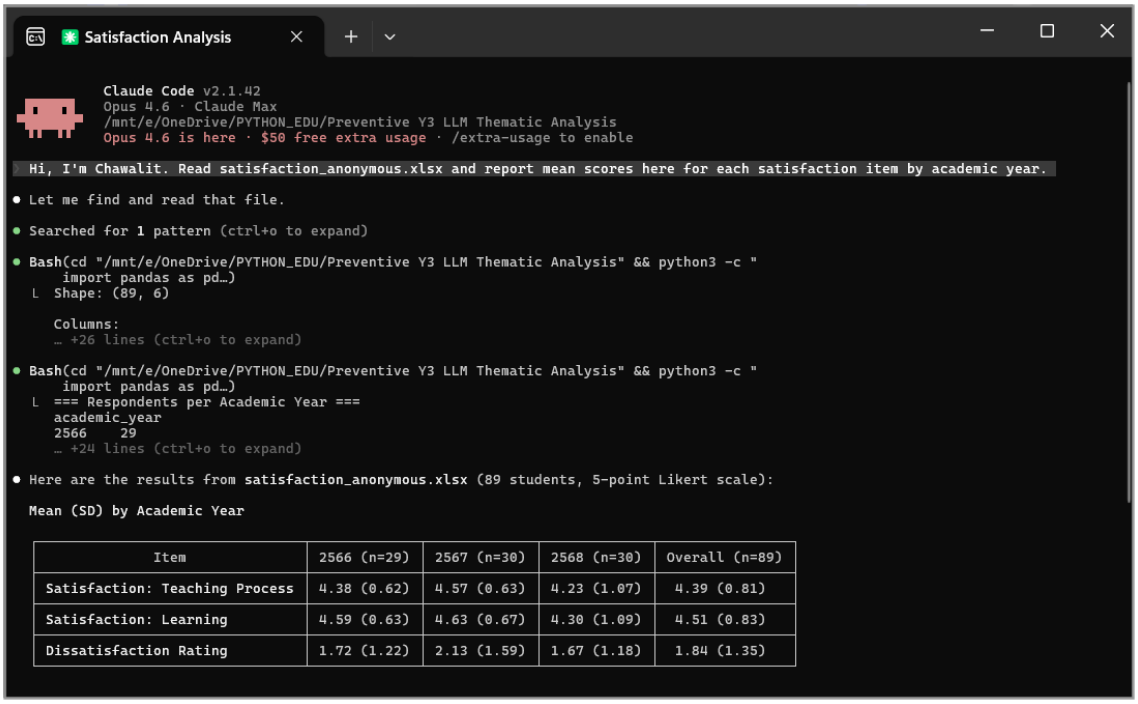


Figure 1: Example command-line interface (CLI) appearance of an agentic coding tool (Claude Code, Anthropic)

2.4 Evaluation of required tasks (R1-R3)

Each run's output files (scripts, reports, and figures) were evaluated against 10 required items spanning three domains. Statistical computation (R1) verified that each run's computed overall means matched values derived from the input data for satisfaction with the teaching process (R1.1, expected 4.39), satisfaction with learning (R1.2, expected 4.51), and dissatisfaction (R1.3, expected 1.84).

The thematic analysis items (R2) assessed whether the model categorized feedback into themes (R2.1), separated positive and negative sentiment (R2.2), presented themes as cross-tabulations by academic year (R2.3), discussed year-over-year trends across academic years 2023-2025 (R2.4), and used LLM-based reading rather than keyword-matching scripts for thematic categorization (R2.5). The year-by-year items (R2.3 and R2.4) tested whether the tools preserved the cohort structure and reported trends across the three academic years rather than collapsing the feedback into a single aggregate. Visualization items (R3) confirmed that satisfaction bar charts (R3.1) and per-metric detail charts with rating frequency distributions (R3.2) were generated.

Evaluating LLM-based thematic reading (R2.5) required a binary judgment about how each model produced its thematic categorization, distinct from R2.1, which only verified whether categorization occurred. For each of the 27 runs, a CLI-

based coding agent (Claude Opus 4.6) first located the relevant files within each run's output set. The model's outputs (Python scripts and execution logs) were then inspected, and each assignment was confirmed against the source files to reach consensus on whether the model (a) read each student's Thai text directly and assigned a category based on semantic interpretation, or (b) used a non-semantic approach, such as generating keyword lists, regex patterns, or dictionary lookups to match substrings in the text.

A run passed R2.5 only under (a), including cases in which a model stored its directly read student-to-theme assignments as fixed lists. A run failed under (b), where categorization relied on hardcoded keyword lists, regular expressions, dictionary lookups, or substring tests such as `any(k in text for k in keywords)`. Runs combining both approaches were classified by the dominant mechanism.

2.5 Theme consolidation

The feedback categories produced across all 27 runs were consolidated and verified into canonical categories by examining every unique theme label against the original student responses, grouping semantically equivalent labels under a common category, and keeping feedback that expressed a distinct facet or a notably different opinion as its own category. Theme labels that did not fit any substantive category were assigned to an uncategorized group. Table 1 shows representative translated examples for selected canonical categories, with all 17 listed in Appendix 1. Theme counts per canonical category were then compared across model, prompt, and repetition conditions.

Table 1: Representative student responses, accepted and translated from Thai, for selected positive and negative feedback categories

Canonical category	Representative student response (translated)
C01 Hands-on Practice	"What I liked was getting to do real hands-on work." (S001)
C02 Teaching Staff Quality	"The instructors taught in a way that was easy to understand and did not put too much pressure on students doing the practical for the first time." (S025)
C03 Learning & Understanding	"…it helped me picture what I had studied and understand it better." (S011)
C04 Professional Motivation	"…this course reminded me that I still love the dental profession, and I have to thank this lab for rekindling my drive." (S042)
C10 Time & Scheduling	"…the downside was too little time, crammed into the end of the semester…" (S005)
C11 Pre-practice Readiness	"…I felt unprepared, for example with the instruments, since I had never actually used them and did not know how to handle the tools." (S046)
C12 Insufficient Supervision	"There were too few supervising instructors, which made it rather chaotic and slow to get questions answered or ask for help." (S002)
C13 Grading Pressure	"…I would prefer no point deductions, only warnings, because when you ask and try something and make a mistake you lose points, which makes you afraid to ask or try." (S018)

2.6 Additional model outputs

All three models also produced analyses not prescribed in the prompts, such as correlation matrices, statistical tests, and additional visualizations. These were recorded as additional tasks to document the extent of extra analyses across runs.

3. Results

3.1 Required and additional tasks

Each agentic analysis run was completed in 2 to 9 minutes (mean 5.00 minutes, range 2.40 to 9.43 across 27 runs). GPT and Gemini were comparable on average (4.14 and 4.26 minutes), and Opus ran longest (mean 6.59 minutes). Required items (R1-R3) achieved high pass rates under the two instructed prompts (Prompts 1 and 2), with compliance declining under the exploratory Prompt 3 (Figure 2). All three models also produced additional analyses beyond the basic commands of the prompts, with Prompt 3 runs producing the widest range.

	Opus			GPT			Gemini		
	P1	P2	P3	P1	P2	P3	P1	P2	P3
R1 Satisfaction means	3	3	3	3	3	3	3	3	1
R2.1 Categorize into themes	3	3	3	3	3	3	3	3	3
R2.2 Separate likes vs dislikes	3	3	3	3	3	0	3	3	2
R2.3 Cross-tabulation tables by year	3	3	3	3	3	2	1	3	0
R2.4 Year trend analysis	3	3	3	3	3	3	3	3	3
R2.5 LLM-based thematic analysis	3	3	1	0	1	0	3	3	2
R3.1 Satisfaction bar charts	3	3	3	3	3	2	3	3	2
R3.2 Per-satisfaction detail charts	3	3	3	3	3	3	3	3	3
IQR / quartile reporting	0	0	2	0	0	2	0	0	3
Correlation analysis	0	0	3	0	0	0	0	0	0
Significance testing	0	0	2	0	0	0	0	0	0
Pairwise post-hoc tests	0	0	2	0	0	0	0	0	0
Effect size reporting	0	0	1	0	0	0	0	0	0
Box plot	0	0	3	0	0	2	0	0	3
Trend line charts	0	0	2	0	0	3	0	0	2
Stacked distribution charts	0	0	3	0	0	2	0	0	0

Figure 2: Task completion by model and prompt (P1-P3), shown as passes out of three repetitions per cell for 10 required tasks (R1-R3) and additional evaluation items

3.2 Statistical computation and visualization (R1, R3)

Under the two instructed prompts (18 runs), all three models achieved 100% (18/18) pass rates on statistical computation (R1.1-R1.3) and both visualization items (R3.1, R3.2). Prompt 1 and Prompt 2 produced consistent multi-panel bar charts across all the models (Figure 3).

Compliance dropped under the exploratory prompt (P3). Statistical computation (R1.1-R1.3) fell to 78% (7/9), with Gemini not computing the all-years combined means in two runs. Satisfaction chart generation (R3.1) also fell to 78% (7/9) because Gemini and GPT did not generate the overview satisfaction chart in one run, even though the prompt specified both overview and per-metric detail charts. Detail charts (R3.2) maintained 100% (9/9). The exploratory prompt also produced the greatest visual variation (Figure 3). All the charts labeled the year axis with Buddhist-Era values, and most used English text for titles and axis labels. Under the exploratory prompt, GPT wrote its chart labels in Thai but the Thai text did not render in the generated figure, appearing as empty boxes rather than legible characters because the default font lacked Thai glyphs.

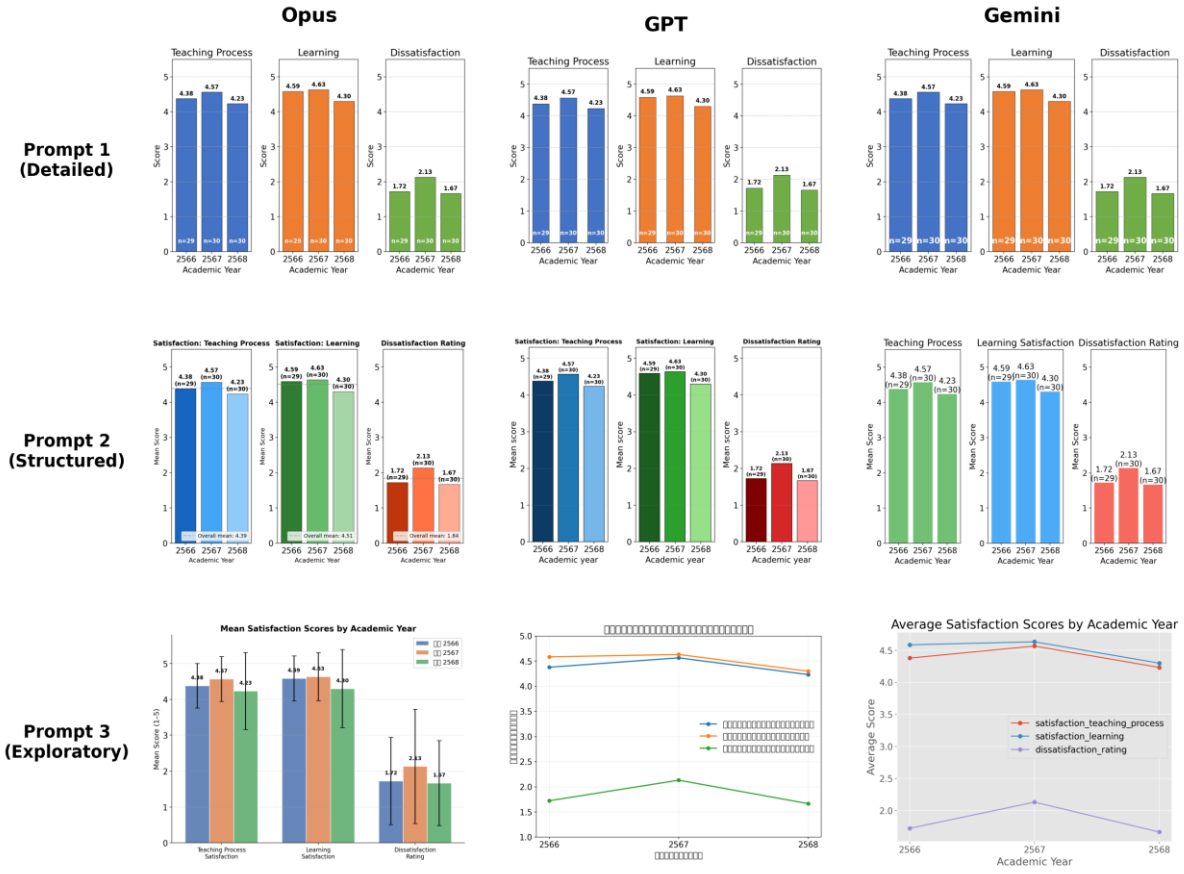


Figure 3: One representative satisfaction chart (R3.1) generated by each model under each prompt, shown as the model's own output. The GPT Prompt 3 panel includes a Thai text-rendering issue

3.3 Thematic analysis (R2)

Theme categorization (R2.1) passed in all 27 runs, with sentiment separation (R2.2) passing at 85% (23/27) and cross-tabulation by academic year (R2.3) at 78% (21/27). Both items fell to 56% (5/9) under Prompt 3. Passing runs produced frequency tables organized by theme and academic year. Table 2 shows one such example, drawn from Opus Prompt 1 runs that generated 12 to 13 bilingual categories using per-year student counts. Gemini failed cross-tabulation in two

Prompt 1 runs, producing theme narratives without year-by-year frequency tables.

Table 2: Representative cross-tabulation of negative feedback themes by academic year (Opus, Prompt 1, Run 1)

Category	2023	2024	2025	Total
Insufficient Time and End-of-Semester Scheduling	5	0	0	5
Insufficient Instructor-to-Student Ratio	4	0	0	4
Lack of Preparedness and Difficulty Understanding	1	3	2	6
Pressure and Point Deduction System	2	0	0	2
Content Sequencing and Teaching Materials	0	3	0	3
Desire for More Practice and Instructor Verification	0	0	2	2

One student may appear in multiple categories. The original table was generated with both Thai and English text. Thai text was removed for presentation clarity

Year-over-year trend discussion (R2.4) passed in all 27 runs, though trend narratives varied in specificity across models. Opus and Gemini cited year-specific frequency shifts. Opus Prompt 1 Run 1 reported time-related complaints in five students in 2023, with the same theme absent in 2024 and 2025 (Table 2). A Gemini trend narrative similarly attributed time-and-scheduling concerns to 2023, equipment readiness to 2024, and reduced negative comments to 2025. GPT's trend sections described general patterns across years without citing year-specific frequency changes, with Gemini Prompt 3 runs also producing recommendation sections linking specific curricular suggestions to individual anonymized student responses.

LLM-based thematic reading (R2.5) passed at 59% (16/27). By model, Gemini passed at 89% (8/9), Opus at 78% (7/9), and GPT at 11% (1/9) (Figure 2). In passing runs, Opus and Gemini read each student's Thai text and assigned it to thematic categories.

GPT generated scripts containing hardcoded keyword dictionaries of Thai terms per category, classifying feedback through substring matching (any(k in text for k in keywords)). GPT's single pass (Prompt 2, Run 1) stored LLM-derived student-to-theme assignments as fixed Python lists ('hands_on': ['S001', 'S004', ...]) rather than applying keyword-matching functions. All Opus and Gemini failures occurred under Prompt 3.

By prompt tier, structured prompts (Prompt 2) achieved the highest LLM-based thematic reading (R2.5) pass rate at 78% (7/9), followed by detailed prompts (Prompt 1) at 67% (6/9). Opus and Gemini passed at identical rates under both tiers (3/3 each). GPT passed once, under Prompt 2. For exploratory prompts (Prompt 3), the pass rate fell to 33% (3/9), with Gemini passing in 2 of 3 runs, Opus in 1 of 3, and GPT in 0 of 3.

3.4 Theme counts and categories

Theme counts varied substantially across models (Table 3). Opus produced the most stable output (mean 12.0, range 9-14). GPT ranged from 6 to 13 themes (mean 9.8). Gemini showed the widest range (mean 9.4, range 4-31). By prompt tier, Prompt 2 produced the highest theme counts across all three models (Opus 13.3, GPT 12.3, Gemini 15.7), and Prompt 3 consistently produced fewer themes (per-prompt averages of 10.3, 7.3, and 5.0 for Opus, GPT, and Gemini).

Table 3: Theme counts by model and prompt tier, shown as mean (range) across 3 runs per cell

	Prompt 1	Prompt 2	Prompt 3	All
Opus	12.3 (12-13)	13.3 (12-14)	10.3 (9-12)	12.0 (9-14)
GPT	9.7 (9-11)	12.3 (11-13)	7.3 (6-8)	9.8 (6-13)
Gemini	7.7 (6-9)	15.7 (7-31)	5.0 (4-6)	9.4 (4-31)
All	9.9 (6-13)	13.8 (7-31)	7.6 (4-12)	10.4 (4-31)

Gemini showed the greatest variation in thematic granularity. In Prompt 2 Run 1 (31 themes), the model split instructor-related feedback into "Teacher Quality" (10 mentions) and "Teaching Technique" (1), sub-categories that the other models and Gemini's own remaining runs consistently reported as a single theme similar to "Teaching Staff Quality". Conversely, in Prompt 3 Run 3 (4 themes), Gemini collapsed all positive feedback into a single "Appreciation for Practical Experience" category and all negative feedback into 3 broad categories. Categories such as teaching staff quality and assessment pressure, present in most other runs, were absent.

After mapping each run's themes to the 17 canonical categories (Figure 4), Opus detected a mean of 11.7 categories per run (range 7 to 14), GPT 9.0 (range 5 to 13), and Gemini 7.7 (range 4 to 17). The complete per-run counts are detailed in Appendix 2. The resulting frequency distribution showed that "Hands-on Practice" and "Time & Scheduling" were detected in all 27 runs as the most frequent positive and negative categories. "Teaching Staff Quality" was next among positive categories at 26 of 27 runs. Several categories appeared at different rates across the models. "Positive Atmosphere" appeared in all 9 Opus runs but only in 2 GPT and 2 Gemini runs. Within identical model-prompt conditions, the three repetitions produced different category counts and student-to-category assignments.

	Opus			GPT			Gemini		
	P1	P2	P3	P1	P2	P3	P1	P2	P3
C01 Hands-on Practice	40–47	40–48	47–52	42–54	48–54	43–46	52–56	36–54	47–53
C02 Teaching Staff Quality	11–12	12–13	13–19	9–19	14–19	20–24	13–15	10–18	12–24*
C03 Learning & Understanding	10*	8*	23–26*	17–18	11–28	23*	13*	2–9*	–
C04 Prof. Motivation	–	6*	7–20*	15–18	17*	13–53*	12*	3–13*	–
C05 Positive Atmosphere	8–9	9–11	8–15	–	9–12*	–	–	6*	12*
C06 Clinical Setting	11–12	13–18	–	–	15*	–	–	8–16*	–
C07 Specific Clinical Skills	14*	6–8	6*	–	8*	–	–	7*	–
C08 Knowledge Integration	7–8*	6–9	–	–	–	7*	12*	3*	–
C09 Course Organization	3–4	3*	11*	–	2*	–	5*	1*	–
C10 Time & Scheduling	5	5	4–7	7–9	5–7	13–14	5–6	3–8	6–13
C11 Pre-practice Readiness	5–6	4–7	10*	4–12	7–15	7*	4–10*	2–6*	9*
C12 Insufficient Supervision	4	4	2–4	4–5	3–4	5–9*	4–12	3–4	3–5*
C13 Grading Pressure	2	2	1–4	4	2–4	7–11*	2	1–2	–
C14 Content Clarity	3*	3*	7*	–	–	12*	6*	2–9	–
C15 Desire for More Practice	1–2	2*	5*	3–5	1–3	–	–	2*	–
C16 Teaching Materials	3–4*	3–4*	–	–	3*	–	–	2*	2–3*
C17 Lecture-Lab Sequencing	–	2–3*	5*	3*	3*	–	–	1*	–

Figure 4: Count ranges for 17 canonical feedback categories (positive and negative, separated by a horizontal line) by model and prompt. Each cell gives the count range across the repetitions that detected the category. A gray cell with an asterisk marks a category detected in only one or two of the three repetitions

Among the category and model-prompt combinations, a category was detected on average in all three repetitions 51% of the time, and in only one or two repetitions otherwise. This all-three detection rate was 57% for Opus, 57% for GPT, and 37% for Gemini. The most frequently reported categories were the most consistent. "Hands-on Practice" and "Time & Scheduling" were detected in all three repetitions for every condition, while lower-frequency categories such as "Teaching Materials" and "Lecture-Lab Sequencing" were not detected in all three repetitions for any condition.

4. Discussion

4.1 End-to-end feasibility

CLI-based agentic coding tools combine two distinct capabilities, functioning as code executors that write and run scripts using locally installed libraries, and as language models that read free-text comments directly to perform qualitative categorization without programmatic mediation. Agentic AI systems have been applied to scientific research tasks such as hypothesis generation and experiment design (Gridach et al., 2025), and LLM-based data agents have addressed complex analytical problems with minimal human intervention (Sun et al., 2025). This

study evaluated whether CLI-based agentic coding tools can reliably perform both capabilities for educational data analysis.

To the best of our knowledge, this study is among the few to address this question in an educational context. Twenty-seven runs were conducted across different models and prompt specificities. Each run produced complete analytical pipelines without human intervention, and the individual runs were completed in 2 to 9 minutes. Statistical computation (R1) and overview chart generation (R3.1) each achieved 93% (25/27), with failures confined to the exploratory prompt (Prompt 3), which listed tasks without specifying procedures or required output formats. Detail charts (R3.2) passed in all 27 runs.

These results mirror the code-execution capability common to agentic coding tools. The primary difference was not statistical computation or visualization but how the models performed thematic analysis (R2), specifically whether they applied language-reading capability to interpret Thai text directly or used a non-semantic approach, such as generating keyword lists, regex patterns, or dictionary lookups to match substrings.

4.2 Model capability and prompt specificity

Text categorization has evolved from rule-based keyword matching through machine learning classifiers requiring manual feature engineering, and deep learning models that encode features and classify text automatically (Li et al., 2022), to direct classification by large language models without task-specific training. Parker et al. (2025) reported that GPT-4 achieved human-level agreement (Jaccard similarity 80.60% vs. human average 81.24%) in zero-shot classification of 2,500 English course comments, and Wen et al. (2026) achieved substantial inter-rater agreement (Cohen's kappa 0.61-0.65) in human-LLM collaborative thematic coding.

Agentic coding tools are powered by such models, giving them the inherent capability to read and interpret text directly, but their code-execution capability also allows them to write programmatic classification scripts. In this study, all three prompts instructed the model to use this reading capability to discover thematic categories from the student feedback. Opus and Gemini followed this instruction in most prompted runs, reading each student's Thai text and assigning categories directly. GPT, however, usually defaulted to hardcoded keyword dictionaries and substring-matching functions. This pattern appeared across prompt tiers, though one Prompt 2 repetition stored LLM-derived assignments.

For LLM-based thematic reading (R2.5), structured, concise prompts (Prompt 2) achieved the highest overall pass rate at 78% (7/9), followed by detailed prompts (Prompt 1) at 67% (6/9) and exploratory prompts (Prompt 3) at 33% (3/9). This ordering concurred with prompt engineering research, showing that LLM outputs are highly sensitive to prompt design, and even minor phrasing changes can substantially affect performance (Schulhoff et al., 2024). Broader prompt-engineering surveys catalogue dozens of task-specific prompting techniques (Sahoo et al., 2024). In LLM-assisted reflexive thematic analysis, evaluators

preferred LLM-generated codes 61% of the time when prompts were well-structured, though LLMs still fragmented data and missed latent meanings even with refined prompts (Martinez Montes et al., 2025).

In this study, the sensitivity to prompt design was more pronounced under Prompt 3, where the lack of procedural specificity led to the lowest R2.5 pass rate at 33% (3/9). These findings suggested that structured prompts are needed for consistent thematic reading from agentic coding tools, while exploratory prompts risk degrading performance. How each model approached the analytical task also affected LLM-based thematic reading (R2.5) outcomes. Opus and Gemini consistently applied LLM-based reading under both detailed and structured prompts, while GPT tended to rely on keyword-matching scripts across all prompt tiers.

Thai text presents specific challenges for automated text categorization. Word meanings depend on context in Thai because word boundaries are ambiguous, allowing the same character sequence to convey a different meaning depending on the surrounding text (Arreerard et al., 2022). The word "ตื่นเต้น" (excited), for instance, appeared in an anxiety keyword list, but in student responses expressed positive anticipation about clinic work. The same word could also convey nervousness about being unprepared, and only the surrounding context disambiguates its meaning.

Individual responses compounded this difficulty by combining multiple sentiments within a single passage. For example, S006 contained praise for hands-on activities, appreciation of instructors, and complaints about scheduling and insufficient instructor-student ratio within one continuous response, requiring the reader to parse which segments conveyed positive versus negative sentiment. In this response, a positive emotion word was embedded inside a scheduling complaint (e.g., "do not want this fun course crammed at end of semester near exams"), but GPT's keyword-matching scripts assigned the response to an enjoyment category based on the keyword match alone.

Student responses also included emojis, Thai English language mixing (e.g., writing the English word "inspiration" within a predominantly Thai sentence), and informal conversational text. The keyword lists were not designed to interpret these occurrences. Despite the availability of dictionary-based segmentation libraries (Phatthiyaphaibun et al., 2023), the keyword-matching scripts generated in this study used naive substring or regex operations without invoking such tools. This limited keyword matching to exact surface-level matches that were not generalized across the context-dependent expressions in Thai student feedback.

Traditional text classification approaches require labeled training data, from machine learning classifiers that depend on manual feature engineering to deep learning models that learn feature representations automatically (Li et al., 2022). To the best of our knowledge, no public dataset exists for Thai educational feedback at this level of domain specificity, and training a task-specific model would require substantial annotation effort. Frontier LLMs bypass this

requirement by processing text through general language capability, though benchmark studies confirm that they exhibit performance gaps on Thai-specific tasks (Kim et al., 2025), with performance declining further on regional dialects where only proprietary frontier models maintain adequate fluency (Limkonchotiwat et al., 2025). One multilingual LLM approach is parameter-frozen prompting, in which the model's general language capability handles non-English input without explicit fine-tuning (Qin et al., 2025).

In this study, Opus and Gemini processed Thai student feedback through direct reading without task-specific training, categorizing multi-topic responses into thematic categories that distinguished positive from negative sentiment within the same passage. These results provide preliminary evidence that frontier LLMs can assist with thematic analysis of non-English educational feedback in languages where specialized NLP resources are limited, though the approach is not uniformly available across models, as GPT's preference for keyword-matching scripts demonstrates. Model selection should, therefore consider deployment context and human-centered evaluation criteria in addition to general benchmark scores (Miller & Tang, 2025; Saleh et al., 2025).

4.3 Output variability across repetitions

In this study, output variability appeared in three forms. The first was thematic granularity as the number of canonical categories a run detected. Opus detected the most and most stably, a mean of 11.7 of the 17 categories per run, while Gemini detected the fewest and varied most, a mean of 7.7 and a range from 4 to 17, because it sometimes split one theme into several sub-categories and sometimes collapsed several into one. The second was the detection consistency of which categories appeared across the three repetitions of a condition. Only about half of all category detections were reproduced in all three repetitions. This rate was lowest for Gemini, and a category identified in one run could be absent from another run under identical settings.

A few high-frequency categories such as Hands-on Practice and Time & Scheduling appeared in every repetition, while lower-frequency categories appeared sporadically. The third was the count variability as the number of student mentions assigned to a detected category. This was largest under GPT's keyword-matching scripts, where one category ranged from 13 to 53 mentions across the repetitions of a single condition, while the semantic-reading models recorded results within narrower ranges. Some of this variability arose from the analytical approach because a model could read semantically in one run and generate keyword-matching scripts in another under the same prompt, as GPT did within its Prompt 2 condition.

Our observed run-to-run variation concurred with broader findings on LLM stochasticity. A large-scale study of 3.4 million LLM outputs found that binary classification achieved near-perfect reproducibility while generative tasks such as summarization exhibited greater variation (Wang & Wang, 2025). Deductive qualitative coding required 40 repeated iterations for response rates to stabilize (Tai et al., 2024). Prior studies of LLM-based educational feedback analysis used

a sampling temperature of zero to suppress this variability (Parker et al., 2025). However, a temperature of zero does not guarantee identical outputs because the computational process can introduce variation even under fixed parameters (Atil et al., 2024). Agentic execution adds a further source of variation because the model can autonomously decide which scripts to write and which analytical steps to take. The CLI applications used in this study did not expose sampling parameters for adjustment. Theme identification proved closer in complexity to summarization than to binary classification, with each student response assigned to multiple themes rather than to a single label.

In reflexive thematic analysis, qualitative coding is inherently interpretive, and different analysts can generate different valid readings of the same data (Braun & Clarke, 2019). Bano et al. (2024) examined LLM use in qualitative research by framing LLMs as assistive tools that require human oversight rather than autonomous coders. Treating model outputs as review inputs, rather than final categorizations, allows evaluation of which themes are robust across repeated analyses. Stochastic LLM variation can, however, differ from genuine human interpretive diversity because it arises from inference-level randomness (Herrera-Poyatos et al., 2025) rather than from differing analytical perspectives.

Multiple runs remain necessary to distinguish robust findings from run-specific artifacts (Herrera-Poyatos et al., 2025; Wang & Wang, 2025). This study demonstrated the same pattern, with output variability observed both across models and within identical model-prompt configurations. Additional canonical category mapping allowed outputs to be compared across runs. Each run captured a different level of thematic resolution, but whether a given run over-fragmented or over-consolidate the data could not be predicted. One practical benefit of CLI-based agentic tools is that they expose intermediate model outputs that can verify the analytical strategy, rather than assessing quality from text output alone. In this study, inspecting these intermediate outputs revealed whether categorization relied on semantic reading or on keyword matching. Researchers conducting LLM-based analysis should, therefore, expect output variability across repeated runs.

4.4 Implications

Several practical implications for educators and education researchers using agentic coding tools to analyze course evaluations follow from these findings. The models processed the feedback differently and produced diverse results; therefore, model choice should be deliberate. Structured prompts that direct the model through the required tasks are preferable to open-ended ones, consistent with the higher thematic-reading pass rates under the structured and detailed prompts. Sampling parameters were not adjustable in the CLI tools, but running the analysis across multiple independent runs made the run-to-run variability visible. The generated scripts and intermediate outputs are available for inspection, showing how each result was produced, how the statistics were computed, and whether categorization relied on keyword matching or on direct reading of the feedback.

5. Limitations

This study had several limitations. The dataset of student evaluations was small and was sourced from a single-institution Thai dental school course, limiting the robustness and the generalizability of the findings to English-language feedback and other non-English writing systems. This anonymized dataset did not include age, gender, or seniority variables, and the findings were interpreted only by academic year and by the experimental variables of model, prompt specificity, and repetition. The three prompt tiers were designed to represent detailed, structured, and exploratory instructions. Different prompt wording within the same level of procedural detail was not investigated. All the runs used single-turn execution, in which one prompt initiated fully autonomous analysis, with no human interaction during the session. This design was necessary for controlled comparison but might underestimate the quality achievable through interactive multi-turn use, which is more common in practice.

Reproducibility is also at risk because of how frontier models are versioned and retired. The tools evaluated here did not expose dated model snapshots. Recent model generations have moved to dateless version identifiers, which makes pinning an exact model state for replication harder. Previous educational-feedback studies reached human-level agreement using GPT-4 and GPT-3.5 (Parker et al., 2025). These models have now been retired from general availability, showing how the specific versions behind published findings become unavailable over time. Reproductions should report the exact tool and model versions used and treat the results as specific to those versions.

The canonical mapping of themes to 17 categories involved subjective post-hoc judgment. A more structured approach, such as a pre-specified category schema or inter-rater agreement with independent human coders, would further strengthen the mapping reliability. This study evaluated how each model performed categorization, specifically whether it read the text directly or matched keywords, rather than the accuracy of the resulting themes against a human-coded gold standard. Comparison with independent human thematic coding was outside the study scope and this is an important direction for assessing coding quality. This study documented run-to-run variability, including raw theme counts that did not depend on the canonicalization step.

Future studies could develop standardized prompt and model toolkits that emulate expert thematic-analysis practices in educational contexts, serving as supportive aids for course-evaluation review. The findings could also be strengthened through additional extensions. Larger studies addressing dataset diversity would improve generalizability. More extensive repetition per condition could reveal the range of possible outcomes, characterize each model's tendencies, and enable inferential analysis across runs.

6. Conclusions

CLI-based agentic coding tools can complete end-to-end analyses of Thai-language course evaluations without human intervention during each run. These tools operate in the local analysis environment, so generated scripts and

intermediate outputs remain available for inspection. The generated analyses varied by model, prompt specificity, and repetition. Statistical computation and visualization were handled more consistently than thematic categorization, while required-output compliance declined when prompts gave little procedural guidance. Repeated runs under identical configurations differed in detected theme categories, category counts, and category assignments, and the models produced diverse thematic results.

In passing runs, the models read the free-text comments directly and assign themes through semantic interpretation, whereas other runs substitute keyword-matching scripts for thematic reading. Structured prompts achieved the highest LLM-based thematic-reading pass rate. These findings suggest that agentic coding tools can assist with qualitative analysis of non-English educational feedback, including languages where specialized language-processing tools are limited. When applying these tools to educational feedback, model comparison, structured prompts, repeated runs, and inspection of generated files are recommended to support consistent use and verification of the analytical process.

7. Data Availability

All data and code used in this study, including the anonymized student evaluation dataset, experiment prompts, evaluation rubrics, analysis scripts, and LLM-generated outputs, are publicly available at https://github.com/superbijk/preventive-y3-llm-thematic-analysis.

Conflicts of Interest

The authors declare no conflicts of interest.

8. Acknowledgements

This study was supported by the University of Phayao. The authors acknowledge the use of AI coding assistants (Claude Opus 4.6 from Anthropic, GPT-5.3-Codex from OpenAI, and Gemini 3 Pro Preview from Google) as experimental subjects and as limited assistive tools for file inspection and manuscript revision checking. The authors retain full responsibility for the research design, manuscript text, analysis, and findings, and all AI-assisted outputs were verified for accuracy.

9. References

Almulla, M., & Ali, S. I. (2024). The changing educational landscape for sustainable online experiences: Implications of ChatGPT in Arab students' learning experience. *International Journal of Learning, Teaching and Educational Research*, *23*(9), 285–306. https://doi.org/10.26803/ijlter.23.9.15

Arreerard, R., Mander, S., & Piao, S. (2022). Survey on Thai NLP language resources and tools. *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, 6495–6505. https://aclanthology.org/2022.lrec-1.697/

Atil, B., Aykent, S., Chittams, A., Fu, L., Passonneau, R. J., Radcliffe, E., Rajagopal, G. R., Sloan, A., Tudrej, T., Ture, F., Wu, Z., Xu, L., & Baldwin, B. (2024). Non-determinism of "deterministic" LLM settings. *arXiv*. https://doi.org/10.48550/arXiv.2408.04667

Bano, M., Hoda, R., Zowghi, D., & Treude, C. (2024). Large language models for qualitative research in software engineering: Exploring opportunities and

challenges. *Automated Software Engineering*, *31*(1), Article 8. https://doi.org/10.1007/s10515-023-00407-8

Braun, V., & Clarke, V. (2006). Using thematic analysis in psychology. *Qualitative Research in Psychology*, *3*(2), 77–101. https://doi.org/10.1191/1478088706qp063oa

Braun, V., & Clarke, V. (2019). Reflecting on reflexive thematic analysis. *Qualitative Research in Sport, Exercise and Health*, *11*(4), 589–597. https://doi.org/10.1080/2159676X.2019.1628806

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., ... Amodei, D. (2020). Language models are few-shot learners. *Advances in Neural Information Processing Systems*, *33*, 1877–1901. https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf

Cai, C., Hong, S., Ma, M., Feng, H., Du, S., Chow, M., Teo, W. L., Liu, S., & Fan, X. (2025). Analyzing the teaching and learning environments through student feedback at scale: A multi-agent LLMs framework. *Education and Information Technologies*, *30*(15), 21815–21847. https://doi.org/10.1007/s10639-025-13633-2

Denny, P., Prather, J., Becker, B. A., Finnie-Ansley, J., Hellas, A., Leinonen, J., Luxton-Reilly, A., Reeves, B. N., Santos, E. A., & Sarsa, S. (2024). Computing education in the era of generative AI. *Communications of the ACM*, *67*(2), 56–67. https://doi.org/10.1145/3624720

El-Hakim, M., Anthonappa, R., & Fawzy, A. (2025). Artificial intelligence in dental education: A scoping review of applications, challenges, and gaps. *Dentistry Journal*, *13*(9), Article 384. https://doi.org/10.3390/dj13090384

Gridach, M., Nanavati, J., Zine El Abidine, K., Mendes, L., & Mack, C. (2025). Agentic AI for scientific discovery: A survey of progress, challenges, and future directions. *arXiv*. https://doi.org/10.48550/arXiv.2503.08979

Herrera-Poyatos, D., Pelaez-Gonzalez, C., Zuheros, C., Herrera-Poyatos, A., Tejedor, V., Herrera, F., & Montes, R. (2025). An overview of model uncertainty and variability in LLM-based sentiment analysis: Challenges, mitigation strategies, and the role of explainability. *Frontiers in Artificial Intelligence*, *8*, Article 1609097. https://doi.org/10.3389/frai.2025.1609097

Jiang, J., Wang, F., Shen, J., Kim, S., & Kim, S. (2026). A survey on large language models for code generation. *ACM Transactions on Software Engineering and Methodology*, *35*(2), Article 58, 1–72. https://doi.org/10.1145/3747588

Jimenez, C. E., Yang, J., Wettig, A., Yao, S., Pei, K., Press, O., & Narasimhan, K. R. (2024). SWE-bench: Can language models resolve real-world GitHub issues? *Proceedings of the Twelfth International Conference on Learning Representations (ICLR)*. https://doi.org/10.48550/arXiv.2310.06770

Jita, T., & Thaanyane, M. E. (2025). Evaluating the validity and impact of teaching practice assessment tools in higher education. *International Journal of Learning, Teaching and Educational Research*, *24*(7), 486–503. https://doi.org/10.26803/ijlter.24.7.24

Kasneci, E., Sessler, K., Küchemann, S., Bannert, M., Dementieva, D., Fischer, F., Gasser, U., Groh, G., Günnemann, S., Hüllermeier, E., Krusche, S., Kutyniok, G., Michaeli, T., Nerdel, C., Pfeffer, J., Poquet, O., Sailer, M., Schmidt, A., Seidel, T., ... Kasneci, G. (2023). ChatGPT for good? On opportunities and challenges of large language models for education. *Learning and Individual Differences*, *103*, Article 102274. https://doi.org/10.1016/j.lindif.2023.102274

Kim, D., Lee, S., Kim, Y., Rutherford, A., & Park, C. (2025). Representing the under-represented: Cultural and core capability benchmarks for developing Thai large language models. In *Proceedings of the 31st International Conference on

Computational Linguistics* (pp. 4114–4129). Association for Computational Linguistics. https://aclanthology.org/2025.coling-main.278/

Li, Q., Peng, H., Li, J., Xia, C., Yang, R., Sun, L., Yu, P. S., & He, L. (2022). A survey on text classification: From traditional to deep learning. *ACM Transactions on Intelligent Systems and Technology*, *13*(2), 1–41. https://doi.org/10.1145/3495162

Limkonchotiwat, P., Masuk, K., Nonesung, S., Mai-On, C., Nutanong, S., Ponwitayarat, W., & Manakul, P. (2025). Assessing Thai dialect performance in LLMs with automatic benchmarks and human evaluation. *arXiv*. https://doi.org/10.48550/arXiv.2504.05898

Lucas, H. C., Upperman, J. S., & Robinson, J. R. (2024). A systematic review of large language models and their implications in medical education. *Medical Education*, *58*(11), 1276–1285. https://doi.org/10.1111/medu.15402

Martinez Montes, C., Feldt, R., Miguel Martos, C., Ouhbi, S., Premanandan, S., & Graziotin, D. (2025). Large language models in thematic analysis: Prompt engineering evaluation and guidelines for qualitative software engineering research. *arXiv*. https://doi.org/10.48550/arXiv.2510.18456

Miller, J. K., & Tang, W. (2025). Evaluating LLM metrics through real-world capabilities. *arXiv*. https://doi.org/10.48550/arXiv.2505.08253

Parker, M. J., Anderson, C., Stone, C., & Oh, Y. R. (2025). A large language model approach to educational survey feedback analysis. *International Journal of Artificial Intelligence in Education*, *35*(2), 444–481. https://doi.org/10.1007/s40593-024-00414-0

Phatthiyaphaibun, W., Chaovavanich, K., Polpanumas, C., Suriyawongkul, A., Lowphansirikul, L., Chormai, P., Limkonchotiwat, P., Suntorntip, T., & Udomcharoenchaikit, C. (2023). PyThaiNLP: Thai natural language processing in Python. In *Proceedings of the 3rd Workshop for Natural Language Processing Open Source Software (NLP-OSS 2023)* (pp. 25–36). Association for Computational Linguistics. https://doi.org/10.18653/v1/2023.nlposs-1.4

Qin, L., Chen, Q., Zhou, Y., Chen, Z., Li, Y., Liao, L., Li, M., Che, W., & Yu, P. S. (2025). A survey of multilingual large language models. *Patterns*, *6*(1), Article 101118. https://doi.org/10.1016/j.patter.2024.101118

Sahoo, P., Singh, A. K., Saha, S., Jain, V., Mondal, S., & Chadha, A. (2024). A systematic survey of prompt engineering in large language models: Techniques and applications. *arXiv*. https://doi.org/10.48550/arXiv.2402.07927

Saleh, Y., Abu Talib, M., Nasir, Q., & Dakalbab, F. (2025). Evaluating large language models: A systematic review of efficiency, applications, and future directions. *Frontiers in Computer Science*, *7*, Article 1523699. https://doi.org/10.3389/fcomp.2025.1523699

Schulhoff, S., Ilie, M., Balepur, N., Kahadze, K., Liu, A., Si, C., Li, Y., Gupta, A., Han, H., Dulepet, P. S., Vidyadhara, S., Ki, D., Agrawal, S., Pham, C., Kroiz, G., Li, F., Tao, H., Srivastava, A., Da Costa, H., ... Resnik, P. (2024). The prompt report: A systematic survey of prompting techniques. *arXiv*. https://doi.org/10.48550/arXiv.2406.06608

Sun, M., Han, R., Jiang, B., Qi, H., Sun, D., Yuan, Y., & Huang, J. (2025). A survey on large language model-based agents for statistics and data science. *The American Statistician*, 1–14. https://doi.org/10.1080/00031305.2025.2561140

Tai, R. H., Bentley, L. R., Xia, X., Sitt, J. M., Fankhauser, S. C., Chicas-Mosier, A. M., & Monteith, B. G. (2024). An examination of the use of large language models to aid analysis of textual data. *International Journal of Qualitative Methods*, *23*, Article 16094069241231168. https://doi.org/10.1177/16094069241231168

Wang, J. J., & Wang, V. X. (2025). Assessing consistency and reproducibility in the outputs of large language models: Evidence across diverse finance and accounting tasks. *arXiv*. https://doi.org/10.48550/arXiv.2503.16974

Wei, X., Cui, X., Cheng, N., Wang, X., Zhang, X., Huang, S., Xie, P., Xu, J., Chen, Y., Zhang, M., Jiang, Y., & Han, W. (2023). ChatIE: Zero-shot information extraction via chatting with ChatGPT. *arXiv*. https://doi.org/10.48550/arXiv.2302.10205

Wen, C., Clough, P., Paton, R., & Middleton, R. (2026). Leveraging large language models for thematic analysis: A case study in the charity sector. *AI & Society*, *41*(1), 731–748. https://doi.org/10.1007/s00146-025-02487-4

Wijnen-Meijer, M., van den Broek, S., Koens, F., & ten Cate, O. (2020). Vertical integration in medical education: The broader perspective. *BMC Medical Education*, *20*(1), Article 509. https://doi.org/10.1186/s12909-020-02433-6

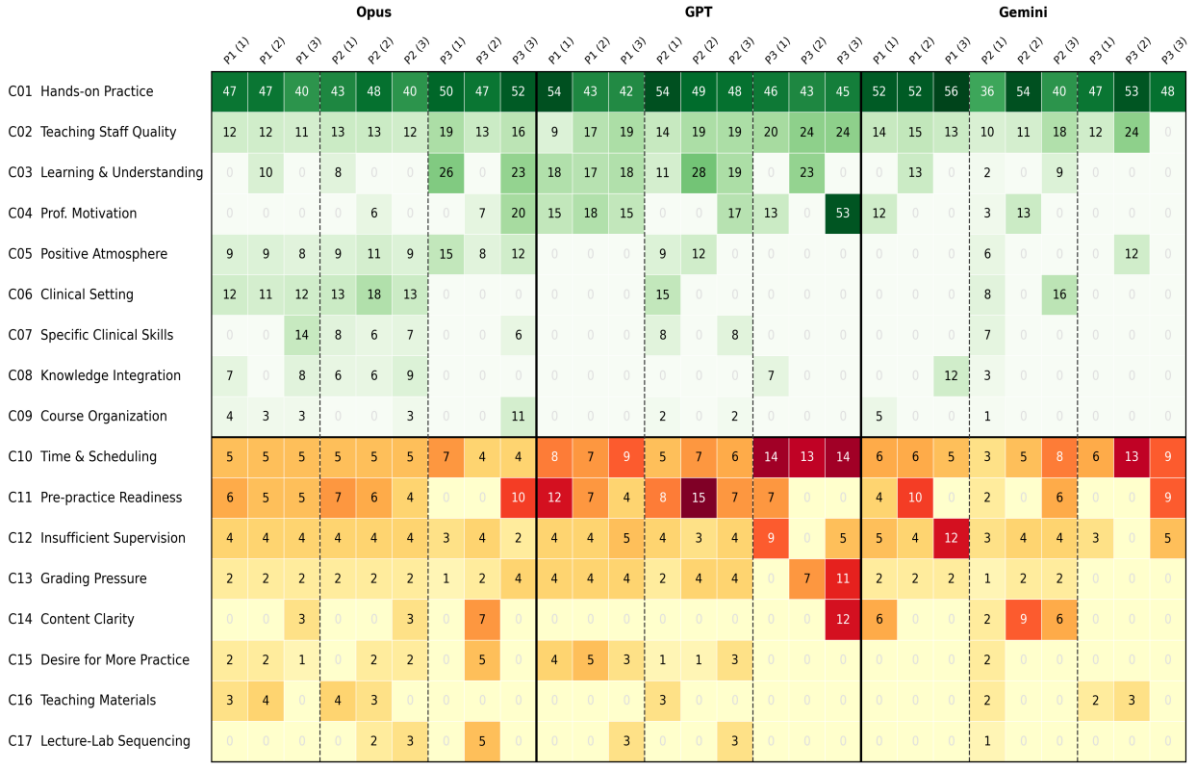
Appendix 1

Representative student responses for the 17 canonical feedback categories

Canonical category	Representative student response (translated)
C01 Hands-on Practice	"What I liked was getting to do real hands-on work." (S001)
C02 Teaching Staff Quality	"The instructors taught in a way that was easy to understand and did not put too much pressure on students doing the practical for the first time." (S025)
C03 Learning & Understanding	"…it helped me picture what I had studied and understand it better." (S011)
C04 Professional Motivation	"…this course reminded me that I still love the dental profession, and I have to thank this lab for rekindling my drive." (S042)
C05 Positive Atmosphere	"I liked that it was taught clearly and without too much pressure, and the lab was fun every time." (S058)
C06 Clinical Setting	"I liked going to learn in the clinic, which made me feel closer to becoming a dentist…" (S008)
C07 Specific Clinical Skills	"I liked getting to practice doing sealants." (S009)
C08 Knowledge Integration	"…I reviewed knowledge from other lectures such as cariology, perio, and dental anatomy…" (S015)
C09 Course Organization	"The schedule was well arranged, each session's content was well balanced and usable in practice, and students' knowledge was assessed appropriately." (S041)
C10 Time & Scheduling	"…the downside was too little time, crammed into the end of the semester…" (S005)
C11 Pre-practice Readiness	"…I felt unprepared, for example with the instruments, since I had never actually used them and did not know how to handle the tools." (S046)
C12 Insufficient Supervision	"There were too few supervising instructors, which made it rather chaotic and slow to get questions answered or ask for help." (S002)
C13 Grading Pressure	"…I would prefer no point deductions, only warnings, because when you ask and try something and make a mistake you lose points, which makes you afraid to ask or try." (S018)
C14 Content Clarity	"…when studying I did not really understand the content, could not see the overall picture, and was still confused about what was what…" (S080)
C15 Desire for More Practice	"…I would like more practice examining caries and calculus…" (S066)
C16 Teaching Materials	"…the instructors could provide detailed Thai teaching handouts…" (S053)
C17 Lecture-Lab Sequencing	"…I would like to alternate lectures and labs, studying a topic one week and doing the lab on that topic the next, so it is less confusing." (S042)

Appendix 2

Per-run feedback-category counts across all 27 runs



Each column represents one run, and each cell gives the mention count for a canonical feedback category. The main-text Figure 4 aggregates these by model and prompt and marks with an asterisk (*) any category not detected in all three repetitions of a cell.

Appendix 3: Prompts

- **Prompt 1 (Detailed Procedural, 372 lines)**

```
# Dental Course Evaluation Analysis — Full Pipeline

You are analyzing anonymized student evaluation data for a preventive dentistry course taught to 3rd-year dental students across three academic years.

**Execute all steps below in order. Write Python code and run it to produce each output file.**

**Python environment**: Use the `dental-eval` conda environment. Run all Python commands with `conda run -n dental-eval python ...` (it has pandas, openpyxl, matplotlib, numpy).

---

## Section A: Data Description

**Input file**: `{{DATA_FILE}}`
- Single sheet named `feedbacks`, 89 rows, 6 columns
- All student IDs are already anonymized (S001-S089)

| Column | Type | Description |
|--------|------|-------------|
| `student_id` | str | Anonymized ID (S001-S089) |
| `academic_year` | int | 2566 (n=29), 2567 (n=30), 2568 (n=30) |
| `satisfaction_teaching_process` | int | Likert 1-5 |
| `satisfaction_learning` | int | Likert 1-5 |
| `dissatisfaction_rating` | int | 1-5 (1 = no dissatisfaction) |
| `likes_dislikes` | str | Thai free-text feedback (some rows are null) |

**First step**: Read the xlsx file with pandas + openpyxl. Confirm the shape and column names before proceeding.

---

## Section B: Output 1 — `{{RUN_DIR}}/statistics_summary.md`

Compute descriptive statistics and write a Markdown report.

### Structure:

```markdown
# Statistical Summary Report

รายงานสรุปผลการวิเคราะห์ความพึงพอใจในการเรียนการสอน

## Data Overview

| รายการ | จำนวน |
|--------|-------|
| จำนวนนิสิตทั้งหมด | XX คน |
| ปีการศึกษา 2566 | XX คน |
| ปีการศึกษา 2567 | XX คน |
| ปีการศึกษา 2568 | XX คน |

---

## Satisfaction Scores Summary

### 1. ความพึงพอใจกระบวนการสอน (Satisfaction with Teaching Process)

| ปีการศึกษา | N | Mean | SD | Min | Max | Median |
|------------|---|------|-----|-----|-----|--------|
```
```

2566	...	...	...	...	...	...
2567	...	...	...	...	...	...
2568	...	...	...	...	...	...
รวมทุกปี	**...**	**...**	**...**	**...**	**...**	**...**
```

Repeat the table for:
- **2. ความพึงพอใจการเรียนรู้ (Satisfaction with Learning)**
- **3. ความไม่พึงพอใจ (Dissatisfaction Rating)**

**Important notes on statistics**:
- Mean rounded to 2 decimal places, SD rounded to 2 decimal places
- Median rounded to 1 decimal place (e.g., 5.0, 4.5, 1.0)
- Min and Max as integers
- N = count of non-null values for that column in that year group
- For dissatisfaction_rating, note that some rows may have null values — use `.dropna()` before computing

### Frequency Distribution (รวมทุกปี)

For each of the 3 metrics, create a table:

```markdown
### ความพึงพอใจกระบวนการสอน

| คะแนน | จำนวน | ร้อยละ |
|-------|-------|--------|
| 1 | ... | ...% |
| 2 | ... | ...% |
| 3 | ... | ...% |
| 4 | ... | ...% |
| 5 | ... | ...% |
```

Percentages should be of the non-null total for that column, rounded to 1 decimal place.

Key Findings section

Write in Thai:

```markdown
## Key Findings

### ผลการวิเคราะห์โดยรวม

1. **ความพึงพอใจกระบวนการสอน**: ค่าเฉลี่ยรวม X.XX (SD=X.XX) อยู่ในระดับ...
   - ปี XXXX มีค่าเฉลี่ยสูงสุด (X.XX)
   - ปี XXXX มีค่าเฉลี่ยต่ำสุด (X.XX)...

2. **ความพึงพอใจการเรียนรู้**: ...

3. **ความไม่พึงพอใจ**: ...

### สรุป

- [3-5 bullet points summarizing the key findings]
```

End with:
```
---

*Report generated from deduplicated data (89 unique students)*
```

```

Section C: Detailed Thematic Analysis (do this BEFORE the summary)

What to do

Read every non-null `likes_dislikes` entry. Each entry is Thai free-text that may contain things the student liked, disliked, or both.

Your task: Discover thematic categories by reading all the feedback. Do NOT use pre-defined categories. Instead:

1. Read through all feedback entries
2. Identify recurring themes for "likes" (สิ่งที่ชอบ) and "dislikes/suggestions" (สิ่งที่ไม่ชอบ/ข้อเสนอแนะ) that naturally emerge from the text
3. Name each category with a Thai label and English translation
4. Assign each student's feedback to the appropriate categories (a student can appear in multiple categories)

Output file: `{{RUN_DIR}}/feedback_analysis_llm.md`

```markdown
# Feedback Analysis (LLM-Based)

การวิเคราะห์ความคิดเห็นจากแบบประเมินความพึงพอใจ โดยใช้ LLM ในการอ่านและจัดกลุ่มข้อความ

---

## Data Summary

| ปีการศึกษา | จำนวนนิสิต | จำนวนความคิดเห็น |
|---|---|---|
| 2566 | ... | ... |
| 2567 | ... | ... |
| 2568 | ... | ... |
| **รวม** | **...** | **...** |
```

Count "จำนวนความคิดเห็น" as non-null entries in `likes_dislikes` per year.

Likes Section

For each discovered category:

```markdown
## สิ่งที่ชอบ (Likes)

### 1. [Thai Category Name] / [English Translation]

**จำนวน: XX คน**

[2-3 sentence Thai description of what students said in this category]

| Student ID | ปี | ความคิดเห็น |
|---|---|---|
| SXXX | 2566 | [relevant excerpt from feedback] |
| SXXX | 2567 | [relevant excerpt] |
| ... | ... | ... |

---
```

Dislikes Section

Same format:

```markdown
```

สิ่งที่ไม่ชอบ/ข้อเสนอแนะ (Dislikes & Suggestions)

1. [Thai Category Name] / [English Translation]

จำนวน: XX คน

[2-3 sentence Thai description]

Student ID	ปี	ความคิดเห็น
SXXX	2566	[relevant excerpt]
...	...	...

```

**Formatting rules**:
- Use the anonymized student IDs (S001-S089) from the data
- For students whose feedback contains both likes and dislikes, show the full text under the relevant "like" category, and use "..." prefix/suffix when showing partial quotes under "dislike" categories
- A single student can appear in multiple categories
- Show the original Thai text as-is — do not translate or modify

---

## Section D: Feedback Summary (write AFTER Section C, using its results)

This is a summary-level report derived from the detailed analysis in Section C. It contains NO individual student rows — only aggregate counts and observations.

### Output file: `{{RUN_DIR}}/feedback_summary.md`

```markdown
# Feedback Summary Report

สรุปภาพรวมความคิดเห็นจากแบบประเมินความพึงพอใจ

## Data Summary

| ปีการศึกษา | จำนวนนิสิต | จำนวนความคิดเห็น |
|---|---|---|
| 2566 | ... | ... |
| 2567 | ... | ... |
| 2568 | ... | ... |
| **รวม** | **...** | **...** |
```

Cross-tabulation tables

Use the **same categories and counts** discovered in Section C. Reproduce the cross-tabulation tables for likes and dislikes (same data, just without individual student rows).

Observations

Write in Thai:

```markdown
## แนวโน้มและข้อสังเกต

1. **จุดเด่นของการจัดการเรียนการสอนและการเรียนรู้**: ...
2. **แนวโน้มในแต่ละปีการศึกษา**: ... (describe what changed or stayed consistent across years)
3. **ข้อเสนอแนะ**: ...
4. [Additional observations if any]

---

*Summary generated using LLM-based text analysis ({{MODEL_NAME}})*
```

Section E: Four PNG Charts

Write Python code to generate 4 chart files. Use `pandas`, `matplotlib`, and `numpy`.

Chart 1: `{{RUN_DIR}}/figures/1_satisfaction_charts.png`

Overview bar chart — 1×3 subplots showing all 3 metrics side by side.

- Figure size: `(14, 5)`
- Main title (bold): `'Satisfaction Survey Results by Academic Year'`
- 3 subplots, one per metric:
  - Subplot 1: `satisfaction_teaching_process` — title `'Teaching Process'`
  - Subplot 2: `satisfaction_learning` — title `'Learning'`
  - Subplot 3: `dissatisfaction_rating` — title `'Dissatisfaction'`
- Each subplot: bars for years 2566, 2567, 2568 (x-axis)
- Colors: `['#4472C4', '#ED7D31', '#70AD47']` (blue for subplot 1, orange for subplot 2, green for subplot 3)
- Bar edge: black, linewidth 0.5
- Y-axis: 0 to 5.5, label `'Score'`
- X-axis: label `'Academic Year'`, ticks at [2566, 2567, 2568]
- Value labels: mean value (2 decimal) on top of each bar, bold, fontsize 10
- Sample size: white bold text `'n=XX'` inside each bar at y=0.3
- Gridlines: horizontal dashed, alpha 0.7
- DPI: 150, white background, tight layout

Charts 2-4: Detailed per-metric charts

Each chart has **1×4 subplots** (figure size `16×4.5`):
- **Left subplot**: mean bars by year (same as Chart 1 for that metric)
- **Right 3 subplots**: rating frequency histograms, one per year

Chart 2: `{{RUN_DIR}}/figures/2_satisfaction_teaching_process.png`
- Color: `#4472C4` (blue)
- Title: `'Satisfaction with Teaching Process'`

Chart 3: `{{RUN_DIR}}/figures/3_satisfaction_learning.png`
- Color: `#ED7D31` (orange)
- Title: `'Satisfaction with Learning'`

Chart 4: `{{RUN_DIR}}/figures/4_dissatisfaction_rating.png`
- Color: `#70AD47` (green)
- Title: `'Dissatisfaction Rating'`

Histogram subplot details:
- X-axis: ratings 1-5, range 0.5 to 5.5
- Y-axis: count of students
- Bar color: gray `#B0B0B0`, edge `#808080`, linewidth 0.5
- Count labels on top of each bar (only if count > 0), fontsize 9
- Vertical line at mean: color = metric color, linewidth 3, solid
- Mean annotation: `f'{mean:.2f}'`, bold, dark text `#333333`, fontsize 11, positioned at 85% of max count height, right-aligned just left of the line
- Title: `f'Year {year}\n(n={count})'`
- X-axis label: `'Rating'`
- Y-axis label: `'Count'`
- Gridlines: horizontal dashed, alpha 0.5

Left subplot title: `'All Years'`
- Has the same bar chart as Chart 1 (means by year with n= labels)

All charts: DPI 150, white background, tight layout.

```

```
## Section F: Execution Order

Follow this exact sequence:

1. **Read data**: Load `{{DATA_FILE}}`, verify shape and columns
2. **Statistics**: Compute all statistics → write `{{RUN_DIR}}/statistics_summary.md`
3. **Detailed thematic analysis**: Read all Thai feedback, discover categories, code each student → write `{{RUN_DIR}}/feedback_analysis_llm.md`
4. **Feedback summary**: Summarize the categories from step 3 → write `{{RUN_DIR}}/feedback_summary.md`
5. **Charts**: Write Python visualization code → execute → save 4 PNGs to `{{RUN_DIR}}/figures/`
6. **Save scripts**: Save all Python code used in the analysis to `{{RUN_DIR}}/` (e.g. `{{RUN_DIR}}/generate_statistics.py`, `{{RUN_DIR}}/generate_charts.py`)
7. **Verify**: List all output files and confirm they exist

**Critical requirements**:
- All student IDs must use the anonymized format from the data (S001-S089)
- All Thai text in feedback must be preserved exactly as-is
- Statistical values must be computed from the data (not hardcoded)
- Thematic categories must be discovered from the data — do not use pre-defined categories
- All chart text (titles, labels, annotations) must be in English

---

## Section G: Completion Gate (MUST PASS BEFORE FINISHING)

Do not stop after writing code. You must execute the scripts and verify all required outputs exist and are non-empty.

Required files (must exist and file size > 0):
- `{{RUN_DIR}}/statistics_summary.md`
- `{{RUN_DIR}}/feedback_analysis_llm.md`
- `{{RUN_DIR}}/feedback_summary.md`
- `{{RUN_DIR}}/generate_statistics.py`
- `{{RUN_DIR}}/generate_feedback_reports.py`
- `{{RUN_DIR}}/generate_charts.py`
- `{{RUN_DIR}}/figures/1_satisfaction_charts.png`
- `{{RUN_DIR}}/figures/2_satisfaction_teaching_process.png`
- `{{RUN_DIR}}/figures/3_satisfaction_learning.png`
- `{{RUN_DIR}}/figures/4_dissatisfaction_rating.png`

At the end, run verification commands and print their output:

```bash
ls -lh {{RUN_DIR}} {{RUN_DIR}}/figures
for f in \
  {{RUN_DIR}}/statistics_summary.md \
  {{RUN_DIR}}/feedback_analysis_llm.md \
  {{RUN_DIR}}/feedback_summary.md \
  {{RUN_DIR}}/generate_statistics.py \
  {{RUN_DIR}}/generate_feedback_reports.py \
  {{RUN_DIR}}/generate_charts.py \
  {{RUN_DIR}}/figures/1_satisfaction_charts.png \
  {{RUN_DIR}}/figures/2_satisfaction_teaching_process.png \
  {{RUN_DIR}}/figures/3_satisfaction_learning.png \
  {{RUN_DIR}}/figures/4_dissatisfaction_rating.png; do
  test -s "$f" || { echo "MISSING_OR_EMPTY: $f"; exit 1; }
done
echo "ALL_REQUIRED_OUTPUTS_PRESENT"
```

If any check fails, continue working and rerun generation until all checks pass.
Only finish after printing `ALL_REQUIRED_OUTPUTS_PRESENT`.
```

- **Prompt 2 (Structured-Concise, 101 lines)**

```
# Dental Course Evaluation Analysis — Structured-Concise

You are analyzing anonymized student evaluation data for a preventive dentistry course taught to 3rd-year dental students across three academic years.

**Execute all steps below in order. Write Python code and run it to produce each output file.**

**Python environment**: Use the `dental-eval` conda environment. Run all Python commands with `conda run -n dental-eval python ...` (it has pandas, openpyxl, matplotlib, numpy).

---

## Data Description

**Input file**: `{{DATA_FILE}}`
- Single sheet named `feedbacks`, 89 rows, 6 columns
- Columns: `student_id` (str, S001-S089), `academic_year` (int: 2566/2567/2568), `satisfaction_teaching_process` (Likert 1-5), `satisfaction_learning` (Likert 1-5), `dissatisfaction_rating` (1-5, 1 = no dissatisfaction), `likes_dislikes` (Thai free-text, some null)
- Year distribution: 2566 (n=29), 2567 (n=30), 2568 (n=30)

Read the xlsx file with pandas + openpyxl. Confirm shape and columns before proceeding.

---

## Output 1: `{{RUN_DIR}}/statistics_summary.md`

Compute descriptive statistics (N, mean, SD, min, max, median) for all 3 satisfaction/dissatisfaction metrics, grouped by academic year and overall. Include frequency distributions for each metric across all years combined. Write key findings in Thai summarizing trends across years. Round means and SDs to 2 decimal places, percentages to 1 decimal place. Handle null values with `.dropna()`.

---

## Output 2: `{{RUN_DIR}}/feedback_analysis_llm.md`

**Do this BEFORE the summary.** Read every non-null `likes_dislikes` entry. Discover thematic categories by reading all feedback — do NOT use pre-defined categories.

For each category, provide:
- Thai category name with English translation
- Count of students in the category
- Brief Thai description of what students said
- Table with Student ID, academic year, and original Thai text (preserved exactly as-is)

Organize into "สิ่งที่ชอบ (Likes)" and "สิ่งที่ไม่ชอบ/ข้อเสนอแนะ (Dislikes & Suggestions)" sections. A student can appear in multiple categories.

---

## Output 3: `{{RUN_DIR}}/feedback_summary.md`

**Write AFTER the detailed analysis, using its results.** Summary-level report with NO individual student rows — only aggregate counts and cross-tabulation tables using the same categories from the detailed analysis. Include Thai observations on trends across years and recommendations. End with: `*Summary generated using LLM-based text analysis ({{MODEL_NAME}})*`

---

## Output 4: Charts in `{{RUN_DIR}}/figures/`

Generate 4 PNG chart files (DPI 150, white background, tight layout):
```

1. **`1_satisfaction_charts.png`** — Overview: 1×3 subplot bar chart showing mean scores by year for all 3 metrics
2. **`2_satisfaction_teaching_process.png`** — Detail: mean bars + per-year rating histograms with mean line
3. **`3_satisfaction_learning.png`** — Detail: same layout as chart 2
4. **`4_dissatisfaction_rating.png`** — Detail: same layout as chart 2

Use distinct colors per metric. Include sample sizes and value labels on bars.

Save Scripts

Save all Python code used to `{{RUN_DIR}}/` (e.g. `generate_statistics.py`, `generate_charts.py`, `generate_feedback_reports.py`).

Completion Gate

Required files (must exist and file size > 0):
- `{{RUN_DIR}}/statistics_summary.md`
- `{{RUN_DIR}}/feedback_analysis_llm.md`
- `{{RUN_DIR}}/feedback_summary.md`
- `{{RUN_DIR}}/generate_statistics.py`
- `{{RUN_DIR}}/generate_feedback_reports.py`
- `{{RUN_DIR}}/generate_charts.py`
- `{{RUN_DIR}}/figures/1_satisfaction_charts.png`
- `{{RUN_DIR}}/figures/2_satisfaction_teaching_process.png`
- `{{RUN_DIR}}/figures/3_satisfaction_learning.png`
- `{{RUN_DIR}}/figures/4_dissatisfaction_rating.png`

Verify all outputs exist and are non-empty before finishing:

```bash
ls -lh {{RUN_DIR}} {{RUN_DIR}}/figures
for f in \
  {{RUN_DIR}}/statistics_summary.md \
  {{RUN_DIR}}/feedback_analysis_llm.md \
  {{RUN_DIR}}/feedback_summary.md \
  {{RUN_DIR}}/generate_statistics.py \
  {{RUN_DIR}}/generate_feedback_reports.py \
  {{RUN_DIR}}/generate_charts.py \
  {{RUN_DIR}}/figures/1_satisfaction_charts.png \
  {{RUN_DIR}}/figures/2_satisfaction_teaching_process.png \
  {{RUN_DIR}}/figures/3_satisfaction_learning.png \
  {{RUN_DIR}}/figures/4_dissatisfaction_rating.png; do
  test -s "$f" || { echo "MISSING_OR_EMPTY: $f"; exit 1; }
done
echo "ALL_REQUIRED_OUTPUTS_PRESENT"
```

Only finish after printing `ALL_REQUIRED_OUTPUTS_PRESENT`.

- **Prompt 3 (Exploratory, 21 lines)**

```
# Dental Course Evaluation Analysis — Exploratory

You are analyzing student evaluation data for a preventive dentistry course.

Data: `{{DATA_FILE}}` — inspect the file to understand its structure.

**Python environment**: Use `conda run -n dental-eval python ...` (has pandas, openpyxl, matplotlib, numpy).

Produce a comprehensive analysis:

1. **Statistical analysis** of satisfaction scores across years — descriptive stats, distributions, trends
2. **Thematic categorization** of Thai free-text feedback — discover categories from the data, show student-level detail with IDs and original text
3. **Summary of findings** with trends and recommendations (write observations in Thai)
4. **Visualizations** that communicate the results effectively — overview and per-metric detail charts

Save all outputs (reports, charts, Python scripts) to `{{RUN_DIR}}/`.
Place chart images in `{{RUN_DIR}}/figures/`.

End with `*Analysis generated using {{MODEL_NAME}}*` in your summary report.

Save the Python scripts you create. Verify outputs at the end.
```