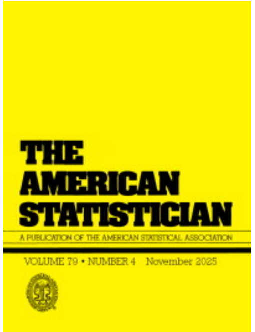

The American Statistician

ISSN: 0003-1305 (Print) 1537-2731 (Online) Journal homepage: www.tandfonline.com/journals/utas20

Facilitating a Collaborative Relationship between Generative AI and the Statistics Student

Richard A. Levine

Facilitating a Collaborative Relationship between Generative AI and the Statistics Student

Richard A. Levine*

Department of Mathematics and Statistics, San Diego State University

*rlevine@sdsu.edu

Abstract

This article examines how students can engage with generative artificial intelligence (genAI) as collaborators in the statistics learning process. Prompt engineering is positioned as a transferable, tool-agnostic competency that reinforces core elements of statistical thinking, including clarity, iteration, and purposeful inquiry. Through illustrative collaborations, we explore applications such as automating and optimizing code, acquiring programming syntax, and designing simulation studies. While these tasks are drawn from upper-level undergraduate and graduate coursework, the running example–a chi-squared test of association–is intended to spur ideas for incorporating genAI into the introductory statistics classroom. Supplementary materials include a) an outline of a learning management module and structure of the discussion and activities during my class periods covering this module on responsible use of generative AI; b) R Markdown files and compiled pdf documents intended to support classroom integration; c) illustrative comparisons across three widely used platforms–ChatGPT, Copilot, and Gemini–to highlight how differences in output style and reasoning can inform instructional design, rather than to rank or evaluate tools technically. The article concludes with a discussion of strategies for promoting ethical, transparent, and inclusive uses of genAI in statistics education.

Keywords: Statistics pedagogy, artificial intelligence, prompt engineering, responsible use of genAI, statistical thinking, algorithm development

1 Introduction

Statistics education has evolved alongside continual advances in technology. From David Moore's reflections in the 1990s (Moore, 1997) on calculators and "multimedia technologies" to today's interactive computing environments, each innovation has prompted instructors to reconsider how best to teach data analysis, modeling, and uncertainty. Simulation-based inference (Rossman and Chance, 2014) exemplifies such reform, using computation to foreground conceptual reasoning. Generative artificial intelligence (genAI) represents the next such pivot point. A parallel position from the National Council of Teachers of Mathematics (2023) affirms that, when used strategically, technology can promote equitable access and meaningful engagement for every learner–a principle that applies equally to the thoughtful integration of genAI into statistics education.

Recent commentaries further underscore both the opportunities and challenges accompanying this shift. Sobo (2023) suggests that tools like ChatGPT could inaugurate a "new golden age" in higher education if adopted thoughtfully. Ellis and Slade (2023) discuss genAI as a catalyst for re-examining the roles of explanation and creativity in statistics instruction. Tu et al. (2024) ask what data-science education should do with large language models, arguing that they move learners toward system-level reasoning and higher-order problem solving. Together these perspectives highlight the urgency of developing pedagogical frameworks that help students use genAI responsibly and meaningfully.

Students are already experimenting with chatbots as study partners and coding assistants. A recent campus survey (Goldberg et al., 2024) found that more than one-third of respondents regularly use AI-powered tools for schoolwork and nearly three-quarters believe AI will be essential in most professions. In my own upper-level Statistical Computing course, every student confirmed using genAI to write or debug code. The challenge for instructors is not whether students will use these tools, but how to guide them toward collaborations that strengthen statistical understanding. My approach treats genAI much like earlier technologies that automated computation: as a collaborator that frees students to focus on reasoning rather than mechanics. When guided appropriately, interactions with genAI can reinforce the core habits of statistical thinking–clarifying questions, articulating assumptions, evaluating evidence, and refining inquiry. These same habits are central to prompt engineering, the process of crafting inputs that elicit accurate and context-appropriate responses from large language models.

The conference paper by Sturdy (2024) offers an instructive precedent from professional practice, illustrating how ChatGPT can assist statistical programmers in automating, optimizing, and documenting code. Building on this idea, the present article extends the notion of collaboration with genAI from the professional to the educational setting, demonstrating how students can learn to partner with these tools to deepen their understanding of statistical concepts and methods.

The goals of this paper are twofold: (1) to illustrate how educators can help students collaborate with genAI so that attention remains focused on statistical reasoning and interpretation, with mechanical coding serving a supportive role; and (2) to demonstrate how prompt engineering and responsible AI use can be integrated into the statistics curriculum as mutually reinforcing components of statistical thinking.

Roadmap for Instructors: I developed a one-week instructional module, delivered during the third week of the semester, on responsible collaboration with generative AI as students begin engaging with course material and assessments. The module sequence–pre-class readings, formative assessment, in-class demonstrations, and the lab practical–structures both my classroom activities and the organization of this paper. The supplemental materials include the complete lesson plan and readings for this module; the lab practical included therein serves as the penultimate assignment of the week. Introducing the module early in the semester allows students to establish a collaborative relationship with generative AI while emphasizing its potential and the ethical responsibilities that accompany its use.

The paper follows this instructional sequence. Section 2 presents the conceptual framework for teaching prompt engineering as statistical thinking, the foundation students need before applying genAI in statistical tasks. Section 3 introduces the chi-square test of association, the dataset used in the lab practical, and the context for class discussions. During the module, class meetings revolve around the lab practical, interweaving ethical use of genAI, prompt engineering, and guided, hands-on collaboration. The lesson plan in the supplemental material outlines the assigned readings, formative assessment, and discussion prompts that structure these class sessions.

The remaining sections of the paper parallel the in-class demonstrations and discussions that complete the module:

• Interactive demonstrations (Section 4): Three in-class examples–exploratory analysis, algorithm development, and inferential reasoning–model how students can use genAI in applied statistical work. The exposition alternates between classroom discussion (prompt design, critique, and revision) and lab practical goals tied to statistical thinking. Subsection titles identify these two pedagogical dimensions: class discussion on prompts and from the lab practical.

• Designing a simulation study (Section 5): The hands-on portion of the module concludes with a demonstration in which I collaborate with genAI to design and code a simulation study assessing the chi-squared test. Although students do not perform this simulation themselves, it extends the discussion of statistical inference from the lab practical and introduces simulation experiments that reappear later in the course.

• Responsible and ethical use (Section 6): The final class discussion explores responsible and ethical collaboration with generative AI, emphasizing statistical reasoning, bias awareness, and professional accountability in data analysis.

The last subsection of Section 6 concludes the paper with final pedagogical reflections on equitable access, transparency, customized chatbots, and directions for future research in statistics education.

2 Prompt Engineering as Statistical Thinking

Building on the motivation outlined in Section 1, this section turns from why we should engage students with generative AI to how we can do so. It presents a conceptual framework that links prompt engineering to the habits of statistical thinking, offering practical strategies for helping students use genAI as a reflective and ethical collaborator.

2.1 Importance of Prompt Engineering in Statistics and Data Science Education

As generative AI (genAI) tools become increasingly integrated into Statistics and Data Science workflows, the ability to interact effectively with these systems has become an essential literacy for students. Prompt engineering–the practice of crafting inputs that elicit useful, accurate, and contextually appropriate responses from genAI–is not merely a technical trick; it is a reflection of clear reasoning, audience awareness, and domain understanding.

Statistics instructors are well positioned to help students develop this skill in a way that is tool-agnostic and transferable. While genAI systems will continue to evolve, the underlying habits of mind that make for effective prompting–clarity, iteration, reflection, and purpose-driven inquiry–will remain central to statistical problem-solving. Although model behaviors change rapidly, broad patterns of strength tend to persist: some systems excel at producing detailed explanations, others at generating or refining code, and still others at summarizing or organizing information. Recognizing these distinctions helps instructors guide students in evaluating and critiquing tools without tying instruction to any particular product or version. Ultimately, teaching prompt engineering offers a unique opportunity to help students collaborate meaningfully with AI while reinforcing core principles of statistics education.

2.2 Foundational Frameworks: CLEAR and Problem Formulation

Two complementary frameworks help structure the teaching of prompt engineering in statistics and data science: the CLEAR model of Lo (2023), which provides a framework for crafting and evaluating prompts, and the problem-formulation perspective of Acar (2023), which grounds prompting in well-defined questions and assumptions. Together they offer practical scaffolds for helping students think statistically while communicating effectively with generative AI.

Lo's CLEAR model highlights five principles for effective prompt design–Concise, Logical, Explicit, Adaptive, and Reflective. These align closely with the instructional priorities already emphasized in statistical communication and can serve as a rubric for evaluating or revising prompts in classroom settings.

Acar (2023), writing from a business perspective, argues that the most impactful uses of genAI begin not with cleverly worded prompts but with well-formulated problems. This view maps directly onto the goals of statistics education: students must learn to identify meaningful questions, frame them in analytic terms, and specify relevant assumptions. Prompt engineering thus becomes a natural extension of the problem-solving process that defines statistical thinking.

2.3 Prompt Engineering as Statistical Thinking

Prompt engineering activates many of the same practices that define statistical thinking. The framework of Wild and Pfannkuch (1999) emphasizes contextualization, transnumeration, and iterative refinement, all mirrored in thoughtful prompt development. Similarly, the GAISE College Report (ASA, 2016) promotes a cycle of formulating questions, exploring data, analyzing results, and communicating findings—stages well suited to genAI-facilitated workflows. Prompting becomes an act of inquiry, interpretation, and critique that parallels the logic of statistical investigation. Saghafian and Idan (2024) describe effective AI use as a "human-algorithm centaur," in which human reasoning is enhanced, not replaced, by AI.

These perspectives align closely with the pedagogy articulated by Grolemund and Wickham (2014), who advocate for a "question-first, tool-second" approach to data science. In their framework, successful use of computational tools emerges from students' ability to define and refine meaningful questions, precisely the skills that prompt engineering aims to develop.

Engaging effectively with genAI also requires distinguishing among query types. Conceptual questions (e.g., "What is a *p*-value?") differ from methodological questions (e.g., "Which model fits this situation?") and from computational ones (e.g., "How do I run this in R?"). Without clarity, genAI may return code instead of explanation or skip vital assumptions.

Prompting also requires contextual and audience awareness. Consider "Run a model." versus "Fit a logistic regression to predict churn using usage, plan type, and support interactions; assume a binary outcome and potential multicollinearity." The latter prompt reflects statistical fluency. Likewise, interpretation prompts should match the audience, whether peer, instructor, or stakeholder.

Students must also learn to treat AI output with healthy skepticism. Generative AI tools can hallucinate facts or misapply statistical procedures. Encouraging students to question outputs with follow-ups such as "What assumptions are you making?" or "What alternatives would be better if assumptions are violated?" reinforces habits of critical appraisal that align with sound statistical reasoning.

These distinctions are illustrated in Table 1, which presents common prompting intents along with effective and vague examples, clarifying their pedagogical implications.

2.4 Pedagogical Strategies for Teaching Prompt Engineering

Prompt engineering should be taught as an iterative, reflective process. Instructors can model prompt revision, highlighting how vague prompts yield poor output and how strategic refinements improve clarity and relevance. Students can document prompt evolution in journals and explore backtracking when outputs are misleading.

For example, when exploring algorithm efficiency for the chi-squared statistic, students might start with a prompt such as "Write R code to compute the chi-squared test." As they refine it to specify input type, desired output, and documentation style, they witness how precise prompting yields code that is both correct and readable. Journaling these iterations–and revisiting an early version that produced an error or an off-target response–helps students diagnose miscommunication with genAI in much the same way they would debug their own scripts. In Section 4, I discuss specific examples of such iterative prompting relative to a chi-squared test exercise.

Technical prompting techniques enhance structured reasoning. Few-shot prompting (Brown et al., 2020) uses embedded examples to guide style and structure: e.g., "Explain the meaning of a logistic regression coefficient. For example: If the coefficient for `OverTime` is 0.7, this suggests that employees who work overtime have higher odds of attrition. Now explain the coefficient of -1.2 for `RemoteWork`." Chain-of-thought prompting scaffolds reasoning by asking for stepwise logic: e.g., "First explain how logistic regression works, then when to use it, then its assumptions." Least-to-most prompting (Zhou et al., 2023) further encourages decomposition of complex problems into tractable steps, an approach well-aligned with statistical modeling.

Reusable prompt templates offer an additional scaffold. These structured formats help students formulate questions with greater clarity and reproducibility. Hsu (2025) demonstrates that templated prompting can support abstraction and decomposition, skills foundational to statistical reasoning. Mollick and Mollick (2024) propose instructional "blueprints" that help guide learners toward clearer analytic articulation.

2.5 Educational Goals and Takeaways

Prompt engineering supports both statistical reasoning and broader educational aims. Students engage in question formulation, assumption-checking, structured iteration, and interpretation, all while developing metacognitive awareness and ethical scrutiny of algorithmic output. When students interrogate AI-generated content–challenging interpretations, testing assumptions, and refining their queries–they practice habits aligned with inquiry-based learning.

Prompt-based workflows also enable inclusive and differentiated instruction. Students can request output in different modalities (e.g., explanation, code, visualization) or levels of complexity, and they can test model behavior across subgroups or scenarios to identify equity issues. When framed as a collaborative process, prompt engineering transforms genAI from a shortcut to a platform for learning.

The following section demonstrates how these principles play out through a worked example involving a chi-squared test of independence. This case illustrates how prompting supports exploratory data analysis, simulation, modeling, and communication, grounding abstract strategies in practice.

3 Motivating problem and illustrative exercise: the chi-squared test

I will motivate potential collaborative opportunities with generative AI through the problem of testing for association between two categorical variables. The problem is one typically part of the introductory Statistics curriculum. Contingency table data also allows for discussions of basic exploratory data analysis, algorithms for computing the chi-squared test statistic, and statistical thinking and reporting when drawing inferences. In the supplemental material, I present the R Markdown file and the compiled pdf document of the lab practical I assign for collaborating with generative AI to study these aspects of the chi-squared test for association.

I encourage the reader to have the lab practical document open for reference while perusing this section and Section 4.

Though the assignment is at the level of a data science class for undergraduate juniors and seniors, in this case either of our required courses *Programming in Data Science* or *Statistical Computing*, I believe the assignment can be pared down for a freshman level elementary statistics course. I also used this assignment for discussing responsible and ethical use of genAI for drawing and reporting statistical inferences in a graduate (MS and PhD) level statistical communication course I teach. In the supplementary materials, I also post an R Markdown file and compiled pdf document of code composed from collaborations with genAI.

Quinnipiac University ran a poll in early 2023 asking a randomly selected group of respondents "Would you support or oppose a national ban of foreign technology such as TikTok, the video-sharing service?" (Malloy and Schwartz, 2023). We wish to determine if support of this ban is associated with age grouped into two categories: adults 18-49 and adults older than 50.

For purposes of notation, and review, I provide the following details to the students. Let us denote the observed 2×2 table as

Observed table	Age <50	Age ≥ 50	row totals
Support ban	o_{11}	o_{12}	$r_1 = o_{11} + o_{12}$
Oppose ban	o_{21}	o_{22}	$r_2 = o_{21} + o_{22}$
column totals	$c_1 = o_{11} + o_{21}$	$c_2 = o_{12} + o_{22}$	*N*, grand total.

Under the null hypothesis of no association between age and support of the ban, the expected counts in the 2×2 table are

Expected table	Age <50	Age ≥ 50	row totals
Support ban	$e_{11} = \frac{r_1 c_1}{N}$	$e_{12} = \frac{r_1 c_2}{N}$	r_1
Oppose ban	$e_{21} = \frac{r_2 c_1}{N}$	$e_{22} = \frac{r_2 c_2}{N}$	r_2
column totals	c_1	c_2	*N*, grand total.

The chi-squared statistic measures the distance between the observed and expected table. If the distance metric is "unusually" large, then the observed table is different than what we would expect under the null hypothesis of no association and conclude that age is associated with support of the ban. The chi-squared statistic is

$$X^2 = \sum_{i,j=1}^{2} \frac{(O_{ij} - E_{ij})^2}{E_{ij}} \dot{\sim} \chi_1^2, \text{ under } H_0, \ (1)$$

where we are using capital letters to denote the observed and expected counts as random variables. The *p*-value for the test is thus

$$p - value = P(\chi_1^2 > x^2 \mid H_0 \text{ true}).$$

I allow students to use any generative AI chatbot of their choosing. Since my University secured licenses with OpenAI ChatGPT, Microsoft Copilot, and Google Gemini versions that allow unlimited uploads and more advanced functionality than the free versions, students gravitate to these three chatbots. At institutions where some students use free versions of chatbots, instructors must take care to create classroom material and write assignments that can be solved with the free version. Of particular note, the free versions often severely limit the number of files that may be uploaded in a day. In this paper, I focus on a data set that may be presented in text or in R code, and stay away from prompts that require files uploaded, be it excel spreadsheets or pdf documents. As academic institutions start broadly purchasing licenses for students to use subscription-based chatbot tools, our collaborations with generative AI may be scaled-up accordingly. However, I stress the importance of working within an inclusive and open access environment in a course, allowing your students to successfully and equitably collaborate with generative AI.

The data in the following code chunk is based on this opinion poll problem. I include this code chunk in my first prompt to generative AI as we will see in Section 4.1.

```
# Quinnipiac poll released 3/15/23: (Malloy & Schwartz, 2023)

# Would you support or oppose a national ban of foreign technology such

# as TikTok, the video-sharing service? Data based on this opinion poll

ct = matrix(c(41, 87, 110, 58), nrow=2 ) # data contingency table

row.names(ct) = c("Support", "Oppose") # rows: support or oppose ban

colnames(ct) = c("Age < 50", "Age > =50") # columns: age groups

rowtot = rowSums(ct) # row sums coltot = colSums(ct) # column sums

N = sum(ct) # grand total
```

4 A collaboration with generative AI

In this section, I illustrate several ways students can collaborate with generative AI in applied statistics work, extending the applications described by Sturdy (2024). These collaborations include automating and documenting code to streamline repetitive tasks; optimizing code for efficiency by minimizing computational expense and conception cost, benchmarking performance, and clarifying syntax and workflow; optimizing code for readability through consistent spacing, indentation, and documentation, as well as explaining code in plain language; learning from generative AI by exploring programming syntax and functionality, writing functions with well-defined inputs and outputs, implementing libraries, and converting code across languages; and using genAI as a debugging partner to diagnose and refine analytical workflows. Each example is framed as an opportunity for students to shift attention from mechanical coding toward statistical reasoning, interpretation, and communication.

4.1 Exploratory data analysis (EDA)

We may collaborate with generative AI on specific data visualizations or for perusing alternative visualizations of the contingency table data. Generative AI can thus be used to bounce around EDA ideas or present relevant R code for a specific visualization. For novice students, either to Statistics or to R, I encourage the use of generative AI as a partner in the learning process. Generative AI eases the burdens of coding, and hopefully on debugging, R visualizations. But in my opinion the students need to understand Statistics and R syntax to error check code produced by generative AI and interpret graphics output. As an advanced R user, I regularly forget the syntax for `ggplot` visualizations and typically do a quick Google search to get me started and/or find the relevant keywords. I have found collaborations with generative AI ease the coding process.

4.1.1 Class discussion on EDA prompts: a stacked bar chart

Let us start with creating a stacked bar chart of the data. I first feed the chatbot the brief data description and R code chunk at the end of Section 3.

Prompts concerning a stacked bar chart

RL: [Paste R code chunk above.] Please provide R code for a stacked bar plot to visualize the contingency table of opinion poll data.

RL: Please explain the code in plain language.

RL: I notice that you use the melt function for the plot. What does the melt function do and why do you need it?

The chatbot provides R code, using `ggplot` (Wickham, 2016), to draw a stacked bar plot of the contingency table data. Here is code from the chatbot.

```
# Data based on this opinion poll

ct = matrix(c(41, 87, 110, 58), nrow=2 )

row.names(ct) = c(”Support”, ”Oppose”)

colnames(ct) = c(”Age < 50”, ”Age > =50”)



ct_melted = melt(ct)

ggplot(ct_melted, aes(x=Var2, y=value, fill=Var1)) +

geom_bar(stat=”identity” ) +

labs(x=”Age Group” , y=”Count” , fill=”Support/ Oppose” ) +
```

```
theme_minimal()
```

I find that chatbots often do a poor job of documenting code. I thus either explicitly ask for documentation, see Section 4.3 for examples, or prompt for a "plain language explanation" of the code as done in my continued prompting in the textbox above. The chatbot provides a very detailed explanation of the code broken up into subsections of text. Once learning that I am creating say a R Markdown lab practical or instructional materials, the chatbot will follow-up asking if I want it to produce an R Markdown template or file for me.

I also ask the chatbot to explain syntax or R functions with which I am not familiar. Here the chatbot uses the `melt` function (Wickham, 2007), which I know will be new to my students. The detailed description explains the difference between long- and wide-formatted data, a construct I explain in class during an EDA module or task. The R Markdown file of code in the supplemental material presents an example of these formats. The response also explains why the chatbot believed the `melt` function was needed here and gives an example of its use. We can discuss changes in R functionality as well since the `pivot_longer` function may be preferred to the `melt` function.

4.1.2 From the lab practical: Task 1 EDA

I ask students to collaborate with generative AI to consider alternative options for visualizing the contingency table data. This task led to three observations.

• `ggplot`: In a first pass, the chatbot presents code using R base graphics. This conversation provides a learning opportunity in prompt engineering: if you want a specific coding construct, in this case `ggplot` code, you should probably explicitly ask the generative AI for it.

• *debugging*: The chatbot suggests a mosaic plot as an option (Jeppson et al., 2021), but has trouble correctly coding the `ggplot` aesthetic for the mosaic geometry. Here is the initial code produced by the chatbot. The error I received when running in R is: Warning: Computation failed in `stat_mosaic()`. Caused by error in `op()`: ! is.call(x) || is.name(x) is not TRUE

```
# Convert matrix to a data frame

ct_df = as.data.frame(as.table(ct))

# Plot mosaic plot using ggmosaic

ggplot(data = ct_df, aes(x = x, fill = y)) +

geom_mosaic(aes(weight = Freq)) +

labs(x=”Age Group” , y=”Support/Oppose” , fill=”Support/Oppose” ) +

theme_minimal()
```

Since I allow students to use any chatbot, I do not assume that they are confronted with the same error, or that their chatbot even suggests a mosaic plot. I give students the erroneous code and, to help in the collaborative debugging process, a correct application of the mosaic geometry I wrote myself.

```
ggplot(data = vizdata) +

# mosaic plot geometry

geom_mosaic(aes(weight=counts, x=product (age), fill=support)) +

labs(x = ”Age”, y = ” ”,

title = ”Support of a TikTok ban by age categories”,

fill = ”Support ban”) + # axis labels and

legend theme(plot.title = element_text(hjust = 0.5)) # center the title
```

In the debugging iteration, the chatbot provides code that left out the contingency table counts, giving me a square with four equally sized colors as the mosaic. It also seems to be hung up on whether I had the most recent `ggmosaic` package installed. I thus have to explicitly tell the chatbot to focus on the aesthetic and include the counts as a weight. For example, here are two of my final prompts. *RL: The mosaic plot from ggmosaic does not look right. It is not weighted by the contingency table counts, "cells" are the same size. Doesn't the geom_mosaic aesthetic need to use the contingency table counts? RL: It still does not work. I think the aesthetic needs to be placed in the geom_mosaic call, not in the original ggplot statement.* Here is the chatbot's response upon finally providing correct code. *Chatbot: You're right–if geom_mosaic isn't displaying correctly with the weight aesthetic in the ggplot call, it should indeed be specified directly within geom_mosaic. Here's how to correctly set up the mosaic plot with ggmosaic, ensuring that the counts are properly reflected: [R code chunk presented here].* Some students of course had an easier debugging collaboration with their chatbot. I share my conversation, emphasizing that first a quality control is required of all code the generative AI provides. Second, do not assume generative AI can debug on its own. It is fallible and, in my experience, often requires one's own analysis of the code and specific prompting to resolve issues.

• *Box plot*: The chatbot suggests a box plot, an inappropriate visualization for a contingency table in my mind. I leverage this opportunity to stress the importance of a scientific evaluation of generative AI output and responses. I also ask students to query the generative AI on the seemingly strange recommendation it made here. Interestingly, in response, the chatbot provides a detailed explanation of why the box plot is not a good visualization of categorical data, contrary to its original recommendation, and suggests more appropriate alternatives.

Statistical thinking: I come back to statistical inference and communication throughout the lab practical, and particularly as part of the collaboration with generative AI. In this EDA task, I ask students to assess all visualizations constructed from the generative AI code (which is best? which is worst? why?) and make an initial assessment on the association of support of a national ban of foreign technology and age group.

4.2 Algorithm development

I have found generative AI very useful for code optimization. For example, the chi-squared test statistic (1) consists of a sum of the distance metric $(observed - expected)^2 / expected$ over each cell of the table. We can think of an algorithm that first computes the expected cell counts and then computes the chi-squared statistic. Students will often code this computation in multiple nested loops as it is conceptually easy, though computationally inefficient in R.

```
# table of expected counts,
# assuming no association between age groups and support of the band
ct_exp = matrix(, nrow = 2, ncol = 2) # expected count contingency table
for (R in 1:2 ){ # computing expected counts one-by-one
for (C in 1:2 ){
ct_exp[R,C] = rowtot[R]*coltot[C]/N
}
}
row.names(ct_exp) = c("Support", "Oppose") # rows: support or oppose ban
colnames(ct_exp) = c("Age < 50", "Age > =50") # columns: age groups


# Compute the chi-squared statistic using a for-loop
chisqstat = 0 # storage for the chi-squared statistic
for (R in 1:2 ){ # loop through each cell
for (C in 1:2 ){
# cumulative sum of distances for each cell
chisqstat = chisqstat + (ct[R,C]-ct_exp[R,C])^2/ ct_exp[R,C]
}
}
```

I provide this code chunk to the students in the lab practical (see the supplemental material). Let us work with generative AI to optimize this code.

4.2.1 Class discussion on prompts: optimizing code for efficiency

I submit the data set code chunk from Section 3 along with the nested-for-loop code chunk above and ask the chatbot to optimize it for efficiency.

Prompts to optimize code for effciency

RL: May I send you R code to optimize for efficiency?

RL: The following is a contingency table of data. [R code chunk of the data presented here]. I have the data presented with R code as a matrix where the rows and columns are labeled. The row, column, and grand totals are also computed. Here is my code to compute the table of expected counts and the chi-squared statistics. [R code chunk of the nested for-loops presented here.] Can you please optimize this code for efficiency?

Though the chi-squared test statistic may be computed in one line, the chatbot chooses to separate computation of the expected cell counts and computation of the chi-squared statistics. I think such a decision helps with code readability.

```
# [Quinnipiac opinion poll data here]

# Compute the table of expected counts assuming no association

ct_exp < - outer(rowtot, coltot) / N

# Compute the chi-squared statistic

chisqstat < - sum((ct - ct_exp)^2 / ct_exp)
```

We can also use generative AI to help us set up R code to benchmark algorithms.

Prompts for benchmarking algorithms

RL: I can think of four ways to code computation of the chi-squared statistic.

1. My code using a for-loop.

2. Your optimized code with vectorized operations.

3. Using matrix manipulations. Here is the R code:

```
chisqstatpieces = as.vector( (ct-ct_exp)/sqrt(ct_exp) )
chisqstat = t(chisqstatpieces)%*%chisqstatpieces
```

4. The R function chisq.test().

Can you provide R code using the R benchmark package to compare computational cost for these four methods?

Chatbot: You can use the rbenchmark package to compare the computational cost of the four methods for calculating the chi-squared statistic. Below is the R code to perform this comparison…

RL: Please use the microbenchmark package instead.

As another example of prompt engineering, I introduce, and want the chatbot to use, the `microbenchmark` package (Mersmann, 2024). However, I inadvertently wrote the prompt with the imprecise instruction "…provide R code using the R benchmark package…" The chatbot responds with code using the package `rbenchmark`. My second prompt specifically cites the `microbenchmark` package; the code produced by the chatbot appears in the supplementary material.

4.2.2 From the lab practical: Task 2 Computing the chi-squared statistics

I leverage this computing exercise to show students how to learn R coding constructs from genAI.

• In the optimized code, the chatbot uses the `outer` function. I have students query the generative AI with the prompt, *"Can you please explain why the outer function correctly computes the expected table of counts?"* The chatbot specifically provides a textbook description of an outer product using matrix manipulations, a small scale example for illustration, and an explanation of why it is useful for optimizing code.

• For a 2×2 contingency table, the chi-squared statistic is equivalent to

$$X^2 = N\cdot\frac{\left(o_{12}o_{21} - o_{11}o_{22}\right)^2}{r_1 r_2 c_1 c_2}.$$

I query the chatbot with the prompt: *"Use the formula for 2x2 tables which takes the ratio of the grand total divided by the product of the row totals and column totals, and then multiplies that number by the square of the difference between the diagonal product and off-diagonal product."* The chatbot finds and codes the formula specific to 2×2 tables, and offers to add this formulation/algorithm to the benchmarking.

Statistical (computing) thinking: I ask students to compare the chi-squared statistic formulations relative to computational expense (the benchmarking), conception cost (time to design and code the algorithms), and statistical efficiency. Here statistical efficiency is not really an issue, though we do need an algorithm that easily generalizes to handle $R\times C$ tables. In Section 4.3, I use generative AI to learn about built-in R functions for drawing statistical inferences. For chi-squared tests, the function is `chisq.test`. Not surprisingly, the optimized chatbot code is benchmarked as the most efficient in all summary statistics of time over 10000 repeated runs of the algorithms. I use this opportunity to discuss why my own matrix computation code that uses matrix multiplication is less efficient than the chatbot optimized code that uses element-by-element matrix computations and the `sum` function–it comes down to `flops` particularly with the `sqrt` function. We also discuss why the built-in R function `chisq.test` is the least computationally efficient–this is not a fair comparison because `chisq.test` is doing a lot more than just computing the chi-squared test statistic.

4.3 Automating and documenting code

Code readability is critical when sharing code with collaborators and the community, as well as when your future self revisits or reuses programs you wrote. One example is organizing code through well documented functions. We often have or compose working code chunks to solve a problem. The task of breaking the code into functions and filling in documentation details is a bookkeeping task I think well suited for generative AI.

The lab practical in the supplemental material asks students to prompt generative AI to compose a function that will run the optimized chi-squared test statistic from Section 4.2 and draw inferences. I recommend an explicit prompt that details the precise inputs and outputs. Here, the function inputs an observed contingency table of any size and outputs the contingency table of expected counts, the chi-squared test statistic, and the *p*-value. We also may have a documentation style in mind, and should explicitly lay the plan out in the prompt. Here, I want the function headed with a description in plain text of the inputs, outputs, function operation, and example of use. I will not provide the function output by the chatbot here as it is a solution to the lab practical I assign the students. In this section, I will thus focus exclusively on Task 3 in the lab practical aimed at automating and documenting code. The class discussion centers on a prompt querying the chatbot to provide a function for a simulation study on the chi-squared test, which I will come back to in Section 5 .

4.3.1 From the lab practical: Task 3 prompts on R functionality

Generative AI may be queried to inform us of R packages and functions appropriate for a statistical analysis or inferential task. Here are prompts about the chi-squared test.

Prompts for a built-in R function for chi-squared testing

RL: What built-in R function can I use to perform a chi-squared test?

RL: Please provide R code that runs a chi-squared test of association between support of a ban and age group. Please use the R function chisq.test. As part of the R code, please use the R pander package to output a table of the chi-squared statistic to 2 decimal points and the p-value to 4 decimal points. Also please use the R pander package to output a table of expected counts. Please provide informative labels of all rows and columns of the tables.

In my experience, chatbots code output presentations as R core dumps using the `print` function. I require my students to present neat, clear, well-labeled tables. I use `pander` for this purpose and thus am in the habit of prompting genAI for tables using the `pander` package (Daroczi and Tsegelskyi, 2022).

The chatbot outputs the following code, which uses the `round` function to format table cells.

```
# Perform the chi-squared test

test_result < - chisq.test(ct)

# Extract the chi-squared statistic and p-value

chisq_stat < - round(test_resultstatistic, 2) # Round to 2 dec
```

```
p_value < - round(test_resultp.value, 4) # Round to 4 dec

# Output the chi-squared statistic and p-value using pander

pander( data.frame(Statistic = chisq_stat, P_value = p_value),

caption = ”Chi-Squared Statistic and P-Value”)
```

We may query generative AI about function options and operation.

Prompts about built-in R functions

RL: How come you prefer using the round function instead of the signif function or the formatC function in R?

RL: I notice that the R function chisq.test() can use simulation to approximate a p-value. How does it do that?

RL: I also notice that the R function chisq.test() has an option to apply a continuity correction. What is this continuity correction? And when should I use it?

RL: What assumptions are required to draw inferences from the chi-squared test?

Generally speaking, coding collaborations with generative AI will teach students alternative syntax and approaches to algorithm development, good or bad. When used as a partner in the learning process, generative AI is also an excellent tool for gaining proficiency in a new language. For example, students fluent in Python and new to R can ask generative AI to translate Python code they wrote into R. I stress that generative AI is not a tool for alleviating the need to learn how to code. The user must evaluate and test code produced by generative AI, in fact a quality control assessment that contributes to the learning of a new programming language.

4.3.2 From the lab practical: Task 3 Inferences

This task focuses on statistical inferences from the chi-squared test, motivating discussions on statistical thinking.

• Rather than copying the whole conversation transcript into a solutions document, I ask students to summarize the chatbot’s, usually extensive, response. In this way, students must read and understand the response.

• An understanding, and assessment of assumptions is important for any inferential or modeling task. I ask students to query generative AI on assumptions for drawing inferences from a chi-squared test.

• As part of the conversation about `chisq.test` , the chatbot brings up Fisher’s exact test as an alternative method for contingency tables with small counts. If unfamiliar with this method, the student can continue down the rabbit hole and learn about Fisher’s exact test.

• I stress the presentation of clear and concise tables for output reporting and statistical communication generally. In Section 4.3.1 , I explicitly ask the chatbot to use `pander` (Daroczi and Tsegelskyi , 2022) and format output to a specific number of decimal places. The chatbot chooses to use the `round` function for the latter task. I also introduce `signif` and `formatC` functions in class. We can ask generative AI to compare these options. The chatbot provides a detailed explanation of why it chose `round` , differences among these three options, examples of use and code, and concluding thoughts. As part of the explanation, the chatbot provides a counter argument that `signif` constrains the number of significant digits, not necessarily decimal places, and `formatC` , though flexible, outputs a character string rather than a numeric value.

• I continue to impress on students the importance of running and checking any code provided by generative AI. A natural assessment here is to have the students run the chi-squared test and draw an inference: Is age associated with support/opposition of the ban? Why?

• The students will find that output from the chatbot optimized function from Section 4.2 and `chisq.test` is most likely a little different. The R function `chisq.test` uses the continuity correction as a default option. In the lab practical and subsequent discussions, I impress on students the importance of not only quality control checks of genAI-produced code, but understanding (default) options of R functions to set up say a gold standard for comparison.

5 Class discussion: Coding a simulation study

Simulation studies are a critical part of statistics research and analysis as we aim to understand new methods and assess assumptions for and efficacy of inferences. Once I have a detailed outline or even pseudocode for a simulation study, I find the coding task to be primarily bookkeeping on loop syntax, storage, and output. I have found generative AI an excellent collaborator for constructing simulation code. In this section, I highlight elements of my class discussion on coding a simulation study.

5.1 Prompts for coding a simulation experiment

Let us assess the level of the chi-squared test. In front of the class, I use the prompts below to ask the chatbot to simulate data replicates under the null hypothesis of no association and compute the proportion of simulated contingency tables where the null hypothesis is rejected. For reference purposes, the simulation study function output by the chatbot appears in the supplemental material R Markdown file of code.

Prompts to code a simulation study

RL: Write R code to perform a simulation study to assess the type 1 error rate of a chi-squared test. Consider a 2 by 2 table with sample size of 100. Generate data under the null hypothesis of no association using a multinomial distribution with equal cell probabilities of 0.25 for each of the four contingency table cells. Report the empirical type 1 error rate as the proportion of tables where the null hypothesis is rejected using a chi-squared test. Output the type 1 error rate to four decimal places. Do not use a continuity correction and do not use a simulation-based p-value.

RL: Write the simulation study code to take as input the number of simulations, the level of the test, and the sample size. Also take as input whether to use the Yates's continuity correction in the chisq.test() R function. Output the type 1 error rate and then print the result to four decimal places. Please provide documentation at the start of the function that states, in plain text, the inputs, outputs, and a brief description of what the function is doing, and a brief example of its use?

RL: Please explain the function in plain language.

Notice the long prompt for simulation study code. I encourage students to explicitly specify the simulation model and the output desired. I hit a number of points in the class discussion:

• *Simulation study design* : For a successful prompt, we must plan out the simulation study ahead of time, and understand the purpose of the study. The preparation work also helps streamline the collaboration with generative AI, focusing exactly on the inputs, outputs, and processes for the simulation study.

• *Simulation study coding construct* : I want a function for the simulation to allow me to easily run it under different input scenarios. Since the chatbot did not provide a function in response to my first prompt, I followed up with a prompt laying out the function parameters and documentation, including options for the `chisq.test` function.

• *Feedback from the chatbot* : The chatbot provides a detailed explanation, allowing the user, in a quality control phase, to check if the prompt was correctly interpreted. The explanation is concise and specific to the code syntax, but even with the documentation, I often find functions hard to read. I thus also ask the chatbot to explain the function in plain language.

5.2 Statistical thinking

I use this simulation experiment as an interactive class discussion rather than a lab practical for students to complete on their own. Some additional discussion points I make relative to statistical inference and communication:

• I assess students on their understanding of the simulation study. Here, why is the proportion output an estimate of the level α ? How does one simulate data under the null hypothesis of no association, say for a 2×2 table?

• It is important to understand the impact of inputs on inferences. Here students may play around with the number of replicated data sets, the sample size, true α , and the continuity correction, and evaluate changes in the type I error rate.

• I also ask students to think about extensions to the simulation study for other models, inferences, or metrics. Here for example, we can wonder how to assess the type II error rate or power of the test.

• I again stress the importance of output reporting and statistical communication. How can we clearly and concisely summarize simulation results for a scientific audience via an output table, visualization, and text?

6 Class discussion: On the responsible use of generative AI

As mentioned in Section 1 , I am finding that all students in my classes have used generative AI extensively as part of their studies. In this section, I highlight aspects of the class discussion in the latter part of the module on responsible use of generative AI. The discussion is centered around the EVERY framework which I will detail first. I then give final thoughts on this module and discussions therein. The lesson plan in the supplemental material outlines logistics in my design of this class discussion period.

6.1 EVERY framework

I impress on students that it is their responsibility to ensure output used from generative AI is accurate, appropriately cited, reliable, and valid. AI for Education (2024) coined the acronym EVERY:

Evaluate generative AI responses to ensure it satisfies the intended purpose of your prompt.

Verify output produced by generative AI be it code, analyses, visualizations, or explanations via your own quality control checks and/or against reliable sources.

Edit your prompt to improve the generative AI response and dig deeper into areas of confusion or knowledge gaps.

Revise the generative AI response relative to your needs.

You are responsible for output you present from generative AI.

I use the EVERY heuristic infographic on the *AI for Education* website and one on ethical use of AI in education by Scott (2024) . Generative AI is a collaborative partner in the learning process through iterative problem-solving. To this end, it is counter-productive to plagiarize generative AI output or use generative AI as a solutions generator without your own input or attempt to understand the output, so-called *assignment outsourcing* . I prompt this discussion by asking students to give examples of each letter in the EVERY heuristic and brainstorm on ethical and unethical uses of generative AI for school work, prior to showing the infographics.

Integrating EVERY into statistical thinking : Instructors may find it helpful to frame EVERY not only as a guideline for ethical use, but also as a mirror of the statistical problem-solving process. Evaluating output reflects assumption-checking. Verifying results reinforces the importance of reproducibility and robustness checks. Editing and revising prompts parallel the model-building and refinement phases in data analysis. And the final “You” emphasizes personal accountability in presenting and interpreting statistical results. This framing helps students internalize responsible AI use as part of their broader statistical training.

Reinforcement beyond the module : Although I introduce these concepts early in the semester, instructors are encouraged to revisit them throughout the course. For instance, when teaching regression, students can critique AI-generated models or explanations using the EVERY framework. During final projects, students can reflect on their AI interactions and assess the quality, bias, and appropriateness of the tools they used. Embedding the EVERY framework across content helps students see ethical AI use as integral to data science rather than an add-on.

Bias in generative AI responses : The discussion on responsible use very naturally leads to interesting statistical discussions of *algorithmic bias* . As with any machine learning tool, generative AI responses are only as good as the data used to train the large language models. The chatbot responses may privilege a group or category based on representation in the data or, in the extreme, construct a logical path that leads to meaningless or counterfactual conclusions, called *hallucinations* . I end the conversation with the discussion prompt: "Should we just not use generative AI due to these potentially very harmful biases?" I view generative AI as a very useful tool in the learning process. We must take great care in responsible and ethical use. The United Nations Educational, Scientific, and Cultural Organization (UNESCO) AI in higher education primer provides an infographic flowchart to answer the question "When is it safe to use generative AI?" (UNESCO , 2023) .

Strategies for teaching responsible use : I encourage instructors to facilitate discussion-based and scenario-based activities. One effective exercise is to present flawed generative AI output and have students use the EVERY framework to identify issues. Another is to assign paired roles in which one student serves as the "AI assistant" and another as the statistician, prompting critical dialogue and revision. These activities promote active learning and demystify generative AI collaboration.

6.2 Final thoughts

Having discussed the EVERY framework and in-class strategies for teaching responsible AI use, I close with broader reflections on what these ideas mean for the future of statistics education. This section turns from classroom-level practice to program- and discipline-level considerations, focusing on equitable access, transparency, and emerging tools, such as customized chatbots, that can extend responsible collaboration with AI beyond this module.

Equity and access : Instructors should be mindful of equity issues. Not all students may have access to the same generative AI tools, especially premium versions. Additionally, language or cultural barriers may affect how students interpret or interact with chatbot output. Providing shared tools and discussing inclusivity in AI design promotes an equitable classroom environment.

Hands-on practice : Students work on a version of the lab practical presented in the supplemental material. The chi-squared test provides an introductory statistics inference method as the driving example. But of course we can plug-and-play depending on the course content and level. For example, my forthcoming book Levine (2026) includes predictive analytics problems (in Chapter 7 of the book) of collaborating with generative AI to build a k -nearest neighbors (kNN) classification algorithm, simulate the impact of k on the kNN classifier performance, debug a kNN classifier and a naive Bayes classifier, and compose a function of classification performance metrics.

Encourage students to try different chatbots : Students will gain varied insights and perspectives by conversing with different generative AI tools. The differences themselves can be instructive, prompting discussion about reliability, verification, and how students might adapt their collaboration style to different systems. GenAI responses vary by platform, version of the large language model (LLM), iterative prompting over time, and user personalization. I purposefully avoid presenting lengthy transcripts or detailed comparisons in the main text, as such content would quickly become outdated. The supplemental materials include a brief document comparing chatbot responses to the same prompts used in Sections 4 and 5 , illustrating how instructors might use these differences to foster critical evaluation of AI-generated output. Bowen and Watson (2024) provide additional guidance on teaching with chatbots, including links to a variety of instructional resources for different classroom contexts.

Transparency and accountability : I impress on students that, much like citing sources and contribution acknowledgements in one's writing, we need to be transparent and honest about our use of generative AI. I require a *design statement* at the start of every assignment submitted explaining processes taken and chatbot tools used in collaboration with generative AI to produce the final product submitted for grading. My university recommends a one-paragraph design statement template taken from the Elsevier journal *Social Science & Medicine* that I use in my courses:

"During the preparation of this work the author(s) used [NAME TOOL / SERVICE] in order to [REASON]. After using this tool/service, the author(s) reviewed and edited the content as needed and take(s) full responsibility for the content of the [report]."

Suggested resources for instructors : Those seeking additional frameworks and readings may consult the UNESCO AI in Education guidelines (UNESCO , 2023) , the Harvard Data Science Review article "Doing Data Science: A Framework and Case Study" (Donoho et al. , 2021) , and the paper "Teaching Responsible Data Science" (Stoyanovich and Lewis , 2021) . These works provide curricular structures, classroom activities, and ethical frameworks that pair well with the EVERY heuristic.

This article focuses on pedagogy rather than the technical workings of large language models or on empirical evaluations of learning outcomes; those areas warrant separate treatment. Future work in Statistics education could explore these directions further, including the development and classroom evaluation of customized chatbots designed to support data analysis and statistical reasoning.

SUPPLEMENTARY MATERIAL

The supplemental material provides resources that mirror the instructional module described in Section 1 and support replication or adaptation of this approach in other Statistics courses. They include (a) a detailed lesson plan outlining pre-class readings, formative assessments, and discussion prompts for the one-week module on responsible use of generative AI; (b) the lab practical assignment and compiled output used in class demonstrations and student work; (c) the R code and chatbot-generated responses associated with the prompts discussed in Sections 4 and 5 ; and (d) a brief comparison of responses from ChatGPT, Copilot, and Gemini to the same set of prompts, illustrating how platform differences can be leveraged for classroom discussion about reliability and model variability.

All collaborations described in this article were conducted using ChatGPT 4o (OpenAI , 2025) , Copilot Quick Response (Microsoft , 2025) , Gemini 2.5 Flash (Google , 2025) , and R 4.5.0 (R-Core-Team , 2025) on an iMac (3.7 GHz Intel 6-core i5, 16 GB RAM).

Responsible use of generative AI module lesson plan:
`responsible_use_genAI_lesson_plan.pdf`

Generative AI lab practical: `genAI_LabPractical.Rmd` & `genAI_LabPractical.pdf` (R Markdown file and pdf file)

Code and output in response to prompts in Sections 4 and 5 : `genAI_code.Rmd` & `genAI_code.pdf` (R Markdown file and pdf file)

Prompting different chatbots: `supplmaterials_genAI_chatbots.pdf` discusses responses to the exact same prompts, without any iterative refinement, by three chatbots: ChatGPT, Google Gemini, and Microsoft Copilot in Sections 4 and 5 .

ACKNOWLEDGEMENTS

I am grateful to SDSU Professors Scott Goldberg and Elisa Sobo, as well as the SDSU Instructional Technology Services unit, for developing the Academic Applications of Artificial Intelligence (AAAI) micro-credential course and training–an initiative that inspired many of the ideas for integrating generative AI into my teaching. I also appreciate the insightful and energizing conversations about educational applications of generative AI with NC State Professor Jason Osborne and SDSU Professors Juanjuan Fan and Chris O'Neill. I thank an Associate Editor and reviewer for their valuable feedback, which contributed to a clearer framing and stronger presentation of the material. This project was supported in part by a California Education Learning Lab Data Science Challenge grant. The study was deemed exempt by the SDSU Institutional Review Board. The author reports there are no competing interests to declare.

References

Acar, O. A. (2023), ‘Ai prompt engineering isn’t the future’, *Harvard Business Review* . https://hbr.org/2023/06/ai-prompt-engineering-isnt-the-future

AI for Education (2024), *How to use AI responsibly EVERY time* . retrieved September 6, 2024. https://www.aiforeducation.io/ai-resources/how-to-use-ai-responsibly-every-time

Bowen, J. A. and Watson, C. E. (2024), *Teaching with AI: A Practical Guide to a New Era of Human Learning* , Hopkins Press. https://josebowen.com/

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A. et al. (2020), ‘Language models are few-shot learners’, *Advances in Neural Information Processing Systems* **33** , 1877–1901. https://arxiv.org/abs/2005.14165

Daroczi, G. and Tsegelskyi, R. (2022), *pander: An R ‘Pandoc’ Writer* . R package version 0.6.5. https://CRAN.R-project.org/package=pander

Donoho, D., DeNero, J. and Yu, B. (2021), ‘Doing data science: A framework and case study’, *Harvard Data Science Review* **3** (1). https://hdsr.mitpress.mit.edu/pub/hnptx6lq

Ellis, A. R. and Slade, E. (2023), ‘A new era of learning: considerations for chatgpt as a tool to enhance statistics and data science education’, *Journal of Statistics and Data Science Education* **31** , 128–133.

Goldberg, D., Sobo, E., Frazee, J. and Hauze, S. (2024), *Generative AI in higher education: insights from a campus-wide student survey at a large public university* , Association for the Advancement of Computing in Education (AACE), Waynesville, NC, pp. 757–766.

Google (2025), *Gemini* . Retrieved Summer 2025. https://https://gemini.google.com

Grolemund, G. and Wickham, H. (2014), *R for Data Science* , O’Reilly Media. https://r4ds.had.co.nz

Hsu, H.-P. (2025), ‘From programming to prompting: Developing computational thinking through large language model-based generative artificial intelligence’, *TechTrends* **69** (4), 485–506.

Jeppson, H., Hofmann, H. and Cook, D. (2021), *ggmosaic: Mosaic Plots in the ‘ggplot2’ Framework* . R package version 0.3.3. https://CRAN.R-project.org/package=ggmosaic

Levine, R. A. (2026), *Statistical Computing for Data Science* , Cambridge University Press.

Lo, L. S. (2023), ‘The clear path: A framework for enhancing information literacy through prompt engineering’, *The Journal of Academic Librarianship* **49** (6), 102720.

Malloy, T. and Schwartz, D. (2023), *Quinnipiac University Poll, released March 15, 2023* . Retrieved September 6, 2024. https://poll.qu.edu/poll-release?releaseid=3869

Mersmann, O. (2024), *microbenchmark: Accurate Timing Functions* . R package version 1.5.0. https://CRAN.R-project.org/package=microbenchmark

Microsoft (2025), *Copilot* . Retrieved Summer 2025. https://copilot.microsoft.com

Mollick, E. and Mollick, L. (2024), 'Stop writing all your ai prompts from scratch'. Harvard Business Publishing Education. https://hbsp.harvard.edu/inspiring-minds/an-ai-prompting-template-for-teaching-tasks

Moore, D. S. (1997), 'New pedagogy and new content: the case of statistics', *International Statistical Review* **65** (2), 123–165.

National Council of Teachers of Mathematics (2023), *Equitable integration of technology for Mathematics learning* . retrieved September 6, 2024. https://www.nctm.org/standards-and-positions/equitable-integration-of-technology/

OpenAI (2025), *ChatGPT* . Retrieved Summer 2025. https://chatgpt.com

R-Core-Team (2025), *R: A language and environment for statistical computing* , R Foundation for Statistical Computing, Vienna, Austria. http://www.R-project.org/

Rossman, A. J. and Chance, B. L. (2014), 'Using simulation-based inference for learning introductory statistics', *WIREs Computational Statistics* **6** , 211–221.

Saghafian, S. and Idan, L. (2024), 'Effective generative ai: The human-algorithm centaur', *Harvard Data Science Review* **6** (Special Issue 5). https://hdsr.mitpress.mit.edu/pub/1yo82mqa

Scott, M. (2024), *Education & AI* . retrieved September 6, 2024. https://scottybreaksitdown.com/ai/

Sobo, E. (2023), 'Could chatgpt prompt a new golden age in higher education?', *Teaching & Learning Anthropology Journal* **6** (1).

Stoyanovich, J. and Lewis, A. (2021), 'Teaching responsible data science: Charting new pedagogical territory', *International Journal of Artificial Intelligence in Education* **31** (4), 618–641. https://link.springer.com/article/10.1007/s40593-021-00241-7

Sturdy, I. (2024), 'Leveraging chatgpt in statistical programming in the pharmaceutical industry', *PharmaSUG 2024 Conference Proceedings* . Paper AP-256. https://www.pharmasug.org/us/2024/proceedings.html

Tu, X., Zou, J., Su, W. and Zhang, L. (2024), 'What should data science education do with large language models?', *Harvard Data Science Review* **6** (Special Issue 5). https://hdsr.mitpress.mit.edu/pub/pqiufdew

UNESCO (2023), *ChatGPT and artificial intelligence in higher education, quick start guide* , UNESCO, Paris, France. https://unesdoc.unesco.org/ark:/48223/pf0000385146.locale=en.

Wickham, H. (2007), ‘Reshaping data with the `reshape` package’, *Journal of Statistical Software* **21** (12), 1–20. http://www.jstatsoft.org/v21/i12/

Wickham, H. (2016), *ggplot2: Elegant Graphics for Data Analysis* , Springer-Verlag New York. https://ggplot2.tidyverse.org

Wild, C. J. and Pfannkuch, M. (1999), ‘Statistical thinking in empirical enquiry’, *International Statistical Review* **67** (3), 223–265.

Zhou, D., Schärli, N., Hou, L., Wei, J., Scales, N., Wang, X., Schuurmans, D., Cui, C., Bousquet, O., Le, Q. and Chi, E. H. (2023), Least-to-most prompting enables complex reasoning in large language models, *in* ‘International Conference on Learning Representations (ICLR)’. https://arxiv.org/abs/2205.10625

Table 1: Examples of effective and ineffective prompts in applied statistics tasks.

Intent	**Purpose**	**Effective Prompt**	**Vague Prompt**	**Common Issue**
Conceptual Understanding	Understand a core statistical idea	"What is the difference between confidence intervals and prediction intervals?"	"What is a confidence interval?"	May result in a generic definition without contrast or context
Exploratory Data Analysis (EDA)	Visualize or summarize data relationships	"Show me how to use R to explore the relationship between `customer churn` (binary) and numeric predictors like `usage time`, `number of logins`, and `support calls`. Include plots and summary statistics."	"Explore this data."	Lacks variables or goals; AI guesses intent or returns unhelpful output
Method Selection	Identify an appropriate method and assess assumptions	"What assumptions must be checked before fitting a logistic regression with categorical predictors?"	"Should I use logistic regression?"	AI may suggest a method without evaluating data type or assumptions
Pedagogical Prompting	Support learning through analogy or scaffolded explanation	"Use a simple analogy to explain how logistic regression estimates parameters."	"Teach me logistic regression."	May return dense or inaccessible technical explanations
Code Generation (R)	Generate R code for a known analysis task	"Write R code using `glm()` to fit a logistic regression with `Attrition` as outcome and `RemoteWork`, `Age`, and `JobLevel` as predictors."	"Give me R code for logistic regression."	May return incomplete or incorrect syntax; lacks variables or function details
Statistical Analysis	Execute a complete analysis task with a dataset	"Using the attached dataset of employee records, fit a logistic regression model predicting `Attrition`, check for multicollinearity, and summarize the results."	"Analyze this data."	Too vague; AI may default to inappropriate methods or skip important steps
Interpretation /	Translate results for a specific	"Explain what an odds ratio of 2.0 for	"Explain this	AI may use technical language

Communication	audience	`RemoteWork` means in plain language for an HR manager.”	number.”	or misinterpret the context