

Coding without Coding: Generative AI Support for Introductory Statistics


Dr ZHU Tianming


*Mathematics and Mathematics Education Department
National Institute of Education, Nanyang Technological University*

5 June 2026

The "Syntax Wall" in Education



 In teaching graphical descriptive statistics, hand-drawing is no longer practical for complex datasets; Just as a calculator is necessary for calculation, programming is the mandatory engine required for modern data visualization.

 The Singapore MOE secondary mathematics syllabus already includes *Data Handling and Analysis* as a core topic. Students are expected to work with real data.

 While programming is essential to statistics, it has become a major roadblock for beginners.

 As students transition from **No-Code** to **High-Code (R/Python)**, syntax errors often halt mathematical discovery and lead to student frustration.

 Limited class time makes it nearly impossible for educators to balance teaching technical programming with deep statistical concepts.

Before Generative AI: Teaching Visualization with R



Traditional Workflow

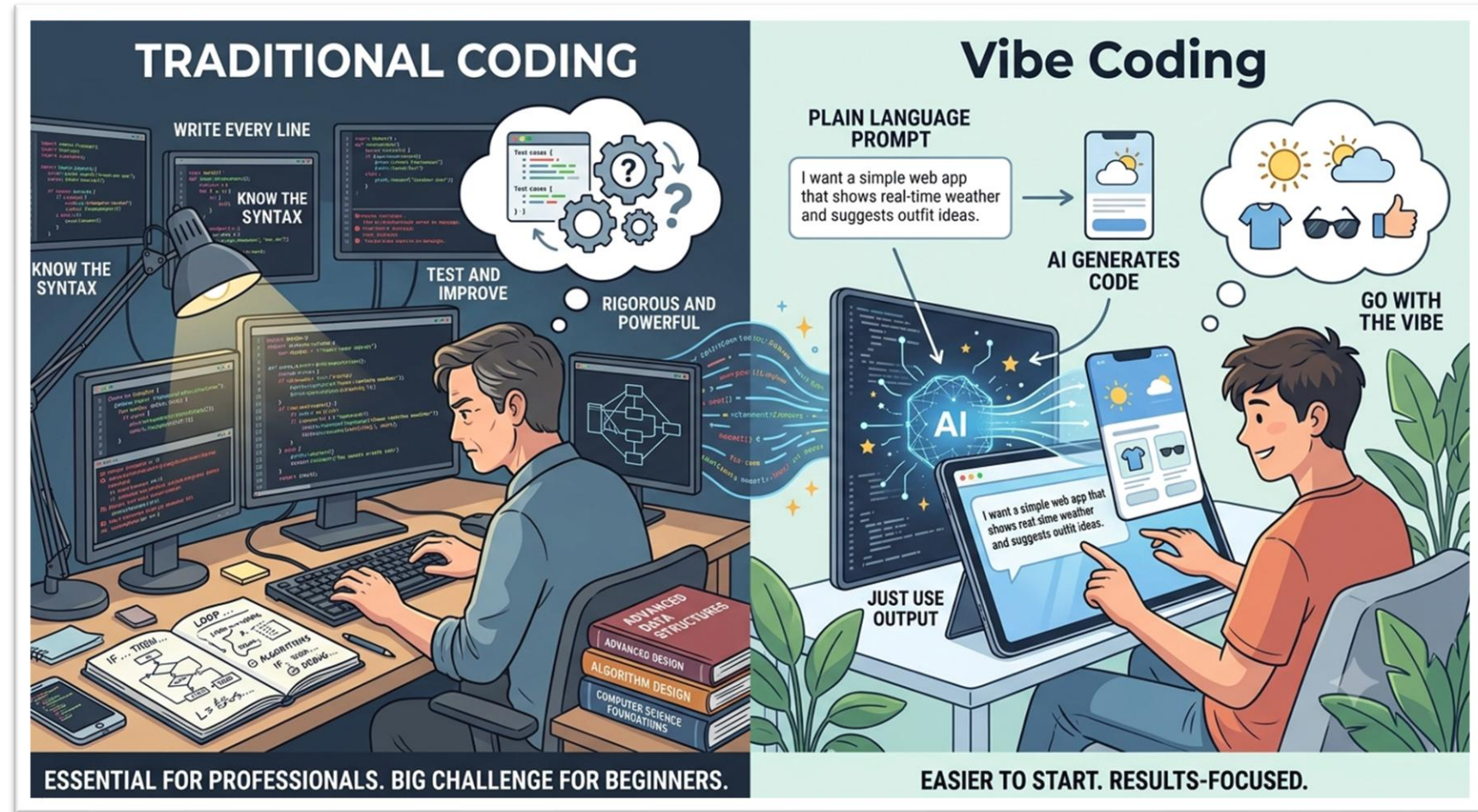


Common Classroom Challenges

- Students struggled with syntax before understanding concepts
- Small typing mistakes caused frustration
- Considerable class time spent debugging code
- Some students disengaged before reaching interpretation

Can we achieve “coding without coding”?

Traditional Coding vs Vibe Coding



We borrow the idea of vibe coding — but design it carefully for learning.

Current Existing AI Code Assistants



| Tool | Limitations for Teaching Beginners | Cost / Accessibility |
|-------------------------|---|--|
| GitHub Copilot | <ul style="list-style-type: none">▪ Cannot restrict to code-only mode easily▪ Students may get explanations and lose the chance to think independently | <ul style="list-style-type: none">• Paid subscription |
| ChatGPT / GPT-4 | <ul style="list-style-type: none">▪ Cannot reliably enforce code-only usage▪ May provide full analysis or even conclusions | <ul style="list-style-type: none">• Free tier limited• Paid for full access |
| Other AI Tutors | <ul style="list-style-type: none">▪ Often not customizable▪ May directly provide complete solutions including interpretations | <ul style="list-style-type: none">• Varies• Many require subscriptions |
| Colab Gemini / Posit AI | <ul style="list-style-type: none">▪ Some customization possible, but not always fine-grained control | <ul style="list-style-type: none">• Colab free• Posit free tier limited |

*Summarized by ChatGPT

Goals of This Approach



Real tools, real data

Use R or Python instead of hand-drawing



Code-only explanation

AI explains code structure, not data meaning



Early confidence building

Quick success with visible outputs



Gradual coding fluency

Repeated exposure builds programming literacy



Why Start with Data Visualization?

❑ Some statistical tasks already have simple tools

- Mean, median, and standard deviation can be calculated using a calculator.

❑ Some tasks require deeper mathematical understanding

- Linear regression and statistical modeling involve more advanced concepts and interpretation.

❑ Data visualization is an ideal starting point

- Real datasets are often too large for hand-drawn graphs.
- Programming becomes necessary for producing meaningful visualizations.
- Students can focus on patterns, distributions, and comparisons without advanced mathematics.



*Image generated by Gemini

Technologies & Capabilities



- **DifyLab Platform:** Built on GPT-4.1 and the [Dify](#) platform, optimized specifically for educational support rather than generic assistance.
- **Restricted Output:** The tool is strictly programmed to provide **code only**. It is forbidden from making decisions or interpreting data, ensuring students must perform their own statistical analysis.
- **Natural Language to Code:** Translate a student's intent (e.g., "I want to draw a histogram") into specific R or Python syntax.
- **Integrated Access:** Seamlessly embedded into the [NTULearn](#) course portal for easy deployment and student access.
- **Web-Based Running:** Works perfectly with [Google Colab](#) and [Posit Cloud](#), meaning students can run complex data analysis on any basic laptop without installing software.



DifyLab Instruction Design



<instructions>

1. You are a **teaching assistant** for first-year students learning data science or statistics.
2. Students describe a programming task without specifying the programming language.
3. **Generate code** in Python or R according to the language setting provided by the platform.
4. Use clear, descriptive variable names.
5. Explain how the student can supply the data required by the requested code.
6. **Do not choose or recommend** graphs, statistical methods, summary statistics, tests, models, or other analytical procedures on behalf of the student. Only implement methods explicitly requested by the student. If the requested method is not specified, ask for clarification instead of making a recommendation.
7. Include code to perform the task, with simple error handling where appropriate.

8. After the code, provide a brief, plain-language explanation for beginners.

9. **Do not perform data analysis or return results.**

10. Only output the code and explanation; no XML or other markup.

</instructions>

A screenshot of a web interface for setting up a new chat. At the top, it says 'New chat setup' with a blue speech bubble icon. Below this is a label 'language' and a text input field containing the word 'language'. At the bottom of the form is a prominent blue button labeled 'Start Chat'.

Case Study: Plotting Categorical Data in R

Example: “On the average day, about how many hours do you personally watch television?” This was measured as a discrete variable, with values 0, 1, 2, and so on. This table is one way of constructing a frequency table for the 905 subjects who responded. Display the frequencies using a pie chart and a bar graph, respectively.

| No. Hours | Frequency |
|------------|-----------|
| 0-1 | 232 |
| 2-3 | 403 |
| 4-5 | 181 |
| 6-7 | 45 |
| 8 and more | 44 |
| Total | 905 |



Conceptual Definition

The student identifies the statistical objective (e.g., visualizing a distribution) rather than the code syntax.



Guided Generation

DifyLab translates the natural language intent into precise R code, following predefined instructional guardrails.



Active Execution

The student runs the code in Posit Cloud, troubleshooting the environment while being shielded from syntax frustration.



Critical Interpretation

The student analyzes the output to draw statistical conclusions—a step the AI is strictly forbidden from performing.

Case Study: Plotting Categorical Data in R

- Then the students can draw the bar plot accordingly...

I want to draw a Bar Graph according to the data

Create vectors for study hours intervals and their frequencies

```
study_hours <- c("0-1", "2-3", "4-5", "6-7", "8 or more")  
hours_frequency <- c(232, 403, 181, 45, 44)
```

Draw a bar graph of study hours

```
barplot(hours_frequency, names.arg = study_hours,  
main = "Bar Graph of Study Hours",  
xlab = "No. of Hours Studied",  
ylab = "Frequency",  
col = "lightblue")
```

This code creates two vectors: one listing the study hours intervals and one listing the corresponding frequencies. It then displays a bar graph showing how many students are in each study hour group. You can type in the data as shown or use a file—just let me know if you need help reading data from a file.

let the bars be of different colours

Define the categories and counts

```
categories <- c("0-1", "2-3", "4-5", "6-7", "8 or more")  
counts <- c(232, 403, 181, 45, 44)
```

Generate different pastel rainbow colors for each bar

```
pastel_rainbow <- rainbow(length(categories), s = 0.5, v = 0.9)
```

Create a bar chart with bars in different colors

```
barplot(counts,  
names.arg = categories,  
main = "Frequency of Number of Hours",  
xlab = "Category",  
ylab = "Count",  
col = pastel_rainbow)
```

This code generates a bar chart in R with each bar shown in a different pastel rainbow color. The `rainbow` function is used with settings to make the colors softer. You type in the category names and counts manually, and each bar will have its own distinct color for easier visual distinction.

Change the colours of the bar. First will be blue, then orange, then green, then pink and then purple

If you don't have `ggplot2` installed, uncomment and run the next line:

```
install.packages("ggplot2")  
library(ggplot2)
```

Manually enter the hours and frequency data

```
hours <- c("0-1", "2-3", "4-5", "6-7", "8 or more")  
frequency <- c(232, 403, 181, 45, 44)
```

Create a data frame with the data

```
frequency_table <- data.frame(hours, frequency)
```

Define custom colours for each bar

```
bar_colours <- c("blue", "orange", "green", "pink", "purple")
```

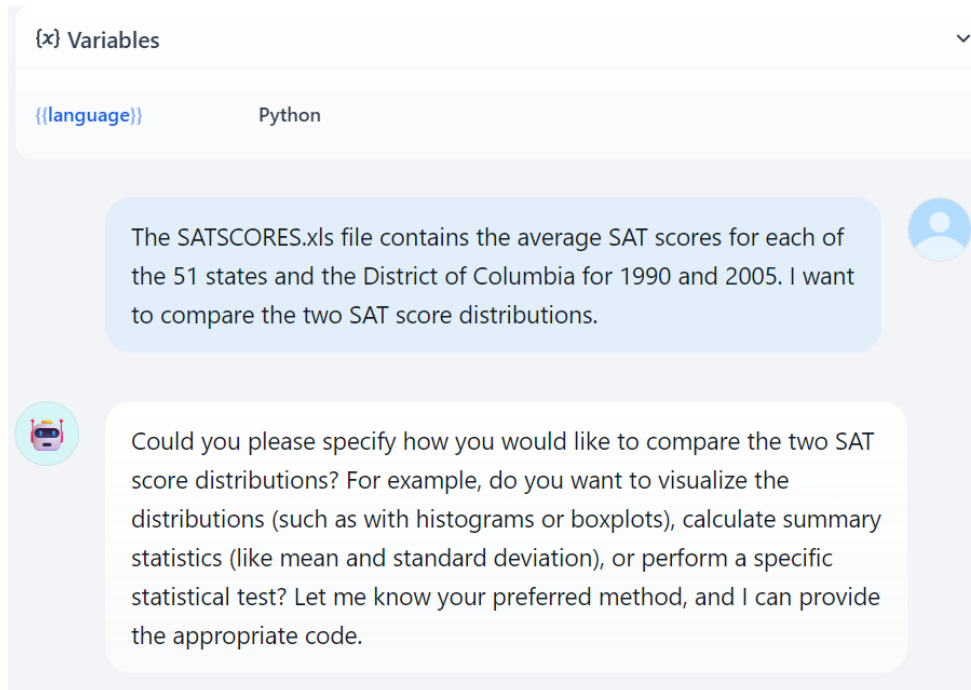
Plot the bar graph with specified colours

```
ggplot(frequency_table, aes(x = hours, y = frequency, fill = hours)) +  
  geom_bar(stat = "identity") +  
  scale_fill_manual(values = bar_colours) +  
  labs(title = "Bar Graph of Study Hours Frequency",  
       x = "Number of Hours",  
       y = "Frequency") +  
  theme_minimal() +  
  theme(legend.position = "none")
```

This code assigns a custom color to each bar in your bar graph. The first bar will be blue, the second orange, the third green, the fourth pink, and the fifth purple. The colours are set using a vector and applied with `scale_fill_manual`. The legend is removed for clarity. The input data is entered manually, so you don't need a file.

Example on using Python: Analyzing Excel Data

- The **SATSCORES.xls** file contains the average SAT scores for each of the 51 states and the District of Columbia for 1990 and 2005. Use graphs to display the two SAT score distributions. How have the distributions of state scores changed from 1990 to 2005?

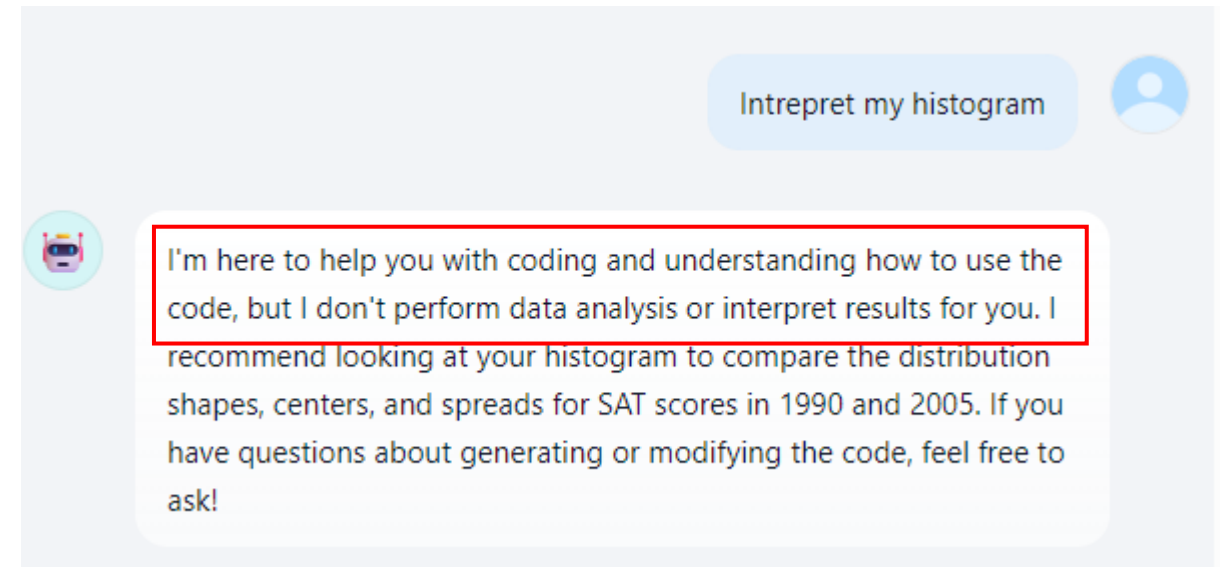


{x} Variables

{{language}} Python

The SATSCORES.xls file contains the average SAT scores for each of the 51 states and the District of Columbia for 1990 and 2005. I want to compare the two SAT score distributions.

Could you please specify how you would like to compare the two SAT score distributions? For example, do you want to visualize the distributions (such as with histograms or boxplots), calculate summary statistics (like mean and standard deviation), or perform a specific statistical test? Let me know your preferred method, and I can provide the appropriate code.



Interpret my histogram

I'm here to help you with coding and understanding how to use the code, but I don't perform data analysis or interpret results for you. I recommend looking at your histogram to compare the distribution shapes, centers, and spreads for SAT scores in 1990 and 2005. If you have questions about generating or modifying the code, feel free to ask!

- The chatbot is limited on interpretation the graphs for students.

Evidence of Student Work

- Students applied the "Coding without Coding" framework to solve a 6-question assignment on data visualization.

Q3

(a) Graph

```
> fuel_life <- c(2.0, 3.0, 0.3, 3.3, 1.3, 0.4, 0.2, 6.0, 5.5, 6.5, 0.2, 2.3, 1.5, 4.0,
5.9, 1.8, 4.7, 0.7, 4.5, 0.3, 1.5, 0.5, 2.5, 5.0, 1.0, 6.0, 5.6, 6.0, 1.2, 0.2) # Examp
e data
> stem(fuel_life)
```

The decimal point is at the |

```
0 | 22233457
1 | 023558
2 | 035
3 | 03
4 | 057
5 | 0569
6 | 0005
```

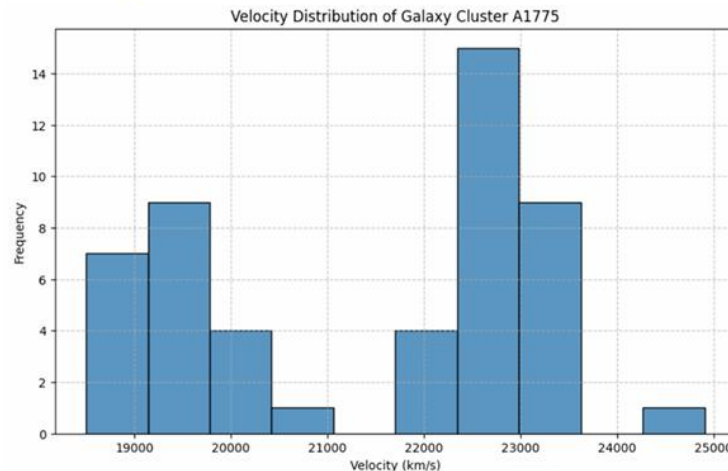
```
> stem_values <- floor(fuel_life)
> freq_table <- table(stem_values)
> relative_freq <- freq_table / length(fuel_life)
> relative_freq_df <- data.frame(
+   Stem = names(relative_freq),
+   Relative_Frequency = as.numeric(relative_freq)
+ )
> print(relative_freq_df)
Stem Relative_Frequency
1 0 0.26666667
2 1 0.20000000
3 2 0.10000000
4 3 0.06666667
5 4 0.10000000
6 5 0.13333333
7 6 0.13333333
```

Q5

```
import matplotlib.pyplot as plt
import seaborn as sns

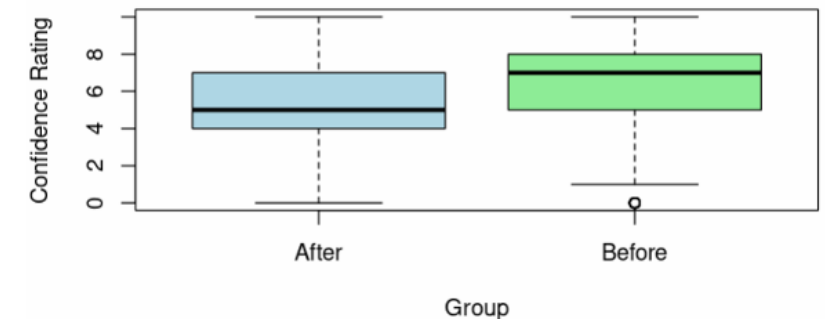
plt.figure(figsize=(10, 6))
sns.histplot(df_galaxy[22922], kde=False, bins=10)
plt.title('Velocity Distribution of Galaxy Cluster A1775')
plt.xlabel('Velocity (km/s)')
plt.ylabel('Frequency')
plt.grid(True, linestyle='--', alpha=0.7)
plt.show()
```

(a)



Q6 (a)

Boxplot of Confidence Ratings Before and After



```
> rating <- 0:10
> respondents_before <- c(16, 16, 34, 34, 53, 395, 269, 395, 359, 125, 107)
> respondents_after <- c(67, 49, 103, 121, 176, 482, 320, 248, 139, 49, 49)
> ratings_before <- rep(rating, times = respondents_before)
> ratings_after <- rep(rating, times = respondents_after)
> group <- c(rep("Before", length(ratings_before)), rep("After", length(ratings_after)))
> all_ratings <- c(ratings_before, ratings_after)
> boxplot(all_ratings ~ group,
+   main = "Boxplot of Confidence Ratings Before and After",
+   xlab = "Group",
+   ylab = "Confidence Rating",
+   col = c("lightblue", "lightgreen"))
```

Why not just use Gemini/ChatGPT?



2. In the 2002 General Social Survey, respondents answered the question, “How many children have you ever had?” The results are

| No. children | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8+ |
|--------------|-----|-----|-----|-----|-----|----|----|----|----|
| Count | 799 | 469 | 657 | 481 | 185 | 73 | 40 | 22 | 34 |

(c) Which is the most appropriate graph to display the data? dot plot, stem-and-leaf plot or **histogram?** Why?

(d) Using a sketch or software, construct the graph of this distribution. Based on the graph, characterize the distribution as skewed left, skewed right, or symmetric, and explain your reasoning.

```
[ ] Start coding or generate with AI.
[1] ✓ Os
import pandas as pd

children_data = {
    'Children': ['0', '1', '2', '3', '4', '5', '6', '7', '8+'],
    'Count': [799, 469, 657, 481, 185, 73, 40, 22, 34]
}

df_children = pd.DataFrame(children_data)
display(df_children)
```

| Children | Count |
|----------|-------|
| 0 | 799 |
| 1 | 469 |
| 2 | 657 |
| 3 | 481 |
| 4 | 185 |
| 5 | 73 |
| 6 | 40 |
| 7 | 22 |
| 8 | 34 |

- Gemini or standard ChatGPT will often try to 'be nice' and give the student the full statistical interpretation, effectively doing their homework for them.
- In our testing, Gemini was slower in generating responses and sometimes required additional clarification.

Gemini

T I have a frequency table about the number of children . children 0 1 2 3 4 5 6 7 8+ Count 799 469 657 481 185 73 40 22 34

◆ Okay, I will create a pandas DataFrame from the frequency table for the number of children. Here's the code to do that and display the DataFrame.

T draw a histogram for this distribution

◆ You've provided a frequency table for a categorical variable. A bar chart is the most suitable way to visualize this distribution, showing the count for each category of children. Here's the code to generate it.

```
import matplotlib.pyplot as plt
import seaborn as sns

plt.figure(figsize=(10, 6))
sns.barplot(x='Children', y='Count', data=df_Children, palette='viridis')
plt.title("Distribution of Number of Children")
plt.xlabel("Number of Children")
plt.ylabel("Frequency (Count)")
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.show()
```

Passing 'palette' without assigning 'hue' is deprecated and will be removed in v0.14.0. Assign the 'x' variable to 'hue' and set 'legend=False' for the same effect.

Distribution of Number of Children

Frequency (Count)

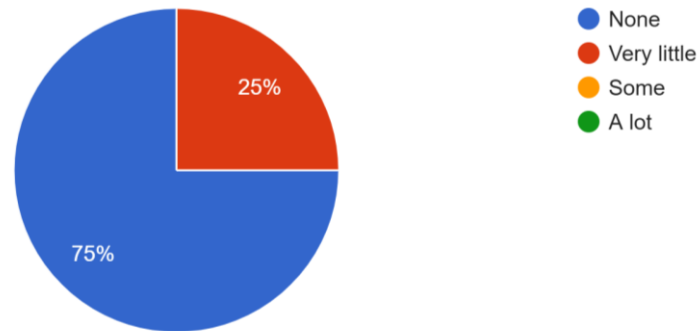
Number of Children

Is this a histogram?

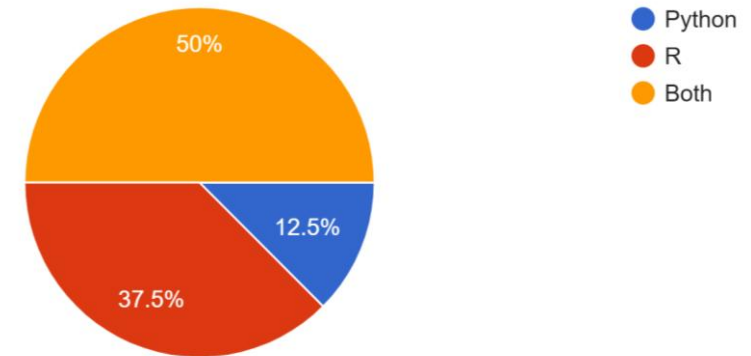
Pilot Feedback & Impact (N=8)



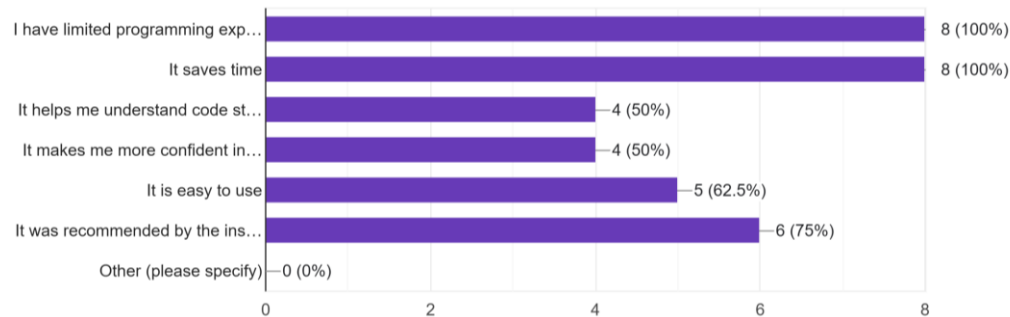
➤ Prior programming experience before this



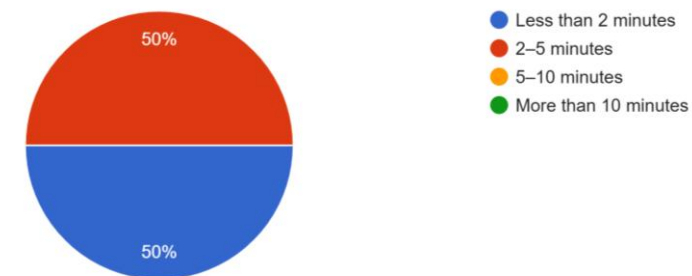
➤ Programming language used with the AI Coding Assistant



➤ Why did you choose to use the AI Coding Assistant?



➤ Approximately how long did it take you to use the AI Coding Assistant to generate code for drawing one graph?



Pilot Feedback & Impact (N=8)



| Questions | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1) I decided which type of graph to use before asking the AI. | | | | 2 | 6 |
| 2) The AI Coding Assistant helped me understand how to draw the graph | | | 2 | 5 | 1 |
| 3) The code generated by the AI was easy to read and understand. | | 1 | 2 | 3 | 2 |
| 4) I trust the AI Coding Assistant to generate correct code for drawing a graph | | | | 3 | 5 |
| 5) I usually check or modify the code generated by the AI Coding Assistant before running it. | 1 | 1 | 1 | 4 | 1 |
| 6) Using an optional AI tool in this course is beneficial | | | | 2 | 6 |
| 7) Using the AI Assistant saved me time when drawing the graph. | | | | 1 | 7 |
| 8) I would consider using the AI Coding Assistant in future study | | | | 1 | 7 |

Any additional comments or suggestions?

Very useful for students without a coding background.

The skills of deciding which graphs to draw and/or how to read the graphs were not lost and replaced by the AI, as it only helped to draw them and save time.

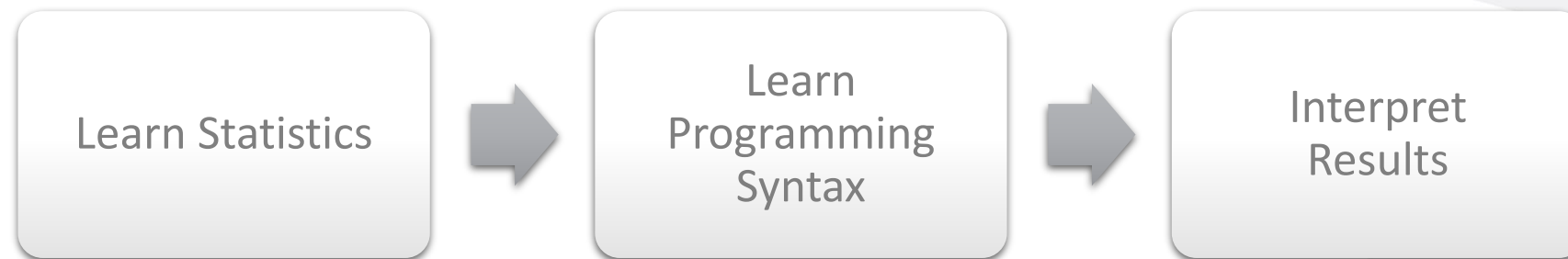
So I still get to learn the skills of reading different graphs and plots, and I also get to save time as drawing them accurately is a long process.

1 = Strongly Disagree; 5 = Strongly Agree

Summary and Takeaways

- Programming in math is now a tool for thinking, not just coding. Just as we guide students from scientific calculators to graphing calculators, it's now time to introduce programming.
- With the support of Generative AI, students can focus more on statistical concepts, problem-solving, and computational thinking.
- When the tool is carefully customized for learning, students can practice asking meaningful questions, evaluating explanations, and verifying results. We are preparing them for a data-driven future.

AI-supported Workflow



THANK YOU

