

Lessons Learned When Teaching Data Analytics with ChatGPT to MBAs in Spring 2023

Robert L. Bray

2023-08-28

Abstract

ChatGPT rendered my old data science class obsolete, so I created a new version that leverages large language models. This paper chronicles my experiences incorporating ChatGPT into the MBA classroom. I learned that the chatbot does not blur the lines between A and B students, as I only needed to marginally increase the quizzes' difficulty to maintain a robust grade distribution. I also found that MBA students are well suited to "programming in English"—i.e., expressing the required data transformations in words and then using a large language model to convert their prose into code. I present informal experimental results that suggest that moderately advanced data-wrangling tasks are more manageable for students when they program in English than when they use Excel. The integration of ChatGPT yielded mixed results: both my elective and compulsory sections seemed more engaged, but only the former gave me higher evaluations—the compulsory section gave me significantly lower teaching scores. Students from both sections also reported working less outside of class.

I thought my class was set. Over the prior two years, I had made an entirely new analytics course. I wrote a textbook on data science with R for MBA students, which revolves around a set of labs that guide the reader through eight empirical operations management studies—from raw data to final results (Bray 2023). In addition, I created a corresponding set of interactive R Markdown slide decks that can execute code. The class went well in the previous year, and I was planning on coasting for the next few years.

How wrong I was. ChatGPT debuted on November 30, a mere 120 days before my first class, and I squandered the first seven weeks because it took me until the middle of January—when I started preparing for my spring teaching—to realize the problem it would pose. I remember the moment the harsh reality set in. I copied the following exercise from my textbook into ChatGPT without further explanation or context:

Exercise 3.43 We will now subject our sample to 10 data filters.

- Start with `alibaba_long` and `group_by() order`.
- `filter()` the grouped tibble accordingly:
- Remove the orders with `any()` "FAILURE" action. For example, you should remove all the `order = 87717` observations since this order's 12th action was a "FAILURE".
- Remove orders without exactly one "ORDER" action, one "SIGNED" action, and one "CONSIGN" action.
- Remove orders that have an action before the "ORDER" action or after the "SIGNED" action. In other words, there should be an "ORDER" action at `time = 0` and a "SIGNED" action at `time = 1`.
- Remove orders that correspond to multiple `shipper` values.
- Remove orders with `day_count > 8`.
- Remove orders with more than 10 or fewer than 5 posted actions.
- Remove observations with "ORDER" and "SIGNED" actions, because their `time` values are degenerate (mechanically being either 0 or 1).

- `ungroup()` the filtered tibble and `<-` it into `alibaba_long`.
- `alibaba_long` should have 102331 rows after this step.

This was the most challenging question from my Alibaba lab, which replicates the analysis from Bray (2020). Last year, around two dozen students visited my office hours to discuss this particular question. So you can imagine my astonishment when the chatbot produced a correct code solution on its first attempt. Regrettably, I can't find the answer it gave me then, but when I give ChatGPT this input now, I receive the following output:

It seems like you're working with R programming and the `dplyr` package, specifically in manipulating a data frame named ``alibaba_long``. I'm assuming that your ``alibaba_long`` data frame has columns like ``order``, ``action``, ``shipper``, ``day_count``, and ``time``. Here's how you could apply the 10 data filters you mentioned:

```
library(dplyr)

alibaba_long <- alibaba_long %>%
  group_by(order) %>%
  filter(!any(action == "FAILURE"), # Filter out "FAILURE" actions
         sum(action == "ORDER") == 1, # Must have exactly one "ORDER" action
         sum(action == "SIGNED") == 1, # Must have exactly one "SIGNED" action
         sum(action == "CONSIGN") == 1, # Must have exactly one "CONSIGN" action
         min(time[action == "ORDER"]) == 0, # "ORDER" action must be at time = 0
         max(time[action == "SIGNED"]) == 1, # "SIGNED" action must be at time = 1
         n_distinct(shipper) == 1, # Should only have one distinct shipper value per order
         day_count <= 8, # Must have day_count <= 8
         n() >= 5, n() <= 10) %>% # Must have 5 to 10 actions
  ungroup() %>%
  filter(!(action %in% c("ORDER", "SIGNED") & time %in% c(0, 1))) # Filter out "ORDER" and "SIGNED" act

# Check if there are 102331 rows left
if(nrow(alibaba_long) != 102331){
  print("The final number of rows in 'alibaba_long' is not 102331.")
} else {
  print("Data filtering completed successfully. 'alibaba_long' now has 102331 rows.")
}
```

Remember to replace the column names if they are different in your actual data. And be sure to run the checks to see if the transformations have been done correctly.

While not especially elegant, this code is nevertheless correct—a fact that's all the more impressive considering `alibaba_long` is undefined. I soon discovered that ChatGPT could solve nearly every class question, be it from a quiz, lecture, or lab. Three weeks before students were slated to bid on it, ChatGPT had rendered my class an obsolete farce.

After the initial dread subsided, however, it dawned on me that I could teach one of the first-ever classes on coding with ChatGPT. What an opportunity! I quickly went from desperation to anticipation.

The first order of business was rewriting my syllabus, which now begins with the following:

Large Language Models (LLMs) such as ChatGPT are powerful. To maximize your productivity—and stay relevant—you should aim to delegate as much of your workflow to

these language engines as possible. This means you should become comfortable processing and analyzing data with a computer language—such as R—that LLMs excel at reading and writing. For instance, ChatGPT can answer nearly every question in my R textbook. Additionally, it's adept at explaining its solutions. Case in point, when I instruct it to “run a regression in R and use the broom package to get its coefficient estimates and fitted values,” it provides the following response:

Here is an example of how to run a linear regression in R using the ``lm()`` function and the broom package to get the coefficient estimates and fitted values:

```
# Import the necessary packages
library(broom)

# Create some example data
x <- rnorm(100)
y <- x + rnorm(100)

# Fit the linear regression model
model <- lm(y ~ x)

# Use the broom package to get the coefficient estimates
coefs <- tidy(model)
print(coefs)

# Use the broom package to get the fitted values
fitted_values <- augment(model)
print(fitted_values)
```

In this example, the ``lm()`` function is used to fit a linear regression model with ``y`` as the response variable and ``x`` as the predictor variable. The ``tidy()`` function from the broom package is then used to extract the coefficient estimates, and the ``augment()`` function is used to extract the fitted values. The ``print()`` function is used to display the results.

You can then visualize the fitted values by running

```
library(ggplot2)
ggplot(fitted_values, aes(x = x, y = .fitted)) + geom_point()
```

Note that in this example, data is generated randomly, you should replace it with your own data.

I couldn't have said it better myself. As you see, switching to R equips you with a software equivalent of a genie in a bottle, capable of implementing and explaining almost every data transformation. The tool is a great equalizer; an MBA with an LLM can accomplish almost anything that an experienced data scientist can. Indeed, MBAs now stand out as the finest data scientists since the critical analytics differentiator has shifted from technical expertise to business insight. MBAs will excel as data analysts because they understand the most meaningful questions to pursue.

This class will teach you how to use LLMs to process and analyze data. The only challenge is transitioning from the spreadsheet to a *language*. However, once you master the lingua franca of data science, you will be capable of communicating and collaborating with a machine of immeasurable power.

After updating the syllabus, I circulated it among the MBA students with the following tagline: “OPNS 451 = Coding with ChatGPT.” This message struck a chord, and the final enrollment in my elective MBA section swelled from 21 students to 55. In addition to this elective MBA section, I had a compulsory Master of Management and Manufacturing (MMM) section, whose total enrollment, 61 students, was predetermined. I distributed a survey before the first class, which asked the students to rate their agreement with several statements on a scale from 1 (“strongly disagree”) to 7 (“strongly agree”). 48 out of 55 MBAs and 36 out of 61 MMMs filled out the survey. Here are their mean responses:

Statement	MBA	MMM
I am excited to learn how to analyze data with code.	6.06	5.42
Large language models are the most exciting business opportunity right now.	5.27	4.14
I find the prospect of programming intimidating.	3.50	3.86
I would like the workload to be high so that I learn as much as possible.	3.40	2.92
I believe AI could make humanity obsolete in my lifetime.	2.48	2.75

Both classes largely agreed that “Large language models are the most exciting business opportunity right now” and were thus “excited to learn how to analyze data with code.” Nevertheless, there was some trepidation: more than a quarter of students “believe[d] AI could make humanity obsolete in my lifetime,” with a conviction of at least 4 out of 7, and nearly half of the students reported “find[ing] the prospect of programming intimidating,” with a conviction of at least 4 out of 7.

I explained that I, too, was nervous since I didn’t know what data science meant in the era of large language models. I proposed using the class as an opportunity to collectively anticipate how analytics education and practice will evolve in response to the transformational technology that is ChatGPT 4.0 (I secured each student a ChatGPT Plus account). I would like to share a few things we learned.

Lessons Learned

Students Will Soon Demand a ChatGPT Retrofit

Consider the following Slack message from one of my elective MBA students:

I thought you’d like this one—I’m currently in a SQL coding class. I went ahead and finished 2.5 hours of labs in 20 minutes. I’ve never used SQL before and this is being taught like it’s a pre-ChatGPT world; knowing what I now know, it’s immeasurably frustrating. I feel like I’m in the film *Don’t Look Up*.

Programming in English is the Sweet Spot

ChatGPT has unlocked a new paradigm for analytics education: “programming in the English language.” Students describe the desired data transformations in words, and ChatGPT translates their descriptions into code. It’s surprisingly effective, and MBA students love it.

To illustrate the power of this approach, I asked the students to use ChatGPT to (i) construct a random forest model in R, (ii) identify its most predictive variables, (iii) test whether it provides a better in-sample fit than OLS, and (iv) test whether it provides a better out-of-sample fit than OLS. We accomplished all of this in 45 minutes *in the very first class*. Most students had no R experience, yet they could still build a random forest model and evaluate its performance using a hold-out sample, all within an hour. I even incorporated missing data in the sample to ensure that the initial random forest run would yield an error. However, that was not a problem because the students could share the error message with ChatGPT, and it would modify its solution accordingly.¹

Here's a second example: ChatGPT enabled my students to create an interactive statistical dashboard, upload it to the cloud, and make it shareable via URL—all within half an hour. The students used the complex and unfamiliar Shiny package, guided only by the following scanty instructions:

- This is an in-class exercise. The only thing you need to do before class is `install.packages("shiny")`.
- Create a new folder in your current R project called `my_shiny_app` and within that folder create a file called `app.R`. All Shiny applications are saved in an `app.R` file.
- Ask ChatGPT to give you an example of an `app.R` file that you could use as a Shiny template. Copy the code it gives you into your `app.R` file, and then execute the code by clicking on the Run App button. A window should pop up with an interactive plot.
- Tell ChatGPT that you would like to create a Shiny app that creates an interactive plot, based on the table of data found here:
`"https://www.dropbox.com/s/eh7f0t7u3rae754/car_data_with_distances.csv?dl=1"`. This table has columns `brand`, `defects`, `distance`, and `year`, among others. The app should have a drop-down menu that specifies a “`brand`”, and then filters the data to the corresponding observations. It should then use the filtered data to create a scatterplot, with the data in the `distance` column as the x-axis and the data in the `defects` column in the y-axis.
- Ask it to add a regression line, and have it report the statistics of the regression below the plot. Also ask it to incorporate a slider that allows you to filter the data by `year`.
- Now ask ChatGPT how you can have ShinyApps.io host your shiny app, so that anyone can interact with it over the web.
- The dashboard we made was pretty ugly. Create a more elegant report that illustrates the relationship between `distance` and `defects`, upload this version to the cloud, and Slack the URL to the class grader. Whoever makes the best dashboard, according to the grader, won't have to do the cars lab.

The students appreciated this activity because (i) it taught them a concrete skill they could use in their internships, and (ii) it illustrated the expansion of possibilities afforded by large language models. Without ChatGPT, the Shiny package would have been too advanced for an MBA class.

In the past, analytics instructors faced a trade-off: R offered more power, while Excel promised simplicity. But now, R is more powerful and more straightforward because students can better leverage large language models if they manipulate their data with a computer language. To illustrate this point, I conducted an experiment. I divided the students into two groups: the first group solved a set of analytics problems with R and ChatGPT and then solved the same problems with Excel and ChatGPT. Conversely, the second group began with Excel and ChatGPT before moving on to R and ChatGPT. Hence, all students attempted all problems with R + ChatGPT and Excel + ChatGPT, with half the class beginning with R and the other half with Excel, to ensure equal treatment for the approaches. I instructed the students to record the number of questions they could correctly answer in 30 minutes with each method. Overall, the students correctly solved more than ten times the number of questions with R than with Excel. Unfortunately, I couldn't get a reliable estimate of the Excel + ChatGPT productivity rate because the students essentially gave up in this case. They quickly realized that they could not elicit sensible Excel instructions from ChatGPT.

¹Unfortunately, this demonstration was quite unpopular, so I will scale it back next year.

For instance, consider the following exchange, in which a student asks ChatGPT for Excel instructions to implement the first filter of Exercise 3.43 (the problem discussed in the introduction):

Student:

I have a table of data called `alibaba_long`, with column names `order` `score` `item` `brand` `category` `sub_category` `merchant` `quantity` `action_num` `action` `time` `facility` `shipper` `day_count`. Remove all observations that correspond to an order with a "FAILURE" action. For example, you should remove all the order = 87717 observations since this order's 12th action was a "FAILURE". Use Excel

ChatGPT:

To remove all observations that correspond to an order with a "FAILURE" action in Excel, follow these steps:

- Open your CSV file in Excel.
- Click on the top left cell (A1) to select the entire dataset.
- Go to the 'Data' tab in the Excel ribbon.
- Click on 'Filter' in the 'Sort & Filter' group. This will add filter arrows to the header row of your dataset.
- Click on the filter arrow in the 'action' column header.
- In the filter dropdown menu, uncheck 'Select All' and then check all options except 'FAILURE'.
- Click 'OK' to apply the filter. This will hide all rows with 'FAILURE' in the 'action' column.
- To remove these rows completely, select the visible dataset (excluding the header row) by clicking and dragging.
- Right-click on the selected dataset and choose 'Delete Row' from the context menu.
- Go to the 'Data' tab and click on 'Filter' again to turn off the filter.
- Save your modified CSV file.

Now, your CSV file will not have any observations with an order that has a "FAILURE" action.

Student:

That is eliminating all the lines that have that FAILURE flag, but not the orders associated with that flag.

ChatGPT:

Apologies for the confusion. To remove all observations corresponding to an order with a "FAILURE" action in Excel, follow these steps:

- Open your CSV file in Excel.
- Add a new column to the right of the 'order' column (column B).
- In cell B2, type the following formula and press Enter: `=IF(C2="FAILURE", A2, "")`. This will mark the orders associated with the "FAILURE" action.
- Click on the bottom-right corner of cell B2, then drag it down to fill the entire column B with the formula.
- Select the entire dataset, including the new column B (click on cell A1, then press `Ctrl+Shift+Right Arrow+Down Arrow`).

- Copy the selected dataset (press Ctrl+C).
- Open a new sheet in Excel.
- Right-click on cell A1 in the new sheet and choose 'Paste Special' > 'Values' to paste the data without the formulas.
- Click on the 'Data' tab in the Excel ribbon.
- Click on 'Filter' in the 'Sort & Filter' group to add filter arrows to the header row of your dataset.
- Click on the filter arrow in the 'action' column header (column C in the new sheet).
- In the filter dropdown menu, uncheck 'Select All' and then check all options except 'FAILURE'.
- Click 'OK' to apply the filter. This will hide all rows with 'FAILURE' in the 'action' column.
- Click on the filter arrow in the new column B header.
- In the filter dropdown menu, uncheck 'Select All', and then check 'Blanks'.
- Click 'OK' to apply the filter. This will show only the rows not associated with the "FAILURE" action orders.
- Select the entire visible dataset (excluding the header row), right-click, and choose 'Copy'.
- Open a new sheet in Excel and paste the data in cell A1 (right-click, choose 'Paste', or press Ctrl+V).
- Delete the extra column B (which was used to mark the orders with "FAILURE" action) by right-clicking on the column header and choosing 'Delete'.
- Save your modified CSV file.

Now, your CSV file will not have any observations corresponding to an order with a "FAILURE" action.

These mousing instructions are both tedious and wrong, as they retain all rows without a "FAILURE" rather than all rows from orders without a "FAILURE". The student attempted to explain this to ChatGPT, but the chatbot repeated the error on its second attempt.

Now contrast the exchange above with the one that follows. For this case, the student replaced "Use Excel" with "Use R" at the end of the prompt:

Student:

I have a table of data called `alibaba_long`, with column names `order score item brand category sub_category merchant quantity action_num action time facility shipper day_count`. Remove all observations that correspond to an order with a "FAILURE" action. For example, you should remove all the order = 87717 observations since this order's 12th action was a "FAILURE". Use R

ChatGPT:

To achieve this, you would need to subset the dataframe to remove the observations that correspond to a "FAILURE" action. Here's how you might do that in R using the `dplyr` package:

Firstly, install the required package if you haven't already.

```
install.packages("dplyr")
```

Then load the package:

```
install.packages("dplyr")
```

Assuming "order" and "action" are two of the columns in your dataframe, you can filter out the "FAILURE" actions like so:

```
alibaba_long <- alibaba_long %>%  
  group_by(order) %>%  
  filter(!any(action == "FAILURE"))
```

This code groups the data by "order", and then removes any groups/orders where any of the "action" values are "FAILURE".

If you want to create a new data frame without modifying the original one, you could do:

```
alibaba_long_without_failures <- alibaba_long %>%  
  group_by(order) %>%  
  filter(!any(action == "FAILURE"))
```

This would leave your original dataframe, `alibaba_long`, unchanged, and create a new dataframe, `alibaba_long_without_failures`, without the "FAILURE" actions.

Now ChatGPT yields an answer that is correct, clear, and easy to implement (just copy three lines of code into R). This experiment convinced the students of the superiority of the “programming with English” approach. Indeed, at the end of that class, I overheard one student telling his partner, “I’m totally sold.” And when I asked the students to rate “How much do you agree with the following sentence: This class has convinced me that it’s better to analyze data with code,” on a scale from 1 (“strongly disagree”) to 7 (“strongly agree”), the average survey response was 5.3 for my compulsory MMM section and 6.0 for my elective MBA section.

The experiment worked because an Excel framing essentially lobotomizes ChatGPT. Why does ChatGPT perform more intelligently when operating in R than Excel? First, R is a more expressive tool with more built-in constructs for data manipulation. For example, the `group_by()` function in R simplifies the removal of all observations of a particular order. However, the concept of `group_by()` doesn’t exist in Excel, so dropping an entire group of orders isn’t straightforward. Second, Excel operates on a point-and-click basis, and you can’t train a language model on ephemeral mouse clicks—while there’s a large corpus of R code online, there’s no corresponding corpus of Excel cursor movements. Third, R solutions are typically shorter because it’s more efficient to communicate through code than through GUI instructions (e.g., “Click on the bottom-right corner of cell B2, then drag it down to fill the entire column B with the formula.”). Shorter solutions are more robust as they offer fewer chances for hallucination or derailment.

I presented the “programming with English” paradigm to a room full of analytics instructors at Columbia’s Future of Analytics Education workshop this May. Few professors seemed aware of this approach—or at least of its full potential. For example, Ciamac Moallemi showcased a set of plug-ins that enable students to use higher-level statistical tools in Excel. He explained that these plug-ins were a necessary compromise since although teaching proper data analytics requires teaching coding, “teaching coding to MBAs is too heavy a lift.” To challenge this claim, I demonstrated how to implement k-means clustering—Ciamac’s primary example—with an English-to-R translation. I performed a live demo, copying the following prompt into ChatGPT:

store_coordinates is a tibble with columns store, long, and lat. Use the k-means clustering function of R to cluster the store values by long, and lat.

And looking through my history, I see that ChatGPT responded with the following:

To cluster the `store` values in the `store_coordinates` tibble using the k-means clustering function in R based on the `long` and `lat` columns, you can follow these steps:

```
# Load the required library
library(dplyr)
library(cluster)

# Perform k-means clustering
k <- 3 # Specify the number of clusters you want

# Select only the relevant columns for clustering
coordinates <- store_coordinates %>% select(long, lat)

# Perform k-means clustering
clusters <- kmeans(coordinates, centers = k)

# Add the cluster labels to the original tibble
store_coordinates <- store_coordinates %>%
  mutate(cluster = clusters$cluster)

# View the resulting tibble
print(store_coordinates)
```

In this example, `k` is set to 3, which means the data will be clustered into three groups. You can modify this value to match the desired number of clusters. The resulting `store_coordinates` tibble will have an additional column `cluster` containing the cluster labels assigned by the k-means algorithm.

Note that in order to run this code, you'll need to have the `dplyr` and `cluster` packages installed in your R environment. If you haven't installed them yet, you can use the following commands:

```
install.packages("dplyr")
install.packages("cluster")
```

Make sure to run these commands before running the code for clustering.

I then executed this code in R to confirm that it properly clusters the stores. The whole process took around 20 seconds.

After the presentation, Dan Iancu suggested that adopting “programming in English” would require teaching code validation. For example, we could instruct the students to subject ChatGPT’s code to test problems for which the correct outputs are known. I will implement this next year.

Use Quizzes, Not Exams

My colleague, Sebastien Martin, has an uncanny understanding of the MBA psyche. Accordingly, I heeded his suggestion when he advised making all formal assessments “open ChatGPT.” He explained that barring the chatbot when the students’ grades were most at stake would lead to resentment since I had pitched this

course as “coding with ChatGPT.” This could have presented a dilemma: since there’s never been an “open ChatGPT” exam, I would not have been able to gauge the difficulty of such a test. Fortunately, however, I had replaced the midterm and final with 16 start-of-the-class quizzes last year. Assessing the students with quizzes enabled me to dynamically adjust the difficulty of the questions in response to the students’ performance.²

ChatGPT Does Not Collapse the Grade Distribution

Before my class, I questioned whether I could derive a meaningful grade distribution from a series of “open ChatGPT” quizzes. Can informative coding exercises be constructed when all students can access ChatGPT’s coding prowess? This was an open question. Surprisingly, however, I found such exercises all too easy to create. In fact, a slight increase in the quiz difficulty was sufficient to ensure that the final quiz scores followed a healthy bell curve (see figure 1).

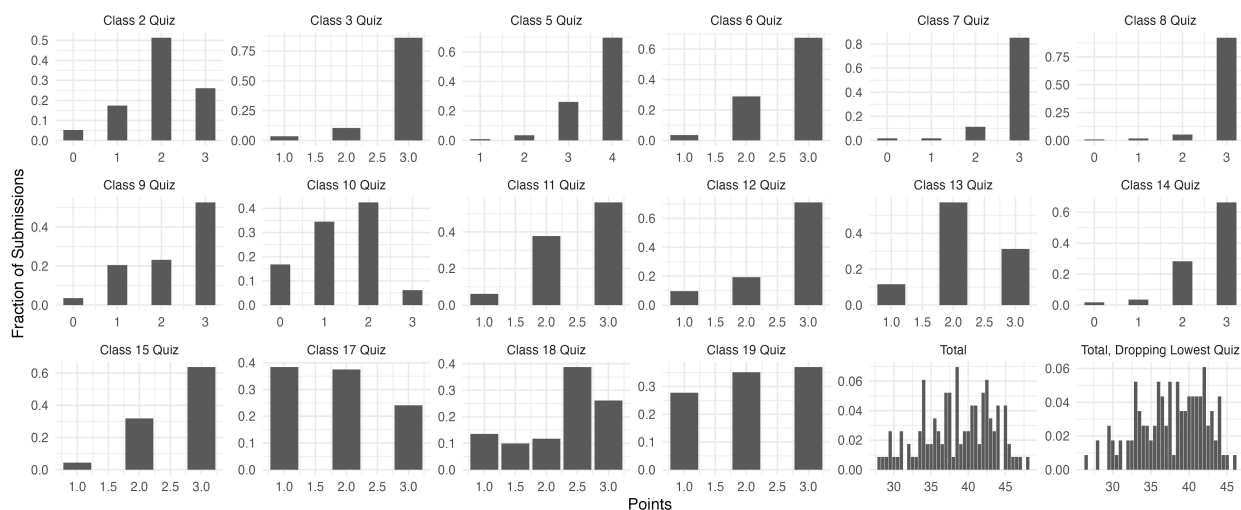


Figure 1: Distribution of Quiz Scores

There were several reasons for the diminished returns to ChatGPT. First, the innate complexity of coding enabled students to consistently discover inventive ways to err. For example, one student told me that she bungled a question because the solution ChatGPT provided began with `install.packages(tidyverse)`, which was a line of code she could not execute within her interactive R Markdown file (I had advised the students to do serious coding in a .R file, but not all of them heeded my advice).

Second, whereas my Spring 2022 students learned to code, my Spring 2023 students learned to *code with ChatGPT*, which proved to be an entirely different proposition. Most of my 2023 students could read the R language, but they could not write it. Accordingly, most of my students were at the chatbot’s mercy, lacking the inclination or capacity to modify its outputs. Even some of my brightest students would use ChatGPT for basic tasks, such as checking the number of rows in a table. This overreliance on ChatGPT led to a brittle performance during quizzes. Since they had yet to write code themselves, they had not made

²Replacing exams with quizzes offers several other advantages. First, it alleviates grading anxiety, as quizzes are low stress for students. For instance, I received only around four regrade requests across all 16 quizzes. In fact, the students were so nonchalant about their quiz performance that I found it necessary to remind them to approach the quiz questions as if they were final exam questions. Second, the students weren’t surprised with their final grades, which progressively unfolded from 16 independent assessments. Third, Dr. Johnson’s insight—“when a man knows he is to be hanged in a fortnight, it concentrates his mind wonderfully”—applies to twice-weekly quizzes. The students pay keen attention when they know they’ll be quizzed in three days. Fourth, the quizzes served as an early detection system, allowing me to swiftly identify and intervene with students who lagged (e.g., I had conducted three quizzes by the end of the second week). Fifth, the students make a point to arrive to class early with their laptops open, knowing that being late would shorten their quiz time. Additionally, students who come late no longer vex me, as I know their quiz scores will reflect their tardiness.

the hundreds of silly mistakes necessary to properly learn the subject—mistakes that invariably cropped up during the quizzes. The course evaluations support this claim: one student reported, “I think most of the class does not know how to code in R after this class and rely mainly on ChatGPT,” and another wrote that

the concept of adding ChatGPT into things was valuable, but i think it came at the cost of actually learning to code. it never felt like we actually had to code, but then there were times where we needed to do things that did require specific skills. it felt like whiplash on the level of specificity. i know this is a new factor in this class, but i think it’s worth thinking about.

Third, abstracting the low-level details with a chatbot may have compromised the students’ high-level understanding. For instance, when I included a tricky problem from the previous homework assignment on a quiz, 58% of students answered it incorrectly. These students failed to recognize a verbatim copy of a problem they had solved just two days prior because rather than engage with the details of the problem, they engaged with a chatbot who did the real engagement for them. Several students expressed regret, in their course evaluations, for outsourcing so much of the thinking to the chatbot:

Since ChatGPT did most of the heavy lifting, I feel like I didn’t learn as much as I wanted. Especially in data–analytics.

Because we relied so heavily on ChatGPT—I truly don’t know what a lot of R even means or what I would use to complete tasks. As well, it was hard to stay engaged.

It was occasionally the case that I would mindlessly complete the quiz without fully knowing what I was doing due to the time constraint, but I got away with it since ChatGPT is so good at coding. If there is a way to effectively force students to think about how to use ChatGPT rather than simply pasting prompts, then that could prove more impactful.

Fourth, students often developed tunnel vision, as crafting GPT prompts would command their undivided attention. Indeed, the students largely ignored the template solutions we covered in class, opting to spend their limited quiz time conversing with the chatbot than perusing their notes. For example, a peculiarity of generalized linear models is that you do not specify the function that maps the linear combination of the independent variables to the mean of the dependent variable; instead, you specify the inverse of this function, known as the “link.” I spent nearly ten minutes explaining this, and in the next class, I quizzed the students on the concept by asking them to explain what the `link = "log"` option in the following code signifies:

```
small_bike %>%
  glm(
    overtime ~ distance + age + duration,
    family = Gamma(link = "log"),
    control = list(maxit = 10^3),
    data = .
  ) %>%
  summary %>%
  {1/pluck(., "dispersion")}
```

After the quiz, most students lamented that they “had to guess” for this question since ChatGPT didn’t know the answer. It led me to question whether the students had forgotten that the quizzes were open ChatGPT and *open notes*.

Fifth, echoing the Peltzman Effect, the students used ChatGPT both to enhance their performance and to *reduce their workload*. The reported weekly hours worked fell from an average of 4.85 and 3.88, for my

MBA and MMM sections, to 3.57 and 2.62. The former drop is statistically insignificant, but the latter is statistically significant at the $p = 0.01$ level. Furthermore, 22% of students reported not studying for quizzes, which would have been inconceivable in 2022.³ One student’s end-of-the-quarter feedback confirms a causal link between ChatGPT and the reduced study times:

I loved how my eyes opened to the possibilities of chat GPT. I wish we had learned more about the fundamentals R before you showcased the power of the chatGPT tool. After seeing that I sort of checked out and knew that I could ask Chat GPT for anything and wouldnt really have to learn on my own.

After teaching the same class both with ChatGPT (easy mode) and without ChatGPT (hard mode), I am left with the impression that regardless of the rigor of the course, a class will revert to the equilibrium whereby A students get As and B students get Bs.

Finally, ChatGPT sometimes struggled with the specialized material we covered in class. For example, ChatGPT doesn’t know how to query the ChatGPT API from R.

ChatGPT Supercharges Homework Productivity

While ChatGPT did not significantly improve performance for the quizzes, it did so for the out-of-class labs. Last year, a four-person team typically required around four to five hours to complete a lab. This year, the self-reported average time spent on an out-of-class lab was 70.0 minutes for the MMMs and 57.8 minutes for the MBAs. One student even said that “Regardless of how much work my group put in, I could get the whole project done solo in an hour or less using ChatGPT.”

This reduction in effort coincided with a marked increase in quality. My grader reported a clear improvement in the students’ lab submissions. For example, in the previous year, it took the students around five weeks to start incorporating the pipe operator, `%>%`, in their submissions. This year, nearly every team began using piping from the outset.

ChatGPT could solve my labs because I provided detailed, step-by-step instructions that students could adapt as prompts. One of the most common suggestions in the course evaluations was to make the homeworks more open-ended:

I love that ChatGPT was rapidly integrated into the syllabus. However, assignments became plug and play. I didn’t need to learn or understand the R material to perform well. In the future, the professor needs to change the focus of the course from coding in R, to learning how to ask the right business questions when presented with data. ChatGPT removes barriers to coding, but if I don’t know what questions to ask I can’t yield this powerful tool.

ChatGPT Atomizes Groups

ChatGPT undermined group work because the students progressed more with the chatbot than with each other. Consider the following tally of the responses to “Did ChatGPT make group work more or less rewarding? Please explain”:

	Elective MBA	Compulsatory MMM
ChatGPT Made Group Work More Rewarding	22	11
ChatGPT Made Group Work Less Rewarding	17	33
No Clear Relationship Reported	5	10

³However, the mean and median study times were a respectable 16.7 and 15.0 minutes per quiz.

While a small plurality of the MBAs found that ChatGPT enhanced group work, an overwhelming majority of MMMs felt it devalued the experience. Most negative comments were along the lines of “there was less discussion and more asking to ChatGPT.”

Nevertheless, ChatGPT also improved group work in several ways. First, a handful of responses explained that ChatGPT put the students on an equal footing, facilitating broader participation in homework tasks. As one student put it, “It evened out where everyone was. Everyone could be helpful.” Second, several students mentioned that ChatGPT streamlined code hand-offs. For instance, one respondent wrote, “It was so helpful to be able to copy and paste someone else’s code into ChatGPT and have it explain what’s going on,” while another noted that ChatGPT “allowed seamless debugging without having to wait for teammates to respond to questions and allowed me to understand what certain functions were meant to accomplish on someone else’s code if there weren’t comments.” And one student explained that their team would copy each student’s code into ChatGPT and ask the bot to compile these versions into a final answer that integrated the best elements of each. Third, students liked ChatGPTing together because “it was interesting to see how my peer’s used the tool differently”; accordingly, they “spent more time comparing ways to best utilize chatgpt and less time troubleshooting menial tasks.” Fourth, the students appreciated the relief from the usual frustrations of coding:

[ChatGPT made group work] Way more rewarding! Such a practical class and when you do get stuck in the code, you have a “get out of jail free” card with chatGPT. Whenever you get stuck in programming, normally you can get stuck for hours. Totally improves the coding experience!

By obviating the lower-level coding issues, ChatGPT enabled some students to focus on the higher-level concepts:

ChatGPT made group work more rewarding by moving us out of troubleshooting mode faster. In my lab assignments I was able to talk to my peers about what was going on in the code and what the data was telling us as opposed to why a syntax error I made threw an error.

ChatGPT Yields New Teaching Possibilities

Last year, I allocated a third of the class time to the labs. However, with the labs now rendered trivial, I had roughly ten hours of additional time to fill. I devoted these hours to ChatGPT to meet the expectations set by promoting the course as “the ChatGPT class.” I developed nine innovations with varying degrees of success, as figure 2 illustrates. This figure presents the students’ responses to a series of survey questions that began with, “How did the following go.” For instance, the top plots depict the distribution of responses to “How did the following go: Student ChatGPT demos,” on a scale from 1 (“very poorly”) to 7 (“very well”).

When I rank the innovations by their mean score, the student demonstrations top the list. To help keep up with the breathless developments in generative AI, I invited the students to give a five-minute presentation to the class if they found something they believed their peers would find intriguing. I awarded the presenters a nominal amount of extra credit, but most student demos were intrinsically motivated—the students wanted to show something cool to their friends. And on that score, they delivered—these were, by far, the best student presentations I have witnessed.

For instance, one student demonstrated how we could use the AskYourPDF plug-in to enable ChatGPT to read the quiz and formulate answers to its three questions without the user needing to open the PDF. Another student showed that we could generate practice quiz questions by copying the pertinent passage from the textbook into ChatGPT. Yet another student used ChatGPT to develop a Python program that reads the top stories on Yahoo News and composes a one-paragraph summary of each in a Word document. Not to be surpassed, another student harnessed ChatGPT to create a Python program to automate a series of video game actions, to grind experience points. As the student explains:

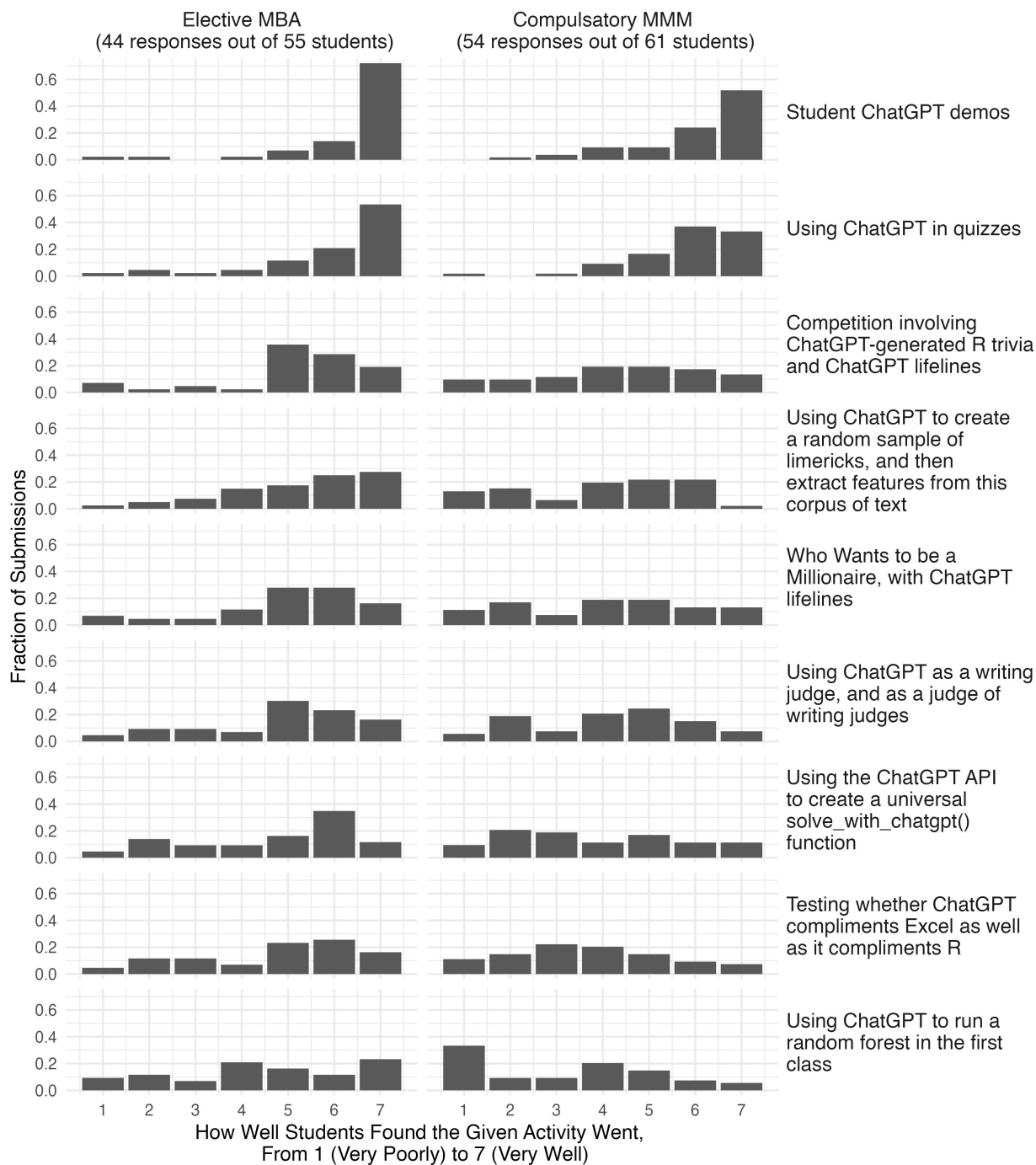


Figure 2: Distribution of Survey Responses

I built two [programs]—one is for training combat skills and the other is to train magic while making money. The first automates gameplay in OldSchool RuneScape to improve gaming efficiency. It autonomously consumes Super Strength and Absorption potions at specified intervals and interacts with the prayer orb to maintain the player’s Hitpoints at an optimal level based on game mechanics, where remaining at exactly 1 hitpoint is the most efficient. The script leverages the PyAutoGUI library to simulate mouse movements and clicks, and incorporates a degree of randomness in these actions to emulate human behavior and avoid detection by the game’s anti-botting systems. Specifically, the script clicks on a random pixel within the possible ranges, whether for the prayer orb or the various potions. To add more variability, the time between click to the prayer orb varies, and each time it is clicked, the script checks to see if a time threshold has been met for either of the potions; if it has, it clicks on the potion, and doesn’t if it has not. The second code is much simpler - it is auto-clicker with time variability between each click. The mouse stays in the same position and clicks over and over, but with sufficiently variable time between each click that makes it seem human-like. What was special about this code was that, since it was so simple, I created a GUI complete with a green- and red-laser eyes on my dog for ‘Start’ and ‘Stop,’ respectively. To make it a bit cooler, I included some calculations to be shown on the GUI, like how much XP I would gain, how many spells would be cast, and how many were left, and if there was a difference between how many I wanted cast, and how many actually were cast, that is also shown.

The students loved this presentation. It was compelling because the presenter had no prior coding experience, being a philosophy and economics major. This presentation thus proved to the class that they, too, could create genuine software.

In summary, the student presentations made one clear: ChatGPT empowers the students to be creative in novel and unexpected ways; allow them to show off their unique creations, and you won’t be disappointed.

The second most popular innovation was integrating ChatGPT into the quizzes, a previously discussed topic. Following that was a gamified version of one of the labs, which pitted the two class sections against one another. The game had multiple rounds, each corresponding to a lab question. To begin a round, I would use ChatGPT-generated R trivia to select two team captains. The captains would then guide the class in collaboratively solving the lab problem. To help with this, I gave the class five one-time-use ChatGPT “lifelines”:

- Automate the Workforce: The team captains can fire the class and replace them with ChatGPT
- Blunder Bot: Everyone can access ChatGPT 3.5
- Hire a Logical MBA: The team captains can select a student to consult ChatGPT, but after that, that student can only answer TRUE or FALSE
- Hire a Technical Technical Writer: Ask ChatGPT to provide a detailed description of the code solution
- Call the Exterminator: Request ChatGPT to debug your code

After the class solved the problem, the team captains would ensure everyone understood the solution. Finally, I would randomly select students and assess their comprehension of the solution with “challenge questions.” This game worked well but would have been somewhat monotonous without the ChatGPT trivia questions and lifelines.

The next most popular innovation was an exercise in which ChatGPT created a text corpus and extracted features from these data. I first showed the students how to have R call ChatGPT to generate a pair of Kellogg-themed limericks and determine which was funnier. I then showed them how to do the API calls in parallel to create 1,000 pairs of limericks, with one from each pair marked as more humorous. I then had the students use logistic regressions to predict which limerick from each pair ChatGPT would find funnier. The students began with features R could easily extract, such as the string length and word count. And then I asked, “Could we include the quality of the rhymes in our logistic regression?” This stumped the class. One student responded affirmatively but added that it would be “really hard to program.” However, I prodded further— “Would it *really* be so hard to program?” —and then students got it: we could use

ChatGPT to score the limericks along abstract dimensions, such as rhyme, rhythm, whimsy, wit, cleverness, and humorousness. This was a significant revelation for both me and the students. With some parallel processing, we extracted these features from each of our 2,000 limericks and then held a Kaggle competition to determine who could best utilize them.

The next most popular innovation was a recreation of “Who Wants to be a Millionaire,” but with ChatGPT lifelines (an idea I actually from ChatGPT itself). This exercise was similar to the previous competition with ChatGPT lifelines, but I also used ChatGPT to create the questions for this one. Specifically, I would input a block of text into ChatGPT, ask it to write a paragraph that correctly explains what the code does, and then ask it to write three analogous incorrect paragraphs. Here’s an example of one of the questions:

Which of the following paragraphs explains what this code does:

```
chess_panel %>%
  inner_join(
    time_to_first_kill %>%
      mutate(
        kill_bucket =
          first_kill_move %>%
            ntile(4) %>%
            as.factor
      )
  ) %>%
  ggplot() +
  aes(x = turns, color = kill_bucket) +
  geom_density() +
  theme_minimal()
```

- (1) This R code takes the chess_panel dataset and performs an outer join with the time_to_first_kill dataset. It then creates a scatter plot using ggplot2, with the x-axis representing the number of turns and the y-axis representing the kill_bucket. Each point is colored based on the kill_bucket (quartiles of the first kill move). The plot uses a minimal theme for better visualization.
- (2) This R code takes the chess_panel dataset and performs an inner join with the time_to_first_kill dataset, which has an additional column kill_bucket created by dividing the first_kill_move into quartiles. The output of the join is then used to create a bar chart using ggplot2. The x-axis represents the number of turns, and the bars are colored based on the kill_bucket (quartiles of the first kill move). The plot uses a minimal theme for better visualization.
- (3) This R code takes the chess_panel dataset and performs an inner join with the time_to_first_kill dataset, which has an additional column kill_bucket created by dividing the first_kill_move into quartiles. The output of the join is then used to create a density plot using ggplot2. The x-axis represents the number of turns, and the density curves are colored based on the kill_bucket (quartiles of the first kill move). The plot uses a minimal theme for better visualization.
- (4) This R code takes the chess_panel dataset and performs an inner join with the time_to_first_kill dataset, which has an additional column kill_bucket created by dividing the first_kill_move into quartiles. The output of the join is then used to create a line plot using ggplot2. The x-axis represents the number of turns, and the lines are colored based on the kill_bucket (quartiles of the first kill move). The plot uses a minimal theme for better visualization.

The students appreciated questions in this format because MBAs generally spend more time reading code than writing it. However, creating these paragraph descriptions would have been prohibitively costly without ChatGPT.

The sixth innovation was an activity that assessed whether I'm a better writer than ChatGPT and whether the students are better writing judges than ChatGPT. For the former comparison, the students provided ChatGPT with the solution code for one of my lab questions. Then they requested the chatbot to write a corresponding bullet-pointed list of instructions that a student could follow to recreate it. The students subsequently determined whether ChatGPT's instructions were more straightforward than those I included in my textbook. Reassuringly, the students largely agreed that my version was clearer. However, when the students asked ChatGPT to judge which piece of writing it found clearer, it typically held the opposite view. Hence, while the students generally favored the man-made text, the chatbot favored the machine-made text. How could we resolve this disagreement? In other words, how could we create an objective measure of writing clarity to determine who's the superior writer—me or ChatGPT—and decide who's the better judge of writing—the students or ChatGPT? It took the students a while to think of the answer: ask ChatGPT to write the code corresponding to both sets of written instructions. Fortunately, my written instructions more frequently yielded the correct answer, allowing us to conclude that humans still reign supreme.

The seventh innovation was a lecture that showed the students how to access the ChatGPT API via the `ask_chatgpt()` function of the `chatgpt` package. We used this function to create the following universal evaluator function:

```
solve_with_chatgpt <-
  . %>%
  str_c(
    "Create R code to do the following: ", #append this message to start of user input
    .,
    sep = "\n\n"
  ) %>%
  ask_chatgpt %>% #pass result into ChatGPT
  str_c(
    "
    Extract code chunk from the following text. #append this message
    Your answer should be a single snippet of code, #to the start of
    with no additional commentary. I should be able to #the ask_chatgpt()
    compile your output as code. #output
    ",
    .,
    sep = "\n\n"
  ) %>% #call ask_chatgpt() a second time to extract the
  ask_chatgpt %>% #code from the previous ask_chatgpt() output
  str_remove("```R|```\|{R\\}|```r|```\|{r\\}") %>% #clean up second
  str_remove_all("```") %>% #ask_chatgpt() output
  parse(text = .) %>% #and try to evaluate it
  eval #as R code
```

This function (i) receives a character string input describing a task, (ii) appends a message to this string to create a sensible prompt, (iii) prompts ChatGPT to generate code to perform the task, (iv) requests ChatGPT to extract the code from the output (as the bot typically intersperses text with code), and then (v) executes the code. For example, when I input "calculate the number of friday the 13ths there will be in the 21st century" %>% `solve_with_chatgpt`, I usually get 172 as the output (although not always, as ChatGPT responses are random). This solution worked pretty well, but one of my students created an even better version of `solve_with_chatgpt()`, which used only one `ask_chatgpt()` call.

Finally, the last two innovations we have already discussed.

ChatGPT Won't Motivate an Unenthusiastic Class

I've always had a harder time engaging my MMM students, who are compelled to take the class, than my MBA students, who choose to do so. As one anonymous student from the 2021 MMM cohort put it,

This should not be a required class for MMMs, or for anybody. . . . Most of the class will never use R again, and this was a wasted course for them. Knowing that, a lot of my classmates just simply didn't put any work into the class, so they didn't learn R, and now they have a bad taste in their mouths about R and coding in general. For that reason, I think this class was actively detrimental to the MMM experience. This does not reflect on Bray necessarily, but again on the administration for making us take an the unnecessary class. Bray's only mistake was thinking that we were going to put in hours outside of class to actually learn how to use R—I agree the only way to learn a coding language is through individual practice and doing labs, but this class (and most business school classes) just aren't going to do that (especially if they are mandatory).

Since receiving this comment, I have endeavored to make data science and R more relevant for the MMMs. I thought ChatGPT was just the ticket, as no one could deny its utility, applicability, and charm. And, indeed, ChatGPT dissipated the sulky 'Why do I need to know this?' atmosphere that previously pervaded the class.

Consequently, I anticipated a considerable increase in my teaching evaluations. I expected the shift in attention from data science to ChatGPT to increase my MBA scores and to increase my MMM scores by even more, since this section is less inclined to data analysis. My MBA scores did rise, but my MMM scores did not rise by more—in fact, they *fell*. Here are my average course evaluations from 2022 and 2023:

	2022 MBA	2023 MBA	2022 MMM	2023 MMM
Overall, how satisfied were you with this course?	4.77	4.95	3.35	3.03
Overall, how satisfied were you with this instructor?	5.15	5.36	3.81	3.21
This course challenged me to engage intellectually and academically.	5.23	5.18	4.31	3.39
This course provided me valuable knowledge, skills and/or analytical frameworks.	4.92	5.59	3.88	3.79

The students responded on a six-point scale, with one representing “extremely dissatisfied” or “strongly disagree” and six representing “extremely satisfied” or “strongly agree.” Most of the differences—from pre- to post-ChatGPT—are statistically insignificant. However, the scores for “This course provided me valuable knowledge, skills and/or analytical frameworks” increased significantly for the MBAs, at the $p = 0.05$ level, and those for “This course challenged me to engage intellectually and academically” dropped significantly for the MMMs, at the $p = 0.01$ level.

The difference between the MBA and MMM reception to ChatGPT was even more pronounced in the written feedback. The MBA responses could be effusive. Here are some examples:

revolutionary course, very bright.

The whole concept of the course was awesome, and I felt like I had to take it to get a leg up on the competition as I graduate and return to the work force. I think the likelihood that I use R in my future work is significantly higher knowing that ChatGPT can assist me when I get rusty.

I loved the cutting-edge nature of the course. Professor Bray is evolving his class to incorporate large-language models that will undoubtedly transform the future of business.

Using ChatGPT was incredibly helpful and such a novel and bold approach given how new the technology is. If anyone from Kellogg administration reads these, look at where you can incorporate ChatGPT and other related tools into other class curriculums, where appropriate. AI tools like this give MBAs superpowers, and it makes me really excited for the next decade.

I really appreciated the way Prof Bray incorporated ChatGPT into the class. It felt like a preview of the way the working world will be for us after graduating (both in the Generative AI sense, and in the “figuring things out as we go” sense). I also appreciated the exercises/activities/competitions in class that broke up any monotony.

Extremely relevant. Enthusiastic professor. Tried things and made an incredible effort to integrate a brand new technology in the class which could be the most impactful and gamechanging breakthrough for B-school students in decades

The MMM students’ positive comments were more measured, and their negative comments were more extreme. Here are some examples:

It felt at times like an experiment, in terms of the class exercises—was not a fan of that. Some were fun, others felt like I was wasting my time.

I felt like this was not applicable to anything I’ll ever do at work. I wish the course was more of an overview of data science, various tools that are used, how to analyze data, etc. than just an “homage to R” class. I’m not sure when else I’ll ever use R and feel like a lot of my time was wasted this term.

I feel like I learned pretty much nothing in class and my valuable time and tuition has been wasted. Honestly, feeling a bit resentful about that. The examples used in class are not relevant in business. As an MBA student, I don’t really care about the professor’s academic research, and don’t feel like doing those examples is going to help me in any way in the future. I also think the goal of this class should’ve been how to analyze data (with R / ChatGPT), what critical questions we should have when analyzing data, instead of how to use R, since we can learn how to use R on our own, with ChatGPT, or even just use other languages such as Excel (which the professor hates vehemently for no reason) or Tableau or Python. I know data cleaning is an important part of data analysis but very few MBA graduate are actually going to clean data on the job so focusing on that seems to be unnecessary. The in-class exercises of showing how wonderful ChatGPT is also seems to miss the point (I signed up to learn analysis not to learn how great ChatGPT is) and I truly felt like I was wasting my time.

This class was by far the biggest waste of my tuition dollars yet. It would have been nice if Bray started off the class with an introduction of why we should learn r, why r is useful etc because I still don’t understand the answer to that. Also organization was HORRENDOUS. In a corporate setting Bray would have been fired immediately for the level of disorganization that goes into his class.

ChatGPT Will Replace Tutors

At the end of the class, I asked the students to rate “How helpful was ChatGPT as an R tutor?” on a scale from 1 (“not at all helpful”) to 7 (“extremely helpful”). Here are the response distributions:

	1	2	3	4	5	6	7
Elective MBA	2.3%	0%	2.3%	2.3%	18.2%	15.9%	59.1%
Compulsatory MMM	0%	7.4%	3.7%	7.4%	14.8%	25.9%	40.7%

70% of responses were either a six or a seven, which explains why my office hours attendance dropped from around six per week to three per quarter. Additionally, I employed four R tutors last year; this year, I found no need to hire any.

ChatGPT Eliminates Cheating

In response to the spike in cheating following the pandemic, I created a unique quiz for each student by randomly drawing from three versions of each question and then shuffling the final arrangement. However, this cheating safeguard was pointless, as I learned in the mid-course feedback: a student noted that nobody would peek at their friend’s answers when they can consult an entity with superhuman coding abilities.

ChatGPT Skills Improve with Practice

In an exit survey I distributed on the last day of class, I asked the students to rate on a scale from 1 (“strongly disagree”) to 7 (“strongly agree”) “How much do you agree with the following sentence: Taking this class has substantially improved my proficiency with ChatGPT, surpassing what it would have been without the class.” I also asked, “How much do you agree with the following sentence: I used ChatGPT in my other classes more than I would have, had I not taken this class.” Figure 3 illustrates the results. The students credit the class with boosting their ChatGPT proficiency, which had spill-over effects for their other courses.

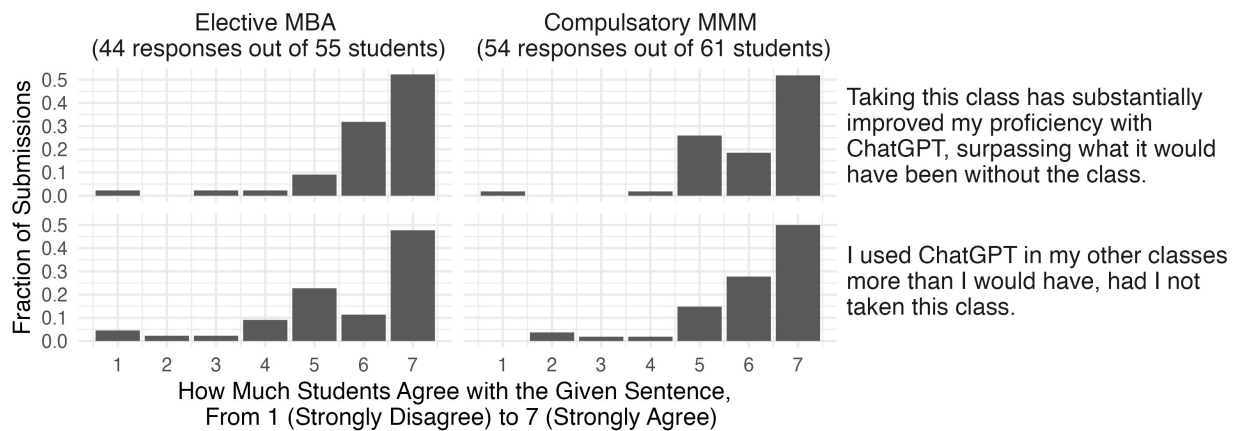


Figure 3: Student Survey Results

Conclusion

Here are the primary takeaways from my quarter teaching data science with ChatGPT to MBAs:

- Students are more productive coding in English than mousing in Excel. My informal in-class experiment suggests that students can be up to an order of magnitude faster with R + ChatGPT than with Excel + ChatGPT. And in a post-class survey, 63% of MMMs and 86% of MBAs reported at least a 5 out

of 7 agreement with the statement, “This class has convinced me that it’s better to analyze data with code.” To better leverage large language models, business schools will likely transition from GUI-based tools, such as Excel or SPSS, to text-based approaches, such as R or Python.

- After hearing me evangelize for the “programming in English” approach, Ciamac Moallemi inquired if it was still necessary to teach data wrangling. I answered affirmatively because the method falls short if students can’t follow ChatGPT’s output—let alone verify its correctness. I learned this the hard way: Asking the students to construct a random forest model in the first session was the least popular ChatGPT innovation (see figure 2). When I expressed dismay and surprise over this, one of my top students offered the following explanation via a Slack:

Btw this is random, but I wanted to share rationale for why me and many other students had a tough time with the random forest experiment at the beginning of the course. We were just starting to get our hands around GPT—lot of nervousness. I didn’t even know the difference between 3.5 and 4 really at the time. When we had GPT assist us with simple code in the first few classes it was great because we could read the output and make some sense of it. Analogous to language immersion with simple words. The random forest was scary because few students had any idea what they were actually doing. I don’t think random forests are part of the standard DECS we all take. So the code it generated was essentially hieroglyphics. It was nerve-wracking trying to “debug” inscrutable code doing something we couldn’t explain. The error messages to console were not helping. And they were caused in retrospect by GPT 3.5’s poor ability to do such complicated tasks. If we did it now, we’d all know to use GPT-4 and could probably even make sense of the code, but it felt like way too much all at once. I was worried after that experience that the “prior knowledge” bar was too high. Quite wrong thankfully! I hope that helps - I remember this experiment was a negative outlier last time we polled student experience and wanted to share the anecdotal perspective.

Since our students don’t want to work with “hieroglyphics,” we must teach them the syntax. And since a lecture that instructs students to read R syntax is analogous to one that instructs them to write R syntax, you should not expect ChatGPT to meaningfully decrease lecture times, as it decreases lab and homework times. In fact, I’ll probably spend *more* time lecturing next year, because the chatbot dulls the students’ understanding, which leads to feedback like this:

I am coming out of this class only being able to do data analytics with the help of ChatGPT. I would have preferred to actually understand it myself but the pace of the course and the encouragement to use ChatGPT made us not actually learn the content. I am paying a lot of money to come to Kellogg and I feel it is a waste of my tuition to be encouraged to use a tool that does not help me to think critically about the underlying logic behind the course content. Don’t get me wrong, I love ChatGPT. But I think this class should have more assignments that force students to learn the concepts on their own first before leveraging ChatGPT as a tool to help us do it faster or more efficiently.

- Don’t take half measures—if you embrace large language models, then allow them on the formal assessments. I had to be talked into doing so, as I harbored concerns that ChatGPT would squash the grade distribution. Thankfully, this did not occur (see figure 1), and allowing the chatbot on the quizzes was the second most well-received ChatGPT innovation I introduced, after the student presentations (see figure 2).
- ChatGPT makes student groups somewhat redundant, as students can more readily get answers from ChatGPT than from each other. One student suggested using teams of two next year since four students running four GPT sessions felt crowded. I suspect four-person data analytics homework groups will soon be an anachronism.
- ChatGPT can trivialize large swaths of your class, especially homework. For instance, ChatGPT turned my labs into straightforward copy-paste exercises because I had provided thorough instructions

for the analyses, instructions that proved to be effective ChatGPT prompts. Indeed, ChatGPT 4.0 can follow almost any data-analysis directions that are simple enough for a team of MBA students, so future assignments must be open-ended. Offloading the lower-level syntactical issues to the chatbot should enable analytics educators to focus on higher-level conceptual issues, which MBA students are more inclined towards. As one anonymous student explains:

Also, I think now that ChatGPT automates a lot of the individual coding steps, there is a big opportunity to have students think more strategically about how they would approach a data wrangling problem. Questions like what is this data actually showing me? Are any relevant columns missing that I would need? Is the data in the right units and format for me? What do I want to group by and filter on? What kind of plot(s) or other visuals would be most effective given my audience? Huge opportunity for critical thinking here, made possible by ChatGPT automating some of the coding.

- Productivity gains are context-dependent, as the previous two points illustrate. ChatGPT greatly simplified the labs—which feature complex, untimed questions backed by comprehensive guidance—but not the quizzes—which consist of concise, timed questions with minimal direction.
- ChatGPT is an exceptional tutor, but students can become overly reliant on it. My students resembled second-generation immigrants who understand their parents’ native language but always respond in English; despite their parents’ wishes, these children will not spontaneously start speaking the mother tongue. Similarly, students who first learn with ChatGPT are unlikely to begin writing unassisted code (at least not within the first ten weeks, which was my observation horizon).
- There is demand for a class on data science with large language models. In a pre-class survey, 47% of MMM students and 77% of MBA students reported at least a 5 out of 7 agreement with the statement, “Large language models are the most exciting business opportunity right now.” Moreover, the enrollment in my elective increased by a factor of 2.6. The students in this elective, which I have so officially renamed to “Data Analytics with Large Language Models,” particularly appreciated the focus on ChatGPT: the score I received for “This course provided me with valuable knowledge, skills, and/or analytical frameworks” increased from 4.92 to 5.59 on a six-point scale, a statistically significant amount. Unfortunately, a class on data analytics with ChatGPT may not hold universal appeal, as the evaluations for my mandatory core section decreased significantly.

Acknowledgements

I want to thank Don Dale, Sarah Forst, and Mohsen Bayati for each telling me to write this paper—it took hearing this suggestion three times to act on it. I would also like to thank Daniel Guetta, Omar Besbes, and the other organizers of the “New MBA: Tech, Data, and Analytics” workshop. Having to give a talk at the end of the quarter on “teaching data science with ChatGPT” compelled me to double down on the ChatGPT content and keep notes of this unique experience. Finally, I thank Ciamac Moallemi for being a good sport, as I ripped him a bit at the workshop. My conversation with Ciamac cemented my intention to write this article, as I only realized how much I had to say on the matter once we spoke.

Bibliography

- Bray, Robert L. 2020. “Operational Transparency: Showing When Work Gets Done.” *Manufacturing & Service Operations Management*, no. November. <https://doi.org/10.1287/msom.2020.0899>.
- . 2023. *Homage to R. Learn the Language of Data Science*. Evanston, IL.