SAS to R - AI Tools

## Intro

Making the jump from SAS to R is a daunting prospect that requires a significant investment in time and effort. However, the advent of AI tools has brought about a new era of automation, making this process more efficient and manageable. Tools like the chattr package and GitHub Copilot, both available in RStudio, are revolutionizing the way we approach such transitions. This blog post aims to guide you on how to leverage these AI tools to facilitate your migration from SAS to R, providing a step-by-step guide to getting started with these powerful resources.

## chattr

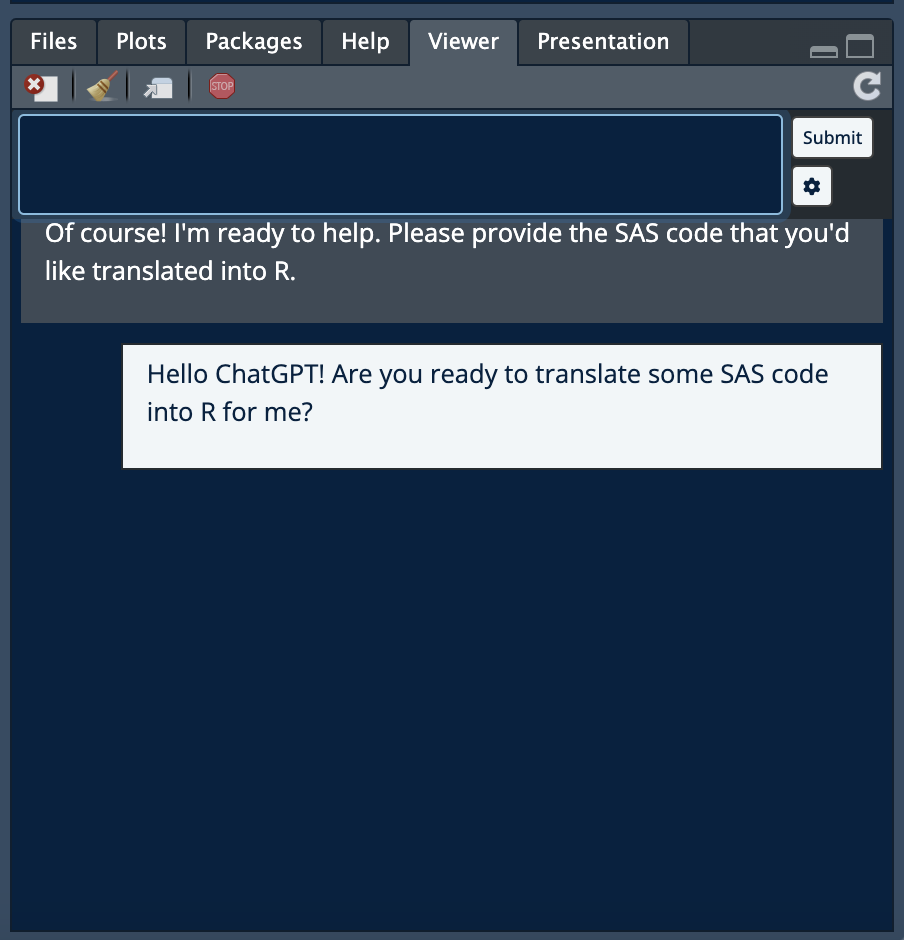
The chattr package provides a way to use Large Language Models (LLMs) within the Rstudio Viewer panel. chattr provides integration with two LLM back-ends, [OpenAI](https://platform.openai.com/docs/introduction) models including GPT 4, 3.5, and DaVinci 3, and [LLamaGPT-Chat](https://github.com/kuvaus/LlamaGPTJ-chat) LLM models available on your local machine. This works similarly to the web portal for Chat GPT, except that it is conveniently integrated within Rstudio and chats can be saved as .r files or on the clipboard.

remotes::install\_github("mlverse/chattr")

### Configuring chattr with Rstudio

[Details on its configuration](https://mlverse.github.io/chattr/articles/openai-gpt.html) are available online. But briefly, you will need to have an account with OpenAI and [then generate an API key](https://platform.openai.com/api-keys) and place this in the [.Renviron](https://support.posit.co/hc/en-us/articles/360047157094-Managing-R-with-Rprofile-Renviron-Rprofile-site-Renviron-site-rsession-conf-and-repos-conf) in your home directory. The following code will verify the configuration and open up the chat window. Code prompts can be copied to your clipboard or saved as a new program.

library(chattr)  
chattr\_test()  
Sys.getenv("OPENAI\_API\_KEY")  
chattr\_app()



## GitHub Copilot

[GitHub Copilot](https://docs.posit.co/ide/user/ide/guide/tools/copilot.html) is an AI pair programmer than is integrated directly into your Rstudio IDE. It is a paid service that starts at $10/month for individuals and $19/user/month for teams. Once subscribed, users can activate the service within the global options manu in Rstudio. Once activated, you can provide prompts to Copilot using the # comment character. After a brief pause, Copilot will provide a suggestion based on your prompt, which you can accept using the Tab key. When integrated into [Quarto](https://quarto.org) or [R Markdown](https://rmarkdown.rstudio.com) documents, Copilot will provide natural language support as well as code generation. GitHub Copilot is already a hit among developers and our team [has successfully used Copilot to build entire Shiny applications](https://www.procogia.com/link_to_mark).

## Incorporating these tools into SAS to R migration

While AI tools like the chattr package and GitHub Copilot were not specifically designed for SAS to R migration, they can be incredibly effective if used correctly. These tools can automate and streamline the process, but their success largely depends on the user’s understanding and skill. Below you will find our experience in how to effectively use these AI tools in a SAS to R migration.

### indexing

GitHub Copilot understands the context of your programs and will use any open programming windows to improve the quality of its suggestions. To expand its scope, you can index files within your [RStudio Projects](https://support.posit.co/hc/en-us/articles/200526207-Using-RStudio-Projects). This is particularly useful when you’re translating SAS code to R. By placing these successful translations in your project, Copilot can reference them to improve its suggestions, making the tool more effective and tailored to your specific needs.

### prompts

Mastering the art of writing effective prompts for AI is a brand new skillset that can significantly enhance your coding efficiency. It’s important to instruct the AI with the code style you prefer. For instance, you might specify “do everything in tidyverse” or “use data.table for this”. This level of instruction helps tailor the AI’s suggestions to your specific needs and preferences.

As you become more familiar with the AI tool, you’ll start to notice common errors. Proactively writing prompts to avoid these errors can save you time and frustration. For example, if you notice that NAs are often produced during merges in character variables, you can instruct R to always recode missing character variables to “” after all joins. This proactive approach can help streamline your SAS to R migration process, making it smoother and more efficient.

### testing

While AI tools like GitHub Copilot and the chattr package can be incredibly useful in automating and streamlining the SAS to R migration process, it’s important to remember that they are not infallible. These tools, like any other, can sometimes return bad code. Therefore, it’s crucial to always test the results to ensure you’re getting what you want.

One effective way to do this is by verifying each step to a given threshold. This can be done using specific R code, which can be incorporated into your workflow. I have found that writing a set of tests using the [testthat package](https://testthat.r-lib.org) after each migration step builds quality into the process and ensures an identical result. I use the following code to compare my migration results and have found that verifies the results to a given threshold.

testthat::test\_that("Validate migration:", {  
 testthat::expect\_equal(  
 sas %>% dplyr::arrange\_all(),  
 r %>% dplyr::arrange\_all(),  
 tolerance = 0.001  
)  
})

## Next steps

The effectiveness of AI tools is largely dependent on the expertise of their users. We specialize in using and training custom AI tools for code migration, and we actively incorporate these bespoke models into our client work. The use of these tools has significantly enhanced our speed and accuracy, allowing us to deliver superior results. We are eager to bring our extensive experience in AI and code migration to your project, ensuring a smooth and efficient transition from SAS to R.