Paper PP25

SAS Programmers Need AI

Craig Parry, MAC Clinical Research / CraigYdwl Ltd., Swansea, Wales

ABSTRACT

When technology advances two things happen, people mention it in every presentation regardless of said technology being relevant and or people worry this is the end of their job. The first point is very true, AI is referenced in almost every presentation I have seen since 2016. But is the second point, redundancy, something to fear? It really depends on how we embrace the advantages and control the disadvantages of AI. I see AI as the super modern version of ask Jeeves, or for people born after the year 2000, Google search with a brain. I have been programming for over a decade, slowly curating a personal portfolio of work, good macros, bad macros, hard to read macros. Yet even after a decade I still find myself with a great idea but limited knowledge of how to implement such an idea, or I am posing a question which sounds correct to me yet it returns something totally irrelevant. Maybe it is because I am a Welsh programmer asking an English question? Anyway, the arrival of AI "chat bots" like Chat GPT are remarkable, we should embrace them not fear them, albeit with a bit of caution. This poster explains how a SAS® programmer coupled with AI can change industry norms, free up programmers to develop new industry norms and show how AI can train the next generation of SAS programmers without taking resources away from billable work. Think of it as AI-led training. The need for AI is far greater than the fear.

INTRODUCTION

Just a prerequisite, this paper heavily relies on the availability of AI chat bots like Chat GPT [1], and maybe you are reading this in 2046, a time where AI chat bots may have been banned. If this is the case, you can consider this paper as a sort of fan fiction for AI chat bots. Anyway, here is the intro. The year was 2022, November 2022 to be precise, England saw their annual “Mozartfest” and Wales saw the Rugby Internationals begin. We also were introduced to our first “mainstream” AI chat bot, Chat GPT [1]. Interestingly, looking at Google’s trends the term “AI” saw a noticeable spike just after the release of Chat GPT on December 4th, 2022, and an even bigger spike in April 2023 when the newest version of Chat GPT, version 4.0 was released [2].

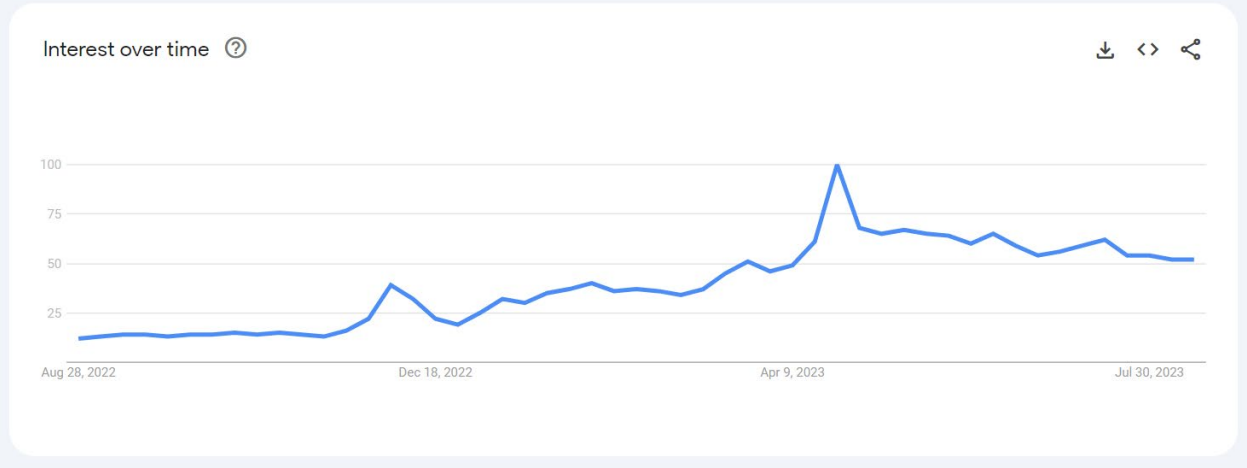

Figure 1: Google trends of the term AI over the last 12 months [2].

This trend initially reminded me of the “3D” movie fad that has appeared multiple times during my lifetime, but this feels different, I think “AI” is here to stay, and we are already seeing companies begin to integrate AI into their systems. Now, I don’t mean integrating AI into every PowerPoint presentation, but real use cases of AI in the clinical industry. I have seen AI successfully process MRI images [3] and I have also seen AI assist with medical coding [4]. Don’t get me wrong there are implementations of AI in other industries some would deem less desirable, like in Cornwall, England where AI is being used to catch bad drivers [5].

Anyway, I digress, how can programmers take advantage of this now? This paper explores this question by explaining how AI can be used in real time, right out of the box. I am mindful nothing is perfect, so we will touch on the obvious pitfalls and hopefully reassure you that programmers like you and I should not fear AI but embrace it.

PROGRAMMERS TO THE RESCUE

We have all been there, you've been handed a study, it was never your study to begin with, but you have been brought in to "save the day", effectively you're on a rescue mission. Now, if I was given ample time to "tidy things up", I thoroughly enjoy a rescue study, for me it's like cleaning your car, the end product is bliss. However, let's be honest, "ample time" and "rescue studies" never appear in the same sentence. This means it becomes less of a rescue mission, more of like a mad scramble. We barely have time to digest the study or even understand the pre-existing programs. This is where AI can step in, and assist. AI is that resource that does not get sick, does not acknowledge stress, a resource which cannot be overburnt, so let's (ab)use it.

FUNCTION VS. FORM

We all know that one programmer who prefers function over form, i.e., they want the program to run above anything else, and I have met several of these programmers. In fact, I recall one programmer once telling me "Runners care about finishing a race, not looking good while doing it!". Maybe this is true, but if I can't read your SAS® code, I'll never finish my work today! This is a great example of where AI can decipher a messy program in seconds. Let's take a look at an awful attempt at programming.

Code Written by a "Programmer"

```
proc sql noprint;
create table cars as
select *
from sashelp.cars
order by make, type;
create table cars_2 as select a.make, b.type, a.max_enginesize, b.total_enginesize
from (select make, max(enginesize) as max_enginesize from cars
group by make) as a left join
(select make, type, sum(enginesize) as total_enginesize
from cars group by make, type) as b on a.make=b.make;
quit;
```

Question to Asked to AI

Could you reformat this code to follow good programming practices without adding anything to the code?

Code Corrected by AI

```
proc sql noprint;
    create table cars as
        select *
        from sashelp.cars
        order by make, type;

    create table cars_2 as
        select a.make, b.type, a.max_enginesize, b.total_enginesize
        from
            (select make, max(enginesize) as max_enginesize
             from cars
             group by make
            ) as a
        left join
            (select make, type, sum(enginesize) as total_enginesize
             from cars
             group by make, type
            ) as b
        on a.make = b.make;
quit;
```

Figure 2: AI correcting code to ensure good programming practices are used.

HYPERREALISM VS. SURREALISM

OK, AI has fixed someone's poor attempt at programming, but it resembles something Salvador Dali painted, i.e., you can see a clock but don't understand why it is melting. Our programs need to be universally understood, this is why we add comments, but some programmers think "no one will ever see this code again, I understand it, so I don't need comments!". Wrong, comments are needed everywhere all the time, and no it is not possible to provide too many comments. If you have written a comment and it's a tad too long, I at the very least still understand what your code is doing. It is easier to tell someone to write less than to write something. So, we have our second problem for AI to solve, the dreaded uncommented code.

Question to Asked to AI
Could you add some comments to the code to help me understand?
Code Corrected by AI

```
proc sql noprint;

    /* Create a new table 'cars' from the dataset 'sashelp.cars'.
       This table sorts the cars based on their make and type. */
    create table cars as...

    /* Create a new table 'cars_2' that will contain:
       - The make of the car
       - The type of the car
       - The maximum engine size for each make
       - The total engine size for each make and type combination */
    create table cars_2 as...

        /* This subquery calculates the maximum engine size for each make.
           The result is aliased as 'a' for reference in the main query. */
        (select make, max(enginesize) as max_enginesize...

        /* This subquery calculates the total engine size for each make and type
           combination. The result is aliased as 'b' for reference in the main
           query. */
        (select make, type, sum(enginesize) as total_enginesize...

quit;
```

Figure 3: AI successfully adding comments to help us understand the code. Some code has been removed for ease of reading.

COMPLICATED VS. SIMPLE

The code is structured well, we have some nice comments, we understand it, what we don't understand is why they decided to complicate such a basic problem. I often see some super elaborate PROC SQL code (i.e., nested PROC SQL especially), and after time I soon realise this "advanced" bit of code is doing something basic but in only a few lines. I am not sure where the idea of less code = good came from. Long programs do not necessarily indicate a bad program, the key thing to remember for any bit of code, is code should be universally understood. I know what some programmers may think, "but you don't understand I did it in one procedure!?". Well done you, gold star…AI could you please knock it down.

Question to Asked to AI
Could you do the same task this code is doing without using PROC SQL?
Code Corrected by AI

```
/* Step 1: Sort the sashelp.cars dataset by make and type */
proc sort data=sashelp.cars out=cars;
```

```
    by make type;
run;

/* Step 2: Calculate max_enginesize for each make */
proc means data=cars noprint max;
    var enginesize;
    by make;
    output out=max_enginesize max(enginesize)=max_enginesize;
run;

/* Step 3: Calculate total_enginesize for each combination of make and type */
proc means data=cars noprint sum;
    var enginesize;
    by make type;
    output out=total_enginesize sum(enginesize)=total_enginesize;
run;

/* Step 4: Merge datasets together based on make */
data cars_2;
    merge max_enginesize (in=a) total_enginesize (in=b);
    by make;

    /* Only keep rows where there's a match from both datasets */
    if a and b;
run;
```

Figure 4: AI simplifying the overly complicated, much better.

LESSONS LEARNT

AI can clearly tidy programs up and help programmers understand a mess, but is this helpful? Or is AI allowing even lazier programmers to follow the Richard Branson ethos, i.e., enabling lazy people to find the quickest solution. For AI to truly be effective, it needs to educate the user not simply do their job.

RATIONALE THOUGHTS PLEASE

It is important to note, in every example above (figures 2, 3 and 4) I did not include the rationale that the AI autogenerated. This rationale is AI both defending and explaining its changes. So, let's revisit figure 2, where we asked the AI to adjust the original code to follow good programming practices and see the rationale is provided.

Question to Asked to AI
Could you reformat this code to follow good programming practices without adding anything to the code?
Rationale for Changes Provided by AI
Here's a summary of what I've done: 1. ***Comments:*** *Provided comments to give context to what each step or subquery does.* 2. ***Indentation:*** *Used consistent indentation for better readability.* 3. ***Spacing:*** *Provided appropriate spacing between different parts of the SQL statements.* 4. ***Alignment:*** *Aligned select, from, and group by clauses for better readability of subqueries.* *Remember that good programming practices often vary by team or organization, so always consider the standards in your specific environment.*

Figure 2B: Rationale provided by AI omitted from figure 2, which explains why the AI did what it did when improving the original codes layout.

This is a great rationale; it is educating me on why the changes were made and the importance of consistency and clarity within a team. What is even more impressive is when we look back at the rationale the AI provided for replacing the complicated PROC SQL with a simpler method, from figure 4. If you recall from figure 4 the code the AI generated included comments prefixed with step 1, step 2, step 3 and step 4. This is because the rationale was written as steps to help the user understand each change made.

Question to Asked to AI
Could you do the same task this code is doing without using PROC SQL?
Rationale for Changes Provided by AI
Yes, you can achieve the same result using SAS DATA steps and other PROCs without the use of PROC SQL. Here's a step-by-step process to achieve this: 1. Sort the **sashelp.cars** dataset by **make** and **type.** 2. Calculate the **max_enginesize** for each **make** using PROC MEANS. 3. Calculate the **total_enginesize** for each combination of **make** and **type** using PROC MEANS. 4. Merge the datasets together based on the **make.**

Figure 4B: Rationale provided by AI omitted from figure 4, which explains what each step was doing in the simplified code.

SPOT THE DIFFERENCE

But wait, this still overlooks the possible issue of a language barrier, for example I know a lot of people have skimmed through this very paper, simply because they have seen walls of text, and thought “this is too much text for me!”. They have likely not even reached this paragraph, but I am certain they have studied the figures. So maybe rationale alone is not the best way for AI to explain the differences between the original code and the AI-corrected code, because rationale assumes everyone’s understanding of the English language is the same, which is not true.

Take SAS®, SAS® is a language, albeit a programming language but a language nonetheless, and like any language we see differences. My English is going to be different to someone else’s English if they were raised in America, since I learnt my English while living in Wales (for those unaware Wales is a country next to England in the UK, I digress). My American counterpart and I both speak English but there are differences, e.g., Haematology vs. Hematology. These subtle differences are also seen in programming, and AI will have its own preference when writing code, so rationale alone is not perfect. So, could AI visually highlight the differences, without using comments or rationale?

Let’s take the original (messy) program and ask AI to do something else. We still want AI to improve the code, but we want AI to also highlight the changes in a universally readable method, ideally without writing out a comment. A visual and clear indication of what AI changed.

Question to Asked to AI
You know the code I originally showed you, and then the improved code you showed me, are you able to visually highlight the differences?
Answer Provided by AI

I'm unable to directly highlight the differences visually within this medium. However, I can provide you with a side-by-side comparison, marking changes with annotations in textual format. For example:
 	
Original:
`create table cars as`
`select *`
 	
Improved:
`    create table cars as`
`    select *`

```
Comparison:
[Added indent]    create table cars as
[Added indent]    select *

Would you like a side-by-side comparison in this style for the entire code?
```

Figure 5: Asking AI to visually illustrate the changes it makes to the code.

OK, I have the AI asking me a question. This may give us an interesting result, so let's see how AI illustrates the changes it made to the original code seen in figure 2. For ease of reading, I will only share the highlighted comparison, please refer to figure 2 for the original and improved code.

~~Question to Asked to AI~~ **Answer to the AI's Question**
Yes please, but could you also highlight any new lines introduced?
Answer Provided by AI

```
proc sql noprint;
[Indent]    create table cars as
[Indent][Indent]    select *
[Indent][Indent]    from sashelp.cars
[Indent][Indent]    order by make, type;
[New Line]
[Indent]    create table cars_2 as
[Indent][Indent]    select a.make, b.type, a.max_enginesize, b.total_enginesize
[Indent][Indent]    from
[Indent][Indent][Indent]    (select make, max(enginesize) as max_enginesize
[Indent][Indent][Indent]     from cars
[Indent][Indent][Indent]     group by make
[Indent][Indent][Indent]    ) as a
[Indent][Indent]    left join
[Indent][Indent][Indent]    (select make, type, sum(enginesize) as total_enginesize
[Indent][Indent][Indent]     from cars
[Indent][Indent][Indent]     group by make, type
[Indent][Indent][Indent]    ) as b
[Indent][Indent]    on a.make = b.make;
quit;
```

Figure 6: Asking AI to show me its visual "spot the difference" attempt of the changes seen in figure 2 between the original code and the AI-improved code.

This is not bad, there is the obvious issue with the comparison though, the original code went from 11 lines to 20 lines, but the above visual comparison mentions the text [New Line] only once. Regardless, AI did visually show us the differences, pair this with the rationale and I think AI did quite well.

FEED IT ERRORS

OK, AI has successfully helped us correct some problematic code. We should reward it with some coding issues, feed it an error or two. Programmers experience programming issues every day, I have certainly never stopped seeing issues in over a decade. Maybe it is because we find ourselves in a SAS® programmer role, with little to no experience and or support. Maybe we are too proud to ask for help, as this is accepting defeat. Let's present AI with a very common issue, date conversion.

The issue, I want to convert my character date variable, of the format DDMMMYYYY (e.g., 11AUG2023) into a numeric variable but my code is not working, let's see if AI can help.

Question to Asked to AI

Slightly off topic, I am trying to convert a character date variable in SAS of the format DDMMMYYYY, to be numeric, using the following code:

date_n = put(date_c, yymmdd10.);

However, I am getting the following note in the log and the variable in my output dataset is not numeric:

NOTE 484-185: Format $YYMMDD was not found or could not be loaded.

Can you help me?

Answer Provided by AI

Certainly. The issue you're facing is due to the use of the **put** function, which is trying to apply a numeric date format (**yymmdd10.**) to a character variable. In doing so, SAS is interpreting it as if you're looking for a character date format, hence the **$YYMMDD** in the log note. This is not what you want.

If you're trying to convert a character date variable of the format **DDMMMYYYY** to a numeric date variable, you should be using the **input** function instead of **put.**

Here's the correct code:

```
date_n = input(date_c, date9.);
```

The **date9.** informat reads the date in **DDMMMYYYY** format and converts it to a numeric date variable that represents the number of days since January 1, 1960.

Figure 7: AI successfully providing a solution to a date conversion issue.

This is correct, and the rationale explains the reason I may have witnessed the error in the first place. But I did have to provide a very precise question, which is not ideal.

PRECISION IS KEY

One thing is abundantly clear, you must be quite precise when posing your problem to AI. Take the date conversion prompt above in figure 6, the question posed is rather detailed:

- The input date format is mentioned i.e., DDMMMYYYY.
- SAS® is explicitly mentioned.
- We have also defined the input variable as character..
- I also show the AI code I am attempting to use.

Without providing the AI with this level of detail, AI typically gets things wrong.

Question to Asked to AI

Slightly off topic, I am trying to convert a date variable however, I am getting the following note in the log and the variable in my output dataset is not numeric:

NOTE 484-185: Format $YYMMDD was not found or could not be loaded.

Can you help me?

Answer Provided by AI

1. **Converting a Character Date to a Numeric Date:** If your date is stored as a character string and you want to convert it to a SAS date (numeric), you'd use the **INPUT** function.

2. **Applying a Format to Display a Numeric Date:** Once you have a numeric date, you can use the **FORMAT** statement to control how it's displayed.

```
data want;
    set have;

    /* Convert character date to numeric date */
    date_num = INPUT(date_char, yymmdd10.);

    /* Apply format to the numeric date */
    FORMAT date_num YYMMDD10.;
run;
```

Figure 8: AI providing an attempt at a solution when I pose a vague question.

Now, I should be clear, the AI is not wrong, what it is saying is correct, however my input date variable is of the format DDMMMYYYY so using the format YYMMDD10 in the INPUT function would not work in my code, the fault is with our question not the AI.

Additionally, AI often likes to provide additional bits of wisdom without being asked to do so, like in the above example in figure 7, I never asked it about adding a format, but it told me about the FORMAT statement anyway.

Conversely, the AI would completely forget snippets of earlier questions, a good example of this is my first question in figure 2 clearly mentioned SAS® programming, but by the time I got to my question in figure 6, the AI ignored this fact and began showing me answers in different programming languages.

Original Code

vbnet Copy code

```
proc sql noprint;
create table cars as
select *
from sashelp.cars
order by make, type;
create table cars_2 as select a.make, b.type, a.max_enginesize, b.total_engi
from (select make, max(enginesize) as max_enginesize from cars
group by make) as a left join
(select make, type, sum(enginesize) as total_enginesize
from cars group by make, type) as b on a.make=b.make;
quit;
```

Figure 9.1: AI presenting solution using visual basic (VB) formatting.

Improved Code

css Copy code

```
proc sql noprint;
[Indent]    create table cars as
[Indent][Indent]    select *
[Indent][Indent]    from sashelp.cars
[Indent][Indent]    order by make, type;
[New Line]
[Indent]    create table cars_2 as
[Indent][Indent]    select a.make, b.type, a.max_enginesize, b.total_engines
[Indent][Indent]    from
[Indent][Indent][Indent]    (select make, max(enginesize) as max_enginesize
[Indent][Indent][Indent]     from cars
[Indent][Indent][Indent]     group by make
[Indent][Indent][Indent]    ) as a
[Indent][Indent]    left join
[Indent][Indent][Indent]    (select make, type, sum(enginesize) as total_eng
[Indent][Indent][Indent]     from cars
[Indent][Indent][Indent]     group by make, type
[Indent][Indent][Indent]    ) as b
[Indent][Indent]    on a.make = b.make;
quit;
```

Figure 9.2: AI presenting solution using CSS formatting.

Now you could argue the colour coding is quite useful, but I am illustrating the issue that one of the key benefits of AI, i.e., that AI can think for itself, can sometimes present unwanted results.

CONCLUSION

The simplest conclusion is AI is more readily available than ever before, and if you sit on the side of the fence of “AI is going to take my job”, then I would argue you are probably in the wrong job and AI should be the least of your worries. I welcome all the weird, funny and slightly alarming stories around AI but I can conclude AI has the power to train and educate the next generation of programmers and aid the current class of programmers. It can improve language barriers, it can take some of the stress away when solving problems, *ac efallai hyd yn oed dysgu rhywfaint o Gymraeg i chi* (I encourage you to ask AI to translate this).

I recently did a mini survey on LinkedIn, posing the very simple question “*How will Artificial Intelligence (AI) impact your job?*” with two answers, “Positively” or “Negatively”. The consensus is AI is an aid to the majority and not the “inevitable” replacement.

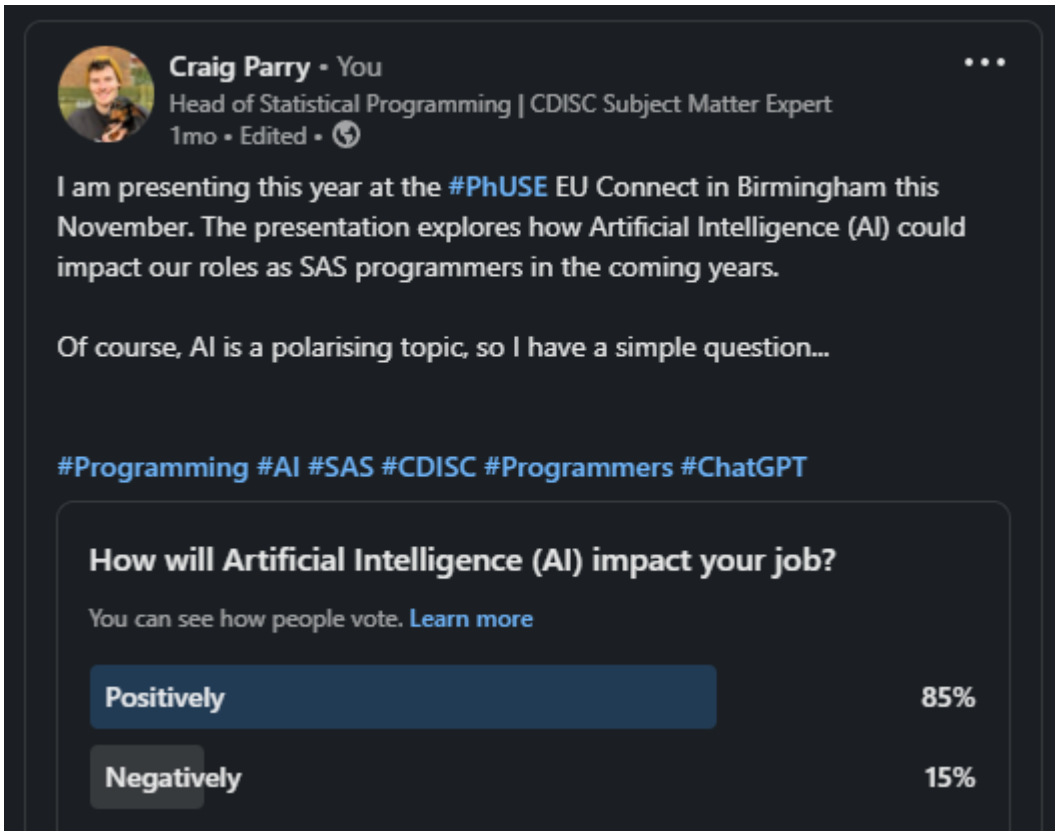

Figure 10: LinkedIn a recent poll looking at how people perceived AI chat bots in relation to their jobs [6].

P.S. I really did reference my own LinkedIn profile [6], I have no shame.

REFERENCES

[1] https://chat.openai.com/
[2] https://trends.google.com/trends/explore?geo=GB&q=ai&hl=en
[3] https://www.ox.ac.uk/news/features/how-artificial-intelligence-shaping-medical-imaging
[4] https://medcitynews.com/2023/06/what-every-health-system-cfo-needs-to-know-before-using-ai-for-medical-coding/#:~:text=Fully%20autonomous%20coding%20means%20that,you%20apply%20across%20the%20board.
[5] https://www.bbc.co.uk/news/uk-england-cornwall-66508840
[6] https://www.linkedin.com/posts/craigparry_phuse-programming-ai-activity-7091887047895867393-ybpA?utm_source=share&utm_medium=member_desktop

ACKNOWLEDGEMENTS

A huge thank you to my programming team, well the ones who reviewed this paper.

RECOMMENDED READING

I can't think of any book which would educate you on AI chat bots, maybe my LinkedIn profile, I do quite like the book "this is going to hurt" by Adam Kay but totally unrelated to AI, great book though.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Craig Parry
MAC Clinical Research / CraigYdwl Ltd.
Swansea, Wales
LinkedIn: https://www.linkedin.com/in/craigparry/