

L'IA per l'analisi dei dati in R

Gli strumenti IA fanno oramai parte del normale modo di lavorare nel contesto dell'analisi dei dati e del *coding*, ed è dunque fondamentale imparare ad utilizzarli in modo consapevole.

Il rischio principale, per il principiante e non solo, è quello di porre eccessive aspettative su strumenti che immaginiamo quasi magici, in grado di fare l'analisi dei dati o di scrivere il codice al posto nostro.

Introduzione

Nell'era dell'Intelligenza Artificiale (IA), il *coding* rimane davvero una competenza essenziale per l'analisi dei dati?

In realtà sì, e anzi secondo me è ancora più strategica di prima, per sfruttare al meglio le potenzialità degli strumenti IA per l'analisi. La capacità di scrivere e leggere il codice è fondamentale per gestire e guidare il flusso di lavoro, e per accedere a funzionalità più avanzate e complesse che sono ora (quasi) a portata di mano.

Per fare le domande

Prima di tutto, gli strumenti IA non sostituiscono la necessità di **comprendere e interpretare il codice**. Essi, infatti, non possono sapere cosa vogliamo fare noi, ma si limitano ad analizzare le richieste (tutt'al più con il contesto aggiuntivo dei file che carichiamo), confrontarle con la loro base di apprendimento, per poi fornire la risposta/soluzione più probabile.

Questo significa che, in generale, per ottenere risposte utili, **dobbiamo essere in grado di formulare richieste** chiare e specifiche, il che richiede una conoscenza di base di R e delle procedure di analisi dei dati.

Per comprendere le risposte

E poi **le IA sbagliano**, per tanti motivi (allucinazioni, ma anche interpretazioni errate del prompt e limiti dei dati di addestramento; vedi oltre) e il principiante potrebbe avere difficoltà a identificare e correggere l'errore.

Capita che i suggerimenti dell'IA siano sin **troppo avanzati e complessi**, rispetto alla capacità dell'utente di comprendere cosa fa il codice e valutare se fa al caso.

Quando poi le due cose si uniscono in **suggerimenti troppo complessi e non funzionali**

agli obiettivi di analisi (e in questo senso 'sbagliati') ... probabilmente si dovrà ricominciare daccapo, dopo aver perso un bel po' di tempo.

Per gestire il processo di analisi

Un consiglio classico è quello di dividere il compito (ad esempio la procedura di analisi) in *steps* che possiamo gestire, e poi farci aiutare dall'IA. Procedere per gradi aiuta a mantenere il controllo del processo.

Naturalmente, anche nel progettare il processo possiamo farci aiutare dall'IA: la cosa importante, però, è che i singoli passi siano comprensibili e gestibili per noi, almeno in linea di principio.

Altro aspetto importante, specie nel contesto accademico, è che gli assistenti IA potrebbero non essere in grado di gestire analisi che richiedono **pacchetti meno usati e molto specializzati**, in quanto, per vasta che sia la base di addestramento, la documentazione di pacchetti di nicchia potrebbe non rientrarvi (è un classico *bias*).

Ciò è tanto più probabile per gli utenti di **lingua** italiana, che è decisamente sottorappresentata in molti LLM.

Le chat IA

Le chat IA sono applicazioni che utilizzano i modelli di linguaggio naturale (LLM) per interagire con gli utenti in modo conversazionale.

Com'è noto, questo significa che è possibile chiedere all'IA: “Crea uno script per calcolare la media di questa variabile” e l'IA fornirà il codice corrispondente.

Le chat sono particolarmente utili **per il principiante**, che potrà porre **domande strutturate** ed ottenere le risposte, con un mix di [markdown](#) e codice, immediatamente ri-utilizzabile.

Nel contesto di un uso di base del codice, le chat possono essere usate non solo per generare il codice necessario, ma anche per

- spiegare il codice e gli output;
- identificare errori e suggerire correzioni;
- (ri)formattare lo *script* applicando convenzioni di stile, indentazioni e spaziature coerenti, rendendolo conforme agli standard (vedi [Buone pratiche del coding in R](#));
- generare documentazione e scrivere i commenti.

Gli assistenti integrati per il codice

Esistono poi **assistenti IA integrati negli ambienti di sviluppo**, come *Codeion* (ora [Windsurf](#)), [GitHub Copilot](#), o [Gemini Code Assist](#), a cui è dedicata la pagina [Gli assistenti al codice IA](#)

Assistono nella scrittura del codice, sono spesso “agentivi”, eseguono cioè azioni complesse in modo autonomo, ma evidentemente richiedono la conoscenza tanto del codice quanto delle IDE nelle quali sono integrati.

A volte sono collegati ad una chat anch'essa integrata, e quindi si riescono ad usare in maniera abbastanza intuitiva già ad un **livello intermedio**

Integrazione degli LLM tramite API

Con l'accesso alle API di un LLM (Large Language Model), è possibile **comunicare direttamente con il modello in R** (programmaticamente, ovvero da script/terminale) ed eseguire numerose funzioni.

I pacchetti per R

Svariati sono i pacchetti disponibili a tale scopo, fra i quali ricordo:

- *ellmer*, pacchetto sviluppato da Posit, per interagire con molti LLM, **inclusi i modelli locali** (si interfaccia con [Ollama](#))¹⁾;
- *chattr* per usare in RStudio le chat dei modelli integrati via *ellmer*²⁾.
- *openai* per le api di OpenAi;
- *gptstudio* e *gpttools* per le api di OpenAi, Gemini, Anthropic e altre;
- *gemini.R* per le api di Gemini;

La comunità di R è molto attiva nello sviluppare nuovi strumenti e pacchetti per sfruttare al meglio le potenzialità degli LLM, che a loro volta cambiano molto velocemente. Questo elenco esemplificativo, oltre che limitato, potrebbe quindi essere già obsoleto (scrivo a settembre 2025).

Uso programmatico: che vuol dire?

In ogni caso, pacchetti specifici a parte, questo tipo di integrazione consiste nell'inviare comandi e ricevere risposte direttamente all'interno dell'ambiente di programmazione o sviluppo, automatizzando flussi di lavoro complessi.

In linea di massima, il funzionamento somiglia a quello a cui ci hanno abituato le chat, ma

prompt e risposte sono file / script, o testo che può poi essere salvato in un file (dipende dalle funzioni dei pacchetti, se non li si usa da terminale).

Si possono ad esempio inviare richieste per analizzare dati o eseguire operazioni di machine learning direttamente da R, e ricevere un output gestibile in R, senza dover passare attraverso un'interfaccia web, esportare o importare, o copia-incollare, file. Oppure, nel campo dell'elaborazione del linguaggio naturale, si può creare uno script R che invia dati a un LLM per l'analisi del sentiment, elabora la risposta dell'LLM e poi utilizza i risultati per generare grafici o report.

Modelli generalisti, non app

I modelli ai quali accediamo in questo modo, però, **non sono “addestrati”** a svolgere un compito specifico, quindi **il prompt deve essere molto strutturato** e circostanziato. Anche **modelli ottimizzati per il codice**, come Code LLaMA, restano generalisti dal punto di vista dell'interazione: non hanno natura conversazionale, e richiedono prompt strutturati. Se la risposta non va bene, bisogna ridefinire il prompt. Non possiamo chiedere: “mi spieghi meglio questo punto?”.

In una parola, l'utente deve sapere molto bene cosa fare, e questo, per l'utente non tecnico, questo può rappresentare una barriera *almeno* quanto il terminale o lo *scripting*.

La gestione del processo richiede quindi una buona familiarità con R (o gli altri linguaggi di programmazione, che si intende utilizzare), con gli LLM e con la gestione delle eventuali chiavi API.

Questo è il motivo per cui, in campo accademico, si stanno diffondendo software che integrano i modelli nel contesto d'uso, pur lasciando all'utente la possibilità di scegliere a quale servizio iscriversi o se installare un modello locale per motivi di sicurezza o privacy. In questo modo, lo scienziato sociale, o lo psicologo, o il medico, potranno dedicarsi alla sola analisi dei dati con il supporto di strumenti avanzati, mantenendo un elevato grado di autonomia nelle decisioni epistemologiche ed etiche.

[RStudio](#), [ide](#), [IA](#), [coding](#)

1)

Per l'uso con i modelli locali: <https://posit.co/blog/setting-up-local-llms-for-r-and-python/>

2)

Non è più disponibile per Copilot Github; per dettagli, si veda il [post sul blog ufficiale](#)

From:

<https://www.agnesevardanega.eu/wiki/> - **Ricerca Sociale con R**

Permanent link:

https://www.agnesevardanega.eu/wiki/r/rstudio/strumenti_ia_analisi_dei_dati

Last update: **15/10/2025 05:32**

