Table S8. Package-specific guardrails and p-value tracking.

Package	Guardrails
pandas	● Override dataframe's runtime *KeyError* to not only list the wrongly used key but also provide a list of all available keys. ● Set dataframe's custom string representation with float formatted to two decimal points. ● Override dataframe's *repr* function to avoid omitting columns. ● Raise runtime error when using unallowed dataframe methods, such as *to_html* and *to_json*. ● Override dataframe's *to_latex* function to add chars escaping in table and caption. ● Do not allow changing the original values of a Series of a dataframe that was read from an external file. Instead issue a rule-based message requesting to create a new series with a sensible name to avoid coding mistakes.
statsmodels	● P-value tracking*. ● Do not allow p-values that are either NaN or 1, which indicate erroneous or ill-defined statistical tests. ● Prevent running the *fit* method of a given instance more than once (multiple calls to this function return the same result object, leading to analysis mistakes when not handled properly by ChatGPT code). ● Allow access only to the structured summary of statistical tests (the *summary2* method), and not the textual summary (the *summary* method). This guardrail prevents ChatGPT errors related to parsing of the textual output of *summary*. ● Require performing at least one statistical test. ● Require incorporating a p-value in one of the saved dataframes. ● Require using string formula notation to run statistical models. ● Check for singularity of the covariance matrix to avoid multicollinearity problems.
scipy	● P-value tracking*. ● Do not allow p-values that are either NaN or 1, which indicate erroneous or ill-defined statistical tests. ● Require performing at least one statistical test. ● Require incorporating a p-value in one of the saved dataframes. ● Prevent unpacking or iterating over results objects. Instead, we only allow direct access to the output variables by name to avoid mistakes in accessing the variables in their correct order.
scikit-learn	● P-value tracking*. ● Do not allow p-values that are either NaN or 1, which indicate erroneous or ill-defined statistical tests. ● Limit the grid size to 30 when using *GridSearchCV*. Limit the number of iterations to 30 when using *RandomizedSearchCV*. ● Override *random_state* parameter for *RandomForestRegressor*, *ElasticNet* and *MLPRegressor* classes to automatically set it, if not provided, to assure reproducibility of the analysis results.

- Limit the size of the *MLPRegressor* hidden layers to 2 layers and 50 neurons per layer to avoid too long runtimes.

* P-value tracking:
Rationale: We implement a method to track p-values in order to give rule-based feedback requiring the LLM-created code to correctly format too small p-values as “<1e-6” (or any other user-chosen limit). Without this guardrail, ChatGPT often creates tables with the raw p-values as output by statistical test functions, which could be extremely close to 0 (or even equal to 0 due to floating point rounding).
Implementation: All statistical functions of the package that output p-values are monkeypatched such that p-values are converted from floats to a custom *PValue* class. Objects of this class behave like a float, yet they can be distinguished based on their type. In the “Data analysis” coding step, when the LLM-created code calls such a monkeypatched function, it gets *PValue* objects, which it may subsequently incorporate into the dataframe tables that the code creates. Then, in the “Table design” coding step, the LLM is instructed to use four custom functions that we wrote (see “Performer mission prompt”, “Table design”, Supplementary Table 1): (a) *to_csv_with_note()* which converts a dataframe to LaTex; (b) *format_p_value()* which converts numeric values to strings, such that values lower than 1e-6 is converted to “<1e-6”; (c) *is_str_in_df()* which allows checking if a given string is found in the dataframe’s columns or index; (d) *split_mapping()* which allows creating the table’s legend and a mapping from abbreviations to full names. Importantly, in addition to their normal functionality, as advertised to the LLM, these functions also check and issue a rule-based feedback message either when the code applies *format_p_value* on non-*PValue* arguments, or when the code calls *to_csv_with_note* with a dataframe containing unformatted *PValue* objects. This way we provide rule-based feedback on code that left some p-values unformatted, as well as on code that erroneously apply *format_p_value* on numeric values that are not p-values.