Handling Missing Values in `predict_response` with `svyglm`

KOUASSI KONAN LEGER & HUANG Ting

March 20, 2025

1 Introduction

When computing predictions with an `svyglm` model, an issue arises when the `newdata` argument contains missing values. If `newdata` includes missing values in predictor variables, the associated weights (`weights`) are incorrectly applied, leading to inaccurate results.

In earlier versions of the `marginaleffects` package (e.g., `0.15.1`), this behavior resulted in an explicit error, preventing users from obtaining biased results. However, in recent versions, no error was raised, potentially leading to misinterpretations.

2 Modifications

We modified the `predict_response_c` function to:

- Remove observations with missing values (default: `na.action = "omit"`).
- Provide mean imputation for missing values (`na.action = "mean_impute"`).
- Raise an error if `na.action = "fail"`.

3 Updated Code

Here is the revised function:

```
predict_response_c <- function(model, terms, margin = "mean_reference",
                               ci_level = 0.95, type = "fixed", condition = NULL,
                               interval = "confidence", back_transform = TRUE,
                               vcov = NULL, vcov_args = NULL, weights = NULL,
                               bias_correction = FALSE, verbose = TRUE, na.rm = FALSE,
                               na.action = "omit", ...) {
  margin <- getOption("ggeffects_margin", margin)
  margin <- insight::validate_argument(argument = margin,
                                       options = c("mean_reference", "mean_mode","
                                           marginalmeans", "empirical", "counterfactual", "
                                           full_data","average", "marginaleffects"))
  model_name <- insight::safe_deparse(substitute(model))
  type <- insight::validate_argument(type, c("fixed", "random", "response", "link"))
  interval <- insight::validate_argument(interval, c("confidence", "prediction"))

  dots <- list(...)
  newdata <- dots$newdata

  if (!is.null(newdata)) {
    predictor_names <- insight::find_predictors(model)$conditional
    required_vars <- c(predictor_names, weights)
    newdata <- newdata[, required_vars, drop = FALSE]
    missing_mask <- !complete.cases(newdata)

    if (any(missing_mask)) {
```

```
      if (na.action == "omit") {
        message(sprintf("      Warning: %d rows in `newdata` with missing values were
            removed.",
                        sum(missing_mask)))
        newdata <- newdata[!missing_mask, , drop = FALSE]
      } else if (na.action == "fail") {
        stop("Error: `newdata` contains missing values. Please handle them before proceeding
            .")
      } else if (na.action == "mean_impute") {
        for (var in required_vars) {
          if (is.numeric(newdata[[var]])) {
            newdata[[var]][is.na(newdata[[var]])] <- mean(newdata[[var]], na.rm = TRUE)
          }
        }
        message("      Warning: Mean imputation applied for missing values.")
      }
    }
  }
  dots$newdata <- newdata
  out <- ggpredict(model, terms = terms, ci_level = ci_level, type = type,
                   condition = condition, back_transform = back_transform,
                   vcov = vcov, vcov_args = vcov_args, interval = interval,
                   bias_correction = bias_correction, verbose = verbose, ...)

  attr(out, "model.name") <- model_name
  out
}
```

4 Validation and Testing

Unit tests were added to validate these modifications:

```
if (!requireNamespace("testthat", quietly = TRUE)) {
  install.packages("testthat")
}
library(testthat)

# Validate missing value handling
test_that("Missing value handling", {
  model <- svyglm(api00 ~ ell + acs.core + awards, design = dstrat)

  expect_message(
    predict_response_c(model, "awards", margin = "empirical",
                       newdata = dstrat$variables, weights = "pw"),
    "rows in `newdata` with missing values were removed"
  )

  expect_error(
    predict_response_c(model, "awards", margin = "empirical",
                       newdata = dstrat$variables, weights = "pw", na.action = "fail"),
    "Error: `newdata` contains missing values"
  )

  expect_message(
    predict_response_c(model, "awards", margin = "empirical",
                       newdata = dstrat$variables, weights = "pw", na.action = "mean_impute"
                           ),
    "Mean imputation applied for missing values"
  )
})
```

5 Conclusion

With this correction, users can now control how missing values are handled in `newdata`, preventing silent errors and ensuring more reliable results.