Fitting and Interpreting MLR Models

Multiple linear regression – Stat 230

In this class, you will begin to explore how to add additional explanatory variables into a regression model in order to explain (or control) more of the variation in the response.

In this activity we will revisit the `Cars` data set described in Sections 3.1 and the first page of Section 3.3 (page 70).

```
cars <- read.csv("https://aloy.rbind.io/kuiper_data/cars.csv")
```

Fitting multiple linear regression (MLR) models

Before fitting your regression model it can be useful to explore the data in hand. With a large number of predictors, creating individual scatterplots is tedious. A solution is to create a *scatterplot matrix*. There are two functions you can use to create scatterplot matrices in R. The "quick" function is `pairs()`, which uses a formula interface to select the variables to include in the scatterplot matrix. Notice, however, that the formula is one sided, so all of the variable names are placed to the right of the `~`.

```
# Load the dplyr package in your setup code chunk to get select()
# cars |>
#   select(-Make, -Model, -Trim, -Type, -Cruise, -Sound, -Leather) |>
  pairs(~ Price + Mileage + Cyl + Liter + Doors, data = cars)
```

If you want a slightly "nicer" version that includes univariate density curves and correlation coefficients, then you can use the `ggpairs()` function found in the {GGally} package. Before we can create a scatterplot matrix, we should remove any categorical variables. The code below removes the `Make`, `Model`, `Trim`, `Type`, `Cruise`, `Sound`, and `Leather` columns using the `select` function and then passes the result into `pairs()`. Notice that a minus sign tells R to omit the column.

```
# Load GGally in the setup chunk!
cars |>
  select(-Make, -Model, -Trim, -Type, -Cruise, -Sound, -Leather) |>
  ggpairs()
```

What you will notice is that the default scatterplot matrix from `ggpairs()` doesn't place `Price` on the y-axis in most of the plots. To adjust the order the variables are displayed, add the `columns` argument. This also allows us to select variables within the plotting function:

```
cars |> ggpairs(columns = c("Mileage", "Cyl", "Liter", "Doors", "Price"))
```

Task 1. Use the above code to create a scatterplot matrix of the quantitative variables in the `Cars` data set. Which appear to be useful predictors of price?

Task 2. Fit the multiple linear regression model of `Price` against the other quantitative variables. This can be done using the `lm()` command and separating the names of the explanatory variables with a `+` sign. Fill in the blanks in the below code chunk to fit this model. Report the equation of the fitted multiple regression model.

```
cars_lm <- lm(Price ~ ___ + ___ + ___ + ___ , data = cars)
```

Task 3. Interpret the following regression coefficients in context:

- the y-intercept
- the slope for `Mileage`
- the slope for `Doors` (think carefully whether you want a 1-door increase here, or some other unit increase)

Task 4. To print the table of regression coefficients and the results of a t-test for each coefficient you can use the `summary()` command. This is a "quick" way to print the table, but it prints too much information if you only want to focus on the coefficients. In this case, the `tidy()` command in the {broom} package is a better function. Based on this table, which variables appear to be associated with the price?

```
# Load the broom package in the setup chunk first!
tidy(cars_lm)
```

Task 5. To calculate confidence intervals for each regression coefficient we still use the `confint()` command. Remember that the `level` argument can be set to change from the default 95% confidence level. Use `confint()` to calculate 90% confidence intervals for each regression coefficient. Interpret the interval for `Mileage` in context.

Task 6. How much variability in `Price` is explained by this multiple regression model?

Checking model assumptions

Now that you have fit your multiple regression model we need to check whether the assumptions are reasonably satisfied.

Task 7. (Adaptation of Activity 7.) To create plots of the residuals vs. the fitted values and each explanatory variable we use the `residualPlots()` function in the {car} package.

```
# Load the car package in the setup chunk!
residualPlots(cars_lm, type = "rstandard", tests = FALSE, quadratic = FALSE)
```

When we assess the assumptions using residual plots in multiple regression we interpret the plots just as we do in simple linear regression, we just need to look at more plots!

Layout of reisudal plots

If you have a large number of residual plots, or just want to arrange them on the page differently, the `layout` argument can be added to the `residualPlots()` command. For example, to have 2 rows and three columns of plots, we add `layout(2, 3)`.

```
residualPlots(cars_lm, type = "rstandard", tests = FALSE, quadratic = FALSE,
              layout = c(2, 3))
```

You can also control which residual plots are displayed using the `terms` argument. This might be necessary if you have a very large number of predictors. For example, we can create one set of residual plots with `Mileage` and `Cyl` and another with `Liter` and `Doors` using these two commands. Notice that we again use a one-sided formula to select the terms.

```
residualPlots(cars_lm, type = "rstandard", tests = FALSE, quadratic = FALSE,
              terms = ~Mileage + Cyl)
residualPlots(cars_lm, type = "rstandard", tests = FALSE, quadratic = FALSE,
              terms = ~Liter + Doors, fitted = FALSE)
```

In the second command, we also add `fitted = FALSE` so that the plot of the standardized residuals vs. fitted values is not reproduced.

(a) Based on these residual plots, does the assumption of linearity appear to be satisfied?

(b) Based on these residual plots, does the assumption of constant error variance appear to be satisfied or is there evidence of heteroscedasticity (which is another word for non-constant error variance)?

Task 8. Now, create a normal Q-Q plot of the standardized residuals using the `qqPlot()` command in the {car} package. Do the residuals appear to follow a normal distribution?

```
qqPlot(cars_lm, type = "rstandard", distribution = "norm")
```

Task 9. You should have detected a couple violations of the assumptions required for the multiple regression model to be valid. Apply a logarithmic transformation (your book uses `log10`) to `Price` and refit the multiple linear regression model. Create the same set of residual plots as above, has the transformation helped remedy the model violations?