Multinomial Logistic Regression: Speed Dating

Signal Data Science

You'll be formally learning about multinomial logistic regression today (sometimes called *softmax regression*[1]).

Previously, you used binomial logistic regression to do *two-class classification*, where you modeled the log-odds associated with a binary outcome as being linearly related to a number of predictor variables. The technique of *multinomial logistic regression* is a straightforward extension of this: our outcome variable has more than two categories, and we model the log-odds associated with falling into each category as being linearly related to our predictor variables.

Using multinomial logistic regression

You can use multinomial logistic regression with `glmnet(x, y, family="multinomial")`, where `x` is a scaled matrix of predictors and `y` is a numeric vector representing a categorical variable. In the following, we can simply use unregularized logistic regression because the number of predictors is relatively low. To do so, **don't** pass in `lambda=0` to `glmnet()`; instead, call `glmnet()` without specifying the `lambda` parameter, and when calling `predict()` or `coef()` on the output of `glmnet()`, simply set `s=0`.

Converting to probabilities

Suppose that we've fit a multinomial logistic regression model to some data and made predictions on the dataset. Now, for each particular row, we have a log-odds L_i associated to each outcome i. We sometimes want to convert to *probabilities* P_i, which ought to be proportional to the exponentiated log-odds $\exp(L_i)$. We can exponentiate and obtain just $\exp(L_i)$, but those values might not necessarily sum to 1: $\sum_i \exp(L_i) \neq 1$. This is a problem, because probabilities have to sum to 1, that is, $\sum_i P_i = 1$.

[1] This comes from the usage of the *softmax function*, which is a continuous approximation of the indicator function.

To resolve this, we divide each $\exp(L_i)$ by the proper *normalization factor*. That is, we can compute

$$P_i = \exp(L_i) / \sum_i \exp(L_i)$$

which makes all the values of P_i sum to 1 as desired while still being proportional to $\exp(L_i)$.

Speed dating dataset

Return to the aggregated speed dating dataset (`speeddating-aggregated.csv` in the `speed-dating` folder).

- Use `table()` on the career code column to find the four most common listed careers in the dataset.
- Restricting to those four careers, predict career in terms of self-rated activity participation and average ratings by other participants. Interpret the coefficients of the resulting linear model. Visualize them with `corrplot()`.
  - You can combine the output of `coef()` with `cbind()`, `do.call()`, and `as.matrix()` as input into `corrplot()`. Be sure to plot just the coefficients, *not* the intercepts of the linear models.
- Write a function `probabilities(preds, rownum)` that takes in a matrix `preds` of predictions generated from multinomial logistic regression (*i.e.*, a matrix of log-odds) and a row number `rownum`, returning row `rownum` converted into *probabilities*. Verify that the output sums to 1 as expected.