Logistic Regression: Speed Dating

Signal Data Science

We'll introduce logistic regression by returning to the Columbia speed dating dataset.

Unregularized logistic regression

First, we'll see how to use unregularized logistic regression.

Loading the data

The dataset you'll be using in this assignment is an aggregated form of the full speed dating dataset; you've worked with a simplified form of this dataset before (with fewer variables). Refer to the documentation in `speeddating-documentation.txt` for a description of the new variables.

- Use `read.csv()` to load `speeddating-aggregated.csv` in the `speed-dating` dataset.
- Use `complete.cases()` to determine the number and proportion of rows in the data with `NA`s. Clean the data by using `na.omit()` to remove all rows with `NA`s.

Using `glm()`

You can run logistic regression with `glm()`. It can be used in the same fashion as `lm()`, except for logistic regression you must pass in the additional parameter `family="binomial"`. Additionally, the column representing the binary class which you want to predict must either be (1) a numeric column taking on values 0 and 1 or (2) a factor.

The `pROC` package provides a function, `roc()`, which plots the receiver operating characteristic (ROC) curve given the results of a logistic regression fit. The output of `roc()` can be passed into `plot()` or directly printed to display the area under the ROC. Note that `roc()` accepts *probabilities* as inputs, but the

predictions made with a logistic regression model will be in the form of *log-odds ratios*, which must be converted into probabilities with

$$P = \frac{\exp L}{1 + \exp L}$$

where L is a log-odds ratio and P is the corresponding probability.

When working through the following questions, examine and interpret the coefficients of each logistic regression model. In addition, examine the area under the ROC curve as well as the shape of the ROC curve itself.

- Predict gender in terms of the 17 self-rated activity participation variables.
- Restrict to the subset of participants who indicated career code 2 (academia) or 7 (business / finance). Predict membership in either class in terms of the 17 activities.
- Restrict to the subset of participants who indicated being Caucasian (`race == 2`) or Asian (`race == 4`). Predict membership in either class in terms of the 17 activities.

Regularized linear regression

In this part of the assignment you'll be predicting decisions (`dec`) of speed dating participants in terms of interactions between partners' attributes.

Starter code is located at `speedDatingDecisionStarter.R` and the associated dataset is `speeddating-full.csv` in the `speed-dating` folder. We'll be using the *unaggregated* version of the dataset, with data about *both* the person being rated *and* each individual person doing the rating.

The first portion of the code creates a data frame with race and career code for both partners on each date, as well as the frequency with which the person making a decision expressed interest in seeing a partner again (`decAvg`), the frequency with which others expressed interest in the partner (`decPartnerAvg`), and the average attractiveness rating of the partner (`attrPartnerAvg`).

We'll be doing our cross validation at the level of speed dating events, so that there no participants appear in each of the train set and test set for any cross validation fold. The function `crossValidate()` in the starter code performs a grid search for `glmnet()` over values of α and λ, returning the area under the ROC for each pair (α, λ), and can correspondingly be used to find the best choice of coefficients.

- Using `dummy.data.frame()` from the `dummies` package, create a data frame `dums1` with dummy variables corresponding to the participant making the decision and another data frame `dums2` with dummy variables

corresponding to the partner being decided on. You only need to expand race and career code out into dummy variables.

- Create a data frame `dums` by calling `cbind()` on `dums1` and `dums2`. To this data frame, add interaction terms for
  - (race of decider) x (attractiveness of partner),
  - (career of decider) x (attractiveness of partner),
  - (race of decider) x (race of partner), and
  - (career code of decider) x (career code of partner),

  with column names formed by calling `paste(name1, name2, sep = ":")`.

  To save on computational time, remove those columns with 20 or fewer entries, with `dums = dums[,colSums(dums) > 20]`.

- Form a `features` data frame by binding `decAvg`, `decPartnerAvg`, and `attrPartnerAvg` to the data frame `dums`.

Keep the following in mind as you work: (1) If your `glmnet()` call returns an error indicating `NA` or `NaN` values in the predictors, use `is.nan()` to check for and filter out columns with `NaN`s. This occurs when `scale()` is used on constant columns (with standard deviation 0). (2) If your `glmnet()` model with a single λ value fails to converge, run `glmnet()` without specifying the `lambda` parameter, and subsequently pass in the desired λ value to `predict()` and `coef()` directly via the `s` parameter.

- For each of males and females, use `crossValidate()` to find the optimal values of α and λ for predicting `dec` in terms of the features. Then inspect the coefficients of the model corresponding to the best values of α and λ and discuss interpretation of the results with your partner.