Gmail **Andrew Ho <kironide@gmail.com>**

Week 4 Day 3 Assignment

1 message

Jonah Sinick <jsinick@gmail.com> Wed, May 25, 2016 at 2:22 AM
To: signal-2016-2@googlegroups.com

The pairings for today are

Melanie <---> Huey
Michael <---> Roko (<---> David)

Richard and Derrick are away.

You'll be working with the 2016 Less Wrong Diaspora survey data. The data is available here and the questions are available in pdf form here.

Data Processing

The data is far from being clean enough to analyze immediately.

- Restrict to users who answered "Yes" to the ResearchConsent question, who entered their age (!is.na(Age)), and who answered the Depression question (with value != ""). This should give you 1956 examples.
- Drop the columns corresponding to answer choices that users wrote in. (Hint: some of the column names are "other", "Write" and "comment", but there are others as well. You can find them by writing a function to return the number of unique values in each column of a dataframe.)
- Also drop the columns with only a single unique value.
- Drop the questions with "Calibration", "CharityDonations" and "Peak" in their titles. (For simplicity.)
- Remove the features corresponding to the write-in choices under the sections "What are the biggest problems with LessWrong now?" and "What would you want from a successor?"
- Convert each feature that's a **factor** into a feature with the same levels, omitting "" and "N/A". (Use the second argument of factor() to specify the levels). Then use addNA() to add a new level corresponding to NA.
- Select the **numeric** columns and use summary() to identify any features with obviously inaccurately recorded (or otherwise unrepresentative) outlier values. Replace such values with NA.
- Replace Income, IncomeCharityPortion and XriskCharity with their transformations under

  x ---> log(x + 1).

- Convert features corresponding to (yes/no) questions to binary (1/0) features.
- Convert the mental health features to numeric features

  No --> 0
  Not formally, but I personally believe I have (or had) it --> 1
  Yes, I was formally diagnosed by a doctor or other mental health professional --> 2

  indicating the strength of affirmative answer.

Do the same with the features with names containing as substrings "SuccessorCommunity", "BlogsRead", "BlogsRead2", "StoriesRead", "SQ001" and "Genetic."

There are a variety of individual questions for which you could do something similar (e.g. "Sequences") but it's arguably time consuming for what you get out of it, so do this only at your description.

Factor analysis

Do factor analysis on the numeric features (excluding Age, Income IncomeCharityPortion and XriskCharity) as follows:

- Compute the correlation matrix using use = "pairwise.complete.obs." Replace any NAs in the resulting matrix with 0's.
- Call VSS.scree() on the correlation matrix to see a plot of the eigenvalues of the principal components, and use this to choose a number of factors to extract. Err on the side of including more factors. You can also try multiple values.
- Use fa() from the psych package with covar=TRUE, rotate = "oblimin" and the number(s) of factors you obtained above. Example the factor loadings with reference to the original survey questions. Choose suitable names for the factors.
- Fill the numeric columns with the column means, and then use predict() on the fa() object obtained above to generate factor scores for each example.
- Form a new dataframe where the numeric features have been replaced by the factors.

Nonlinear classification

Choose a categorical variable to predict. Some ideas are

- NumberPartners
- Profession
- SexualOrientation
- Depression (converting back to a categorical variable)

First do a binary prediction task (which may require simplifying your target variable, e.g. NumberPartners might become "Zero" vs. "Nonzero", and Depression might become "Yes" vs. "No")

- First train a random forest model, and examine feature importance via $importance to see what the most predictive features are. Before proceeding, exclude features that are "too obviously" related to the feature that you're trying to predict (e.g. for depression, you might want to drop OCD and AnxietyDisorder).

  Train a random forest again. You can measure the predictive power of the forest by accessing $votes and interpreting the portion of votes as the probability assigned to the class, and calculating the area under the ROC curve using library(pROC).

- For a point of comparison, use cv.glmnet() with family = "binomial," first converting categorical predictors to dummy variables using dummy.data.frame() from the "dummies" package. Example the coefficients. Compare the magnitudes of the coefficients with the feature importances that came out of the random forest.
- For an apples to apples comparison, use the caret package with model = "rf" and model = "glmnet" and and compare the cross validated areas under the ROC. Then try model = "gbm", again comparing cross validated areas under the ROC.
- Try to make model that uses **as few variables as possible** while retaining a large majority of the predictive power of the original model. You'll probably find the coefficients from glmnet() and/or the variable importances from randomForest() and/or gbm() to be helpful

--