Principal Components Analysis

Use the `prcomp(..., scale=TRUE)` function to explore the principal components of various datasets.

A couple things to keep in mind: Let `p = prcomp(df, scale=TRUE)` be the result of PCA run on a data frame `df`. Then:

- `p$rotation` gives a matrix with rows corresponding to columns of `df`, columns corresponding to principal components, and entries corresponding to the *loadings* of a particular column of `df` on a particular principal component. Put another way, each principal component is formed out of a linear combination of the variables of `df`, and the column corresponding to each principal column corresponds to the *coefficients* of that linear combination.
- `p$x` gives the *principal component scores* for each row of `df`, that is, the *actual value* of each principal component for every row in `df`.
- `p$sdev` gives the *eigenvalues of the covariance matrix* of the data. One can interpret the `n`th value in `p$sdev` as corresponding to the relative proportion of the variance in the data explained by the `n`th principal component. Put another way, `p$sdev[n] / sum(p$sdev)` is the proportion of the variance in the data explained by the `n`th principal component.

For supplementary reading on the theory behind PCA, consult *Introduction to Statistical Learning* or *Applied Predictive Modeling*.

Write up your findings in an "R Markdown" file. (It's one of the options in RStudio when you go to make new files.) At the end, click on the "Knit HTML" button in RStudio, select "Knit PDF", and generate a PDF which you can upload to Github! (Now you can share your findings with friends and family in a more accessible form!) It's pretty intuitive how this works, but for additional guidance check out Yihui Xie's blog (he's the guy who developed `knitr`, the package that makes this all work in the background.)

PCA on the `msq` dataset

Prepare the data in this fashion:

- Extract the columns `active:scornful` from the `msq` dataset.
- Look at the number of `NA`s in each column (hint: use `colSums()` in conjunction with `is.na()`). For simplicity's sake, throw out the columns with a huge number of missing values and subsequently remove all the rows with any `NA`s.

Afterward, run PCA on the remaining variables.

- Write a function `top(n)` that prints out the top 10 *loadings* of the `n`th principal component, ordered by absolute value.
- Look at the PCA loadings for the first 5-10 principal components. Use `corrplot()` on the loadings (with the `is.corr=FALSE` option) to visually explore these relationships. After doing so, interpret and assign concise names to the principal components which seem to represent something coherent.
- Plot the eigenvalues obtained via `prcomp(...)$sdev`. How do their relative magnitudes relate to the interpretability of each principal component?
- Suppose that we use the first nth principal components to predict Extraversion and Neuroticism using a simple, unregularized linear model. Calculate a cross-validated RMSE for $n = 1, 2, \ldots$, plot them against n, and compare to the cross-validated RMSE which you got in the self-assessment when using regularized linear regression with all of the original variables. Interpret the results. (You can either calculate the RMSE by implementing cross-validation yourself, like you did in the self-assessment but simpler, or use `CVlm` from the `DAAG` package.)
- Read about the history of trait theories. How do the principal components relate to Extraversion and Neuroticism?

PCA on the speed dating dataset

Return to the aggregated speed dating dataset from last week, which you used for logistic regression.

- Drop the rows corresponding to people who didn't answer the activities questions.

Run PCA on the columns corresponding to the activity questions.

- Perform the same analysis which you didd with the `msq` dataset: looking at and interpreting the loadings of each principal component, visualizing them with `corrplot()`, and looking at the associated eigenvalues. Since there aren't very many variables, you can `cbind()` the activities to the principal component scores and then use `cor()` $\rightarrow$ `corrplot()` for easy visualization. As before, assign appropriate names to the principal components which seem to have a coherent meaning or interpretation.
- Predict gender, race (restricting to whites and Asians), and career code (restricting to academia and business / finance) using the principal components with logistic regression. Do so with the 1st principal component, the 1st and the 2nd, the 1st, 2nd, and 3rd, ... all the way up to every principal component. (You can just use `glm()` so you get the p-values too.) Interpret the coefficients.
- For the above regressions, use the `pROC` package to calculate the associated areas under the ROC curve. Compare to the results of using stepwise or regularized regression on all of the activities for the same predictions.