prVis --
A Novel Method for Visual Dimension Reduction

Wenxuan Zhao
Norman Matloff
Tiffany Jiang
Robert Tucker
University of California, Davis
Overview

- Motivation
- The Big Picture
- Features
  - Data Preprocessing
  - Processing
  - Result Processing
  - Producing Output
- Helper Functions
Motivation

Goals:
- Discover unknown patterns (Swiss Roll)
- Separation between known components (Spam Dataset)

Partial list of methods:
PCA, t-sne, UMAP, etc

Nice overview paper:

Authors: Lan Huong Nguyen, Susan Holmes
https://doi.org/ 10.1371/journal.pcbi.1006907
Motivation

Swiss roll:
The dataset was created to test various dimensionality reduction algorithms.

The idea was to create several points in 2d, and then map them to 3d with some smooth function, and then to see what the algorithm would find when it mapped the points back to 2d.
Motivation

How many components?

Plain PCA:
Motivation

How many components?

1?
Motivation

How many components?
Motivation

How many components?
5?
7?
Motivation

How many components?

prVis:
Motivation

4! prVis:
Motivation

**Spam**: A Kernlab built-in data set which has 57 predictors that predict whether a E-mail is spam
Motivation

Spam: A Kernlab built-in data set which has 57 predictors that predict whether a E-mail is spam
Motivation

**Spam**: A Kernlab built-in data set which has 57 predictors that predict whether a E-mail is spam
Motivation

Spam: A Kernlab built-in data set which has 57 predictors that predict whether a E-mail is spam
Many More Examples

Our github page:

https://github.com/matloff/prVis

Please refer to our gallery in the link below for dozen more examples:

https://github.com/matloff/prVis/tree/master/inst/gallery
The Big Picture

- Polynomial Expansion + PCA
- Gnanadesikan and Wilk, 1969
- Captures the non-linearity
- Simple but Powerful
Features

- Grouped features based on their functionality

- Based on needs we have in real life
<table>
<thead>
<tr>
<th>Data Preprocessing</th>
<th>Processing Stage</th>
<th>Result Processing</th>
<th>Producing Output</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Features: sub-sampling

Issue with large data sets:
- Dense Plot
- Time Consuming
- Space Consuming

nSubSam: option to subsample the data by specifying the number of rows we want.
Features: nInterval

nInterval: partitions one of the continuous variables into n intervals, each of them corresponds to one label (one color).

Features: scale

scale: Transforming the data to comparable scales
<table>
<thead>
<tr>
<th>Process</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Preprocessing</td>
<td>Processing Stage</td>
</tr>
<tr>
<td>Result Processing</td>
<td>Producing Output</td>
</tr>
</tbody>
</table>
Features: degree & maxInteractDeg

- Options for the subroutine `getPoly` powered by `polyreg`
- `degree`: specifies the highest degree for polynomial terms
- `maxInteractDeg`: specifies the highest degree for interaction terms
Features: handling large dataset

- Powered by package bigstatsr
  - Uses memory-mapping
  - Provides PCA for large matrices

- Enable users to handle “big” data set:
  - Data set with many columns
  - User specifies high `degree` & `maxInteractDeg`
  - Or both
Features: pca methods

- prcomp
- RSpectra
- By benchmarking the two implementations of PCA on various datasets, we have gained about 4-5 times speedup on average when using RSpectra.
<table>
<thead>
<tr>
<th>Data</th>
<th>Preprocessing</th>
<th>Processing Stage</th>
<th>Result Processing</th>
<th>Producing Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>Result Processing</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Features: outlier removal

- Outlier removal by class if any
- Uses Mahalanobis Distance
Removes no outlier

Removes 5% outliers
Removes outliers not by class

Removes outliers by class
<table>
<thead>
<tr>
<th>Data Preprocessing</th>
<th>Processing Stage</th>
<th>Result Processing</th>
<th>Producing Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>Producing Output</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Features: saving outputs

- For future replication
- Can be passed as argument to helper functions
- By default automatically saves the latest prVis output
A typical screen

# Typical Workflow
data(spam)  # loads the data
library(prVis)  # loads the library
# "scale" standardizes the data
# "labels" will label the data based on the category
# "pcaMethod" specifies which PCA method to be used
# "outliersrRemoved" removes 5% of the outliers
# "alpha" uses alpha blending provided by ggplot2
prVis(spam, scale=T, labels=T, pcaMethod="RSpectra", outliersRemoved=5, alpha=0.2, saveOutputs="lastPrVisOut")
Helpers: colorCode

Programmer and Engineer dataset:
Records age, gender, occupation, education level, and salary information of the programmers and engineers in the bay area.
Helpers: colorCode

- Display color coding for user-specified expressions

```r
colorCode(exps="age < 65")
colorCode(exps="age < 45")
colorCode(exps="age < 25")
```
More complex expressions

colorCode(exps="occ3==1")
colorCode(exps="occ3==1 * sex==1")
colorCode(exps="occ3==1 * sex==0")
Helpers: addRowNums

- Chooses \texttt{np} points at random from the \texttt{prVis} output, writing their \texttt{row numbers} on the plot.
- User can specify a vector that has 4 numbers, corresponding to percentages of the graph from left to right and bottom to top.
  - e.g. \texttt{c(0,1,0,1)} specifies the entirety of the graph.
  - \texttt{c(0,0.5,0.5,1)} specifies upper-left quadrant.
Thank you!

Merci beaucoup!

Our github page:

https://github.com/matloff/prVis
Questions?