Dimension Reduction

Montgomery Davis, Luke Andrade,
Anthony Chey, and Dominick Gambino

Completed
4/16/2021

Abstract

Abstract

We attempted to reduce the dimensionality of the Carseats data set. We did this so that data was simpler to interpret and to potentially remove multicollinearity. To do so, we used 4 different methods: Principal Component Analysis, Singular Value Decomposition, Sliced Inverse Regression, and Linear Discriminant Analysis. We gathered the results from each of these different methods. Upon analysis, we saw many different answers. We demonstrated multiple means of reducing dimensionality through 4 different methods, all with varying degrees of success. In the end, we conclude that you must see what will work best with the data that you have to find the best way to reduce the dimensionality of your data set.

Introduction

Introduction

Often, data sets will have many variables. When a data set has many variables, it becomes burdensome to work with and analyze. In order to eliminate the drawbacks of having many variables (high dimensional data), we seek to reduce the dimensionality of the data. There are many ways to do so, and reducing dimensionality has many benefits: removal of multicollinearity, simpler to visualize data, reduced complexity, and more easily interpreted data.

Introduction

We have chosen 4 methods that can reduce dimensionality for our chosen data set: Carseats (from the ISLR library). The 4 methods we chose are:

1. Principal Component Analysis
2. Linear Discriminant Analysis
3. Singular Value Decomposition
4. Sliced Inverse Regression

Materials and Methods/ Results

Principal Component Analysis

Principal Component Analysis

PCA is used to reduce large data with many highly correlated variables while still containing as much information as possible - preserve as much variability - within the original data set.

The purpose of reducing the data set is to make it easier to visualize and analyze the data.

Follows the steps of standardizing the variables, creating covariance matrix, computing eigenvectors and using eigenvalues of the covariance matrix to find the principal components.

PCA Code

m=1

```
> fa.data
Principal Components Analysis
Call: principal(r = S, nfactors = 1, covar = TRUE, rotation = "none")
Unstandardized loadings (pattern matrix) based upon covariance matrix
                    PC1      h2      u2      H2      U2
Sales              0.14 2.1e-02   7.955 2.6e-03 1.0e+00
CompPrice         -1.47 2.2e+00 232.987 9.2e-03 9.9e-01
Income            -0.22 5.0e-02 783.168 6.4e-05 1.0e+00
Advertising        1.77 3.1e+00  41.095 7.1e-02 9.3e-01
Population       147.38 2.2e+04   0.043 1.0e+00 2.0e-06
Price             -0.31 9.5e-02 560.490 1.7e-04 1.0e+00
ShelveLocBad       0.02 2.9e-04   0.183 1.6e-03 1.0e+00
ShelveLocMedium   -0.02 4.1e-04   0.248 1.7e-03 1.0e+00
ShelveLocGood      0.00 1.0e-05   0.168 6.1e-05 1.0e+00
Age               -0.70 4.9e-01 261.963 1.9e-03 1.0e+00
Education         -0.28 7.8e-02   6.789 1.1e-02 9.9e-01
Urban             -0.02 5.7e-04   0.208 2.7e-03 1.0e+00
US                 0.03 8.5e-04   0.229 3.7e-03 1.0e+00

                    PC1
SS loadings    21725.80
Proportion Var     0.92

 Standardized loadings (pattern matrix)
                 V   PC1      h2      u2
Sales            1  0.05 2.6e-03 1.0e+00
CompPrice        2  -0.1 9.2e-03 9.9e-01
Income           3 -0.01 6.4e-05 1.0e+00
Advertising      4  0.27 7.1e-02 9.3e-01
Population       5     1 1.0e+00 2.0e-06
Price            6 -0.01 1.7e-04 1.0e+00
ShelveLocBad     7  0.04 1.6e-03 1.0e+00
ShelveLocMedium  8 -0.04 1.7e-03 1.0e+00
ShelveLocGood    9  0.01 6.1e-05 1.0e+00
Age             10 -0.04 1.9e-03 1.0e+00
Education       11 -0.11 1.1e-02 9.9e-01
Urban           12 -0.05 2.7e-03 1.0e+00
US              13  0.06 3.7e-03 1.0e+00

                 PC1
SS loadings    11.96
Proportion Var  0.92

Mean item complexity =  1
Test of the hypothesis that 1 component is sufficient.

The root mean square of the residuals (RMSR) is  25.61

Fit based upon off diagonal values = 1
```

m=2

```
> fa.m2
Principal Components Analysis
Call: principal(r = S, nfactors = 2, rotate = "none", covar = TRUE)
Unstandardized loadings (pattern matrix) based upon covariance matrix
                   PC1   PC2      h2      u2      H2      U2
Sales             0.14  0.70 5.1e-01   7.469 0.06355 9.4e-01
CompPrice        -1.47 -4.85 2.6e+01 209.423 0.10940 8.9e-01
Income           -0.22 26.53 7.0e+02  79.412 0.89861 1.0e-01
Advertising       1.77  0.32 3.2e+00  40.995 0.07308 9.3e-01
Population      147.38 -0.03 2.2e+04   0.042 1.00000 1.9e-06
Price            -0.31 -8.58 7.4e+01 486.804 0.13161 8.7e-01
ShelveLocBad      0.02  0.03 1.4e-03   0.181 0.00756 9.9e-01
ShelveLocMedium  -0.02 -0.02 9.3e-04   0.247 0.00376 1.0e+00
ShelveLocGood     0.00 -0.01 1.1e-04   0.168 0.00067 1.0e+00
Age              -0.70  0.73 1.0e+00 261.424 0.00391 1.0e+00
Education        -0.28 -0.15 1.0e-01   6.766 0.01469 9.9e-01
Urban            -0.02  0.01 6.3e-04   0.208 0.00303 1.0e+00
US                0.03  0.03 1.9e-03   0.228 0.00844 9.9e-01

                          PC1    PC2
SS loadings          21725.80 802.16
Proportion Var           0.92   0.03
Cumulative Var           0.92   0.95
Proportion Explained     0.96   0.04
Cumulative Proportion    0.96   1.00

 Standardized loadings (pattern matrix)
                item   PC1   PC2      h2      u2
Sales              1  0.05  0.25 0.06355 9.4e-01
CompPrice          2 -0.10 -0.32 0.10940 8.9e-01
Income             3 -0.01  0.95 0.89861 1.0e-01
Advertising        4  0.27  0.05 0.07308 9.3e-01
Population         5  1.00  0.00 1.00000 1.9e-06
Price              6 -0.01 -0.36 0.13161 8.7e-01
ShelveLocBad       7  0.04  0.08 0.00756 9.9e-01
ShelveLocMedium    8 -0.04 -0.05 0.00376 1.0e+00
ShelveLocGood      9  0.01 -0.02 0.00067 1.0e+00
Age               10 -0.04  0.05 0.00391 1.0e+00
Education         11 -0.11 -0.06 0.01469 9.9e-01
Urban             12 -0.05  0.02 0.00303 1.0e+00
US                13  0.06  0.07 0.00844 9.9e-01

                 PC1  PC2
SS loadings     1.11 1.21
Proportion Var  0.09 0.09
Cumulative Var  0.09 0.18
Cum. factor Var 0.48 1.00

Mean item complexity =  1.4
Test of the hypothesis that 2 components are sufficient.

The root mean square of the residuals (RMSR) is  31.4

Fit based upon off diagonal values = 1
```

PCA - Code (cont.)

```
> fa.m3
Principal Components Analysis
Call: principal(r = S, nfactors = 3, rotate = "none", covar = TRUE)
Unstandardized loadings (pattern matrix) based upon covariance matrix
                   PC1   PC2   PC3      h2      u2     H2      U2
Sales             0.14  0.70 -0.72 1.0e+00   6.945 0.1292 8.7e-01
CompPrice        -1.47 -4.85 10.25 1.3e+02 104.316 0.5564 4.4e-01
Income           -0.22 26.53  8.90 7.8e+02   0.129 0.9998 1.6e-04
Advertising       1.77  0.32  0.42 3.4e+00  40.816 0.0771 9.2e-01
Population      147.38 -0.03  0.14 2.2e+04   0.022 1.0000 1.0e-06
Price            -0.31 -8.58 21.43 5.3e+02  27.505 0.9509 4.9e-02
ShelveLocBad      0.02  0.03  0.00 1.4e-03   0.181 0.0076 9.9e-01
ShelveLocMedium  -0.02 -0.02 -0.01 1.1e-03   0.247 0.0045 1.0e+00
ShelveLocGood     0.00 -0.01  0.02 3.5e-04   0.167 0.0021 1.0e+00
Age              -0.70  0.73 -2.87 9.3e+00 253.170 0.0354 9.6e-01
Education        -0.28 -0.15 -0.03 1.0e-01   6.766 0.0148 9.9e-01
Urban            -0.02  0.01  0.03 1.5e-03   0.207 0.0071 9.9e-01
US                0.03  0.03  0.04 3.7e-03   0.226 0.0160 9.8e-01

                          PC1    PC2    PC3
SS loadings           21725.80 802.16 652.67
Proportion Var            0.92   0.03   0.03
Cumulative Var            0.92   0.95   0.98
Proportion Explained      0.94   0.03   0.03
Cumulative Proportion     0.94   0.97   1.00

 Standardized loadings (pattern matrix)
                item   PC1   PC2   PC3     h2      u2
Sales              1  0.05  0.25 -0.26 0.1292 8.7e-01
CompPrice          2 -0.10 -0.32  0.67 0.5564 4.4e-01
Income             3 -0.01  0.95  0.32 0.9998 1.6e-04
Advertising        4  0.27  0.05  0.06 0.0771 9.2e-01
Population         5  1.00  0.00  0.00 1.0000 1.0e-06
Price              6 -0.01 -0.36  0.91 0.9509 4.9e-02
ShelveLocBad       7  0.04  0.08  0.00 0.0076 9.9e-01
ShelveLocMedium    8 -0.04 -0.05 -0.03 0.0045 1.0e+00
ShelveLocGood      9  0.01 -0.02  0.04 0.0021 1.0e+00
Age               10 -0.04  0.05 -0.18 0.0354 9.6e-01
Education         11 -0.11 -0.06 -0.01 0.0148 9.9e-01
Urban             12 -0.05  0.02  0.06 0.0071 9.9e-01
US                13  0.06  0.07  0.09 0.0160 9.8e-01

                 PC1  PC2  PC3
SS loadings     1.11 1.21 1.48
Proportion Var  0.09 0.09 0.11
Cumulative Var  0.09 0.18 0.29
Cum. factor Var 0.29 0.61 1.00

Mean item complexity =  1.7
Test of the hypothesis that 3 components are sufficient.

The root mean square of the residuals (RMSR) is  6.99

Fit based upon off diagonal values = 1
```

m=3

Full Computation

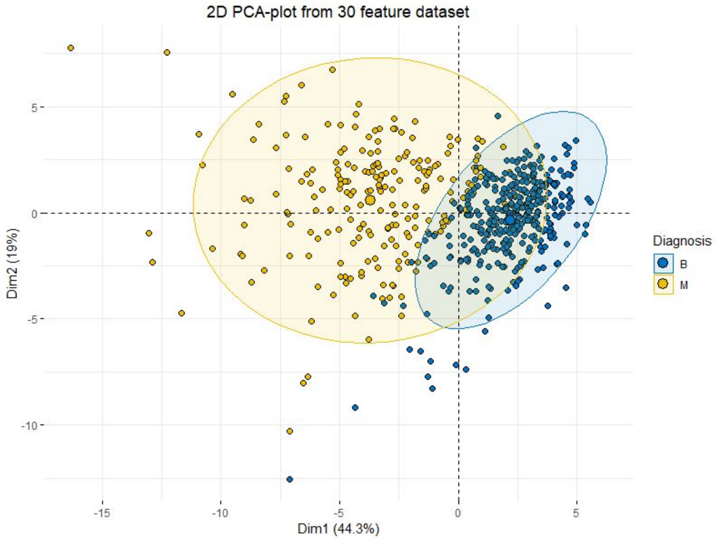
```
> prcarseats$rot
                          PC1           PC2           PC3           PC4           PC5           PC6
Sales           -0.0009729965 -0.0246209184  2.832952e-02 -0.0556537838 -0.1210162795 -0.1429791033
CompPrice        0.0099719488  0.1713933746 -4.012988e-01  0.0008125488 -0.8946542795 -0.0042976509
Income           0.0015231479 -0.9366584414 -3.485329e-01  0.0143633103 -0.0218153034  0.0178152933
Advertising     -0.0120071309 -0.0111477695 -1.658045e-02  0.0072588803  0.0249010361 -0.9880861965
Population      -0.9998613450  0.0009611157 -5.579034e-03  0.0049441911 -0.0085333973  0.0120107237
Price            0.0020878031  0.3030830477 -8.388816e-01  0.1243773303  0.4245825577  0.0090911257
ShelveLocBad    -0.0001158022 -0.0011659882  6.152095e-05 -0.0011587048  0.0006515217  0.0050882689
ShelveLocMedium  0.0001374789  0.0008077928  5.436437e-04  0.0016464650 -0.0005168975  0.0004173695
ShelveLocGood   -0.0000216767  0.0003581954 -6.051646e-04 -0.0004877602 -0.0001346242 -0.0055056384
Age              0.0047303864 -0.0259259896  1.124607e-01  0.9905264908 -0.0591997134 -0.0022054899
Education        0.0018919903  0.0053684516  1.075835e-03  0.0004720819 -0.0011283418  0.0020941451
Urban            0.0001613913 -0.0002874015 -1.137440e-03  0.0010255140 -0.0014520401 -0.0037420401
US              -0.0001978153 -0.0011640979 -1.632168e-03  0.0007347755  0.0009729155 -0.0509926905
                         PC7           PC8           PC9          PC10          PC11          PC12
Sales            0.082260928 -0.9539382617  0.0037427401 -2.347839e-02 -0.0149280527  2.070069e-01
CompPrice       -0.006543610  0.0929158596 -0.0007535787  4.045123e-03  0.0007987454 -1.910480e-02
Income          -0.006387232  0.0117615882  0.0012426618  8.680260e-04 -0.0004257505 -3.502003e-03
Advertising     -0.014789787  0.1378888988  0.0028037176  6.122289e-03 -0.0497831246 -2.671650e-02
Population      -0.001826008 -0.0003484737  0.0001199633 -1.994860e-04  0.0004546691 -5.767842e-05
Price            0.008120590 -0.0901091778  0.0006189756 -1.980068e-03 -0.0018226357  1.995114e-02
ShelveLocBad    -0.009784279  0.1368926810 -0.4798964934 -4.206233e-03 -0.0894506082  6.399082e-01
ShelveLocMedium -0.002207599  0.0224462770  0.8075997530 -6.724984e-02  0.0452952761  8.580370e-02
ShelveLocGood    0.011991878 -0.1593389580 -0.3277032596  7.145607e-02  0.0441553321 -7.257119e-01
Age              0.004275707 -0.0434738389 -0.0018428010  3.706271e-05 -0.0011536613  9.636712e-03
Education       -0.996207483 -0.0855332710 -0.0019803883 -7.137819e-03  0.0122434492  3.240348e-03
Urban            0.006243578  0.0111269238 -0.0768651198 -9.947747e-01  0.0112099372 -6.486867e-02
US               0.011423000  0.0116145892 -0.0644408652  1.078249e-02  0.9924791098  8.857283e-02
                        PC13
Sales            3.244258e-16
CompPrice       -5.847334e-17
Income          -2.194514e-18
Advertising     -8.241602e-17
Population      -1.442777e-18
Price            4.655896e-17
ShelveLocBad    -5.773503e-01
ShelveLocMedium -5.773503e-01
ShelveLocGood   -5.773503e-01
Age             -1.470995e-17
Education        7.038374e-18
Urban            2.558434e-17
US              -4.292010e-17
> summary(prcarseats)
Importance of components:
                             PC1      PC2      PC3     PC4      PC5     PC6     PC7     PC8     PC9
Standard deviation     147.3967 28.32237 25.54742 16.0488 11.41438 6.45261 2.60569 1.91564 0.61042
Proportion of Variance   0.9197  0.03396  0.02763  0.0109  0.00552 0.00176 0.00029 0.00016 0.00002
Cumulative Proportion    0.9197  0.95371  0.98134  0.9922  0.99776 0.99953 0.99981 0.99997 0.99998
                          PC10   PC11   PC12      PC13
Standard deviation     0.45260 0.3401 0.2413 2.499e-16
Proportion of Variance 0.00001 0.0000 0.0000 0.000e+00
Cumulative Proportion  0.99999 1.0000 1.0000 1.000e+00
>
```

PCA - Analysis

- The first principal component nearly explains 92% of the variance - to perform any further analysis with our data we can continue with just using the PC1
- Since PC2 gives us an eigenvalue greater than 1 and explains an additional 3% of variance, we will also use PC2
- PC1 and PC2 explain over 95% of the variance of our data.
  - It is not necessary to include additional principal components past PC1 and 2.

Linear Discriminant Analysis

Why LDA?



(Example Analysis of a Tumor)

- What we're seeing here is a "clear" separation between the two categories of 'Malignant' and 'Benign' on a plot (PCA).

- Using the two dimensions in a PCA plot (like the one shown) could probably get some *pretty good estimates* but higher-dimensional data is difficult to grasp (but also accounts for more variance), thankfully that's what **LDA** is for, *it'll try to find the 'cutoff' or 'decision boundary'* at which we're most successful in our classification.

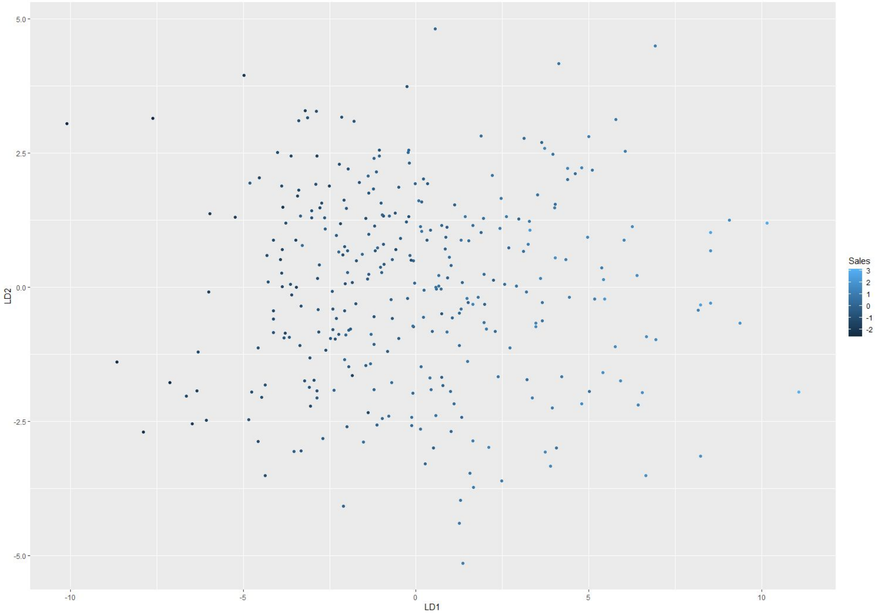
Tackling our Dataset Using LDA

```
library(tidyverse)
library(caret)
library(ISLR)
# Load the data
data("Carseats")
# Split the data into training (80%) and test set (20%)
set.seed(123)
training.samples <- Carseats$Sales %>%
  createDataPartition(p = 0.8, list = FALSE)
train.data <- Carseats[training.samples, ]
test.data <- Carseats[-training.samples, ]
# Normalize the data. Categorical variables are automatically ignored.
# First, estimate preprocessing parameters
preproc.param <- train.data %>%
preProcess(method = c("center", "scale"))
# Transform the data using the estimated parameters
train.transformed <- preproc.param %>% predict(train.data)
test.transformed <- preproc.param %>% predict(test.data)
library(MASS)
# Fit the model
model <- lda(Sales~., data = train.transformed)
# predictions
predictions <- model %>% predict(test.transformed)
predictions

# model output
model
library(ggplot2)
lda.data <- cbind(train.transformed, predict(model)$x)
ggplot(lda.data, aes(LD1, LD2)) +   geom_point(aes(color = Sales))
```

Output

```
Coefficients of linear discriminants:
                        LD1         LD2        LD3          LD4         LD5         LD6          LD7         LD8         LD9        LD10        LD11
CompPrice       1.528425014  0.02253499 -0.8328300  0.718253366  0.54265017 -0.29937468  0.320920468 -0.07314790  0.26094927  0.09383285 -0.42920518
Income          0.030363620 -0.99249284  0.1881183  0.006922891 -0.25026665  0.39057751  0.307639913 -0.45362097  0.01308935  0.36493422 -0.16666812
Advertising     0.829020050 -1.27498757  0.1938255 -0.542875881 -0.41411719 -0.74169131  0.001434326  0.73098014 -0.13260678 -0.01273796 -0.35882178
Population     -0.125247963  0.43131005 -0.8807566  0.230648339 -0.85196167 -0.05826723  0.136424437 -0.29559658  0.14804021  0.02344323  0.22615979
Price          -2.943176524 -0.19046437  0.3319141  0.057870596 -0.56912140  0.65216630 -0.362169554  0.33123135 -0.06459400 -0.15971589 -0.07625481
ShelveLocGood   7.415063413  0.87446462  0.5358860 -0.176594816 -0.71236194  1.97479301  0.142378770  0.79517374  0.48724891 -0.31652879  0.01360740
ShelveLocMedium 2.520409435  0.48586173 -0.3429302 -1.826335671 -0.05312354  1.43079392  0.571430521 -0.22774949  0.38867870 -0.92610577 -0.51722278
Age            -0.563453427  0.74416270  0.6648971  0.193446665 -0.32071214 -0.32820206  0.192246582 -0.03180263  0.48097805  0.09068095 -0.22306135
Education       0.003020899  0.82300529 -0.0958337 -0.036357026 -0.19948314  0.11346377  0.525353454  0.13208847 -0.52133286  0.26794026 -0.11913615
UrbanYes        0.379763893 -0.85682083  0.6414446  1.021285223 -0.07001794  0.05748899  1.333990535  0.00293837 -0.28963040 -1.12072837  0.47003883
USYes           0.575514853  2.69372515  0.1891031  1.057917010  0.17377730  0.57205800 -0.936061897 -1.89148979 -0.61984050 -0.46347830 -0.18590332

Proportion of trace:
   LD1    LD2    LD3    LD4    LD5    LD6    LD7    LD8    LD9   LD10   LD11
0.4895 0.1226 0.0750 0.0694 0.0617 0.0458 0.0372 0.0345 0.0249 0.0213 0.0181
```

- Here in this ggplot we see the percentage separation achieved by each discriminant function in sales.



Judging from the plot there is not a clear or cut separation in variation between different sales that the linear discriminant analysis has displayed which could give us a better understanding of how our predictors interact with our sales (response variable) and most likely do not differ largely

Singular Value Decomposition

Singular Value Decomposition

Singular value decomposition takes a matrix A, and breaks it up into the three separate matrices U, V, and d.

$$A=UdV^T$$

U and V are orthogonal unit vectors and d is a diagonal matrix.

U: columns are the eigenvectors of AA^T

V: columns are the eigenvectors of A^TA

D: the singular values which are the square roots of the non-zero eigenvalues of AA^T and A^TA.

Singular Value Decomposition
svd() function

```
#Read car seat data set from ISLR package
library(ISLR)
cardata=Carseats
#Variables ShelveLoc, Urban, and US must be converted to numeric
cardata$ShelveLoc=as.numeric(cardata$ShelveLoc)
cardata$Urban=as.numeric(cardata$Urban)
cardata$US=as.numeric(cardata$US)


#Use svd package on the fully numeric cardata
library(svd)
carsvd=svd(cardata)
carsvd
#the original data was factored into 3 different matrices
V = carsvd$v
U = carsvd$u
d = diag(carsvd$d)
#these three matrices equal the original data when multiplied (V transposed)
round((U %*% d %*% t(V)),2)
```

svd() output

```
> options(max.print=220)
> V
              [,1]         [,2]          [,3]         [,4]         [,5]         [,6]          [,7]          [,8]          [,9]         [,10]         [,11]
 [1,] -0.020091200  0.027116677 -0.0253754140  0.022885888 -0.158224552  0.145289096 -0.1260435351  0.9543441408 -1.557983e-01  0.0191829204  0.0157674128
 [2,] -0.333095944  0.553829070  0.1959791272  0.095043720 -0.713635452 -0.015794735  0.1031081497 -0.1206988038  9.831132e-03  0.0068877036  0.0032727572
 [3,] -0.183973596  0.309165452 -0.9173681698 -0.156954715  0.058464063 -0.020888689  0.0129726137 -0.0182488235  3.142442e-03  0.0019224711  0.0014263704
 [4,] -0.019565522  0.009030102 -0.0102526060 -0.003527071  0.019829052  0.987513728  0.0292385773 -0.1421189889  1.797956e-02  0.0102070217  0.0485941205
 [5,] -0.858154372 -0.513189287  0.0007392289 -0.006668005  0.002147048 -0.012516072  0.0008118356 -0.0009200655  7.706962e-05  0.0000015337 -0.0002202659
 [6,] -0.310598787  0.527545342  0.3443563585 -0.404990528  0.577994860 -0.006970597 -0.0121539068  0.0901444767 -1.471798e-02  0.0022542223  0.0018930827
 [7,] -0.006119215  0.009754795  0.0020952989  0.008852450 -0.006752466  0.005998445 -0.0428230727  0.1534348474  9.867847e-01 -0.0227243831 -0.0081629933
 [8,] -0.141709832  0.226926540 -0.0255672817  0.894402689  0.354221855 -0.005681999  0.0346716796  0.0305456612 -1.173667e-02  0.0050639059  0.0029749174
 [9,] -0.036804991  0.059332269  0.0061518875  0.040861167 -0.047389125  0.007863581 -0.9843333466 -0.1462452420 -2.107538e-02  0.0179841778  0.0065282903
[10,] -0.004528828  0.007266712 -0.0001367591  0.004465651 -0.004869968  0.004478693 -0.0167478392  0.0091786661 -2.373835e-02 -0.9864159674  0.1609914304
[11,] -0.004423941  0.006477814 -0.0010032054  0.003130816 -0.002713322  0.051638192 -0.0090337629  0.0073587631 -1.382289e-02 -0.1599677536 -0.9855644467
> U
             [,1]          [,2]          [,3]         [,4]          [,5]          [,6]          [,7]          [,8]         [,9]         [,10]         [,11]
 [1,] -0.04872261  1.872939e-02  4.963999e-04 -0.028389413 -0.0438138224  0.0330296442 -0.0388511324 -0.0236356473 -0.086523809 -0.0149189307 -0.0020918124
 [2,] -0.04361818  1.488618e-03  7.861833e-03  0.072760396 -0.0259581380  0.0827563699  0.0437470473  0.0570403683 -0.031023139 -0.0430611051  0.0111080049
 [3,] -0.04421267 -4.694918e-03  2.828306e-02  0.067544635 -0.0490682232  0.0370995824  0.0072327419  0.0354045859  0.038636622 -0.0556287373 -0.0369270705
 [4,] -0.07096333 -4.662403e-02 -6.471991e-02  0.008238591 -0.0110177129 -0.0425785475 -0.0052095036 -0.0254673718  0.058684403 -0.0456064327 -0.0702746292
 [5,] -0.05670184  2.862202e-04  2.161207e-02 -0.044367004 -0.0378782521 -0.0402587355  0.0351984332 -0.1003821923 -0.041204784 -0.0249875629  0.0641774156
 [6,] -0.07533507 -5.715716e-02 -9.989644e-02  0.088557334 -0.0525581549  0.0241341604 -0.0054255232 -0.0581573782 -0.076119441  0.0983482038 -0.0121584288
 [7,] -0.02011179  8.729470e-02 -6.797926e-02  0.040489283  0.0379100659 -0.0354223748 -0.0230215970  0.0172701588  0.037045816 -0.0219372791  0.0625966712
 [8,] -0.06762164 -2.260565e-02 -1.435803e-02  0.025764577 -0.0015021571  0.0493089350  0.0884044668  0.0663110824 -0.051200853 -0.0079542578  0.0293248677
 [9,] -0.02957423  8.012159e-02 -6.065083e-02  0.035515599  0.0362746879 -0.0459519459  0.0914962690  0.0180280401  0.039343420  0.0928204067  0.0484308855
[10,] -0.03250942  7.389501e-02 -6.532929e-02  0.034477249  0.0369722766 -0.0499259838 -0.0198391039 -0.0539073960  0.047996444  0.0860634959 -0.0875972547
[11,] -0.03131542  4.371871e-02 -2.513986e-02 -0.050580110 -0.0617225540  0.0319883237  0.0421567387 -0.0081276555 -0.053142432  0.0483364706 -0.0606949493
[12,] -0.07512069 -6.055749e-02 -5.675945e-02  0.001340115 -0.0279524392 -0.0397861054 -0.0006994895  0.0751411714 -0.040915493 -0.0404507037 -0.0651870608
[13,] -0.06240747 -2.147935e-02  6.600803e-02  0.012968912  0.0568572555 -0.0473171451 -0.0755282854 -0.0051453450  0.034212720 -0.0266793511  0.0552515493
[14,] -0.01483651  6.898174e-02  4.394344e-02  0.054740571 -0.0544389843  0.0702028946 -0.0990344830  0.0391963639 -0.041215190 -0.0406669202 -0.0154599847
[15,] -0.03281177  5.634140e-02 -8.350882e-02 -0.026069869  0.0571289634  0.0437757255 -0.0994347688  0.0967498768 -0.058872443 -0.0208213535  0.0035408831
[16,] -0.06679387  7.236570e-04 -1.797337e-02  0.019784176  0.0307670589 -0.0338084504 -0.0171631951  0.0166801772  0.003707003  0.1320623240  0.0957264741
[17,] -0.04759671  1.685309e-03  5.328699e-02  0.046308583  0.0088628585 -0.0460120923 -0.0037078074  0.0608617729 -0.038762310 -0.0378677101  0.0405928388
[18,] -0.04677749  3.407249e-02  7.971645e-03 -0.014264718 -0.0308362462  0.0512293618  0.0906690052  0.0712875068 -0.052305251 -0.0080703048  0.0223527243
[19,] -0.06232198 -3.954633e-02 -1.008808e-01  0.014069493 -0.0709786072 -0.0589372411 -0.0798356456  0.0753844555 -0.042954372  0.0611151519 -0.1127060405
[20,] -0.02218469  8.643548e-02 -8.487675e-03  0.036676324  0.0198903635  0.0907519552  0.0443263497  0.0201541187  0.030123852 -0.0166866721  0.0381518588
 [ reached getOption("max.print") -- omitted 380 rows ]
> d
          [,1]     [,2]     [,3]     [,4]     [,5]     [,6]     [,7]     [,8]   [,9]   [,10]    [,11]
 [1,] 6989.104    0.000   0.0000   0.0000   0.0000   0.0000  0.00000  0.00000  0.000 0.00000 0.000000
 [2,]    0.000 1686.997   0.0000   0.0000   0.0000   0.0000  0.00000  0.00000  0.000 0.00000 0.000000
 [3,]    0.000    0.000 565.5921   0.0000   0.0000   0.0000  0.00000  0.00000  0.000 0.00000 0.000000
 [4,]    0.000    0.000   0.0000 363.4692   0.0000   0.0000  0.00000  0.00000  0.000 0.00000 0.000000
 [5,]    0.000    0.000   0.0000   0.0000 262.4075   0.0000  0.00000  0.00000  0.000 0.00000 0.000000
 [6,]    0.000    0.000   0.0000   0.0000   0.0000 128.9966  0.00000  0.00000  0.000 0.00000 0.000000
 [7,]    0.000    0.000   0.0000   0.0000   0.0000   0.0000 59.35756  0.00000  0.000 0.00000 0.000000
 [8,]    0.000    0.000   0.0000   0.0000   0.0000   0.0000  0.00000 39.72733  0.000 0.00000 0.000000
 [9,]    0.000    0.000   0.0000   0.0000   0.0000   0.0000  0.00000  0.00000 15.957 0.00000 0.000000
[10,]    0.000    0.000   0.0000   0.0000   0.0000   0.0000  0.00000  0.00000  0.000 9.43689 0.000000
[11,]    0.000    0.000   0.0000   0.0000   0.0000   0.0000  0.00000  0.00000  0.000 0.00000 7.129577
> 
```

U * d * V^T compared to data set

```
> options(max.print=110)
> cardata
   Sales CompPrice Income Advertising Population Price ShelveLoc Age Education Urban US
1   9.50       138     73          11        276   120         1  42        17     2  2
2  11.22       111     48          16        260    83         2  65        10     2  2
3  10.06       113     35          10        269    80         3  59        12     2  2
4   7.40       117    100           4        466    97         3  55        14     2  2
5   4.15       141     64           3        340   128         1  38        13     2  1
6  10.81       124    113          13        501    72         1  78        16     1  2
7   6.63       115    105           0         45   108         3  71        15     2  1
8  11.85       136     81          15        425   120         2  67        10     2  2
9   6.54       132    110           0        108   124         3  76        10     1  1
10  4.69       132    113           0        131   124         3  76        17     1  2
 [ reached 'max' / getOption("max.print") -- omitted 390 rows ]
> round((U %*% d %*% t(V)),2)
       [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
  [1,]  9.50  138   73   11  276  120    1   42   17     2     2
  [2,] 11.22  111   48   16  260   83    2   65   10     2     2
  [3,] 10.06  113   35   10  269   80    3   59   12     2     2
  [4,]  7.40  117  100    4  466   97    3   55   14     2     2
  [5,]  4.15  141   64    3  340  128    1   38   13     2     1
  [6,] 10.81  124  113   13  501   72    1   78   16     1     2
  [7,]  6.63  115  105    0   45  108    3   71   15     2     1
  [8,] 11.85  136   81   15  425  120    2   67   10     2     2
  [9,]  6.54  132  110    0  108  124    3   76   10     1     1
 [10,]  4.69  132  113    0  131  124    3   76   17     1     2
 [ reached getOption("max.print") -- omitted 390 rows ]
> 
```

The data is the same as the multiplication of SVD components.

Manual SVD

```
#performing svd manually where we call the original data X
#step 1: compute XTX and XXT
X = as.matrix(cardata)
XTX = t(X) %*% X
XXT = X %*% t(X)
#step 2: compute V (there will be some sign changes when compared to the V from the svd() fuction)
manV = eigen(XTX)$vectors
manV[,2] = manV[,2]*-1
manV[,6] = manV[,6]*-1

#step 3: compute U (remove all collums with value 0 and change some signs)
manU = eigen(XXT)$vectors
manU = manU[,1:11]
manU[,5:7] = manU[,5:7]*-1
manU[,9:10] = manU[,9:10]*-1

#step 4: compute d
mand = sqrt(eigen(XTX)$values)
mand = mand * diag(length(mand))

#step 5: multiply the SVD components (V transposed) to show it equals the original data
round((manU %*% mand %*% t(manV)),2)
```

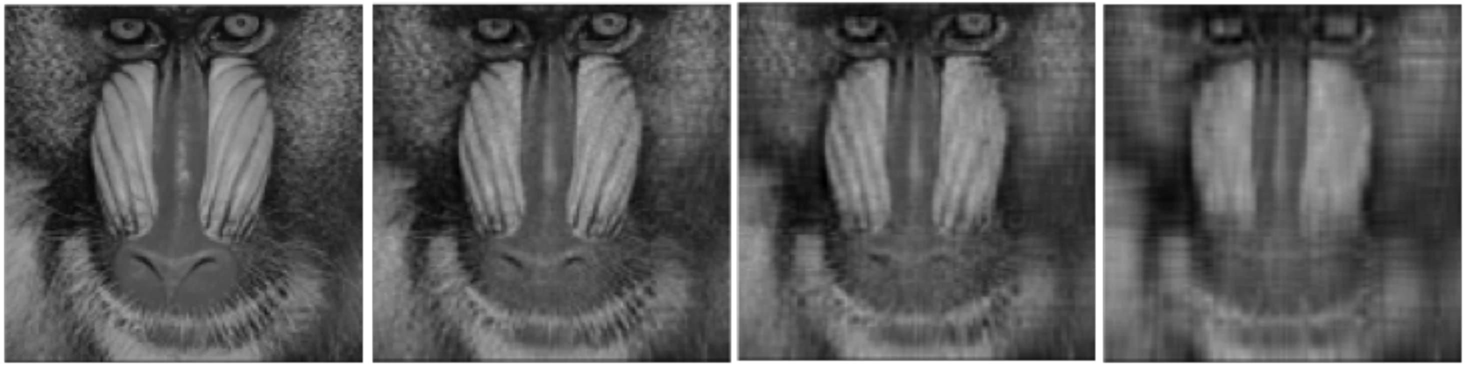
The results are the exact same as if the svd() function was used but some extra data manipulation was required.

Singular Value Decomposition Applications

SVD is used to reduce the dimensions of a matrix to show a simpler representation of the meaningful data by creating 3 simpler matrices.

It is used to compress images and by companies to recommend products to their consumers based on rating data given by previous consumers of said products.

SVD Image Compression



As a matrix’s rank keeps getting reduced by SVD (decreasing the amount of singular values d), the quality of the image will decrease. However, less data is stored meaning less memory is being used.

SVD in Recommendation Systems

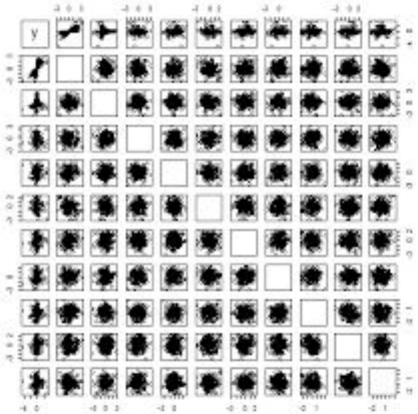
An example would be movie ratings. Suppose you had a matrix of movie ratings given to movies by viewers. If you were to perform SVD on the matrix, the U matrix would represent much how each viewer liked the genre of the movie, V would represent the strength of the genre each movie has(how well it fits into a genre), and d would represent the the strength that each genre has(which has more information). We can see how each person rates movies based on what the genre is so it narrows down what kind of genres of movies they enjoy watching.

(example was found in an honors thesis linked in sources)

Sliced Inverse Regression

Sliced Inverse Regression (SIV)

Sliced Inverse Regression (SIV) is the process of using an inverse regression curve to reduce the dimensionality of a regression. SIV uses this curve to perform a weighted PCA, which gives insight as to how to reduce dimensions for the dataset. The k-value is equivalent to the number of dimensions we are looking to have for the regression. If this k value results in a low p-value through SIV testing, then we can reduce the dimensions of the regression to k dimensions.



SIV in R

The relevant package needed to perform SIV in R is ICtest, and is called using the code: library(ICtest) once it is installed.

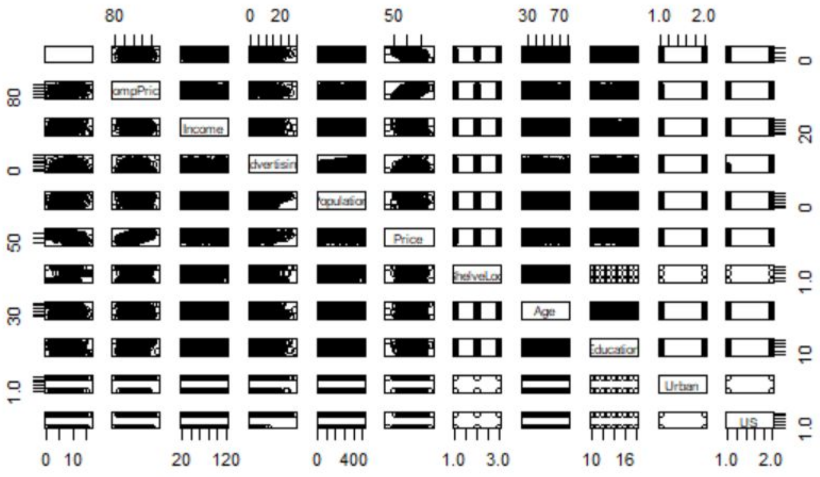
Two methods are available to perform SIV in R: SIRasymp(X,Y,k) and SIRboot(X,Y,k). X is the set of predictor variables, Y is the response variable, and k is the number of dimensions we are looking to have for the regression. The next couple of slides show an example of SIV using the Carseats data set, and utilizes both methods, which come to similar conclusions.

SIV R Code

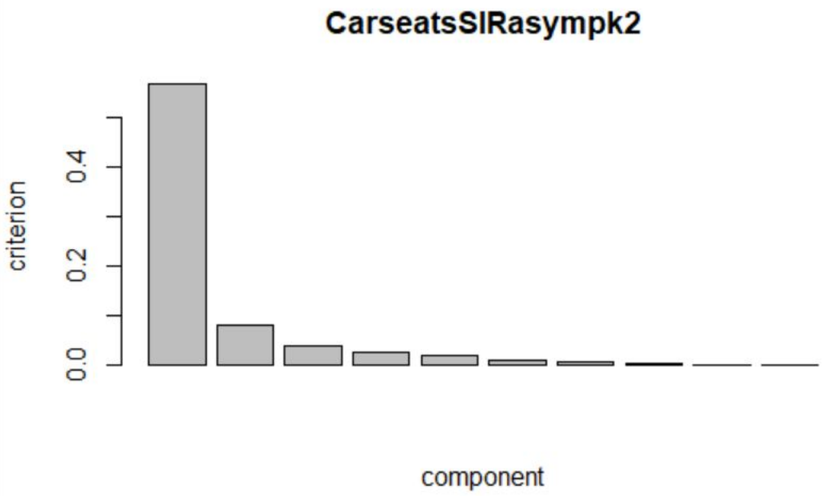
```
head(Carseats)
Carseats$ShelveLoc = as.numeric(Carseats$ShelveLoc)
Carseats$Urban = as.numeric(Carseats$Urban)
Carseats$US = as.numeric(Carseats$US)
X <- Carseats[,2:11]
X = as.matrix(X)
Y <- Carseats[,1]
Y = as.matrix(Y)
pairs(cbind(Y,X))
CarseatsSIRasympk2 <- SIRasymp(X,Y,3)
screeplot(CarseatsSIRasympk2)
CarseatsSIRasympk2
CarseatsSIRbootk2 <- SIRboot(X,Y,3)
CarseatsSIRbootk2
```

SIV R Output (Graphs)

`pairs(cbind(Y,X))`



`screeplot(CarseatsSIRasympk2)`



SIV R Output (Results)

```
> CarseatsSIRasympk2

        SIR test for subspace dimension

data:  X
T = 24.317, df = 42, p-value = 0.9868
alternative hypothesis: the last 7 eigenvalues are not zero

> CarseatsSIRbootk2

        SIR bootstrapping test for subspace dimension

data:  X
T = 0.060793, replications = 200, p-value = 0.9453
alternative hypothesis: the last 7 eigenvalues are not zero
```

SIV Analysis

Based on the above p-values, we fail to reject the null hypotheses. What this means is that the k-value we entered into the SIVasymp() and SIVboot() functions (which was 3) is not equal to the number of significant predictors, and thus we should not reduce dimensionality to 1 response variable and 3 predictor variables. With the Carseats data set, SIV fails to locate a possible decrease in dimensionality after testing all possible values of k (0-10).

Discussion

Discussion

From what we have gathered:

- PCA with m=2 explains over 95% of the variance in the Sales data of the Carseats data set with RMSE = 31.4.
- Fisher's linear discriminant finds out a linear combination of features that can be used to discriminate between the target variable classes. In Fisher's LDA, we take the separation by the ratio of the variance between the classes to the variance within the classes.
- SVD can simplify data into 3 less complex matrices that are simpler to analyze.
- SIR can aid in reducing the dimensionality of a regression, although it failed to do so for Carseats.

Discussion

Overall, some methods work better than others depending on what data set you are working with. SIR did not succeed in reducing dimensionality of the Carseats data set, while the PCA showed that we can explain a majority of the variance in the Sales data using only 2 principal components. Based on these results, it appears that some methods are more aptly built to reduce dimensionality in certain data set as opposed to others. Each method that can be applied has certain limitations and conditions where it can be either more effective or less effective at reducing dimensionality than other methods, and it depends on the data that you are working with. Depending on the data you are studying, some methods work better than others, but all of them work to achieve the same goal.

Contributions

Contributions

Dominick did the segment on PCA

Anthony did the segment on LDA

Luke did the segment on SVD

Montgomery did the segment on ISVR

Everyone contributed the completion of the the presentation.

3 Questions

3 Questions

1. True or False? Principal components are eigenvectors of a data sets covariance matrix.
2. How many matrices are created as a result of SVD?

   a. 1 b. 2

   c. 3 d. 4
3. What is the purpose of performing dimension reduction on a dataset?

Answers

1. True

2. c. 3

3. Dimension reduction of a data set creates a representation of the data that is simpler to visualize and interpret, as well as remove collinearity, and reduce complexity.

Literature Cited

Literature Cited

https://cran.r-project.org/web/packages/ICtest/vignettes/SIR.html

https://towardsdatascience.com/dimensionality-reduction-approaches-8547c4c44334#:~:text=Advantages%20of%20dimensionality%20reduction,Reduce%20space%20complexity.

http://www.math.ucsd.edu/_files/undergraduate/honors-program/honors-program-presentations/2017-2018/Zecheng_Kuang_Honors_Thesis.pdf

https://rpubs.com/aaronsc32/singular-value-decomposition-r

https://support.minitab.com/en-us/minitab/18/help-and-how-to/modeling-statistics/multivariate/how-to/principal-components/interpret-the-results/key-results/

https://builtin.com/data-science/step-step-explanation-principal-component-analysis