Principal Component Analysis

D2K Course Staff

2/21/2019

Introduction

Often when we are working with big data sets, we may have significantly more information than we need. Whether that is redundant information or just uninformative data, it can be helpful for us to reduce the dimensions of the original data set. This can provide many advantages such as visualization, increased density of important information in your variables, and in some cases increased predictive ability when performing regression or classification tasks.

There are several methods that allow us to reduce the dimension of our original dataset. In this module we cover one of the most popular and widely used methods, Principal Component Analysis (PCA).

```
# Load libraries
library(dplyr)
#library(devtools)
#install_github("vqv/ggbiplot") # install from source if necessary
library(ggbiplot)
```

Principal Component Analysis (PCA)

PCA is one of the most widely used dimension reduction techniques for exploratory analysis, visualization, and pattern recognition. The objective of PCA is to find the low-dimensional representation of the data that captures the most variance in the data (i.e. the best low-dimensional representation of the data). In practice, this means determining the linear combinations of our features that preserve the most variance, or in other words, information gained. We often have this objective given a situation where our data has large p (recall our data is defined as a matrix $X_{n \times p}$ with n observations and p features) or when we wish to understand the underlying patterns in the data.

We calculate the *first* principal component as the linear combination of variables that maximizes the variance. Mathematically, we can describe is as the following:

$$\underset{v}{\text{maximize}} \ \text{Var}(Xv) \text{ subject to } ||v||_2 = 1$$

$$\underset{v}{\text{maximize}} \ v^T \text{Var}(X) v \text{ subject to } ||v||_2 = 1$$

$$\underset{v}{\text{maximize}} \ v^T \Sigma v \text{ subject to } ||v||_2 = 1$$

where $\Sigma = Cov(X)$.

We then calculate the next k principal components by finding the subsequent linear combinations that are orthogonal to the previous combinations. Mathematically, we describe it as:

$$\underset{v_k}{\text{maximize}} \ v_k^T \Sigma v_k \text{ subject to } ||v_k||_2 = 1 \ \& \ v_k^T v_j = 0 \ \forall j < k$$

We can arrive at the solution for the above optimization problem either through eigenvalue decomposition of the covariance matrix or singular value decomposition of the data matrix. We show the implementation of

this solution in R below on the *mtcars* data set. Since PCA is used for continuous data, we will only select the numeric variables for our analyses.

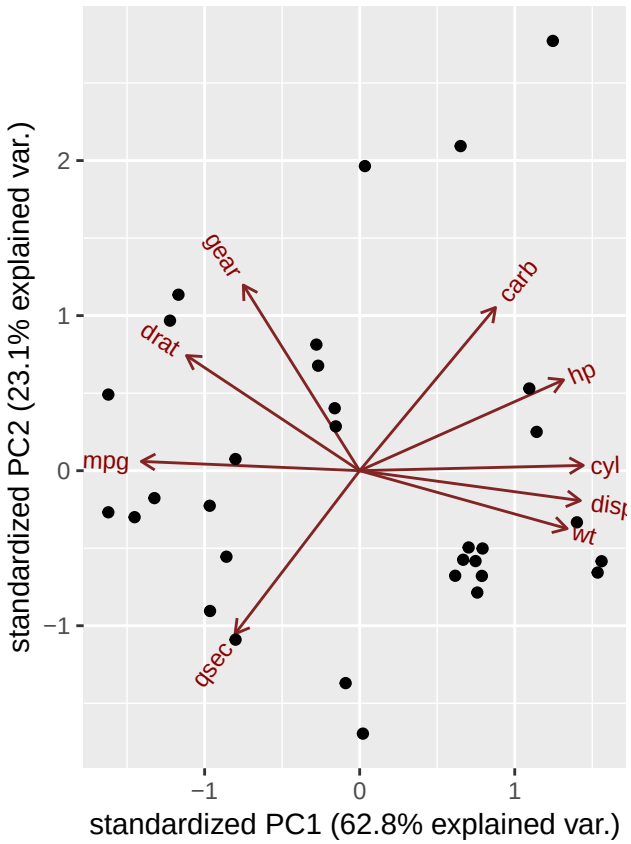
```
# Load data
data(mtcars)
head(mtcars)
```

```
##                   mpg cyl disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
## Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
```

```
# Select numeric variables
mtcars_x <- mtcars %>%
  select("mpg", "cyl", "disp", "hp", "drat", "wt", "qsec", "gear", "carb")
```

Note: one should always center features before performing PCA. It is also a good idea to scale features if they are measured differently. **Don't scale if features are measured in the same way!**

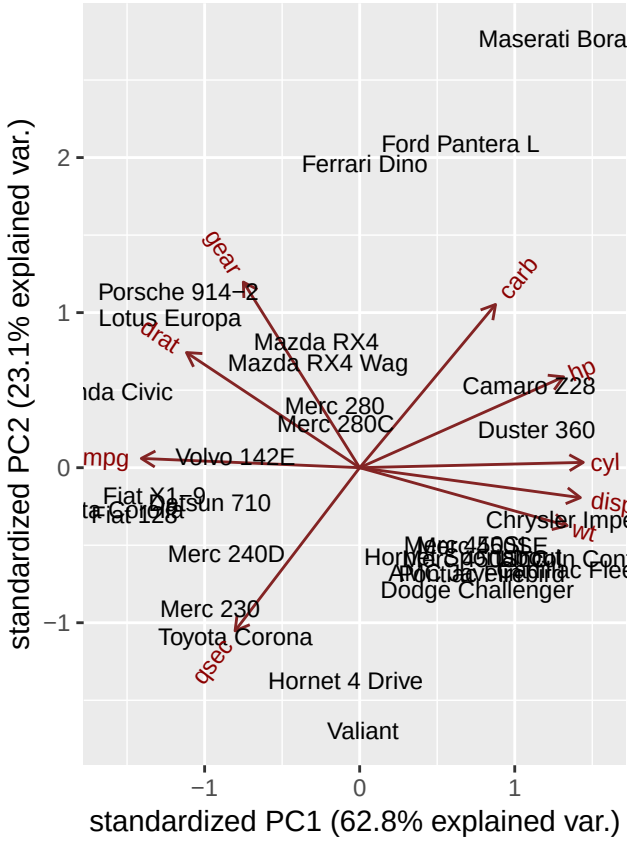
```
# Set seed for reproducibility
set.seed(123)

# Center and scale data
mtcars_x <- scale(mtcars_x, center = TRUE, scale = TRUE)

# Perform PCA with built-in function
mtcars_pca <- prcomp(mtcars_x)

# Visualize 1st and 2nd principal components
par(mfrow=c(1,1))
ggbiplot(mtcars_pca)
```



The above visualization shows how each variable contributes to the first and second principal components of the data. You can see that *cyl*, *disp*, and *wt* contribute significantly to the first principal component in the positive direction, and that *mpg* contributes significantly to the first principal component in the negative direction.

We can come to understand the underlying structure of the data more when we add labels to the data points. We replace each data point with the name of the car model and again visualize the first and second principal components.

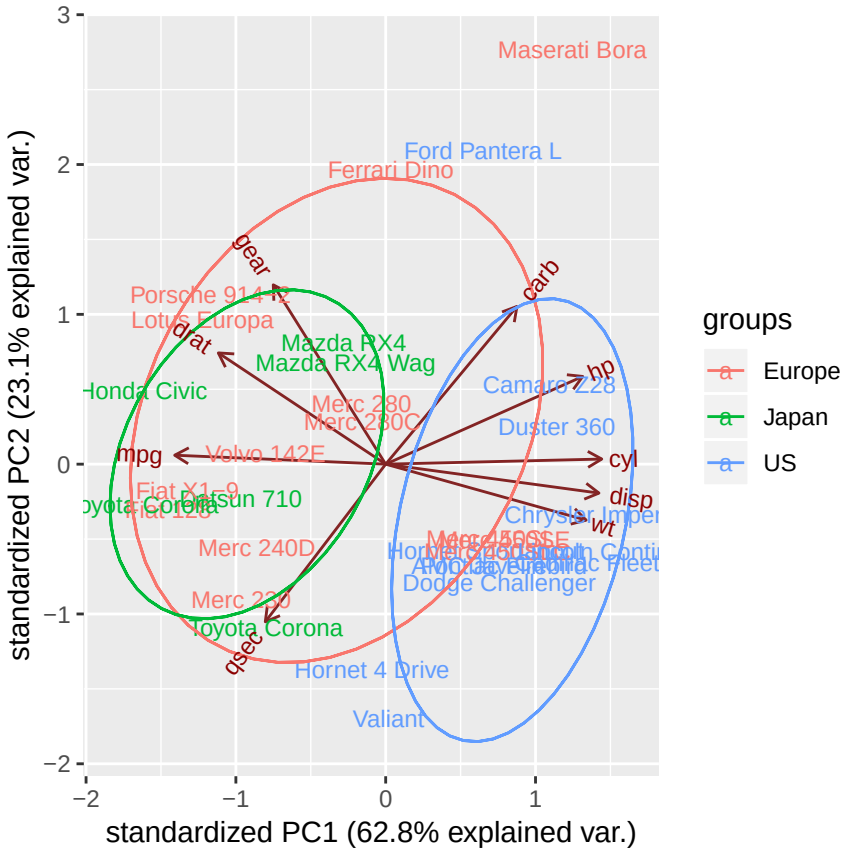
```
# Visualize 1st and 2nd principal components with labels
ggbiplot(mtcars_pca, labels = rownames(mtcars_x))
```



You can see patterns start to arise, like the Maserati Bora, Ford Pantera L, and Ferrari Dino all clustered together. This makes sense all of these models are sports cars that have high values of the features *carb* and *gear*.

Additionally, we can use group labels to understand underlying patterns in the data. We look to see if any clusters arise based on the place of manufacturing of the different model cars.

```
# Define place of manufacturing
mtcars_country <- c(rep("Japan", 3), rep("US",4), rep("Europe", 7),rep("US",3), "Europe", rep("Japan", 3

# Visualize 1st and 2nd principal components
ggbiplot(mtcars_pca, ellipse = TRUE, labels = rownames(mtcars_x), groups = mtcars_country)
```



Very interesting! We see some clear clusters arise based on the place of manufacturing of the model vehicle. This shows us that cars manufactured in the same places have similar characteristics.

As previously stated, PCA allows us to accomplish many different objectives that are common during the exploratory analysis phase of a data science project. It allows us visualize multivariate data sets in low-dimensional space easily, it allows us to generate new features that can increase our predictive ability, and it allows us to determine underlying patterns in our data. If you are working with numeric variables, PCA should be one of the first techniques you try in the exploratory phase.

Advantages

- Unique and global solution
- Ordered and orthognal components
- Best low-rank approximation of the data
- Yields best linear dimension reduction possible
- **Visualization!!!**

Disadvantages

- Computationally expensive for large square matrices
- Doesn't identify non-linear patterns
- Doesn't perform well in ultra high-dimensional spaces

Variance Explained

You may have noticed before on the plot above that next to each axis label is the "% explained variance". As explained earlier, PCA finds the linear combinations of the features that maximizes the variance. Thus, each principal component can be thought of as "explaining" a certain percentage of the overall variance in the data. We can make the following calculations:

Variance explained by the kth principal component:

$$d_k^2 = v_k^T X^T X v_k$$

Total variance of the data:

$$\Sigma_{k=1}^{n} d_k^2$$

Proportion of variance explained by kth principal component:

$$\frac{d_k^2}{\Sigma_{k=1}^{n} d_k^2}$$

Cumulative variance explained by first r principal components:

$$\frac{\Sigma_{k=1}^{r} d_k^2}{\Sigma_{k=1}^{n} d_k^2}$$

References

Hastie, Trevor, et al. The Elements of Statistical Learning Data Mining, Inference, and Prediction. Springer, 2009.

"PCA Analysis in R." DataCamp Community, www.datacamp.com/community/tutorials/pca-analysis-r.