Data Science for Public Policy

Aaron R. Williams and Alena Stern - Georgetown University

Unsupervised Machine Learning – Dimension Reduction

Reading

- Hands on Machine Learning with R Chapter 17
- Reenvisioning Rural America

Unsupervised Learning: A process of summarizing data without a "target", "response", or "outcome" variable.

Dimension reduction: Reducing the number of variables in a data set while maintaining the statistical properties of the data.

Clustering: Grouping observations into homogeneous groups.

Dimension Reduction

Background

Problems:

1. We want to generally understand large data sets with many variables
2. We want to visualize the relationships between more than two or three variables
3. We want to estimate a model but we have many correlated predictors

Solution

Dimension reduction: Reducing the number of variables in a data set while maintaining the statistical properties of the data.

1. **Variable/Predictor/Feature selection:** Process of selecting a subset of variables/predictors/features that contribute the most to a model, statistic, or data visualization
2. **Variable/Predictor/Feature extraction:** Process of combining variables/predictors/features into a reduced subset of *new* informative and non-redundant variables/predictors/features

What does it mean for a variable to be informative and non-redundant?

Sample variance: A measure of data dispersion. Average of squared deviations from the mean.

Sample covariance: A measure of the strength and direction of a linear relationship between two variables.

Covariance matrix (variance-covariance matrix): A pxp symmetric matrix with variances on the main diagonal and covariances off of the main diagonal:

$$\begin{bmatrix} s_1^2 & s_{12} & s_{13} \\ s_{21} & s_2^2 & s_{23} \\ s_{31} & s_{32} & s_3^2 \end{bmatrix}$$

Total sample variance: The total amount of dispersion within a data set. The sum of the diagonal terms in the covariance matrix (trace).

Orthogonal: Perpendicular. Orthogonal variables are linearly uncorrelated.

We can think of a data set as having a total amount of variance. If variables are identical or highly correlated, then it is easy to capture all or most of the total variance with a subset or linear combination of the variables. We want these new variables to be orthogonal. 因为要uncorrelated new variables, 所以要求orthogonal

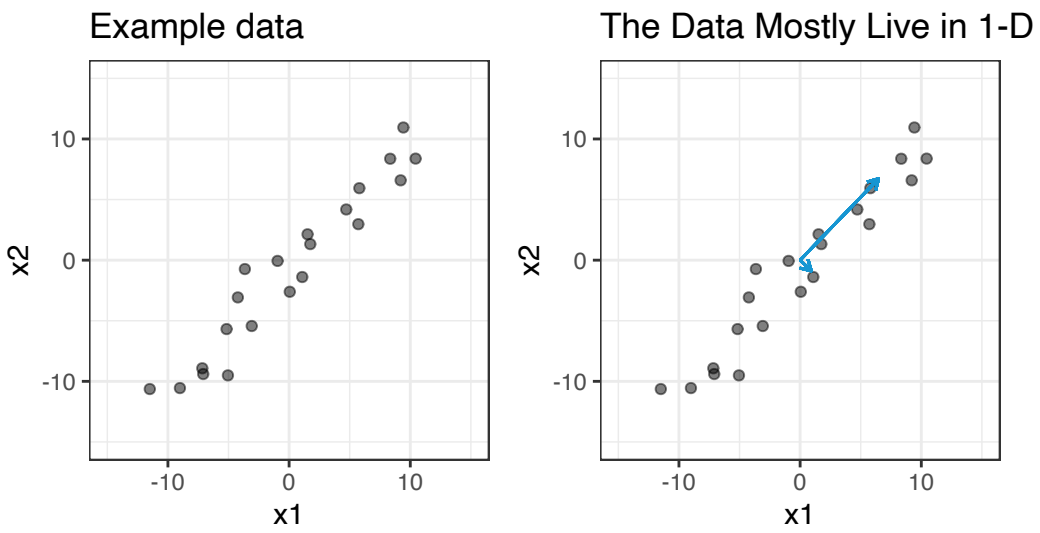
Principal Component Analysis 主成分分析，其实是生成新的、互不相关的variable哦

Principal Component Analysis: A process that transforms many possibly linearly correlated variables into a set of linearly uncorrelated variables called principal components. The components are ordered such that the first component explains the most variance in the original data, the second explains the second most variance in the original data, and so on. A subset of principal components can often be used to represent the original data with limited information loss.

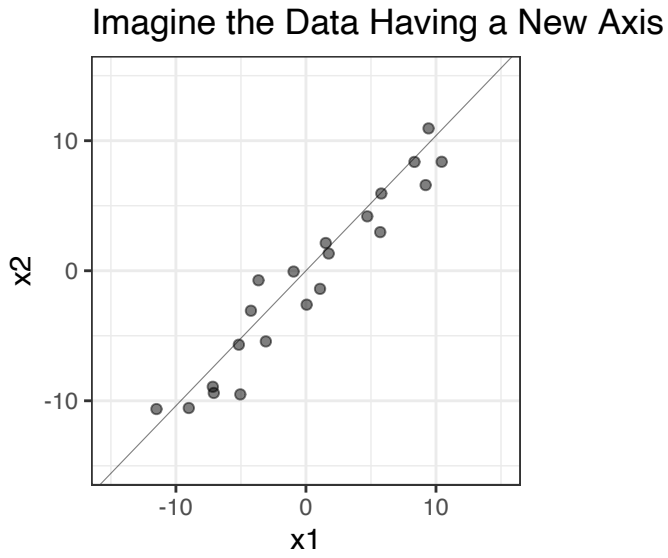
Visual Explanation

Assume there are two variables x_1 and x_2. x_1 has mean zero and x_2 has mean zero.

x_1 and x_2 are highly correlated. In fact, the data are mostly dispersed in one direction, from southwest to northeast, because the data are so highly correlated.



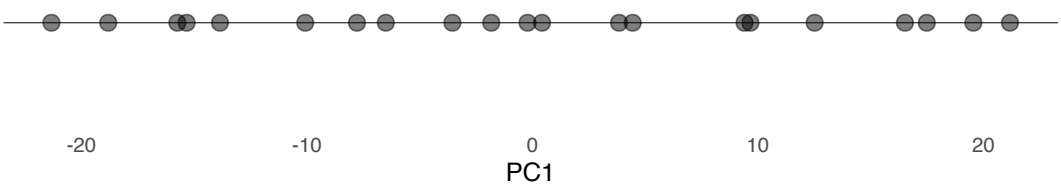
Imagine adding a new axis through the main pattern in the data. This axis should minimize the sum of squared distances from each point to the line.



If we project the data onto the main diagonal, then we can dramatically simplify the data without losing much of the variation. In this case, that means moving the points in 2-D to their closest points on the diagonal and then putting them into 1-D.

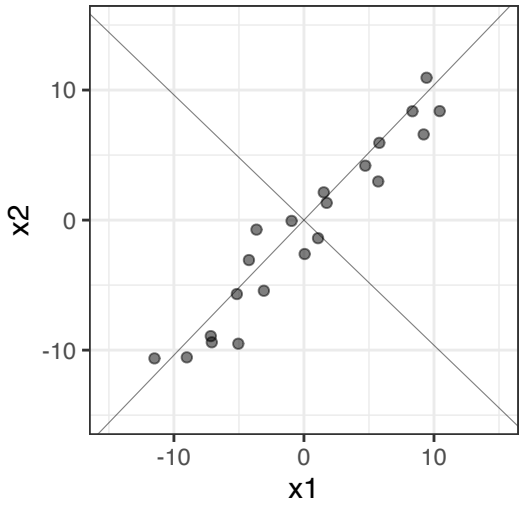
This plot represents the first principal component and it captures most of the variation in the data.

Project the Data On To the New Axis



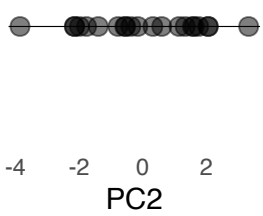
Imagine adding a new axis, that is orthogonal to the first PC, through the second pattern in the data. This axis should minimize the sum of squared distances from each point to the line.

Imagine the Data Having a New Axis



If we project the data onto the second diagonal, then we can further simplify the data–though less than with the first project. Again, this means moving the points in 2-D to their closest points on the diagonal and then putting them into 1-D.

A 2nd Projection 投影



Mathematical Explanation

Principal components are weighted combinations of variables in the original data set.

$$PC_1 = \phi_{11}x_1 + \phi_{21}x_2 \text{ where } \phi_{11}^2 + \phi_{21}^2 = 1$$

ϕ_{11} and ϕ_{21} are called loadings and they come from the dominant eigenvector from an eigenvector decomposition of the covariance matrix of the original data set. The eigenvectors specify the orientation of the principal components relative to the original variables. Each eigenvector has an eigenvalue, or the variance explained by the corresponding principal component. The eigenvalues must decrease from the first principal component (corresponding to the dominant eigenvector) to the last.

$$PC_2 = \phi_{12}x_1 + \phi_{22}x_2 \text{ where } \phi_{12}^2 + \phi_{22}^2 = 1$$

ϕ_{12} and ϕ_{22} are called loadings and they come from the second dominant eigenvector from an eigenvector decomposition of the covariance matrix of the original data set.

Take a simple example with 10 observations of `body_weight` and `height`. All observations are centered and scaled.

body_weight	height
-1.34	-1.49
-1.16	-1.02
-0.77	-0.56
-0.51	0.37
-0.22	-0.09
0.02	-0.56
0.28	0.37
0.80	1.30
1.32	-0.09
1.58	1.77

We want to generate weights to combine `body_weight` and `height` into an entirely new variable called `PC1`.

```
cov_bmi_ex <- cov(bmi_ex)
cov_bmi_ex
```

```
##             body_weight   height
## body_weight    1.000000 0.805714
## height         0.805714 1.000000
```

特征

```
eigen_bmi_ex <- eigen(cov_bmi_ex)
eigen_bmi_ex
```

```
## eigen() decomposition
## $values
## [1] 1.805714 0.194286
##
## $vectors
##           [,1]       [,2]
## [1,] 0.7071068 -0.7071068
## [2,] 0.7071068  0.7071068
```

eigenvalue：其实没用上？？

eigenvector, 是相对于本来的vector而言的
默认还会再生成一个垂直的PC2?

```
pca_bmi_ex <- bmi_ex %>%
  mutate(PC1 = body_weight * 0.7071068 +
           height * 0.7071068) %>%
  mutate(PC2 = body_weight * -0.7071068 +
           height * 0.7071068)

knitr::kable(pca_bmi_ex, digits = 2)
```

body_weight	height	PC1	PC2
-1.34	-1.49	-2.00	-0.10
-1.16	-1.02	-1.54	0.10
-0.77	-0.56	-0.94	0.15
-0.51	0.37	-0.09	0.62
-0.22	-0.09	-0.22	0.09
0.02	-0.56	-0.38	-0.41
0.28	0.37	0.46	0.07
0.80	1.30	1.49	0.36
1.32	-0.09	0.87	-1.00
1.58	1.77	2.37	0.13

The principal components aren't necessarily interpretable but they capture all of the variance in the data and are orthogonal (linearly uncorrelated). However, in this example, `PC1` is essentially the "latent" variable size and `PC2` is essentially body-mass index.

Variance Explained

So how much variance does a principal component explain? How much variance do the first three principal components explain?

It depends on the nature of the data. If variables in the data set have strong linear relationships, then the first few principal components can explain much of the variation.

Proportion of variance explained (PVE): The proportion of variation captured by the k^{th} principal component

Cumulative variance explained (PVE): The proportion of variation captured by the 1^{st} through k^{th} principal components

Take the body weight and height example from above. `PC1` captures more than 90 percent of variation in the original data. This is excellent. We took a two variable data set and captured 90 percent of its variation with one variable!

```
summary(prcomp(bmi_ex))
```

```
## Importance of components:
##                             PC1     PC2
## Standard deviation     1.3438 0.44078
## Proportion of Variance 0.9029 0.09714
## Cumulative Proportion  0.9029 1.00000
```

2nd Interpretation

The principal components create the p-dimensional space that is closest to the observations in the data.

The line in the visiual explanation above is the line that is closest to all of the data. Two principal components in a data set with 50 variables create a plane that is closest to all observations in the data in 50 dimensions.

Implementation

PCA is implemented in R with `prcomp()`.

PCA requires numeric data. Categorical variables need to be recoded with techniques like dummy variable encoding. Mixing continuous variables and dummy variables in the same PCA is possible but can complicate interpretation.

The scale of variables matters. If one variable has a range of $[0, 1000000]$ and another has a range of $[0, 0.1]$, then the first variable will dominate the second variable. Sometimes this makes sense if the units are the same or the ranges matter. Other times it does not make sense.

PCA always mean centers data. In many cases, variables are standardized by dividing by their standard deviations or are min-max scaled (more below).

Tutorial 1 (continuous variables)

`state_data` includes numeric variables about state household population (`hhpop`), state home ownership rate (`horate`), and state median household income (`medhhincome`). The objective is to plot the three variables in two dimensions.

```r
# load statedata from library(urbnmapr)
state_data <- urbnmapr::statedata

knitr::kable(head(state_data), digits = 3)
```

year	state_fips	state_name	hhpop	horate	medhhincome
2015	01	Alabama	1846380	0.681	44700
2015	02	Alaska	250183	0.631	70600
2015	04	Arizona	2463012	0.621	51000
2015	05	Arkansas	1144657	0.655	42000
2015	06	California	12895471	0.537	64600
2015	08	Colorado	2074517	0.639	63500

```r
# select numeric variables and calculate the PCs with scaled data
state_data_numeric <- state_data %>%
  select(hhpop, horate, medhhincome) %>%
  mutate(
    hhpop = scales::rescale(hhpop),
    horate = scales::rescale(horate),
    medhhincome = scales::rescale(medhhincome)
  )

# create a correlation matrix
cor(state_data_numeric)
```

```
##                   hhpop     horate medhhincome
## hhpop        1.00000000 -0.2926963  0.01883672
## horate      -0.29269634  1.0000000 -0.37477714
## medhhincome  0.01883672 -0.3747771  1.00000000
```

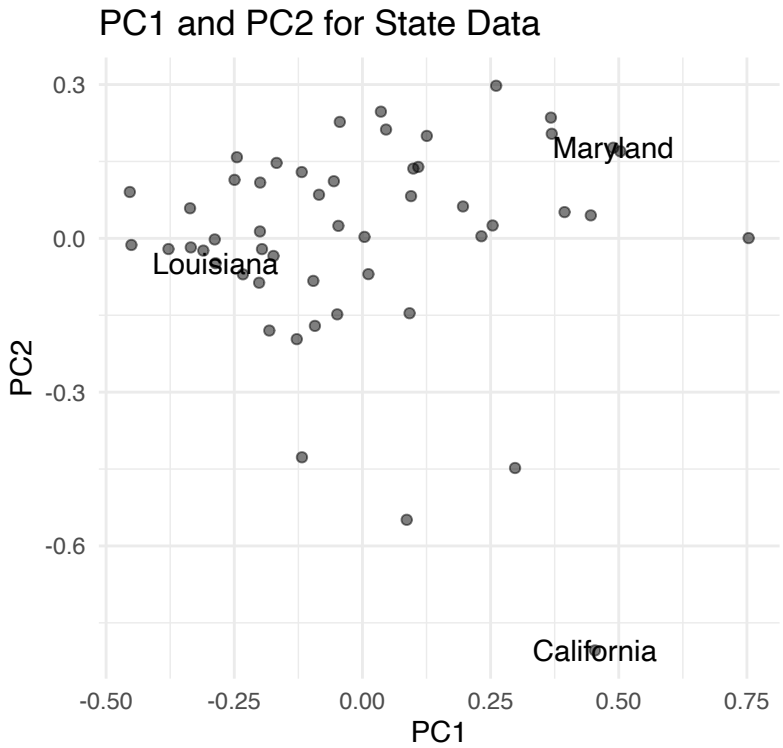
```r
# create a covariance matrix
cov(state_data_numeric)
```

```
##                    hhpop      horate   medhhincome
## hhpop        0.0384419039 -0.01015988  0.0009692593
## horate      -0.0101598767  0.03134277 -0.0174130223
## medhhincome  0.0009692593 -0.01741302  0.0688754957
```

```r
# apply PCA
principal_components <- state_data_numeric %>%
  prcomp()

# plot
state_data_pca <- bind_cols(
  state_data,
  as_tibble(principal_components$x)
)
```

```
ggplot() +
  geom_point(
    data = state_data_pca,
    aes(PC1, PC2),
    alpha = 0.5
  ) +
  geom_text(
    data = filter(state_data_pca, state_fips %in% c("06", "22", "24")),
    aes(PC1, PC2, label = state_name)
  ) +
  labs(title = "PC1 and PC2 for State Data") +
  coord_equal() +
  theme_minimal()
```




```
# view the amount of variance explained
summary(principal_components)
```

```
## Importance of components:
##                           PC1    PC2    PC3
## Standard deviation     0.2762 0.2060 0.1412
## Proportion of Variance 0.5502 0.3060 0.1438
## Cumulative Proportion  0.5502 0.8562 1.0000
```

We can look at the loadings (ϕ_{ij}s) to determine how much each variable contributes to each principal component. Large loadings (positive or negative) indicate that the variable

has a strong relationship to the principal component. The sign of the loading indicates the direction of the relationship. Here, `medhhincome` contributes the most the `PC1` and `hhpop` contributes the most to `PC2`.

```
principal_components$rotation
```

```
##                    PC1        PC2       PC3
## hhpop        0.1262199 -0.8848076 0.4485354
## horate      -0.3829869  0.3736307 0.8448202
## medhhincome  0.9150899  0.2784163 0.2917101
```

Tutorial 2 (categorical variables)

Here we have a data set with 493 votes from two years of the 114th Senate (2015-2017). The data set has 100 rows and 495 variables. An affirmative vote is `1`, a negative vote is `-1`, and an abstention is `0`. The data are from Bradley Robinson and this example is based on an earlier analysis by Professor Sivan Leviyang.

Understanding the data or visualizing the data would be a time consuming task. Here are the first few columns and rows:

```
votes <- read_csv(here::here("tutorials", "11_unsupervised-ml-dimension-reduction", "votes.csv"))

votes[1:6, 1:6]
```

```
## # A tibble: 6 x 6
##   name               party    v1    v2    v3    v4
##   <chr>              <chr> <dbl> <dbl> <dbl> <dbl>
## 1 Lamar Alexander    R         1    -1    -1     1
## 2 Kelly Ayotte       R        -1    -1     1     1
## 3 Tammy Baldwin      D        -1     1    -1    -1
## 4 John Barrasso      R         1    -1     1     1
## 5 Michael Bennet     D        -1     1    -1    -1
## 6 Richard Blumenthal D        -1     1    -1    -1
```

Let's use PCA to create principal components of the 493 votes variables.

```
# select the numeric variables
votes_numeric <- votes %>%
  select_if(is.numeric)

# run PCA
votes_pca <- prcomp(votes_numeric)

# extract the principal components
```

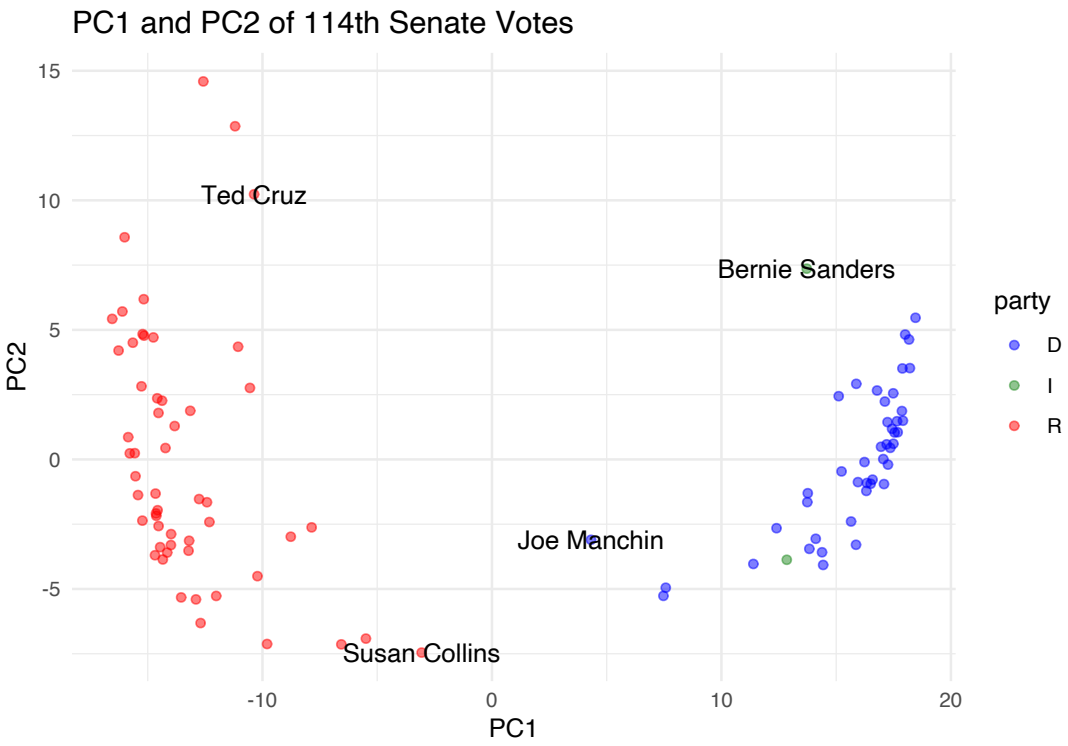
```
votes_pcs <- votes_pca %>%
  .$x %>%
  as_tibble()

# combine the pcs to the names and parties
votes_pcs <- bind_cols(
  select(votes, name, party),
  votes_pcs
)
```

The first principal component explains more than 60 percent of the variation.

```
summary(votes_pca)
```

PC	SD	PVE	CVE
PC1	14.713	0.627	0.627
PC2	4.141	0.050	0.677
PC3	2.598	0.020	0.697
PC4	2.371	0.016	0.713
PC5	2.308	0.015	0.728
PC6	2.239	0.015	0.743
PC7	2.098	0.013	0.756
PC8	1.954	0.011	0.767
PC9	1.866	0.010	0.777
PC10	1.793	0.009	0.786



How many principal components should you use?

By default, conducting PCA with `prcomp()` will return as many principal components as there are variables. Because because our goal is dimensionality reduction, we need a way to determine how many principal components to keep for analysis. Several different criteria have been suggested:

- Ignore principal components at the point at which the next principal component offers little increase in the total explained variation.
- Include all principal components up to a predetermined total percent explained variation, such as 90%.
- Ignore components whose explained variation is less than the average variation explained, with the idea being that such a principal component offers less than one variable's worth of information.
- Ignore the last principal components whose explained variation are all roughly equal.

Source: Holland, 2021