Topic05.Feature reduction selection/extraction

資料科學 Data Science

張家銘 Jia-Ming Chang

政治大學資訊科學系

Copyright declaration 版權說明

- Some of the figures in this presentation are taken from "An Introduction to Statistical Learning, with applications in R" (Springer, 2013) with permission from the authors: G. James, D. Witten, T. Hastie and R. Tibshirani.
- The web site of the book
- The credit of individual is indicated in the bottom part of the slide.
  - ie.,

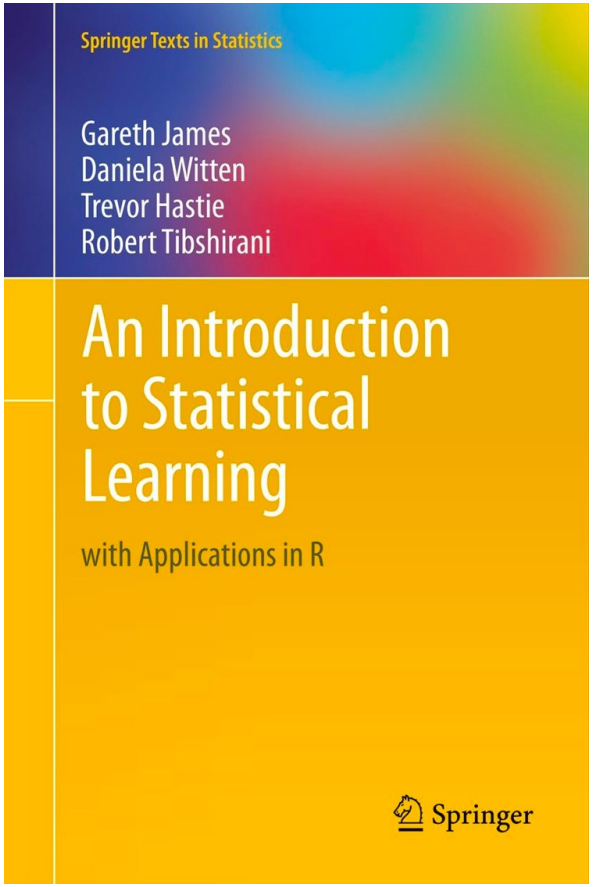


Figure 3.18, *An Introduction to Statistical Learning with Applications in R,* by Gareth James, Daniela Witten, Trevor Hastie and Robert Tibshirani

Copyright declaration 版權說明

- Some of the figures in this presentation are taken from "Practical Data Science with R (Manning, 2019)"
- The web site of the book
- The credit of individual is indicated in the bottom part of the slide.
  - ie.,

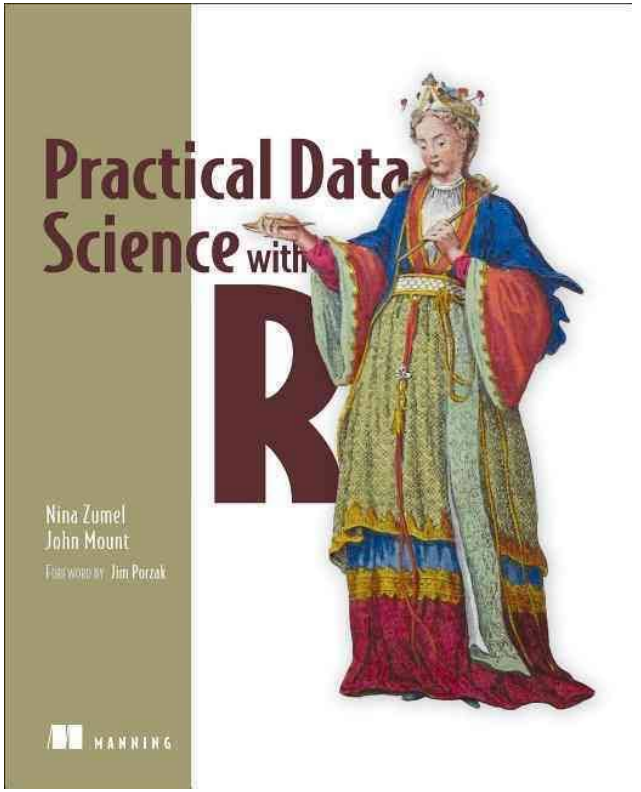
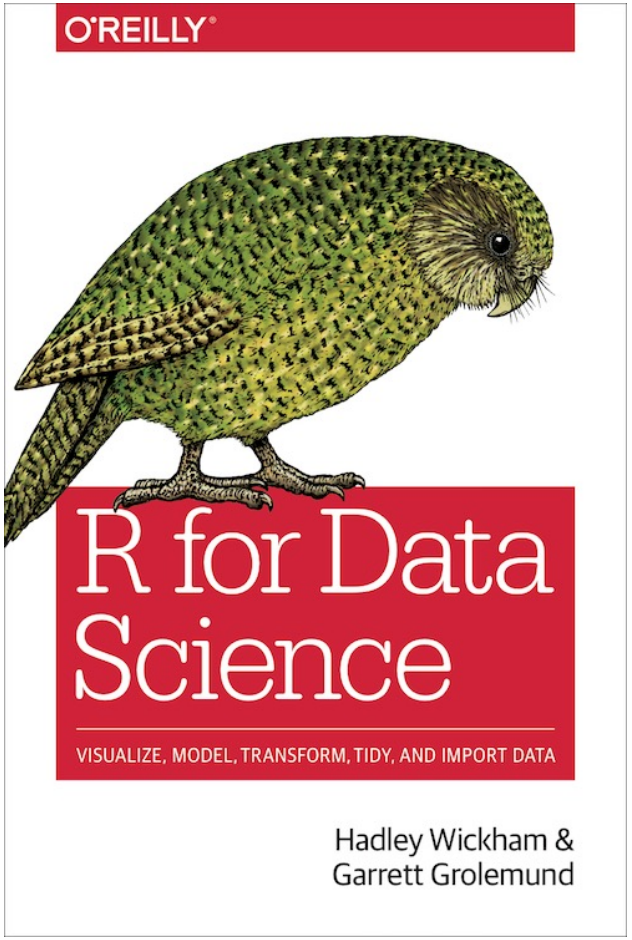


Figure 7.6, *Practical Data Science with R* by Nina Zumel and John Mount

Copyright declaration 版權說明

- Some of the figures in this presentation are taken from "R for Data Science" under the Creative Commons Attribution-NonCommercial-NoDerivs 3.0 License.
- The web site of the book
- The credit of individual is indicated in the bottom part.
  - ie.,

R for Data Science by Garrett Grolemund, Hadley Wickham

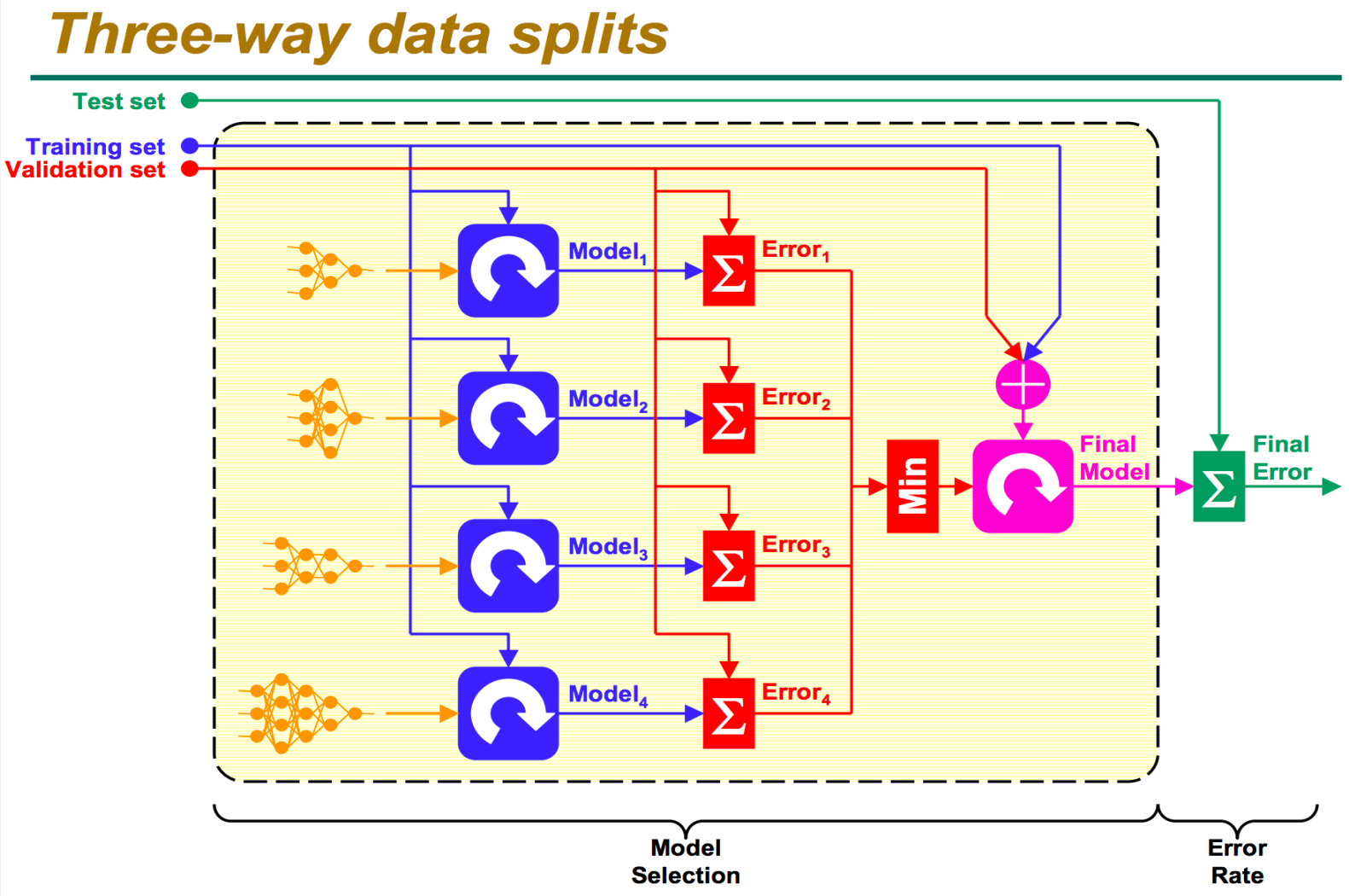


Recap for the last week

Bias-variance decomposition

- $E[(y[i]-f(x[i,]))^2] = \text{bias}^2 + \text{variance} + \text{irreducibleError}$
  - Model bias : your chosen modeling technique will never get right
  - Model variance : your modeling technique gets wrong due to incidental relations in the data.
  - Irreducible error : truly unmodelable portion of the problem given the current variables

Three-way data splits

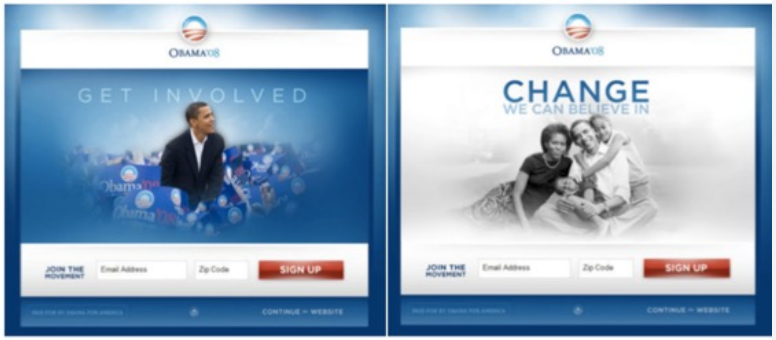


Summarizing R's distribution naming conventions

- dDIST(x, ...)
  - the *distribution function* (PDF) that returns the probability of observing the value *x*.
- rDIST(n, ...)
  - the *random number generation* function that returns *n* values drawn from the distribution DIST.
- qDIST(p, ...)
  - the *quantile function* that returns the *x* corresponding to the *p*th percentile of DIST.
- pDIST(x, ...)
  - the *cumulative distribution function* (CDF) that returns the probability of observing a value less than *x*.

Significance testing

- *null hypothesis* D = (err.null - err.model) == 0
- *p-value* is the probability of null hypothesis
  - ***p* < 0.05 =>** reject the null hypothesis



Summary	Original trait	Family trait
Visitors	51,794	51,696
Sign-up	4,425	4,996
Conv. Rate	8.54%	9.66%
		✔

A/B test

- Fisher's test for independence
  - the null hypothesis : conversion is independent of group , A=B
  - fisher.test(tab)
- Frequentist significance test
  - assume that *A* and *B* come from an identical distribution with a common conversion rate => how likely it would be that the B group scores as high as it did by mere chance!

Statistical test power

- probability of rejecting the null hypothesis when the null hypothesis is false = 1 - *p*-value

Parameter	Meaning	Value for our example
`confidence` (or `power`)	This is how likely you want it to be that the test result is correct. We'll write `confidence = 1 - errorProb`.	0.95 (or 95% confident), or `errorProb=0.05`.
`targetRate`	This is the conversion rate you hope the B treatment achieves: the further away from the A rate, the better.	We hope the B treatment is at least 0.045 or a 4.5% conversion rate.
`difference`	This is how big an error in conversion rate estimate we can tolerate.	We'll try to estimate the conversion rate to within plus/minus 0.4%, or 0.004, which is greater than the distance from our `targetRate` and our historical A conversion rate.

seeing a B conversion rate in the range of 4.1–4.9% if the true B conversion rate were in fact 4.5% => how many customers to the B treatment?

Sample size estimate

```
estimate <-
function(targetRate,difference,errorProb) {
    ceiling(-
log(errorProb)*targetRate/(difference^2))
}
(est <- estimate(0.045,0.004,0.05))
```

- We need about 8,426 visitors to have a 95% chance of observing a B conversion rate of at least 0.041 if the true unknown B conversion rate is at least 0.045.

Design experiments

```
estimate <-
function(targetRate,difference,err
orProb)
```

```
estimate(0.045,0.004,0.05)
```

- Performance difference vs confidence
  - measure large performance differences with high confidence

    ```
    estimate(0.045,0.005,0.04)
    ```

  - measure small performance differences with even moderate confidence.

    ```
    estimate(0.045,0.003,0.06)
    ```

Exact binomial sample size calculation

- We need about 8,426 visitors to have a 95% chance of observing a B conversion rate of at least 0.041 if the true unknown B conversion rate is at least 0.045.

```
pbinom(ceiling((0.045-0.04)* 8426), size= 8426,
prob=0.045)
```

- The failure odds are around 4% (under the 5% we're designing for), which means the estimate size was slightly high.

Venue shopping reduces test power

- *multiple tests* : if you run 20 treatments, each with a *p*-value goal of 0.05, you would expect one test to appear to show significant improvement, even if all 20 treatments are useless
- *p* cutoff: *p* / the # of tests you intend to run

Feature reduction

Feature reduction

- Feature selection
  - select the most influential features from the feature set => improve the recognition rate as well as reduce the computation load
- Feature extraction
  - find a linear transformation to transform the original feature vector into a new one => maximize the separation between different classes

Feature extraction vs. feature selection

- Feature extraction: Transform the existing features into a lower dimensional space
- Feature selection: Select a subset of the existing features without a transformation

selection

$$\begin{bmatrix} x_1 \\ x_2 \\ \\ \\ x_N \end{bmatrix} \rightarrow \begin{bmatrix} x_{i_1} \\ x_{i_2} \\ \\ \\ x_{i_M} \end{bmatrix}$$

extraction

$$\begin{bmatrix} x_1 \\ x_2 \\ \\ \\ x_N \end{bmatrix} \rightarrow \begin{bmatrix} y_1 \\ y_2 \\ \\ \\ y_M \end{bmatrix} = f\left(\begin{bmatrix} x_1 \\ x_2 \\ \\ \\ x_N \end{bmatrix}\right)$$

Feature selection

Feature Selection: Goal & Benefits

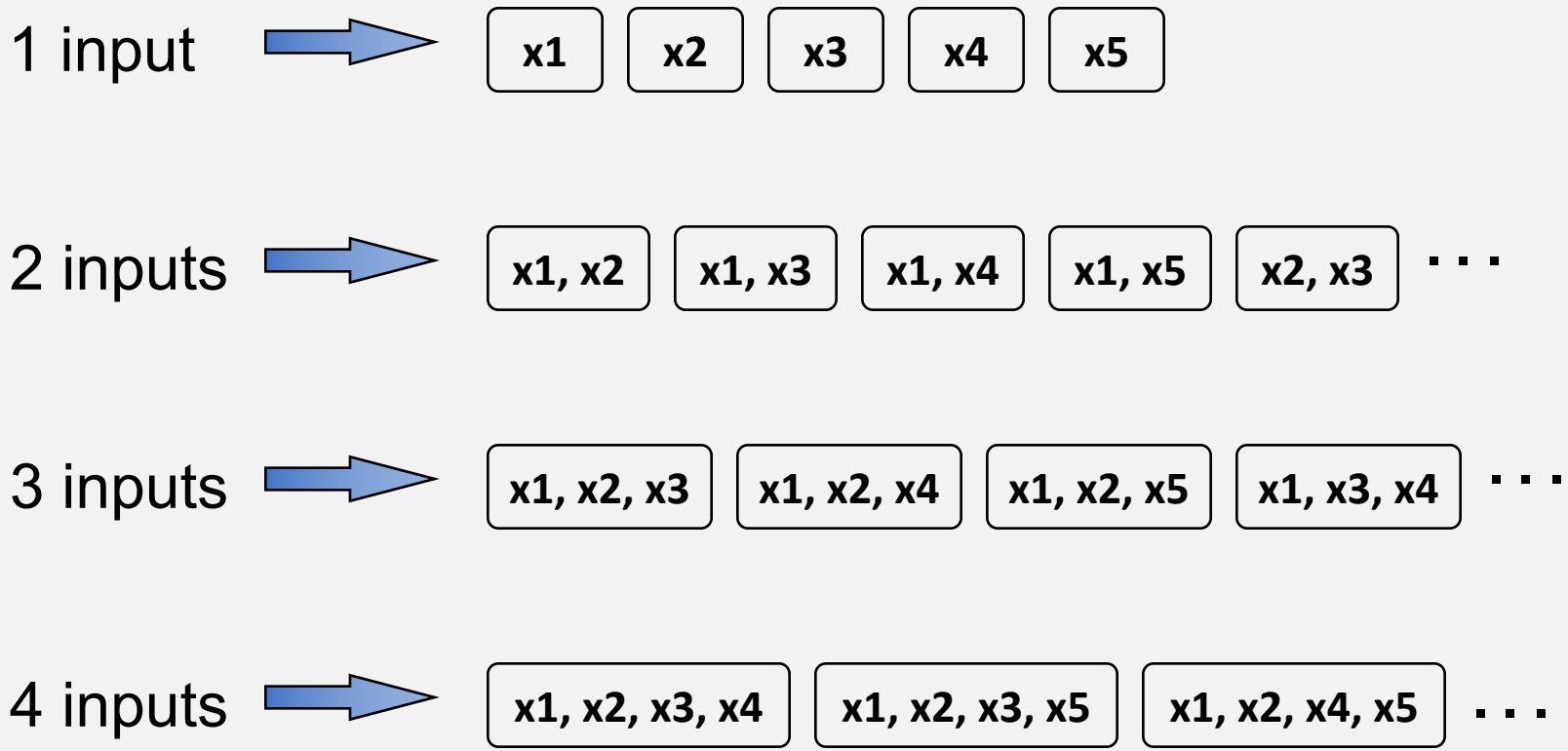
- Goal
  - To select a subset out of the original feature set for better prediction accuracy
- Benefits
  - Improve recognition rate
  - Reduce computation load
  - Explain relationships between features and classes

Exhaustive Search

- Steps for direct exhaustive search
  - Use *k*-nearnest neighboring classifier (KNNC) as the classifier, leave-one-out (LOO) test for R^2 estimate
  - Generate all combinations of features and evaluate them one-by-one
  - Select the feature combination that has the best R^2.

Exhaustive Search

- Direct exhaustive search



Exhaustive Search

- Drawback
  - Time consuming
  - *d* features => 2^d-1 procedures for performance evaluation
    - d = 10 ➔ 1023 CV for evaluation
- Advantage
  - The optimal feature set can be identified.

# of proper subsets for one w/ cardinality = d

Exhaustive Search

- Characteristics of exhaustive search for feature selection
  - The process is time consuming, but the identified feature set is optimum.
  - It's possible to use classifiers other than KNNC.
  - It's possible to use performance indices other than LOO.

Heuristic Search

educated guess

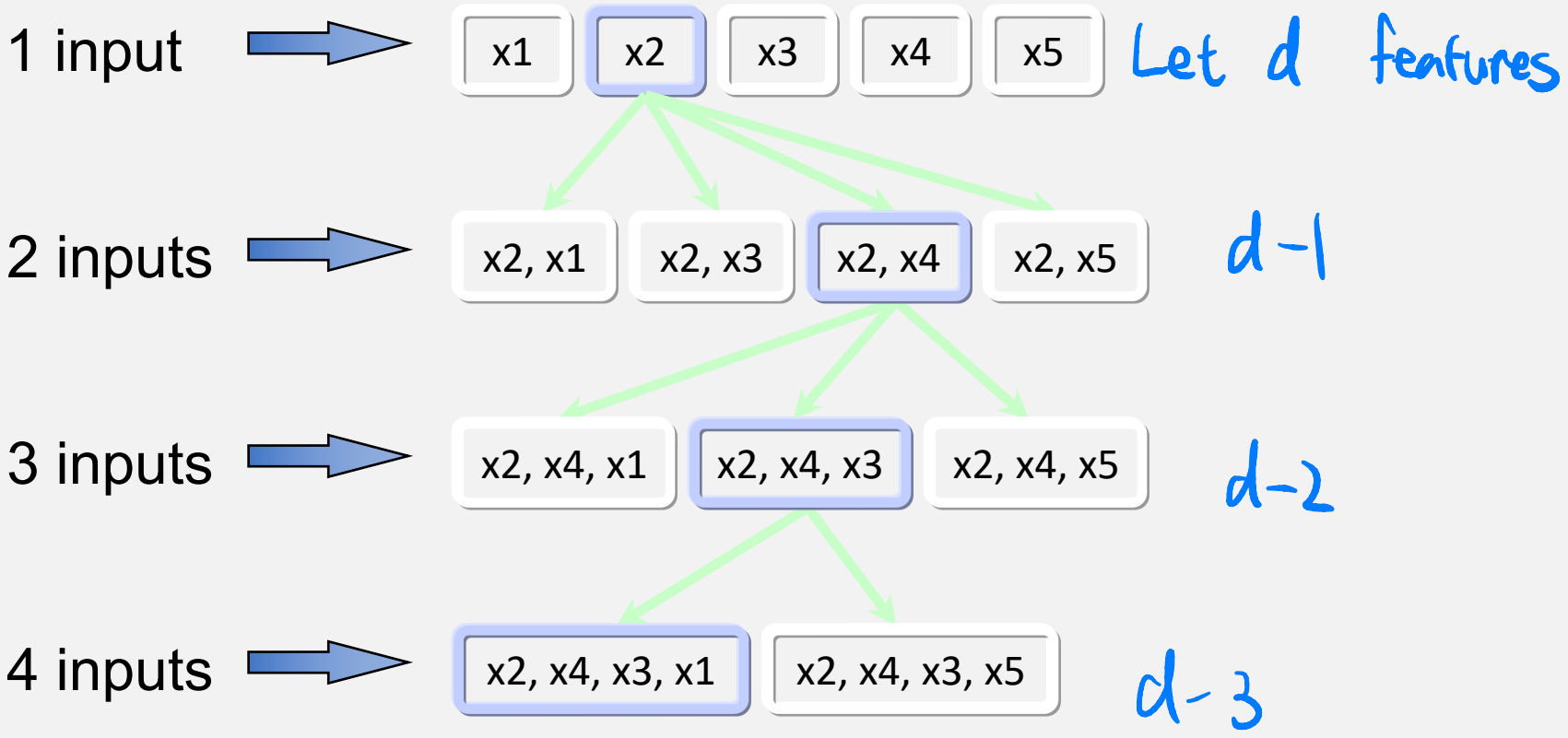
- Heuristic search for input selection
  - One-pass ranking
  - Sequential forward selection
  - Generalized sequential forward selection
  - Sequential backward selection
  - Generalized sequential backward selection
  - ‘Add *m*, remove *n*’ selection
  - Generalized ‘add *m*, remove *n*’ selection

1. **One-pass ranking**: This method ranks features individually based on certain criteria like correlation with the target variable, and selects features in a single pass without revisiting the selection.
2. **Sequential forward selection**: Starting with no features, this process adds one feature at a time. At each step, it adds the feature that, when combined with the features already selected, improves the model's performance the most until no further improvements are possible.
3. **Generalized sequential forward selection**: An extension of sequential forward selection, this method allows for a look-ahead by adding multiple features at a time before finalizing the selection, which can avoid local optima.
4. **Sequential backward selection**: This method starts with all available features and removes one feature at a time. At each step, it removes the feature whose absence least affects or even improves the performance of the model.
5. **Generalized sequential backward selection**: Similar to generalized forward selection but in reverse; it removes sets of features iteratively, allowing multiple features to be removed at once while checking the impact on performance.
6. **'Add m, remove n' selection**: This method allows both addition and removal of features within the same iteration: 'm' features are added and 'n' less important features are removed in each round to find a good feature subset.
7. **Generalized 'add m, remove n' selection**: It generalizes the 'add m, remove n' method by potentially adding and removing different numbers of features ('m' and 'n') at each iteration, and it may use additional criteria to guide these additions and removals.

Sequential Forward Selection

- Steps for sequential forward selection
  - Use KNNC as the classifier, LOO for R^2 estimate
  - Select the first feature that has the best R^2.
  - Select the next feature (among all unselected features) that, together with the selected features, gives the best R^2.
  - Repeat the previous step until all features are selected.

Sequential Forward Selection



Sequential Forward Selection

- Advantage
  - A lot more efficient
  - *d* features, we need to perform d(d+1)/2 CV

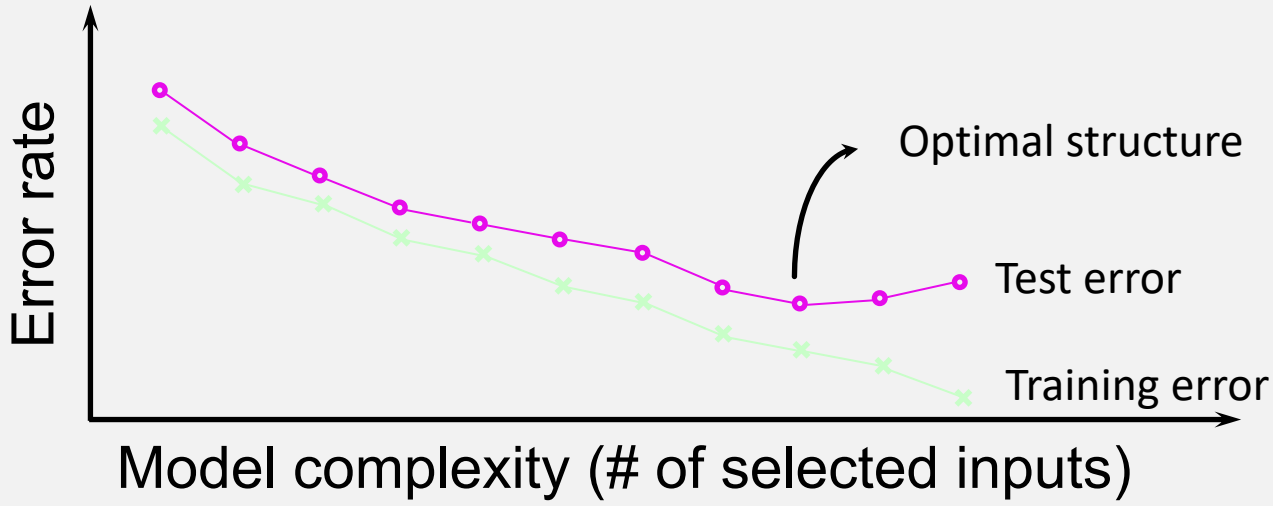
$d + (d-1) + (d-2) + \ldots + 1$

cross validations

- Drawback
  - The selected features are not always optimal.

Use of Input Selection

- Common use of input selection
  - Increase the model complexity sequentially by adding more inputs
  - Select the model that has the best test RR (R^2)
- Typical curve of error vs. model complexity
  - Determine the model structure with the least test error



how would a given example score differently if it had different attributes?

Local interpretable model-agnostic explanations for explaining model predictions

iris example

- A dataset of petal and sepal measurements for three varieties of iris.
- The object is to predict whether a given iris is a setosa based on its petal and sepal dimensions.
  - 4 measurements
    - sepal length
    - sepal width
    - petal length
    - petal width
  - 3 different species
    - Setosa
    - versicolor
    - virginica

Credit by https://www.math.umd.edu/~petersd/666/html/iris_pca.html

Loading the iris dataset : LIME.R

```
iris <- iris

iris$class <- as.numeric(iris$Species == "setosa")

set.seed(2345)
intrain <- runif(nrow(iris)) < 0.75
train <- iris[intrain,]
test <- iris[!intrain,]

head(train)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species class
## 1          5.1         3.5          1.4         0.2  setosa     1
## 2          4.9         3.0          1.4         0.2  setosa     1
## 3          4.7         3.2          1.3         0.2  setosa     1
## 4          4.6         3.1          1.5         0.2  setosa     1
## 5          5.0         3.6          1.4         0.2  setosa     1
## 6          5.4         3.9          1.7         0.4  setosa     1
```

Fitting a XGBOOST model to the iris training data

```
source("lime_iris_example.R")

input <- as.matrix(train[, 1:4])
model <- fit_iris_example(input, train$class)
```

Building a LIME explainer from the model and training data

```
library(lime)
explainer <- lime(train[,1:4],
                  model = model,
                  bin_continuous = TRUE,
                  n_bins = 10)
```

Explaining the iris example

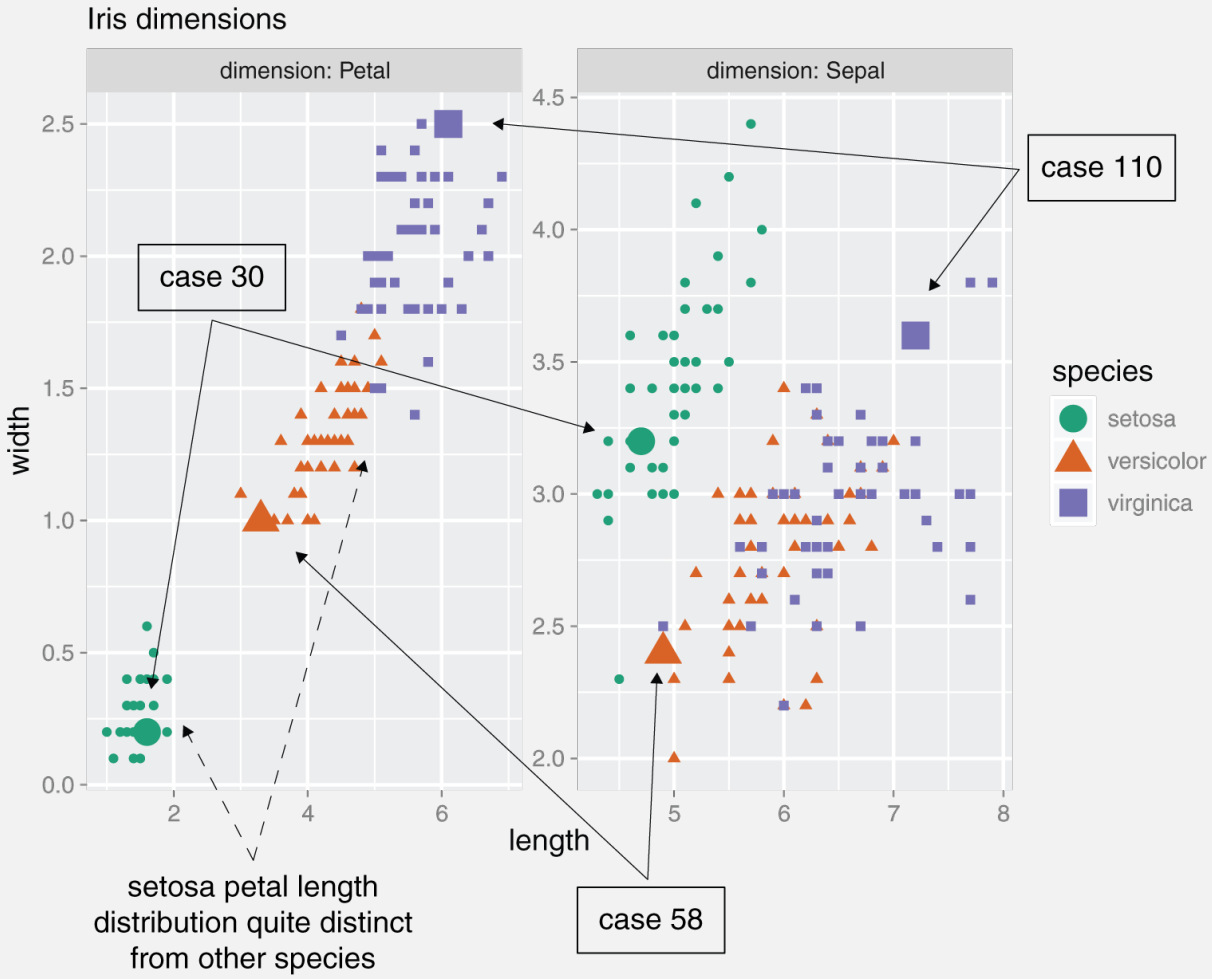
```
(example <- test[5, 1:4, drop=FALSE])
explanation <- lime::explain(example,
                             explainer,
                             n_labels = 1,
                             n_features = 4)
plot_features(explanation)
```

More iris examples

```
(example <- test[c(13, 24), 1:4])
test$class[c(13,24)]
explanation <- explain(example,
                       explainer,
                       n_labels = 1,
                       n_features = 4,
                       kernel_width = 0.5)

plot_features(explanation)
```

Distributions of petal and sepal dimensions by species



Credit by Figure 6.25, *Practical Data Science with R* by Nina Zumel, John Mount

How does LIME work?

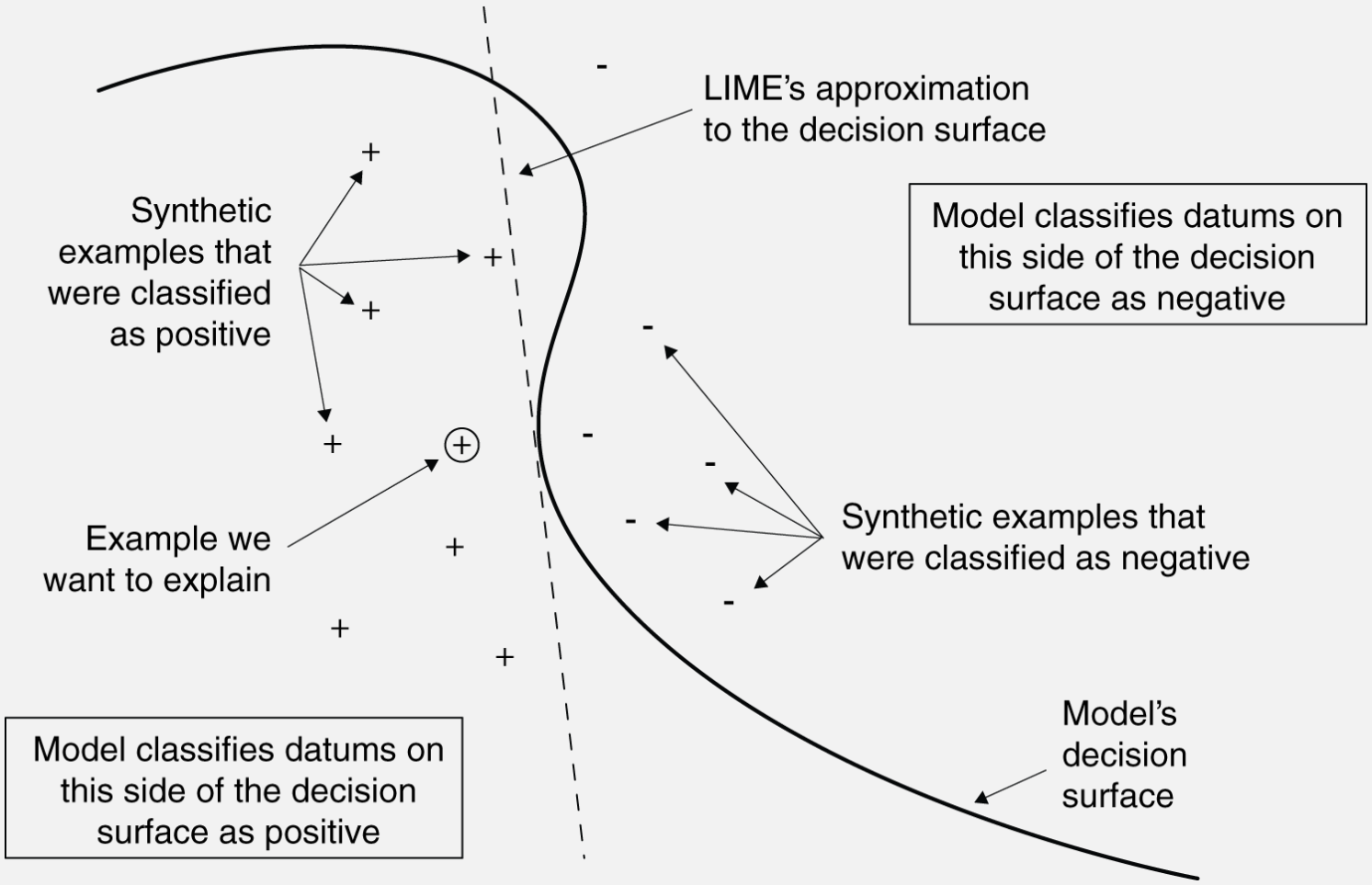
Local

Interpretable

Model - agnostic

Explanations

Notional sketch of how LIME works



Credit by Figure 6.23, *Practical Data Science with R* by Nina Zumel, John Mount

synthetic data points

- Original data point

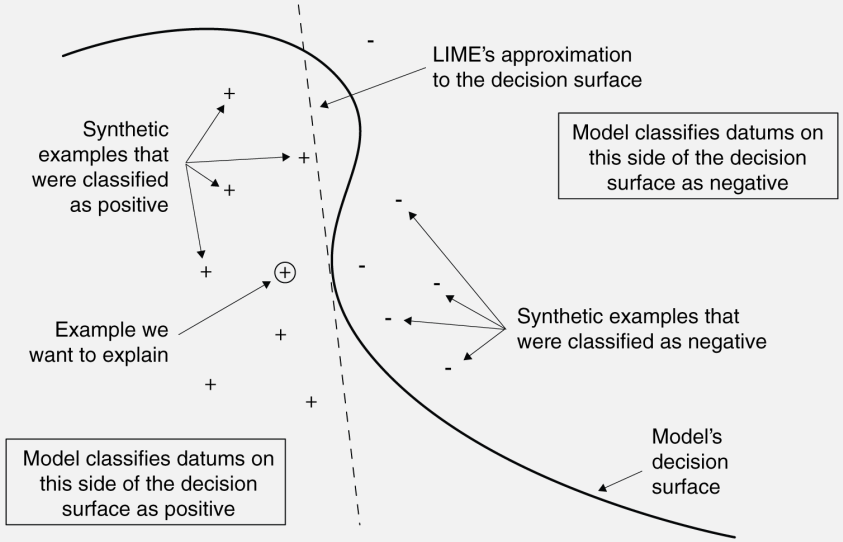
```
Sepal.Length Sepal.Width Petal.Length Petal.Width
         5.1         3.5          1.4         0.2
```

- a jittered point

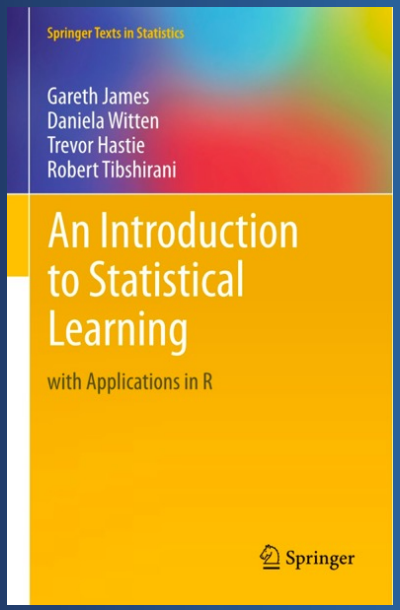
```
Sepal.Length Sepal.Width Petal.Length Petal.Width
    5.505938    3.422535       1.3551   0.4259682
```

The model's decision surface

- LIME's estimate of the decision surface near the example we are trying to explain.
- Fit an *m*-dimensional linear model for {y_i} as a function of {s_i}. The linear model is LIME's estimate of the original model's decision surface near example (a dashed line).



Feature extraction



10. Unsupervised Learning

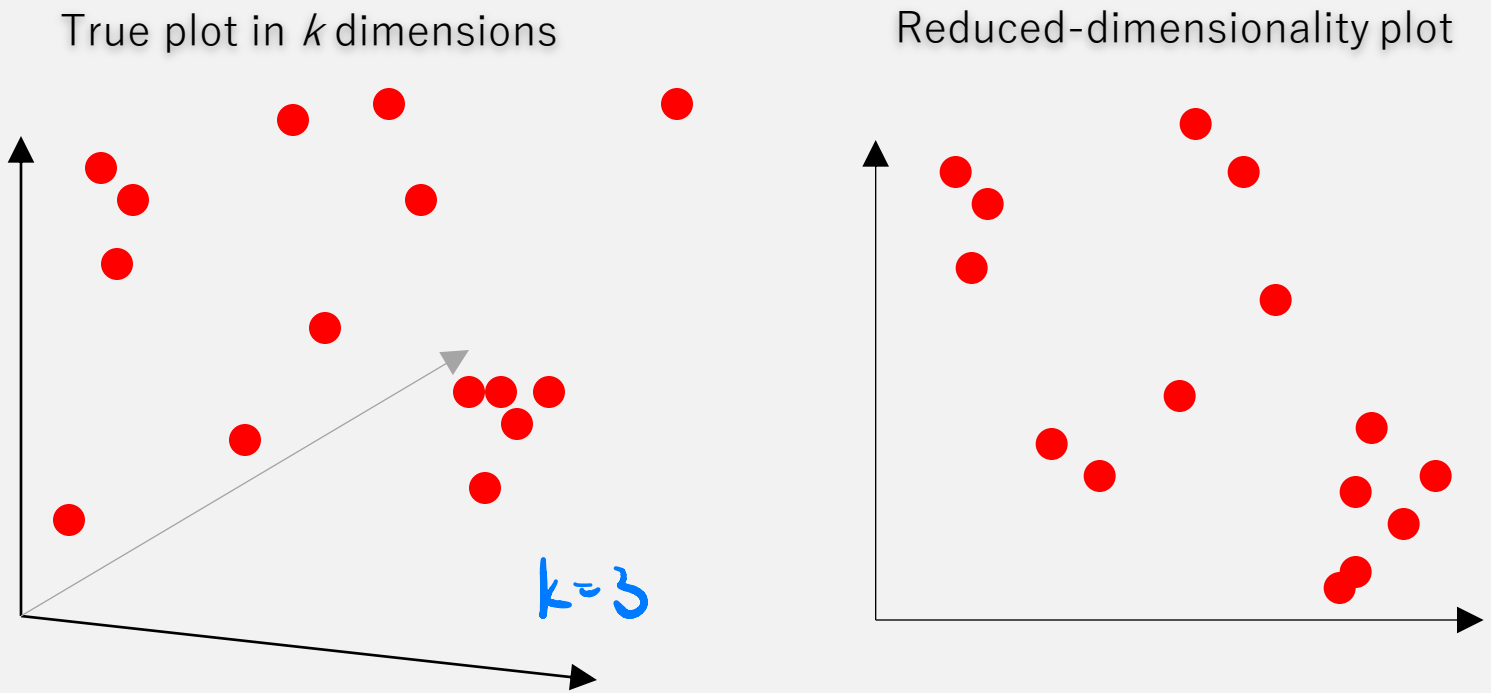
10.2 Principal Components Analysis

Feature extraction

- We wish to visualize n observations with measurements on a set of p features, $X_1, X_2, \ldots, X_p$, as part of an exploratory data analysis.
- We could do this by examining two-dimensional scatterplots of the data.
  - A two-dimensional representation of the data that captures most of the information, then we can plot the observations in this low-dimensional space.

Feature extraction

- a best choice of axes – shows most variation in the data.
- Found by linear algebra: Singular Value Decomposition (SVD)



Principal Component Analysis (PCA)

Principal Component Analysis (PCA)

dimensionality reduction

- An effective method for reducing dataset dimensions while keeping spatial characteristics as much as possible
- Characteristics
  - For unlabeled data
  - A linear transform with solid mathematical foundation
- Applications
  - Line/plane fitting
  - Face recognition
  - Many many more…

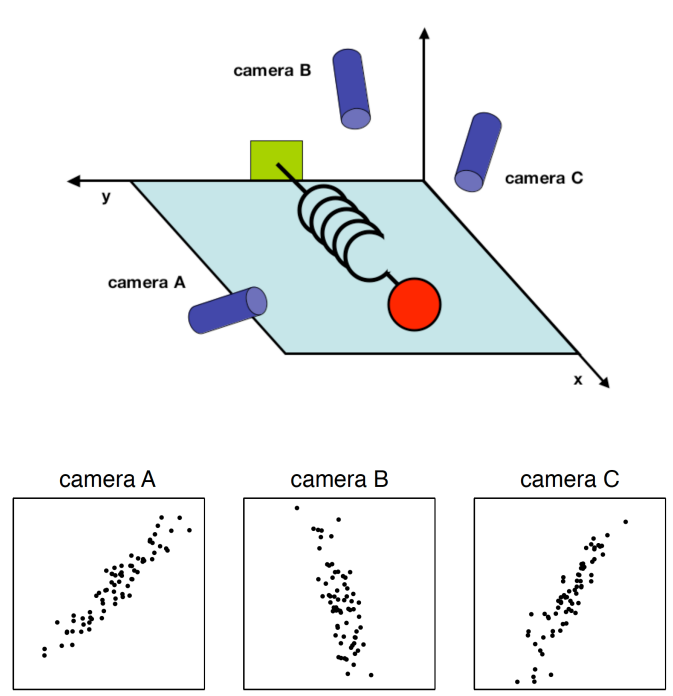
What are principal components?

- PCA seeks a small number of dimensions that are as **interesting** as possible
  - the concept of interesting = the amount that the observations **vary along each dimension.**
- Dimensions found by PCA is a linear combination of the p features

Assumes linear combination ∃ might not perform well when the underlying relationship ∃ not linear.

A toy example for PCA

- Is there another basis, which is a linear combination of the original basis, that best re-expresses our data set?



$$\vec{X} = \begin{bmatrix} x_A \\ y_A \\ x_B \\ y_B \\ x_C \\ y_C \end{bmatrix}$$

6 features

Change of Basis

- X : be the original data set, ***m* x *n***
  - m features
  - n points
- P : linear transformation
- Y : a new representation of that data set

· each column is an observation

· each row is a feature

$$X = \begin{bmatrix} | & | & & | \\ x_1 & x_2 & \cdots & x_n \\ | & | & & | \end{bmatrix}$$

m

n

Credit: A Tutorial on Principal Component Analysis, Jonathon Shlens

Change of Basis

each column vector (y_i) represents the transformed version of the corresponding column vector (X_i)

- Y be another matrix related a linear transformation P.

$$PX = \begin{bmatrix} p_1 \\ \cdots \\ p_m \end{bmatrix} [X_1 \ldots X_n] = \begin{bmatrix} p_1 \cdot X_1 & \cdots & p_1 \cdot X_n \\ \vdots & \ddots & \vdots \\ p_m \cdot X_1 & \cdots & p_m \cdot X_n \end{bmatrix} = Y, \text{ where } y_i = \begin{bmatrix} p_1 \cdot X_i \\ \vdots \\ p_m \cdot X_i \end{bmatrix}$$

  - Geometrically, P is a rotation and a stretch which again transforms X into Y.
  - The rows of P, $\{p_1,\ldots,p_m\}$, are a set of new basis vector for expressing the columns of X.

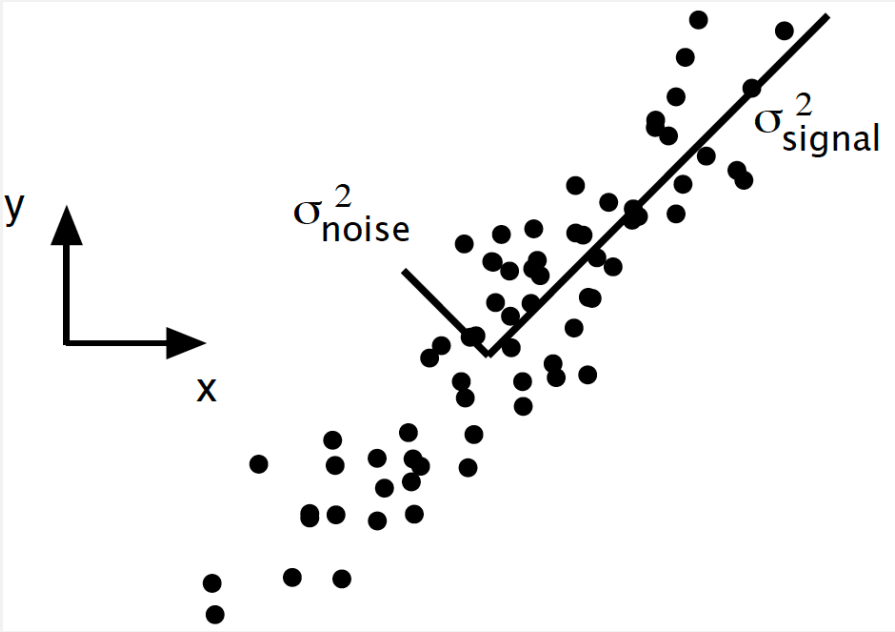
- Therefore, the rows of P are a new set of basis vectors for representing of columns of X.

What is the best way to re-express X?

- What is a good choice of basis P?
- Two key points
  - Noise and Rotation
  - Redundancy

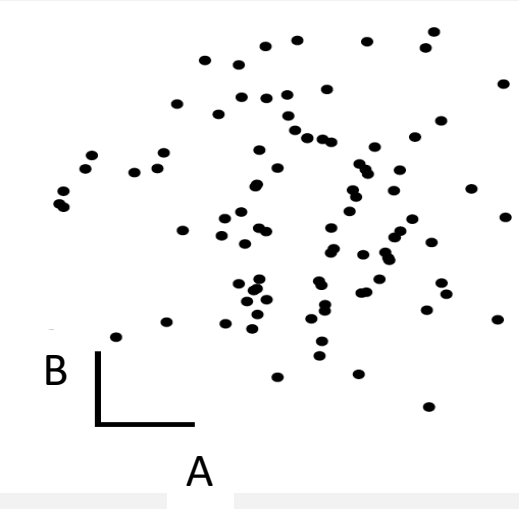
Ratio of variances

- signal-to-noise ratio (SNR) = $\frac{\sigma^2_{signal}}{\sigma^2_{noise}}$

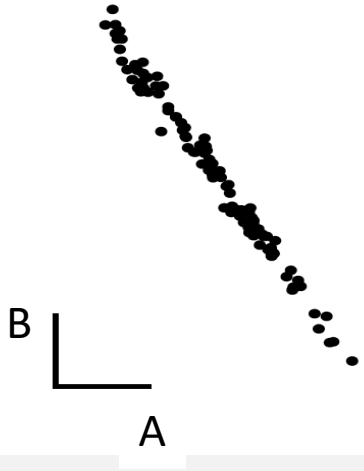


Credit: A Tutorial on Principal Component Analysis, Jonathon Shlens

Redundancy



low redundancy



high redundancy

In the context of the slides and PCA, redundancy refers to the overlap in information provided by the features (dimensions) of a dataset. When there is low redundancy, it means that each feature contributes unique information that is not available from the other features. Therefore, in a low redundancy scenario:

- The features are less correlated or uncorrelated.
- The dataset cannot be accurately described by fewer dimensions without losing significant information.
- Variance (information) is spread out across all dimensions, meaning that each dimension adds some new information about the data distribution.

On the other hand, high redundancy indicates that:

- Some features are highly correlated.
- Much of the information contained in one feature can be predicted or inferred from others.
- The dataset can be compressed into fewer dimensions with minimal loss of information.

Redundancy measured using *Covariance*

zero mean assumption

- Consider two sets of measurements with zero means

$$A = \{a_1, a_2, \dots, a_n\},\ B = \{b_1, b_2, \dots, b_n\}$$

- The variance of *A* and *B*

$$Var(X) = \sum_i^n p_i \cdot (x_i - \mu)^2$$

$$\sigma_A^2 = \frac{1}{n}\sum_i a_i^2\ ,\ \sigma_B^2 = \frac{1}{n}\sum_i b_i^2$$

- Covariance of *A* and *B*

$$cov(X,Y) = \frac{1}{n}\sum_i^n (x_i - \overset{0}{E(X)})(y_i - \overset{0}{E(Y)})$$

$$\sigma_{AB}^2 = \frac{1}{n}\sum_i a_i b_i$$

Credit: A Tutorial on Principal Component Analysis, Jonathon Shlens

Redundancy measured using *Covariance*

- Convert *A* and *B* into corresponding row vectors

$$a = [a_1\ a_2\ \ldots\ a_n],\ b = [b_1\ b_2\ \ldots\ b_n]$$

- Covariance of *A* and *B*

$$\sigma_{ab}^2 = \frac{1}{n} ab^T$$

scalar

nx1

1xn

- A **high positive covariance** indicates that the two variables increase together and thus are redundant, as increases in one predict increases in the other.
- A **high negative covariance** implies that as one variable increases, the other decreases.
- A **covariance near zero** suggests that the variables are independent, indicating low redundancy.

Covariance matrix C_X

$$X = \begin{bmatrix} X_1 \\ \dots \\ X_m \end{bmatrix}$$

$$\sigma^2_{ab} = \frac{1}{n} ab^T$$

$$C_X \equiv \frac{1}{n} XX^T \equiv \begin{bmatrix} \sigma^2_{X_1X_1} & \sigma^2_{X_1X_2} & \dots \\ \sigma^2_{X_2X_1} & \sigma^2_{X_2X_2} & \dots \\ \vdots & \vdots & \ddots \end{bmatrix}$$

- C_X is a square symmetric m x m matrix
  - The diagonal terms : the variance of particular measurement types
  - The off-diagonal terms : the covariance between measurement types

Credit: A Tutorial on Principal Component Analysis, Jonathon Shlens

Diagonal vs off-diagonal terms @ C_X

- the diagonal terms
  - the variance of particular measurement types
  - large values correspond to **interesting** structure
- the off-diagonal terms
  - the covariance between measurement types
  - large magnitudes correspond to **high redundancy**

$$C_X \equiv \begin{bmatrix} \sigma^2_{X_1X_1} & \sigma^2_{X_1X_2} & \cdots \\ \sigma^2_{X_2X_1} & \sigma^2_{X_2X_2} & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix}$$

Credit: A Tutorial on Principal Component Analysis, Jonathon Shlens

The goal of PCA

- Find some orthonormal matrix P in $Y = PX$
- What is the covariance matrix C_Y?
- $C_Y = P C_X P^T$?

$$C_Y \overset{\text{def. of covariance}}{=\!=} \frac{1}{n} YY^T \overset{Y=PX}{=\!=} \frac{1}{n}(PX)(PX)^T$$

$$= \frac{1}{n}(PX)(X^T P^T) = P\left(\frac{1}{n}XX^T\right)P^T$$

$$\overset{\text{def. of covariance}}{=\!=} C_X$$

Diagonalize C_Y

- All off-diagonal terms in C_Y should be zero => C_Y must be a diagonal matrix
- Y is decorrelated (P decorrelates the data)
- Each successive dimension in Y should be rank-ordered according to variance

$$C_Y \equiv \begin{bmatrix} \sigma_1 & 0 & \dots \\ 0 & \sigma_2 & \dots \\ \vdots & \vdots & \ddots \end{bmatrix}$$

variance of the 1st principal component

variance of the 2nd principal component

How to diagonalize C_Y?

- Find some orthonormal matrix P in $Y = PX$ such that $C_y = \frac{1}{n} YY^T$ is a diagonal matrix.
- What is P?
  - The rows of P = the **principal components** of X

= the eigenvectors of the covariance matrix of X (C_x)

Vector space

linear algebra

Scalar multiples

- a scalar λ

$$x = \begin{bmatrix} 1 \\ 3 \\ 4 \end{bmatrix} \text{ and } y = \begin{bmatrix} -20 \\ -60 \\ -80 \end{bmatrix}$$

$$x = \lambda y$$

Credit by https://en.wikipedia.org/wiki/Eigenvalues_and_eigenvectors

Linearly dependent/independent

- In the theory of vector spaces, a set of vectors is said to be
  - linearly dependent : if at least one of the vectors in the set can be defined as a linear combination of the others
    - $a_1 \underset{v_1}{\rightarrow} + a_2 \underset{v_2}{\rightarrow} + \cdots + a_k \underset{v_k}{\rightarrow} = 0$, where $a_1, a_2, \ldots, a_k$ are non−zeros
  - linearly independent : if no vector in the set can be written in this way

Credit by https://en.wikipedia.org/wiki/Linear_independence

Orthogonal vectors

- 2 vectors are orthogonal if they are perpendicular to each other
- the dot product of the two vectors is zero
  - $\begin{pmatrix} 1 \\ 0 \\ -1 \end{pmatrix} \begin{pmatrix} 1 \\ \sqrt{2} \\ 1 \end{pmatrix} \begin{pmatrix} 1 \\ -\sqrt{2} \\ 1 \end{pmatrix}$

Principal Component Analysis (PCA)

Eigenvalues and eigenvectors

- linear transformation of *n*-dimensional vectors defined by an *n* x *n* matrix *A*

$$Av = w$$

$$\begin{bmatrix} A_{11} & \cdots & A_{1n} \\ \vdots & \ddots & \vdots \\ A_{n1} & \cdots & A_{nn} \end{bmatrix} \begin{bmatrix} v_1 \\ \vdots \\ v_n \end{bmatrix} = \begin{bmatrix} w_1 \\ \vdots \\ w_n \end{bmatrix}$$

n x n n x 1 n x 1

Eigenvalues and eigenvectors

- If it occurs that *v* and *w* are scalar multiples, that is if

$$Av = w = \lambda v$$

  - *v* is an **eigenvector** of the linear transformation *A*
  - the scale factor λ is the **eigenvalue** corresponding to that eigenvector
  - If $\lambda_1,\ldots, \lambda_k$ are different, then $v_1,\ldots,v_k$ are linearly independent

THEOREM 7.6 Sufficient Condition for Diagonalization

If an $n \times n$ matrix A has n *distinct* eigenvalues, then the corresponding eigenvectors are linearly independent and A is diagonalizable.

PROOF

Let $\lambda_1, \lambda_2, \ldots, \lambda_n$ be n distinct eigenvalues of A with corresponding eigenvectors $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n$. To begin, assume the set of eigenvectors is linearly dependent. Moreover, consider the eigenvectors to be ordered so that the first m eigenvectors are linearly independent, but the first $m + 1$ are linearly dependent, where $m < n$. Then $\mathbf{x}_{m+1}$ can be written as a linear combination of the first m eigenvectors:

$$\mathbf{x}_{m+1} = c_1\mathbf{x}_1 + c_2\mathbf{x}_2 + \cdots + c_m\mathbf{x}_m \qquad \text{Equation 1}$$

where the c_i's are not all zero. Multiplication of both sides of Equation 1 by A yields

$$A\mathbf{x}_{m+1} = Ac_1\mathbf{x}_1 + Ac_2\mathbf{x}_2 + \cdots + Ac_m\mathbf{x}_m.$$

Now $A\mathbf{x}_i = \lambda_i\mathbf{x}_i$, $i = 1, 2, \ldots, m + 1$, so you have

$$\lambda_{m+1}\mathbf{x}_{m+1} = c_1\lambda_1\mathbf{x}_1 + c_2\lambda_2\mathbf{x}_2 + \cdots + c_m\lambda_m\mathbf{x}_m. \qquad \text{Equation 2}$$

Multiplication of Equation 1 by λ_{m+1} yields

$$\lambda_{m+1}\mathbf{x}_{m+1} = c_1\lambda_{m+1}\mathbf{x}_1 + c_2\lambda_{m+1}\mathbf{x}_2 + \cdots + c_m\lambda_{m+1}\mathbf{x}_m. \qquad \text{Equation 3}$$

Subtracting Equation 2 from Equation 3 produces

$$c_1(\lambda_{m+1} - \lambda_1)\mathbf{x}_1 + c_2(\lambda_{m+1} - \lambda_2)\mathbf{x}_2 + \cdots + c_m(\lambda_{m+1} - \lambda_m)\mathbf{x}_m = \mathbf{0}$$

and, using the fact that the first m eigenvectors are linearly independent, all coefficients of this equation must be zero. That is,

$$c_1(\lambda_{m+1} - \lambda_1) = c_2(\lambda_{m+1} - \lambda_2) = \cdots = c_m(\lambda_{m+1} - \lambda_m) = 0.$$

All the eigenvalues are distinct, so it follows that $c_i = 0$, $i = 1, 2, \ldots, m$. But this result contradicts our assumption that $\mathbf{x}_{m+1}$ can be written as a linear combination of the first m eigenvectors. So, the set of eigenvectors is linearly independent, and from Theorem 7.5, you can conclude that A is diagonalizable.

Two-dimensional matrix example

- $Av = w = \lambda v$
- $(A - \lambda I)v = 0$
- Given $A = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$, what are eigenvalues and eigenvectors?

$$|A - \lambda I| = \begin{vmatrix} 2-\lambda & 1 \\ 1 & 2-\lambda \end{vmatrix} = 3 - 4\lambda + \lambda^2 = 0$$

$$v_{\lambda=1} = \begin{bmatrix} 1 \\ -1 \end{bmatrix}, \; v_{\lambda=3} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

Two-dimensional matrix example

- Given $A = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$, then $v_{\lambda=1} = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$, $v_{\lambda=3} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$

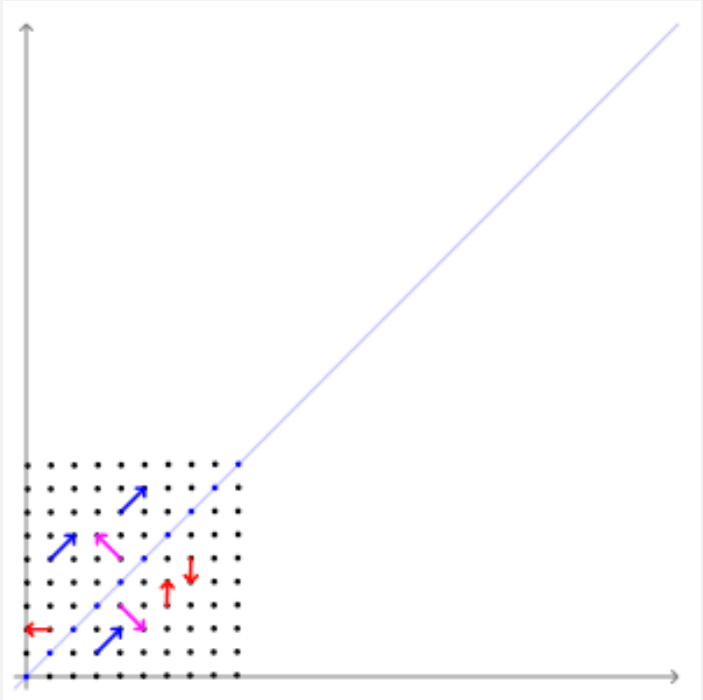
A υ λ υ

- $Av_{\lambda=1} = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \end{bmatrix} = \begin{bmatrix} 1 \\ -1 \end{bmatrix} = 1 \begin{bmatrix} 1 \\ -1 \end{bmatrix}$

- $Av_{\lambda=3} = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 3 \\ 3 \end{bmatrix} = 3 \begin{bmatrix} 1 \\ 1 \end{bmatrix}$

Two-dimensional matrix example

- Given $A = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$, then $v_{\lambda=1} = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$, $v_{\lambda=3} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$



Credit by https://en.wikipedia.org/wiki/Eigenvalues_and_eigenvectors

https://www.youtube.com/watch?v=HWnCv4iHCDc&list=WL&index=2&t=2s

Scalar form $Av_1 = \lambda_1 v_1$ to Matrix form $AE = ED$

- $Av_1 = \lambda_1 v_1$
- How about $[Av_1 \quad \dots \quad Av_n]$?
- $[Av_1 \quad \dots \quad Av_n] = [\lambda_1 v_1 \quad \dots \quad \lambda_n v_n]$ ... ①
- Suppose
  - $\mathrm{E} = [v_1 \quad \dots \quad v_n]$
  - $\mathrm{D} = \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \ddots \end{bmatrix}$
- $[Av_1 \quad \dots \quad Av_n] = AE$
- $[\lambda_1 v_1 \quad \dots \quad \lambda_n v_n] = [v_1 \quad \dots \quad v_n] \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \ddots \end{bmatrix} = [v_1 \quad \dots \quad v_n]\mathrm{D} = \mathrm{ED}$ ... ②
- $AE = ED$ (∵ ① ②)

Credit: A Tutorial on Principal Component Analysis, Jonathon Shlens

Eigenvector Decomposition

- $AE = ED$
- $A = EDE^{-1}$

Eigenvector Decomposition

- For a symmetric matrix A, $A=EDE^T$, where
  - D is a diagonal matrix
  - E is a matrix of eigenvectors of A arranged as columns
- Based on
  - Theorem1: a matrix is symmetric *if and only if* it is orthogonally diagonalizable
  - Theorem2: a symmetric matrix is diagonalized by a matrix of its orthogonal eigenvectors

Credit: A Tutorial on Principal Component Analysis, Jonathon Shlens

Fundamental Thm. of Symmetric Matrices (L.A. P. 373)

Theorem1: a matrix is symmetric *if and only if* it is orthogonally diagonalizable

- => : if A is orthogonally diagonalizable, then A is a symmetric matrix

if A is orthogonally diagonalizable => $A = EDE^T$, then

$$A^T = (EDE^T)^T = E^{TT}D^TE^T = EDE^T = A$$

A is a symmetric matrix → def: $A = A^T$

A matrix A is orthogonally diagonalizable when there exists an orthogonal matrix P s.t.

$P^{-1}AP = D$ is diagonal

PROOF

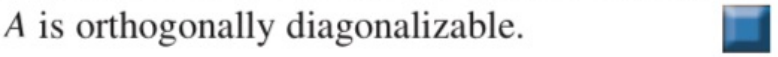
The proof of the theorem in one direction is fairly straightforward. That is, if you assume A is orthogonally diagonalizable, then there exists an orthogonal matrix P such that $D = P^{-1}AP$ is diagonal. Moreover, $P^{-1} = P^T$, so you have

$$\begin{aligned} A &= PDP^{-1} \\ &= PDP^T \end{aligned}$$

which implies that

$$\begin{aligned} A^T &= (PDP^T)^T \\ &= (P^T)^T D^T P^T \\ &= PDP^T \\ &= A. \end{aligned}$$

So, A is symmetric.

The proof of the theorem in the other direction is more involved, but it is important because it is constructive. Assume A is symmetric. If A has an eigenvalue λ of multiplicity k, then by Theorem 7.7, λ has k linearly independent eigenvectors. Through the Gram-Schmidt orthonormalization process, use this set of k vectors to form an orthonormal basis of eigenvectors for the eigenspace corresponding to λ. Repeat this procedure for each eigenvalue of A. The collection of all resulting eigenvectors is orthogonal by Theorem 7.9, and you know from the orthonormalization process that the collection is also orthonormal. Now let P be the matrix whose columns consist of these n orthonormal eigenvectors. By Theorem 7.8, P is an orthogonal matrix. Finally, by Theorem 7.5, $P^{-1}AP$ is diagonal. So, A is orthogonally diagonalizable. 

Theorem2: a symmetric matrix is diagonalized by a matrix of its **orthogonal** eigenvectors

- A symmetric matrix always has orthogonal eigenvectors (LA P.372)
- For a symmetric matrix A, let λ_1 and λ_2 be distinct eigenvalues for eigenvectors e_1 and e_2
  - $\lambda_1 e_1 \cdot e_2 = \lambda_2 e_1 \cdot e_2$
  - $(\lambda_1 - \lambda_2) e_1 \cdot e_2 = 0$
  - $e_1 \cdot e_2 = 0$
  - The eigenvectors of a symmetric matrix are orthogonal.

THEOREM 7.9 Property of Symmetric Matrices

Let A be an $n \times n$ symmetric matrix. If λ_1 and λ_2 are distinct eigenvalues of A, then their corresponding eigenvectors $\mathbf{x}_1$ and $\mathbf{x}_2$ are orthogonal.

PROOF

Let λ_1 and λ_2 be distinct eigenvalues of A with corresponding eigenvectors $\mathbf{x}_1$ and $\mathbf{x}_2$. So, $A\mathbf{x}_1 = \lambda_1\mathbf{x}_1$ and $A\mathbf{x}_2 = \lambda_2\mathbf{x}_2$. To prove the theorem, use the matrix form of the dot product, $\mathbf{x}_1 \cdot \mathbf{x}_2 = \mathbf{x}_1^T\mathbf{x}_2$. (See Section 5.1.) Now you can write

$$\begin{aligned}
\lambda_1(\mathbf{x}_1 \cdot \mathbf{x}_2) &= (\lambda_1\mathbf{x}_1) \cdot \mathbf{x}_2 \\
&= (A\mathbf{x}_1) \cdot \mathbf{x}_2 \\
&= (A\mathbf{x}_1)^T\mathbf{x}_2 \\
&= (\mathbf{x}_1^TA^T)\mathbf{x}_2 \\
&= (\mathbf{x}_1^TA)\mathbf{x}_2 \qquad A \text{ is symmetric, so } A = A^T. \\
&= \mathbf{x}_1^T(A\mathbf{x}_2) \\
&= \mathbf{x}_1^T(\lambda_2\mathbf{x}_2) \\
&= \mathbf{x}_1 \cdot (\lambda_2\mathbf{x}_2) \\
&= \lambda_2(\mathbf{x}_1 \cdot \mathbf{x}_2).
\end{aligned}$$

This implies that $(\lambda_1 - \lambda_2)(\mathbf{x}_1 \cdot \mathbf{x}_2) = 0$. Also, $\lambda_1 \neq \lambda_2$, so it follows that $\mathbf{x}_1 \cdot \mathbf{x}_2 = 0$, which means that $\mathbf{x}_1$ and $\mathbf{x}_2$ are orthogonal.

$E^{-1} = E^T$?

(the def. of an orthogonal matrix)

<Def> Orthogonal Matrix

A square matrix is orthogonal when it is
① invertible ② $P^{-1} = P^T$

- Theorem: the inverse of an orthogonal matrix is its transpose
- $\mathrm{E} = [v_1 \quad \cdots \quad v_n]$, where v_i is the i^{th} columns vector
- The ij^{th} element of E^TE

$$(E^TE)_{ij} = {v_i}^T v_j = \begin{cases} 1 \text{ if } i = j \\ 0 \text{ otherwise} \end{cases}$$ I

$$E^TE = I$$

$$E^{-1} = E^T$$

- $A = EDE^{-1} = EDE^T$

Solving PCA by Eigenvector Decomposition

1. For a symmetric matrix A, $A=EDE^T$, where
   - D is a diagonal matrix
   - E is a matrix of eigenvectors of A arranged as columns.
2. $C_x = EDE^T$
3. We select the matrix P to be a matrix where each row p_i is an eigenvector of C_X
4. Let $Y = PX$, then $\boldsymbol{C_Y = D}$?

Credit: A Tutorial on Principal Component Analysis, Jonathon Shlens

Another Interpretation of Principal Components

- The first principal component loading vector has a very special property
  - it is the line in p-dimensional space that is closest to the n observations (using average squared Euclidean distance as a measure of closeness)
- the first M principal component score vectors and the first M principal component loading vectors provide the best M-dimensional approximation (in terms of Euclidean distance)

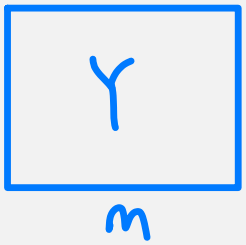
Summary of PCA

- The principal components of X are the eigenvectors of C_X
- The i^{th} diagonal value of C_Y is the variance of X along p_i
- In practice
  - Subtracting off the mean of each measurement type
  - Computing the eigenvectors of C_x

Credit: A Tutorial on Principal Component Analysis, Jonathon Shlens

Single Value Decomposition (SVD)

A more general solution using SVD



- Y be an arbitrary ***n*** **x** ***m*** matrix
- Y^TY be a rank *r*, square, symmetric *m* x *m* matrix

$$(Y^TY)\hat{v}_i = \lambda_i \hat{v}_i$$

mxm mx1 mx1

for Y^TY (

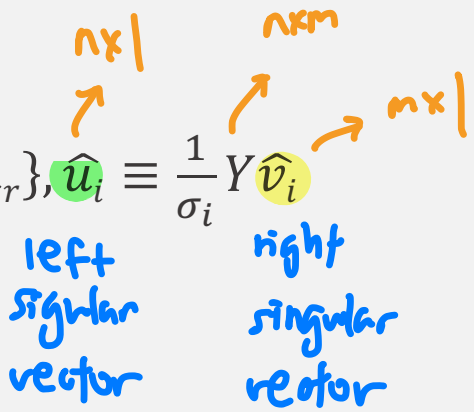
- $\{\hat{v}_1, \hat{v}_2, \ldots, \hat{v}_r\}$: the set of orthonormal *mx1* eigenvectors
- $\{\lambda_1, \lambda_2, \ldots, \lambda_r\}$: associated eigenvalues

A more general solution using SVD

$$(Y^T Y)\hat{v}_i = \lambda_i \hat{v}_i$$

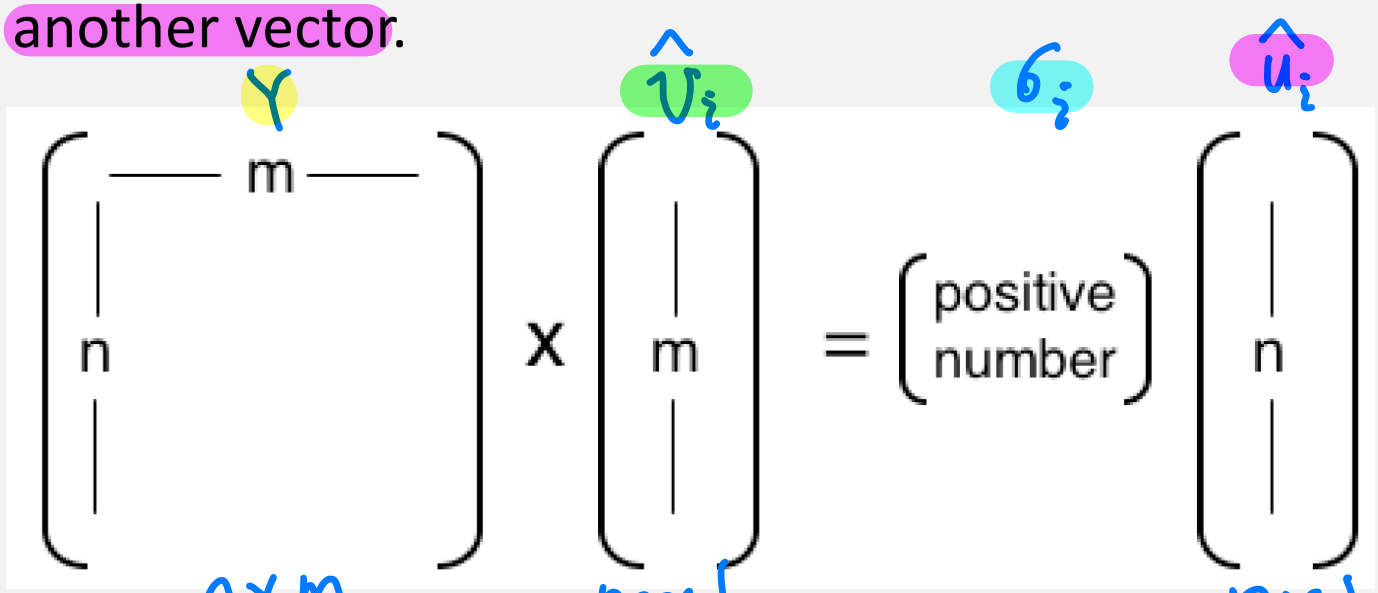
- $\{\hat{v}_1, \hat{v}_2, \ldots, \hat{v}_r\}$: the set of orthonormal *mx1* eigenvectors
- $\{\lambda_1, \lambda_2, \ldots, \lambda_r\}$: associated eigenvalues

- *singular values* : positive real, $\sigma_i = \sqrt{\lambda_i}$
- the set of *n* x 1 vectors defined $\{\hat{u}_1, \ldots, \hat{u}_r\}, \hat{u}_i \equiv \frac{1}{\sigma_i} Y \hat{v}_i$



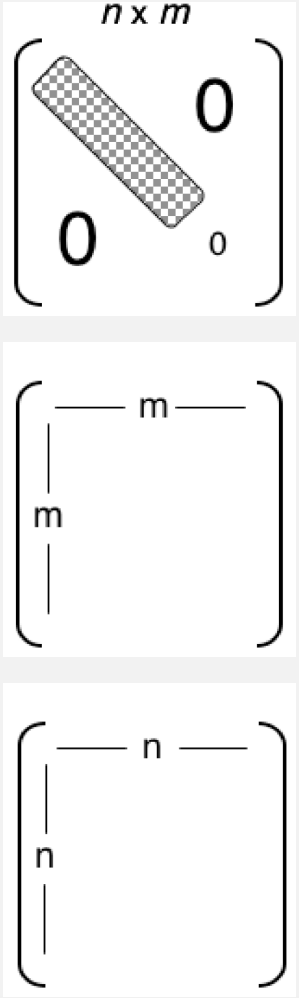
The scalar form of SVD

- $Y\hat{v}_i = \sigma_i \times \left(\frac{1}{\sigma_i} Y\hat{v}_i\right) = \sigma_i\, \hat{u}_i$
  - where $\hat{u}_i \equiv \frac{1}{\sigma_i} Y\hat{v}_i$
- Y multiplied by an eigenvector of Y^TY = a scalar times another vector.



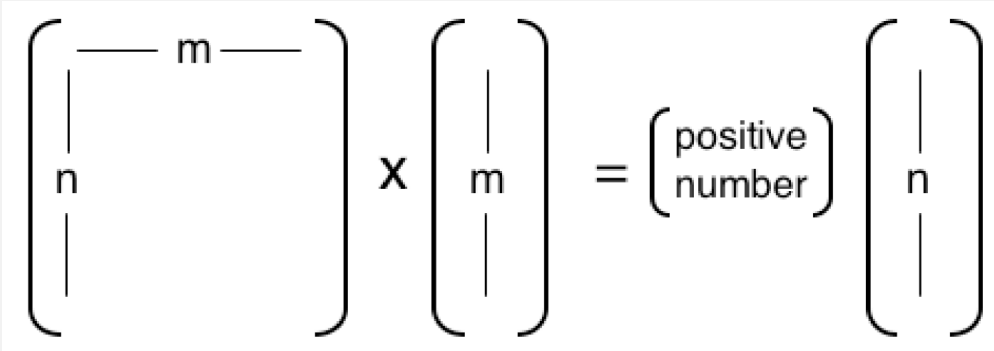
Scalar form $Y\hat{v}_i = \sigma_i \hat{u}_i$ => Matrix form $YV = U\Sigma$

- $\sum \equiv \begin{bmatrix} \sigma_{\tilde{1}} & 0 & 0 \\ 0 & \sigma_{\tilde{2}} & \vdots \\ 0 & \cdots & \ddots \end{bmatrix}$
  - $\sigma_{\tilde{1}} \geq \sigma_{\tilde{2}} \geq \cdots \geq \sigma_{\tilde{r}}$
- $V = [\hat{v}_{\tilde{1}}\ \hat{v}_{\tilde{2}} \cdots \hat{v}_{\tilde{m}}]$
- $U = [\hat{u}_{\tilde{1}}\ \hat{u}_{\tilde{2}} \cdots \hat{u}_{\tilde{n}}]$

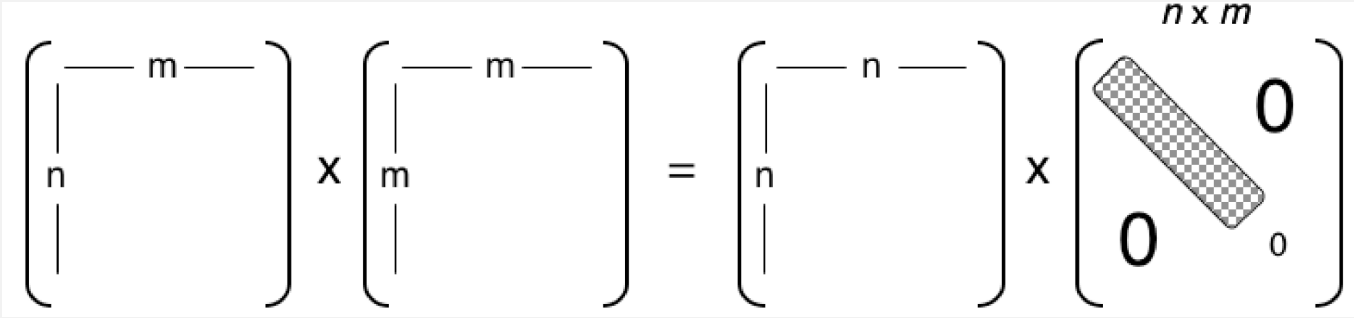


Construction of the matrix form of SVD from the scalar form

- $Y\hat{v}_i = \sigma_i \hat{u}_i$



- $YV = U\Sigma$

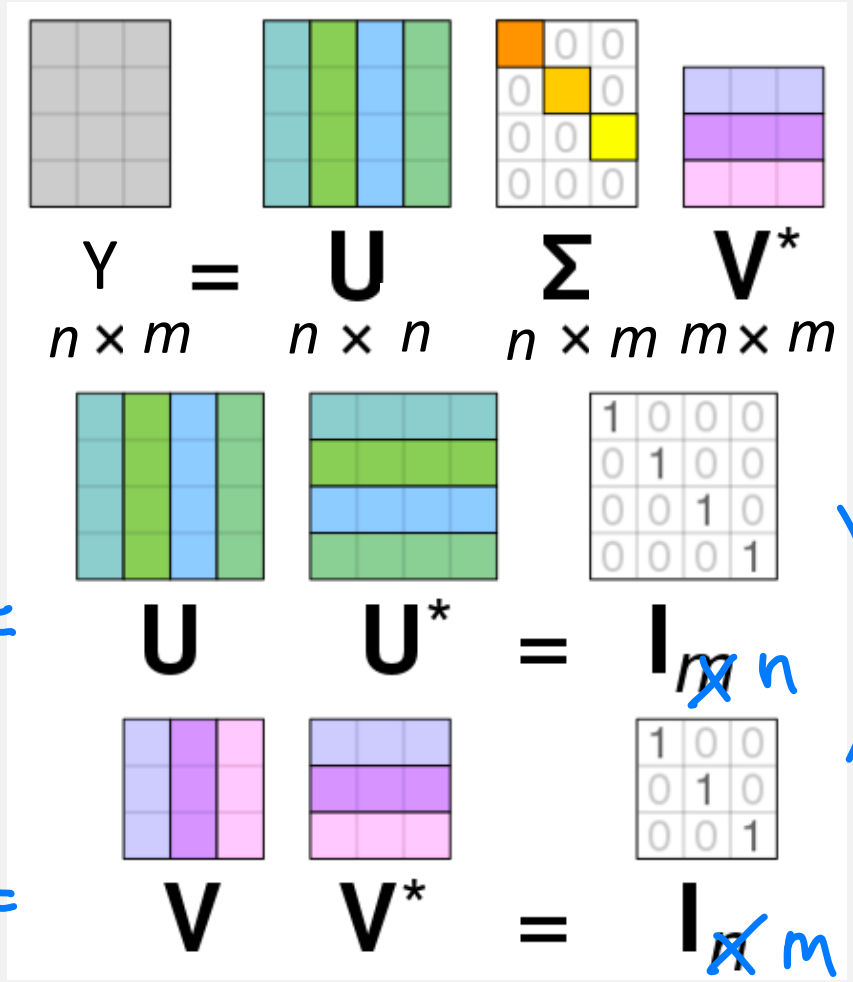


Credit: A Tutorial on Principal Component Analysis, Jonathon Shlens

Single Value Decomposition (SVD)

- $YV = U\Sigma$ ⇒ $Y = U\Sigma V^{-1}$
- $V^{-1} = V^T$
- $Y = U\Sigma V^T$
- any arbitrary matrix *Y* can be converted to
  - an orthogonal matrix
  - a diagonal matrix
  - another orthogonal matrix
  - a rotation, a stretch and a second rotation

Visualization of the matrix multiplications in SVD



$U^* U = I_n$

$V^* V = I_m$

Properties of orthogonal (unitary) matrices

$AA^T = I$

$A^{-1} = A^T$

Interpreting SVD : column space

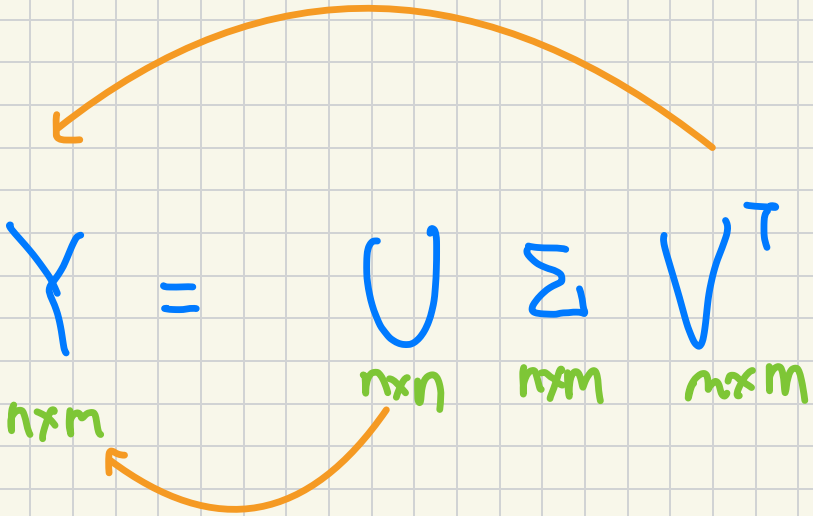
- $Y = U\Sigma V^T$
- $U^T Y = \Sigma V^T \equiv Z$
  - $U = [\hat{u}_{\tilde{1}}\ \hat{u}_{\tilde{2}} \cdots \hat{u}_{\tilde{n}}]$, rows in U^T ($\hat{U}_i$: principal components)
  - $PX = Y$, P : each row p_i is an eigenvector of C_X
- U^T is as change of basis from Y to Z (linear transformation matrix)
- U^T is a basis that spans the columns of Y (the column space of Y) (n-dimensional space)

Interpreting SVD : row space

- $Y = U\Sigma V^T$
- $YV = U\Sigma$
- $(YV)^T = (U\Sigma)^T$
- $V^T Y^T = \Sigma U^T \equiv Z$
- V^T is as change of basis from Y^T to Z
- V^T is a basis that spans the rows of Y (the row space of Y)

$$V^T Y^T = \Sigma U^T$$

V^T Spans the row space of Y



U^T Spans the column space of Y

$$U^T Y = \Sigma V^T$$

Why Y be an arbitrary n x m matrix?

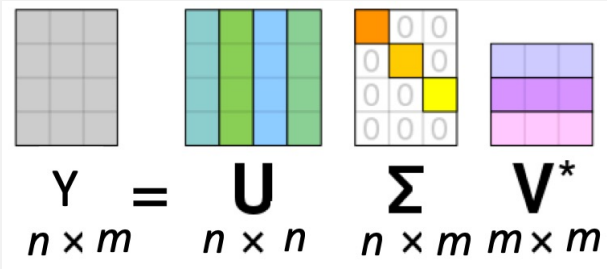
- The original m x n matrix X
- Why define a new matrix n x m, $Y \equiv \frac{1}{\sqrt{n}} X^T$?
- $Y^T Y = \left(\frac{1}{\sqrt{n}} X^T\right)^T \frac{1}{\sqrt{n}} X^T$

$$= \frac{1}{\sqrt{n}} X \frac{1}{\sqrt{n}} X^T$$

$$= \frac{1}{n} X X^T$$

$$= C_X$$

SVD and PCA

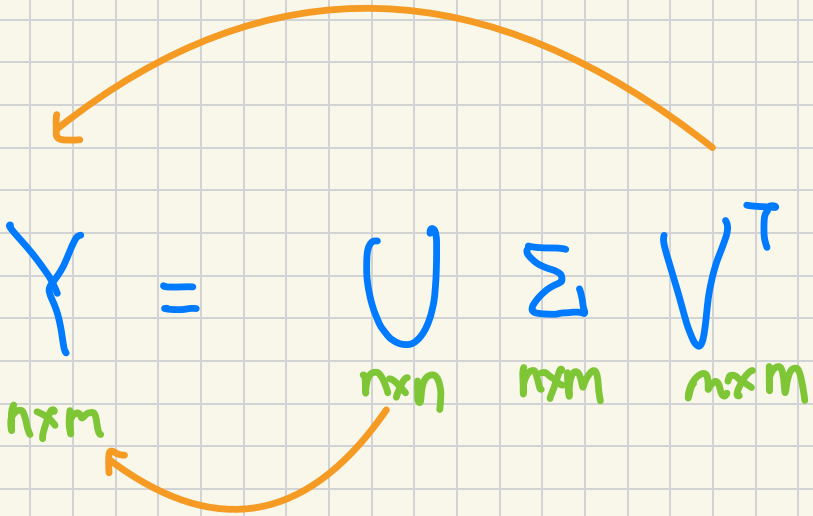


- SVD
  - $Y = U\Sigma V^T$
  - The columns of matrix *V* contain the eigenvectors of Y^TY
- PCA
  - The principal components of *X* = the eigenvectors of C_X
- $Y^TY = C_X$
- The columns of matrix *V* are the principal components of *X*

(Review)

$$V^T Y^T = \Sigma U^T$$

V^T Spans the row space of Y



U^T Spans the column space of Y

$$U^T Y = \Sigma V^T$$

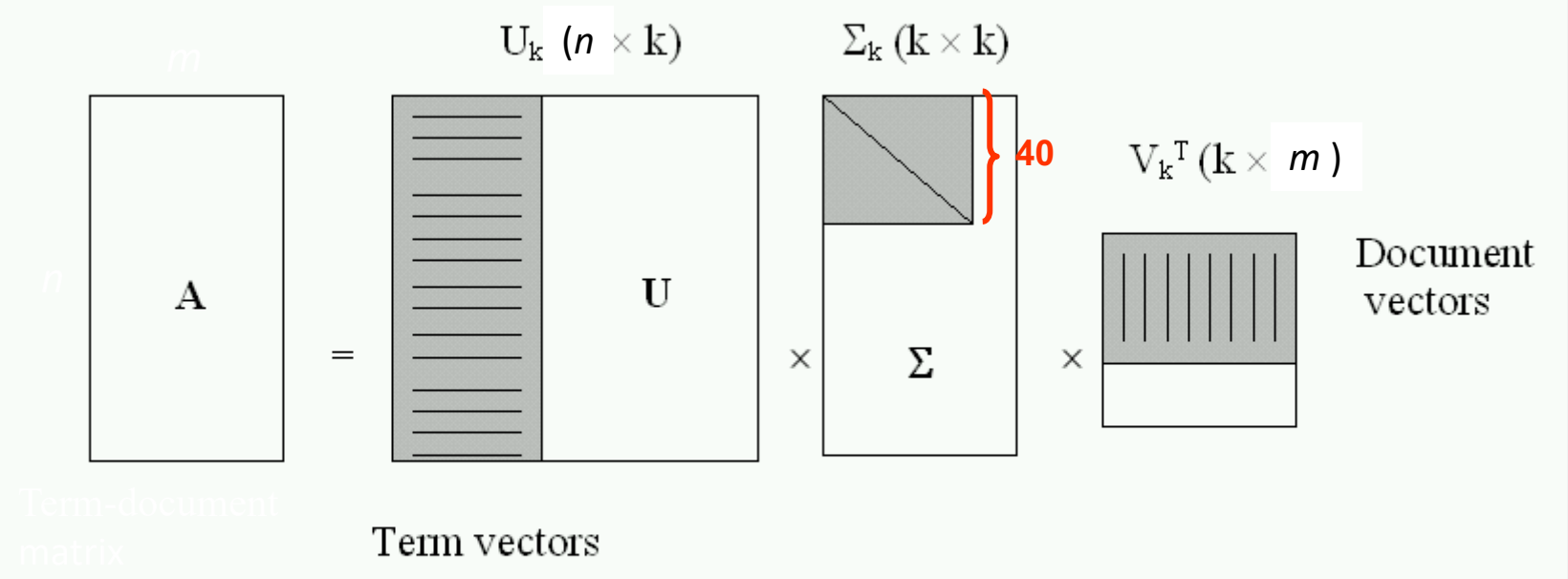
SVD and PCA

- V^T spans the row space of $Y \equiv \frac{1}{\sqrt{n}} X^T$
- V^T spans the column space of $\frac{1}{\sqrt{n}} X$
- Find the principal components = find an orthonormal basis that spans the column space of X

$$\begin{cases} Y = U \Sigma V^T \\ Y = \frac{1}{\sqrt{n}} X^T \Rightarrow Y^T Y = C_X \end{cases}$$

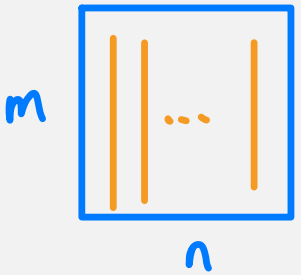
Singular Value Decomposition

- $Y = U\Sigma V^T$
- Reduced feature size = 40 features



Quick Summary of PCA

1. Organize data as an m x n matrix, X
   - m : the number of measurement types
   - n : the number of samples



2. Subtract off the mean for each measurement type
3. Calculate the SVD on $\frac{1}{\sqrt{n}} X^T$
4. the eigenvectors of the covariance C_X

Assumptions behind PCA

- Linearity
- Large variances have important structure
- The principal components are orthogonal

Credit: A Tutorial on Principal Component Analysis, Jonathon Shlens

PCA practice

pca.R

```
# Load data
data(iris)
head(iris, 3)
# log transform
log.ir <- log(iris[, 1:4])
ir.species <- iris[, 5]
# apply PCA - scale. = TRUE is highly advisable, but
default is FALSE.
ir.pca <- prcomp(log.ir,center = TRUE, scale. = TRUE)
```

Credit by thiagogm, https://www.r-bloggers.com/computing-and-visualizing-pca-in-r/

`prcomp` function

- Performs a principal components analysis on the given data matrix
  - The calculation is done by a *singular value decomposition* of the (centered and possibly scaled) data matrix, not by using *eigen on the covariance matrix*.
    - Center: a logical value indicating whether the variables should be shifted to be zero centered, i.e. the mean of a variable is subtracted from each observation of that variable.
    - Scale: a logical value indicating whether the variables should be scaled to have unit variance before the analysis takes place (i.e. a variance of 1, and hence a standard deviation of 1).

Scaling the Variables

- The variance of Murder, Rape, Assault, UrbanPop = 18. 97, 87. 73, 6945. 16, and 209. 5, respectively.

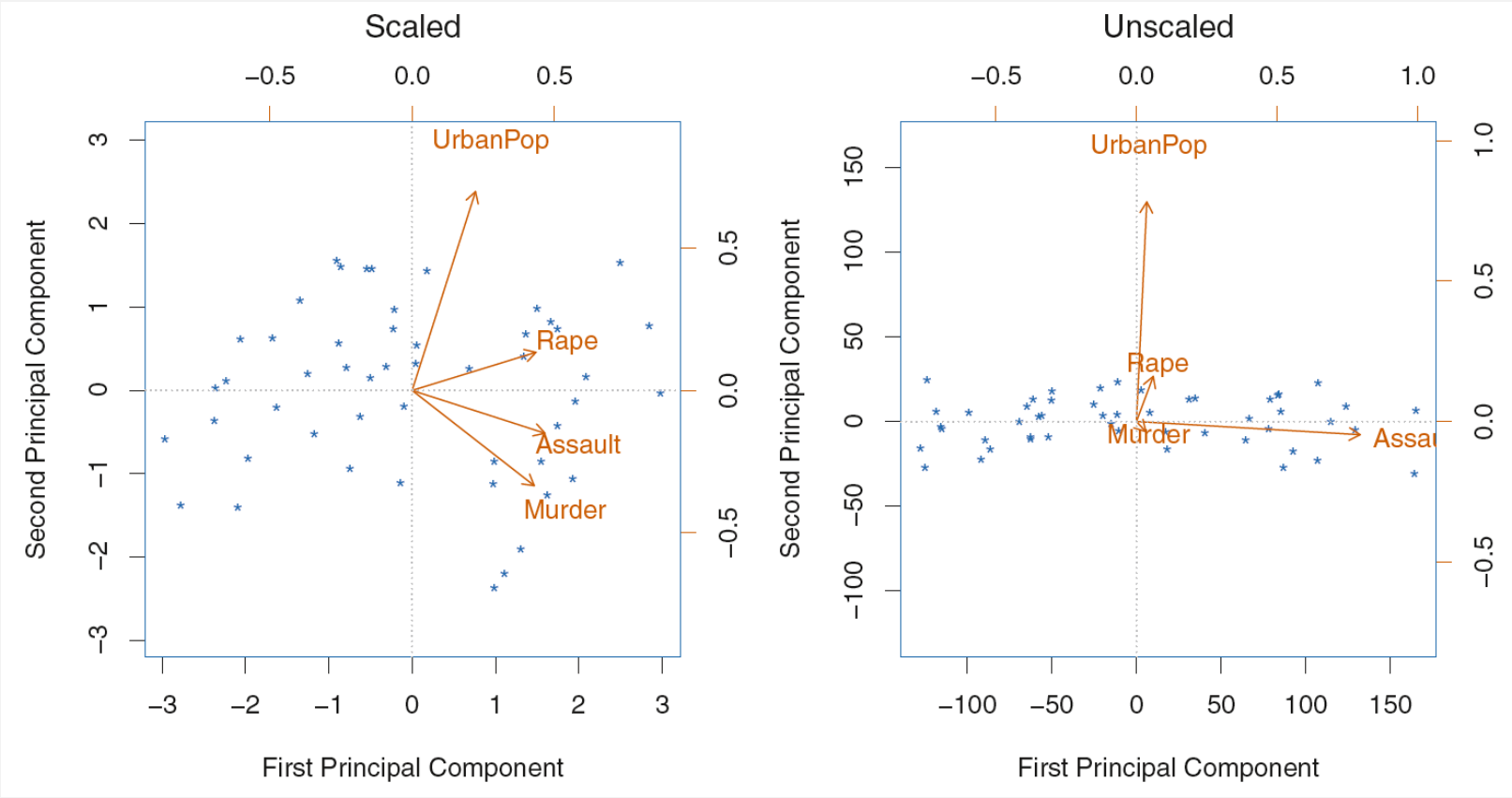


Figure 10.3, *An Introduction to Statistical Learning with Applications in R* by Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani

The class of `prcomp`

`print(ir.pca)`

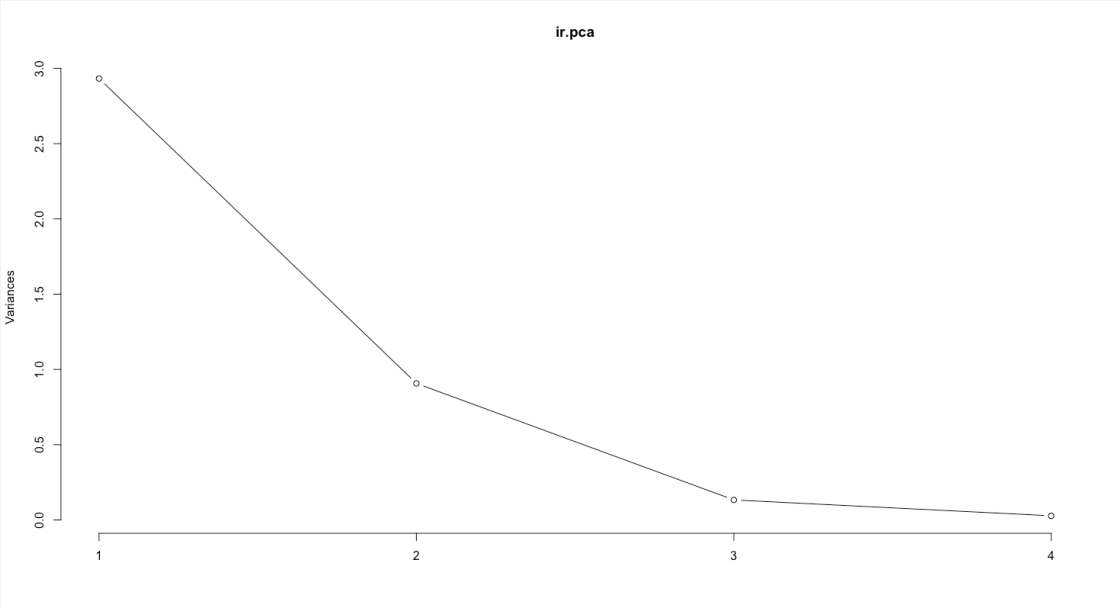
- Sdev: the standard deviations of the principal components (i.e., the square roots of the eigenvalues of the covariance/correlation matrix).
- Rotation: the matrix of variable loadings (i.e., a matrix whose columns contain the eigenvectors).

```
Rotation:
                    PC1         PC2        PC3         PC4
Sepal.Length  0.5038236 -0.45499872  0.7088547  0.19147575
Sepal.Width  -0.3023682 -0.88914419 -0.3311628 -0.09125405
Petal.Length  0.5767881 -0.03378802 -0.2192793 -0.78618732
Petal.Width   0.5674952 -0.03545628 -0.5829003  0.58044745
```

PCA : Analyzing the results

```
# summary method
summary(ir.pca)
# plot method
plot(ir.pca, type = "l")
```

Scree plot



proportion of variance explained (PVE)

ϕ_{jm}: the loading of the j-th variable on the m-th principal component

x_{ij}: the value of the j-th variable for the i-th observation

- $\text{PVE} = \dfrac{\sum_{i=1}^{n}\left(\sum_{j=1}^{p}\phi_{jm}x_{ij}\right)^2}{\sum_{j=1}^{p}\sum_{i=1}^{n}x_{ij}^2}$

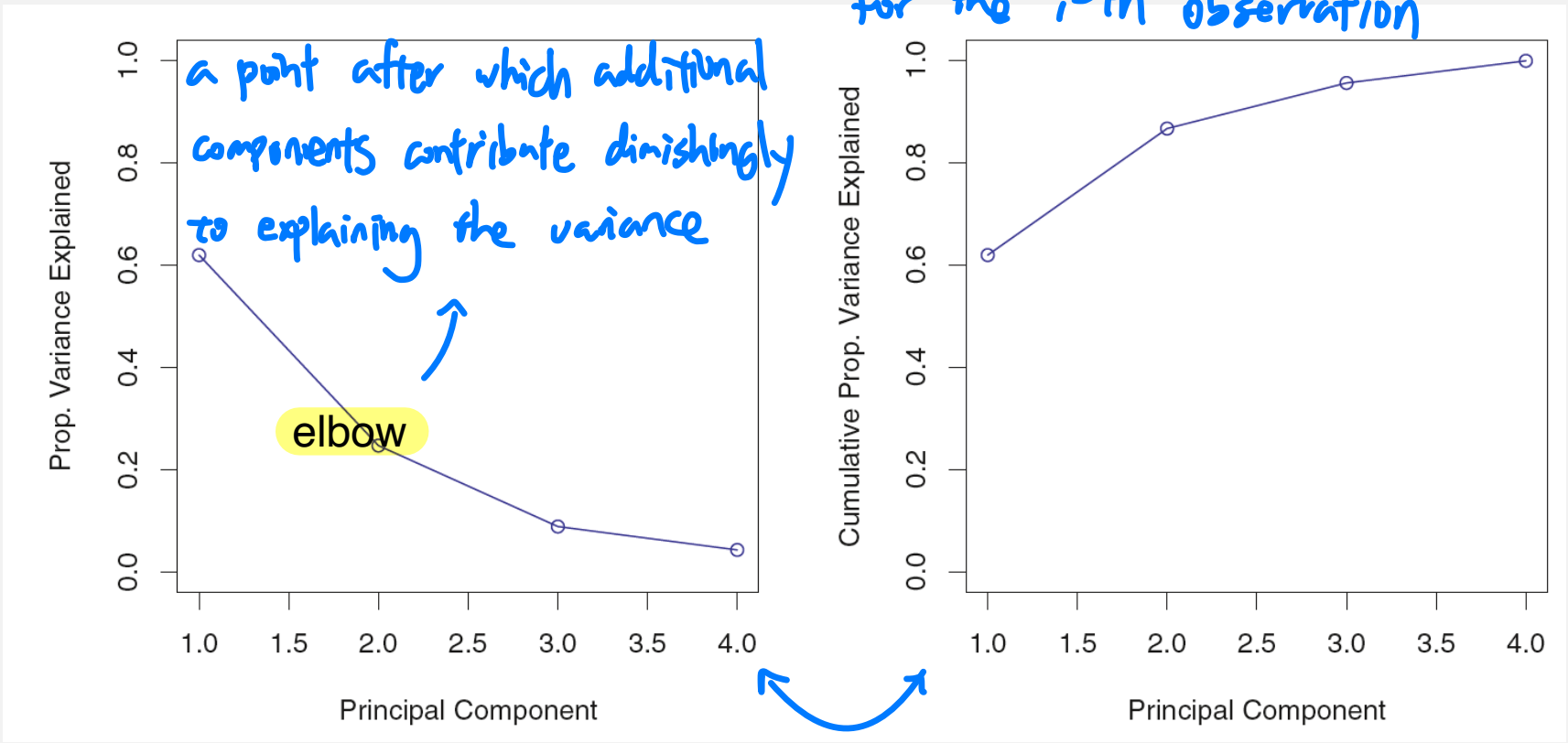
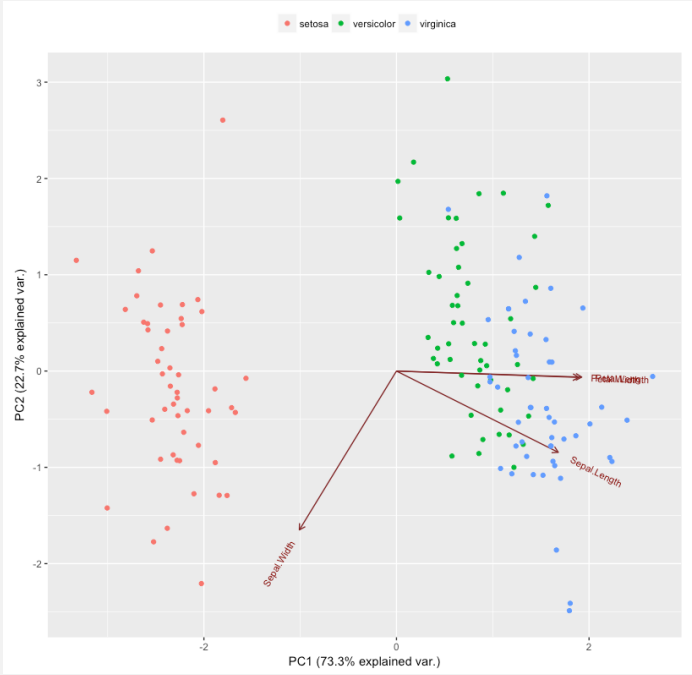


Figure 10.4, *An Introduction to Statistical Learning with Applications in R* by Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani

Biplot for Principal Components using ggplot2

```
library(ggbiplot)
g <- ggbiplot(ir.pca, obs.scale = 1, var.scale = 1, groups = ir.species)
g <- g + scale_color_discrete(name = '')
g <- g + theme(legend.direction = 'horizontal', legend.position = 'top')
print(g)
```



Credit by thiagogm, https://www.r-bloggers.com/computing-and-visualizing-pca-in-r/

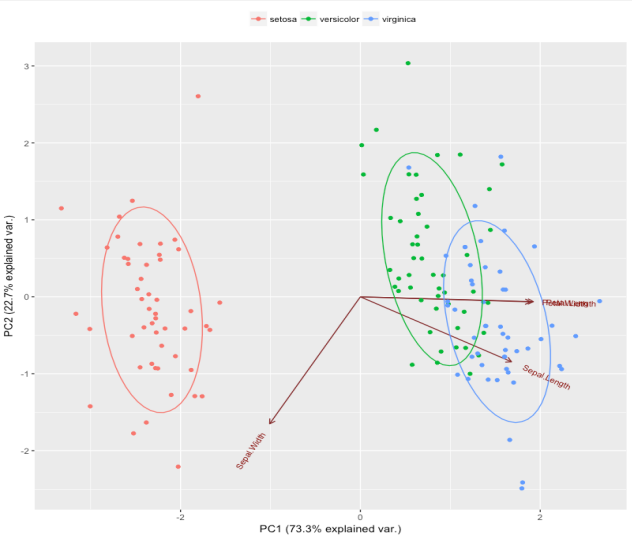
PCA : visualization

```
Rotation:
                PC1         PC2        PC3         PC4
Sepal.Length  0.5038236 -0.45499872  0.7088547  0.19147575
Sepal.Width  -0.3023682 -0.88914419 -0.3311628 -0.09125405
Petal.Length  0.5767881 -0.03378802 -0.2192793 -0.78618732
Petal.Width   0.5674952 -0.03545628 -0.5829003  0.58044745
```

Credit by thiagogm, https://www.r-bloggers.com/computing-and-visualizing-pca-in-r/

PCA : visualization

```
g <- ggbiplot(ir.pca, obs.scale = 1, var.scale = 1, groups = ir.species,
ellipse = TRUE, circle = TRUE)
g <- g + scale_color_discrete(name = '')
g <- g + theme(legend.direction = 'horizontal',legend.position = 'top')
print(g)
#ellipse: draw a normal data ellipse for each groupellipse.prob = 0.68, why?
```



PCA : visualization

```
g <- ggbiplot(ir.pca, obs.scale = 1, var.scale = 1,
              groups = ir.species, ellipse = TRUE,
              circle = TRUE)
g <- g + scale_color_discrete(name = '')
g <- g + theme(legend.direction = 'horizontal',
               legend.position = 'top')
print(g)
```

- circle: draw a correlation circle (only applies when `prcomp` was called with scale = TRUE and when var.scale = 1)
  - sum(ir.pca$rotation[1,]*ir.pca$rotation[1,])

Credit by thiagogm, https://www.r-bloggers.com/computing-and-visualizing-pca-in-r/

PCA for prediction

```
# Predict PCs
predict(ir.pca, newdata=tail(log.ir, 2))
```

USArrests

- For each of the 50 states in the United States, the data set contains the number of arrests per 100,000 residents for each of three crimes: Assault , Murder, and Rape. We also record UrbanPop (the percent of the population in each state livingin urban areas).
- $n = 50$, $p = 4$

biplot display

plotting rows & columns in the same space

- principal component scores + principal component loadings

	PC1	PC2
Murder	0.5358995	−0.4181809
Assault	0.5831836	−0.1879856
UrbanPop	0.2781909	0.8728062
Rape	0.5434321	0.1673186

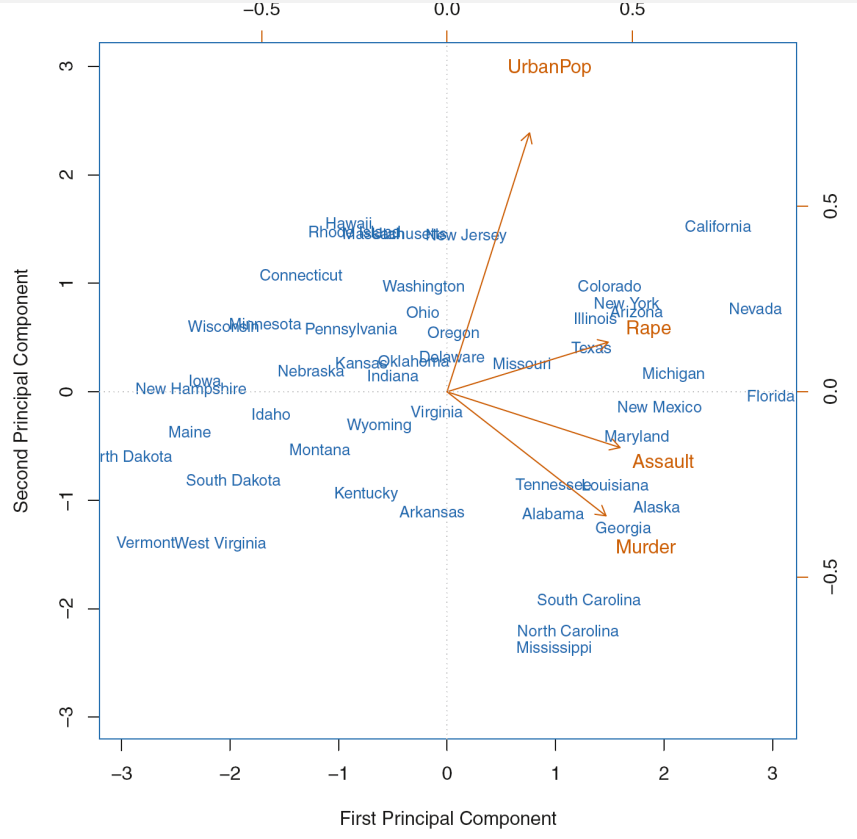
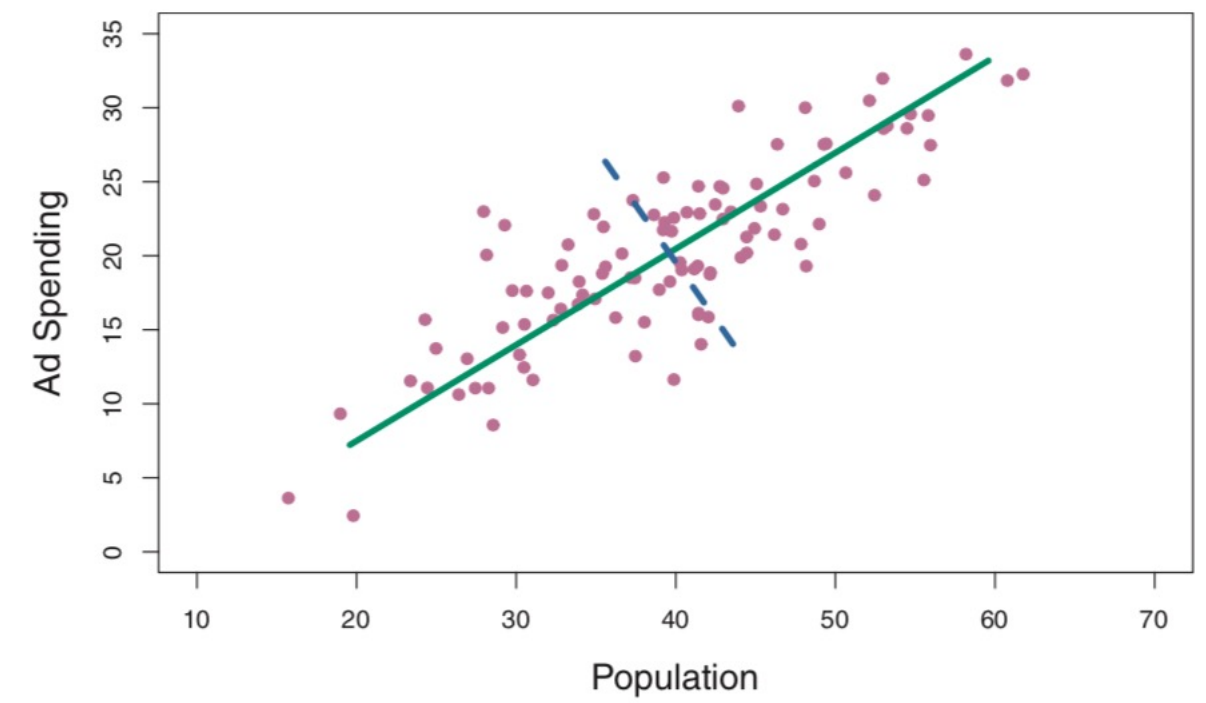


Figure10.1, *An Introduction to Statistical Learning with Applications in R* by Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani



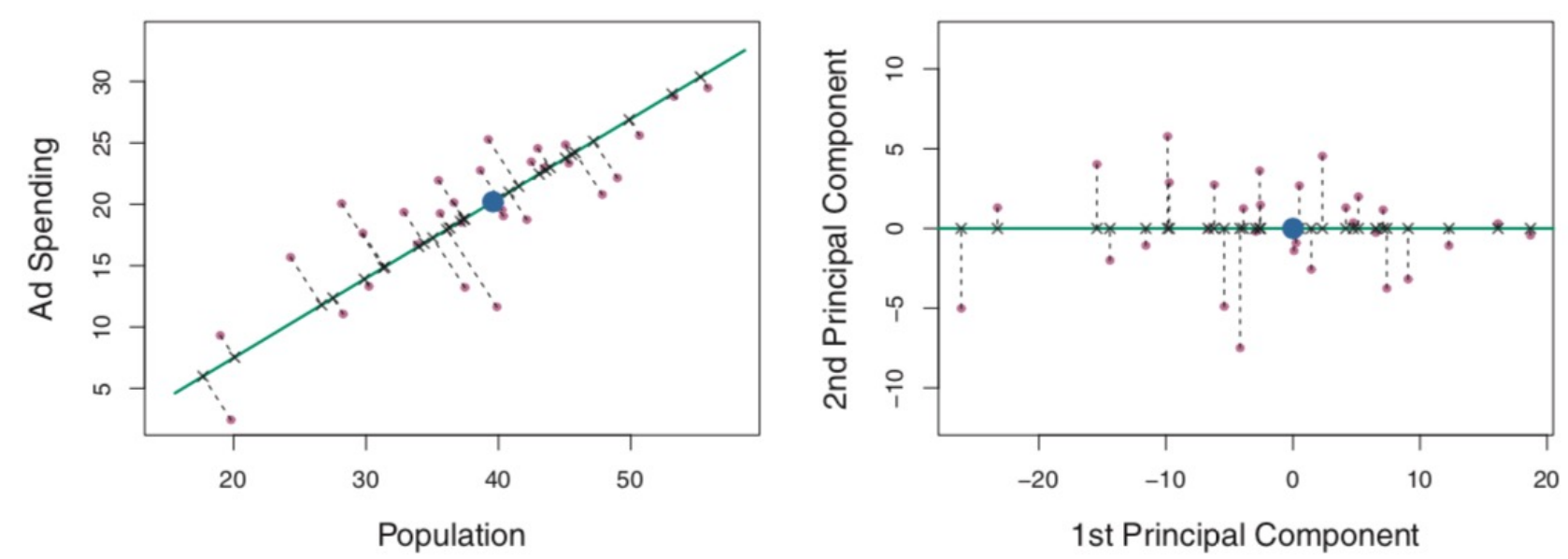
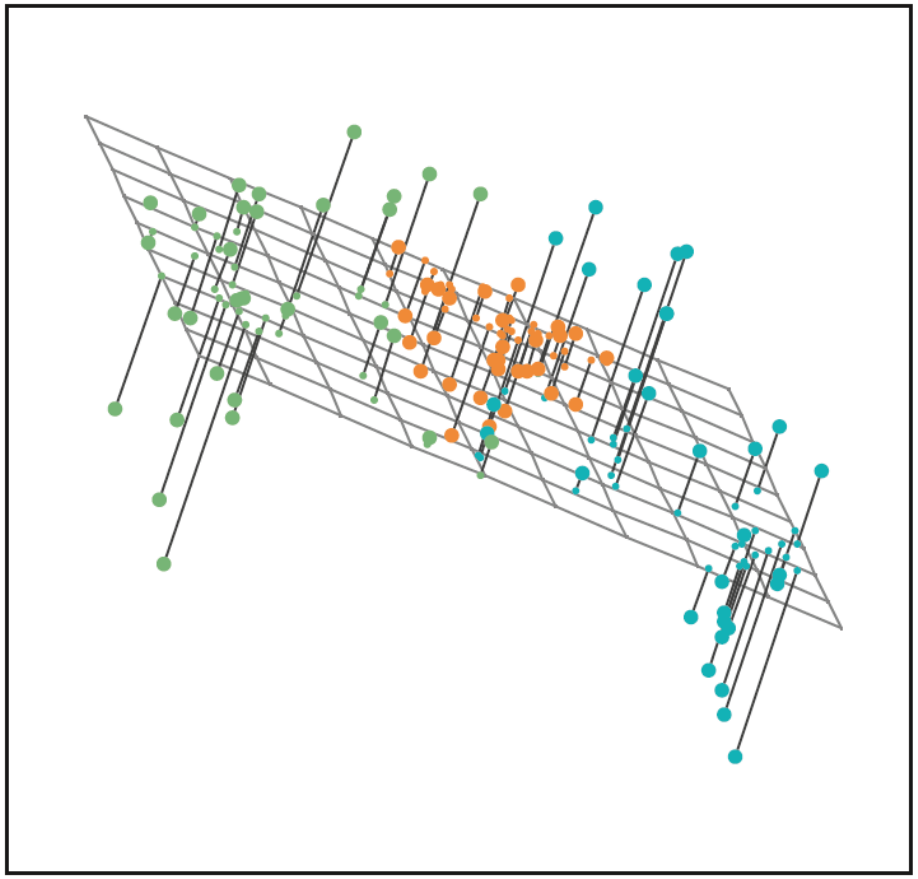


Figure6.14 & 6.15, *An Introduction to Statistical Learning with Applications in R* by Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani



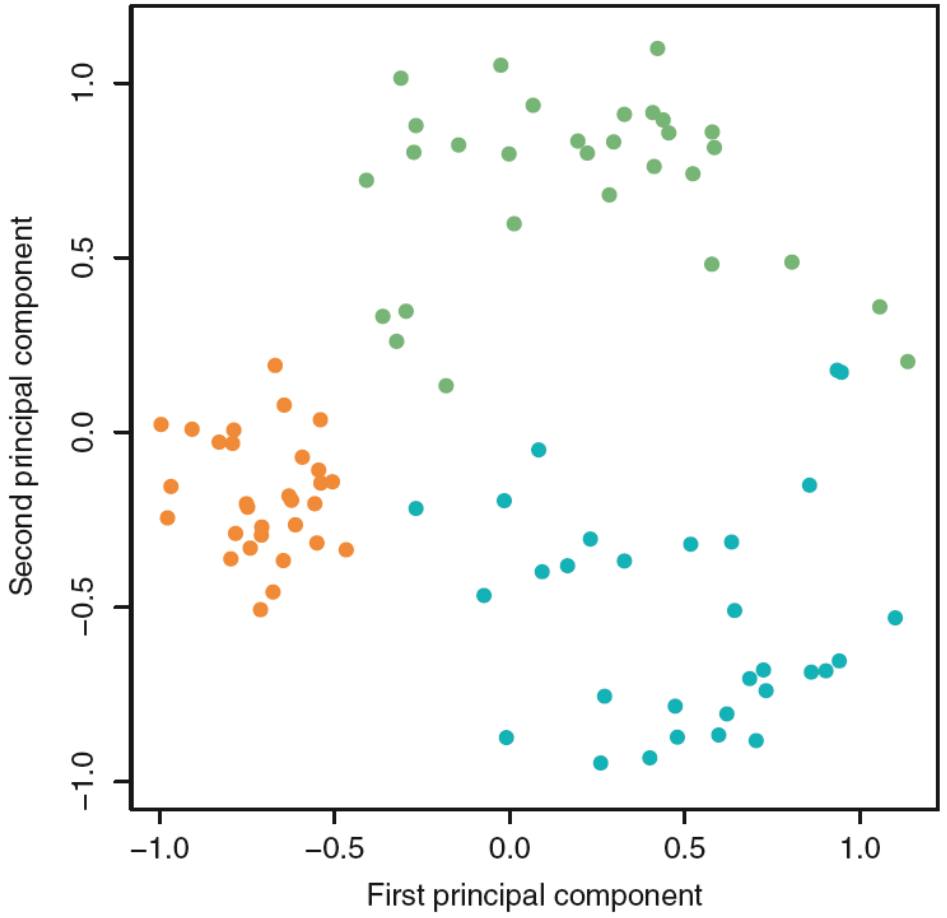


Figure 10.2, *An Introduction to Statistical Learning with Applications in R* by Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani

Correspondence Analysis (CA)

How writers punctuate?

Table 1: The punctuation marks of six French writers (from Brunet, 1989).

	Period	Comma	All the other marks
Rousseau	7836	13112	6026
Chateaubriand	53655	102383	42413
Hugo	115615	184541	59226
Zola	161926	340479	62754
Proust	38177	105101	12670
Giraudoux	46371	58367	14299

Aloz = [2699 5675 1046]

Credit : HervLe Abdi & Lynne J. Williams, Correspondence Analysis

The matrix of deviations

$$\begin{bmatrix} .2905 & .4861 & .2234 \\ .2704 & .5159 & .2137 \\ .3217 & .5135 & .1648 \\ .2865 & .6024 & .1110 \\ .2448 & .6739 & .0812 \\ .3896 & .4903 & .1201 \end{bmatrix}$$

- row profile, $R =$
- Barycenter, $C^T = [.2973 \ .5642 \ .1385]$
- matrix of deviations, $Y = R - (1_{I\times 1} \times C^T)$

$$\begin{bmatrix} -.0068 & -.0781 & .0849 \\ -.0269 & -.0483 & .0752 \\ .0244 & -.0507 & .0263 \\ -.0107 & .0382 & -.0275 \\ -.0525 & .1097 & -.0573 \\ .0923 & -.0739 & -.0184 \end{bmatrix}$$

1. **Row Profile,** R: The row profile matrix consists of proportions that describe how the data in each row is distributed across the columns. Each element in the row profile is computed by dividing the value of a cell by the total of its row.
2. **Barycenter,** C^T: The barycenter or centroid of the data (sometimes called the column average or profile) is a vector that contains the averages of each column. It represents the "average" row profile and is used as a reference point to measure deviations.

1. R: This is the row profile matrix. It's composed of proportions that show how the data in each row is distributed across the columns. Each element in R at position (i, j) is calculated as $\frac{O_{ij}}{\sum_j O_{ij}}$, where O_{ij} is the observed frequency in the cell, and $\sum_j O_{ij}$ is the total of the i-th row.
2. C^T: This is the transpose of the column centroid vector C, which represents the average of each column. Each component j of C is $\frac{\sum_i O_{ij}}{n}$, where n is the grand total of all observations.
3. $1_{Ix1} \times C^T$: The notation 1_{Ix1} indicates a column vector of ones with I rows, where I is the number of rows in R. When this vector is multiplied by the transpose of C (which is a row vector), the result is a matrix where each row is a replica of C^T. This operation essentially broadcasts the average column profile across all rows, creating a matrix of the same size as R.
4. **Subtraction** $R - (1_{Ix1} \times C^T)$: The matrix of deviations Y is obtained by subtracting the matrix that replicates the average column profiles from the original row profiles. This subtraction is performed element-wise. The resulting matrix Y reveals how each row deviates from the column averages. A negative value in Y means the observed proportion in R is less than the average proportion, while a positive value suggests it is greater.

This deviation matrix Y emphasizes differences from the average behavior, highlighting anomalies or particularities in the data which are critical for further analysis, such as through singular value decomposition (SVD) to identify major patterns and dimensions that summarize these deviations.

In the language of CA, the sums of the rows and the columns of the table of proportions are called masses.

Masses (rows)

Observation: $\begin{bmatrix}1 & 1\end{bmatrix}\begin{bmatrix}2 & 4\\ 6 & 8\end{bmatrix} = \begin{bmatrix}2+6 & 4+8\end{bmatrix}$, $\begin{bmatrix}2 & 4\\ 6 & 8\end{bmatrix}\begin{bmatrix}1\\ 1\end{bmatrix} = \begin{bmatrix}2+4\\ 6+8\end{bmatrix}$

Summarize columns — Summarize rows

- the mass of each row is the proportion of this row in the total of the table.
- => The mass of each row reflects its importance in the sample

$(X : I \times J)$

$$\mathbf{m} = \underbrace{\left(\underset{1\times I}{\mathbf{1}} \times \mathbf{X} \times \underset{J\times 1}{\mathbf{1}}\right)^{-1}}_{\text{Inverse of the total of } \mathbf{X}} \times \underbrace{\underset{J\times 1}{\mathbf{X}\,\mathbf{1}}}_{\text{Total of the rows of } \mathbf{X}} = \begin{bmatrix}.0189 & .1393 & .2522 & .3966 & .1094 & .0835\end{bmatrix}^T$$

1×1 — reciprocal — $I\times 1$ — $I\times 1$

Summarize rows + summarize columns = total of the matrix

summarize rows

sums to 1

mass of 1st row

mass of 2nd row

Weights (columns)

- The weight of each column reflects its importance for discriminating between the authors.
- The idea is that columns that are used often do not provide much information, and columns that are used rarely provide much information.

(this is akin to the notion of TF-IDF)

$$\mathbf{w} = [w_j] = [c_j^{-1}] = \begin{bmatrix} \frac{1}{.2973} \\ \frac{1}{.5642} \\ \frac{1}{.1385} \end{bmatrix} = \begin{bmatrix} 3.3641 \\ 1.7724 \\ 7.2190 \end{bmatrix}$$

(X: I x J)

J x 1

Correspondence analysis (CA)

m: masses (rows)

c^T: weights (columns)

- CA may be defined as a special case of principal components analysis - eigenvector methods
- $Y = (N - mc^T) = P\Delta Q^T$, where (we apply SVD on the matrix of deviation Y.)

$$P^T D_m^{-1} P = Q^T D_c^{-1} Q = I$$

  - *N* denote the matrix *X* divided by the sum of all its elements (raw frequencies → proportions)
  - D_m (respectively D_c) denote the diagonal matrices with the elements of *m* (respectively *c* on the diagonal)

$1 \times J \quad I \times J \quad I \times I \quad K \times J$

residual $\underset{\sim}{Y} = N - m\, C^T$

y

outer product (the result is the expected proportions)

(N: observed proportions

m: row masses

C: column masses)

$I \times K \quad K \times K \quad K \times J$

$= P \Delta Q^T$

left singular vectors

right singular vectors

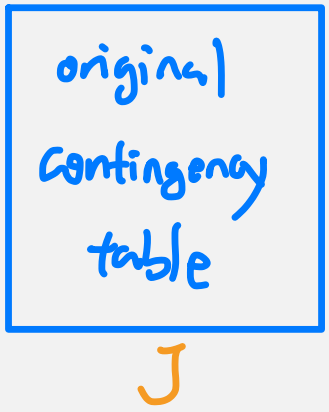
$P^T D_m^{-1} P = I_{K \times K}$ $\quad$ $Q^T D_c^{-1} Q = I_{K \times K}$

$K \times I \quad I \times I \quad I \times K$ $\quad$ $K \times J \quad J \times J \quad J \times K$

The factor scores for rows and columns

(D_m^{-1}, D_c^{-1} : normalization, Δ: scaling)

- The factor scores for
  - the rows : $F = D_m^{-1} P \Delta$ (I×K, I×1, J×K, K×K)
  - the columns : $G = D_c^{-1} Q \Delta$ (J×K, J×J, J×K, K×K)



K : the dimension of the reduced space

* The scores represent coordinates in a reduced dimensional space where rows and columns of the contingency tables are plotted.

Factor Scores for Rows (F)

1. **Definition**:

$$F = D_m^{-1} P \Delta$$

Where:

- D_m^{-1} is the inverse of the diagonal matrix containing the masses of the rows. Each row's mass is proportional to its total relative to the grand total of the table. This matrix serves to weight each row according to its overall contribution to the total data, normalizing the effect of different row sizes.
- P contains the left singular vectors derived from the singular value decomposition (SVD) of the normalized residuals matrix of the table. These vectors capture the underlying structure and patterns across the rows.
- Δ is a diagonal matrix containing the singular values, which scale the row profiles relative to their importance in capturing the variability in the data.

2. **Intuitive Explanation**:

- $D_m^{-1} P$ adjusts the row vectors for the size of each row, ensuring that larger rows don't disproportionately influence the analysis.
- **Multiplying by** Δ then scales these adjusted profiles according to the strength of each dimension in capturing the data's variance.
- The resulting factor scores (F) represent each row's position in the reduced space, with each dimension reflecting a key pattern or gradient in the data.

Factor Scores for Columns (G)

1. **Definition**:

$$G = D_c^{-1} Q \Delta$$

Where:

- D_c^{-1} is the inverse of the diagonal matrix containing the masses of the columns.
- Q contains the right singular vectors, analogous to P but capturing the column structures.
- Δ (the same as used for rows) scales the column profiles.

2. **Intuitive Explanation**:
   - $D_c^{-1} Q$ adjusts the column vectors for the size of each column.
   - **Multiplying by** Δ scales these profiles by the importance of the dimensions.
   - The resulting factor scores (G) place each column in the reduced space, helping to visualize how different column categories relate to one another and to the row categories.

Overall Purpose and Use

- The purpose of using F and G is to facilitate a visual examination of the data through plots where both rows and columns are displayed simultaneously.
- By plotting rows and columns in the same space (biplot), you can interpret the proximity (closeness) and the angle (directionality) between categories across both dimensions. Close proximity suggests a strong association between categories, while orthogonal (right angle) positioning suggests independence.

The variance of row factor scores

row factor score

- $F = D_m^{-1} P \Delta$) Def.
- $P^T D_m^{-1} P = I$
- $F^T D_m F$ (the variance of f)
- $= \Delta P^T D_m^{-1} D_m D_m^{-1} P \Delta$ ($F = D_m^{-1} P \Delta$)
- $= \Delta P^T D_m^{-1} P \Delta$ ($P^T D_m^{-1} P = I$)
- $= \Delta^2$

The variance of column factor scores

column factor score

- $G = D_c^{-1} Q \Delta$) Def.
- $Q^T D_c^{-1} Q = I$
- $G^T D_c G$ ≠ (the variance of G)
- $= \Delta G^T D_c^{-1} D_c D_c^{-1} G \Delta$
- $= \Delta G^T D_c^{-1} G \Delta$
- $= \Delta^2$

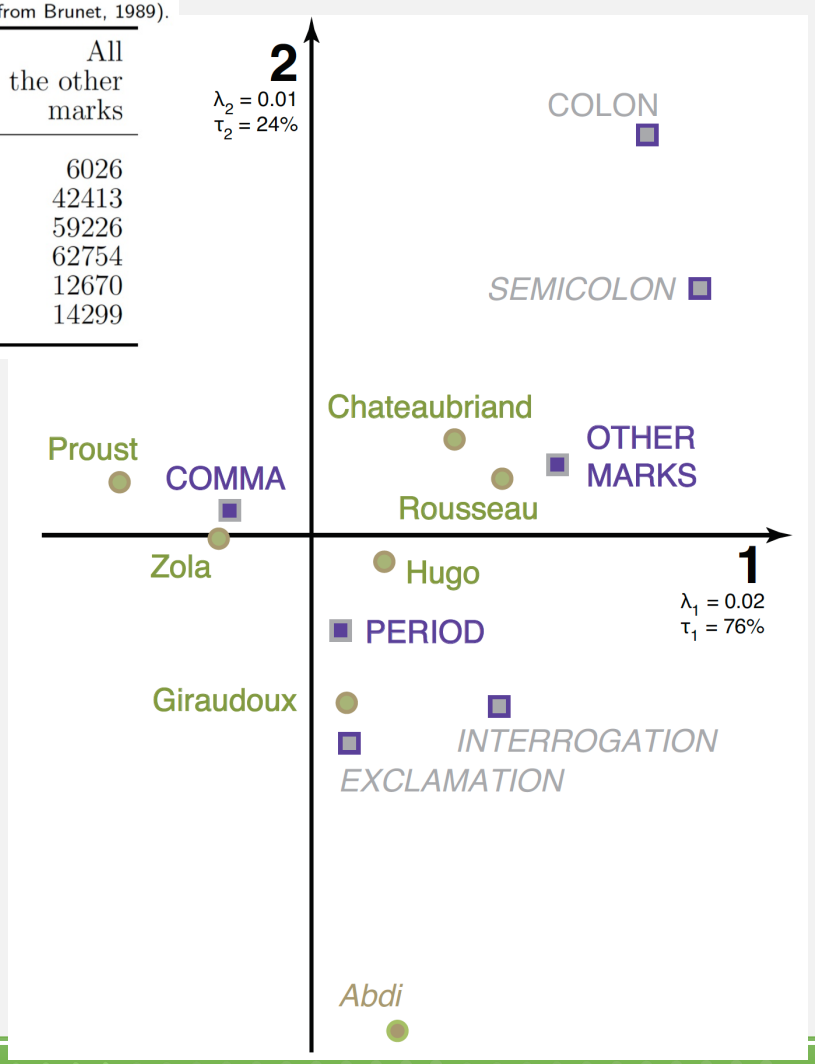
Correspondence analysis

- Because the factor scores obtained for the rows and the columns have the same variance Δ^2(*i.e.,* they have the same “scale”), it is possible to plot them in the same space.

Correspondence analysis of the punctuation of six authors, Comma, Period, and Others

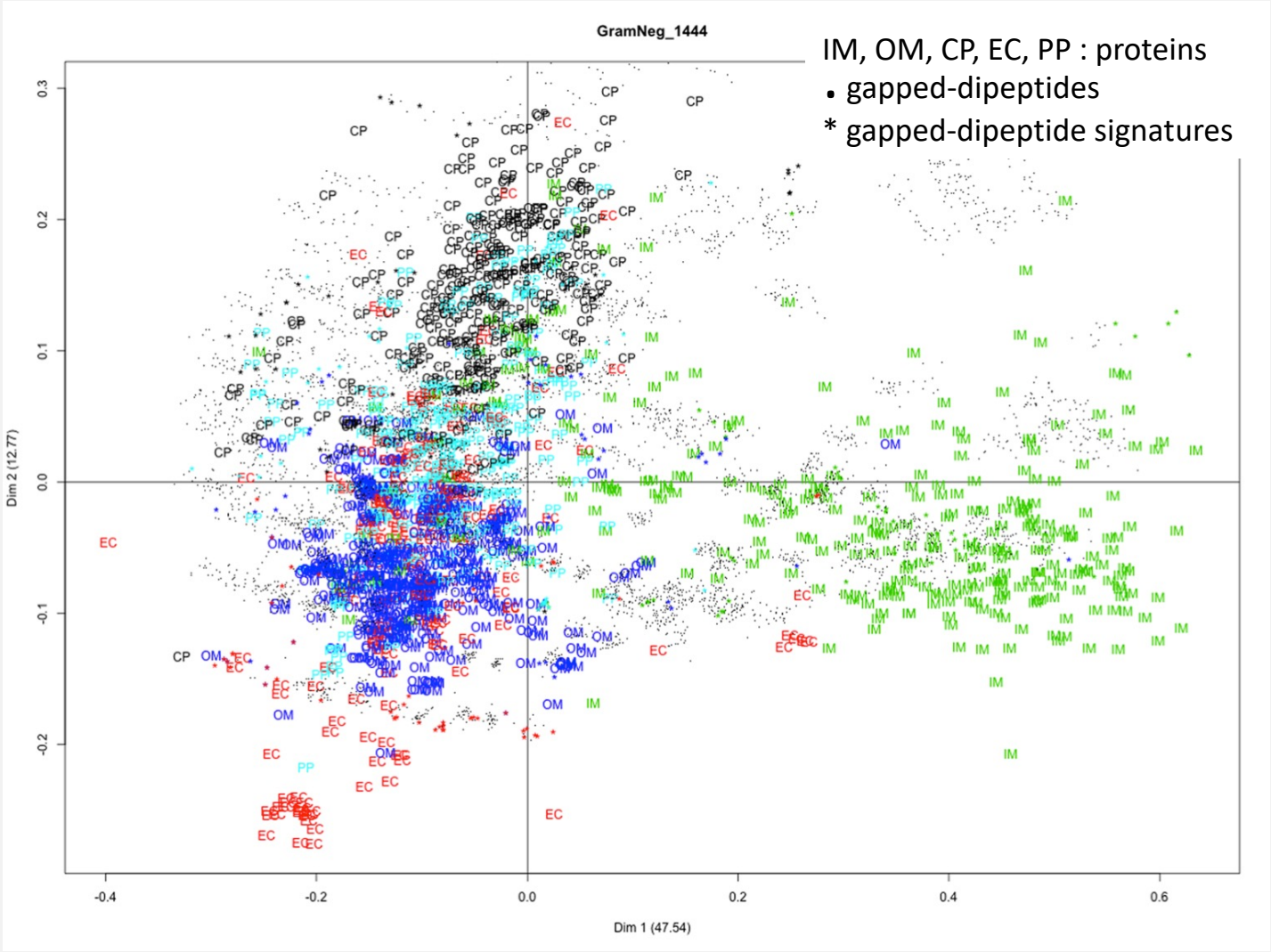
Table 1: The punctuation marks of six French writers (from Brunet, 1989).

	Period	Comma	All the other marks
Rousseau	7836	13112	6026
Chateaubriand	53655	102383	42413
Hugo	115615	184541	59226
Zola	161926	340479	62754
Proust	38177	105101	12670
Giraudoux	46371	58367	14299



Credit : HervLe Abdi & Lynne J. Williams, Correspondence Analysis

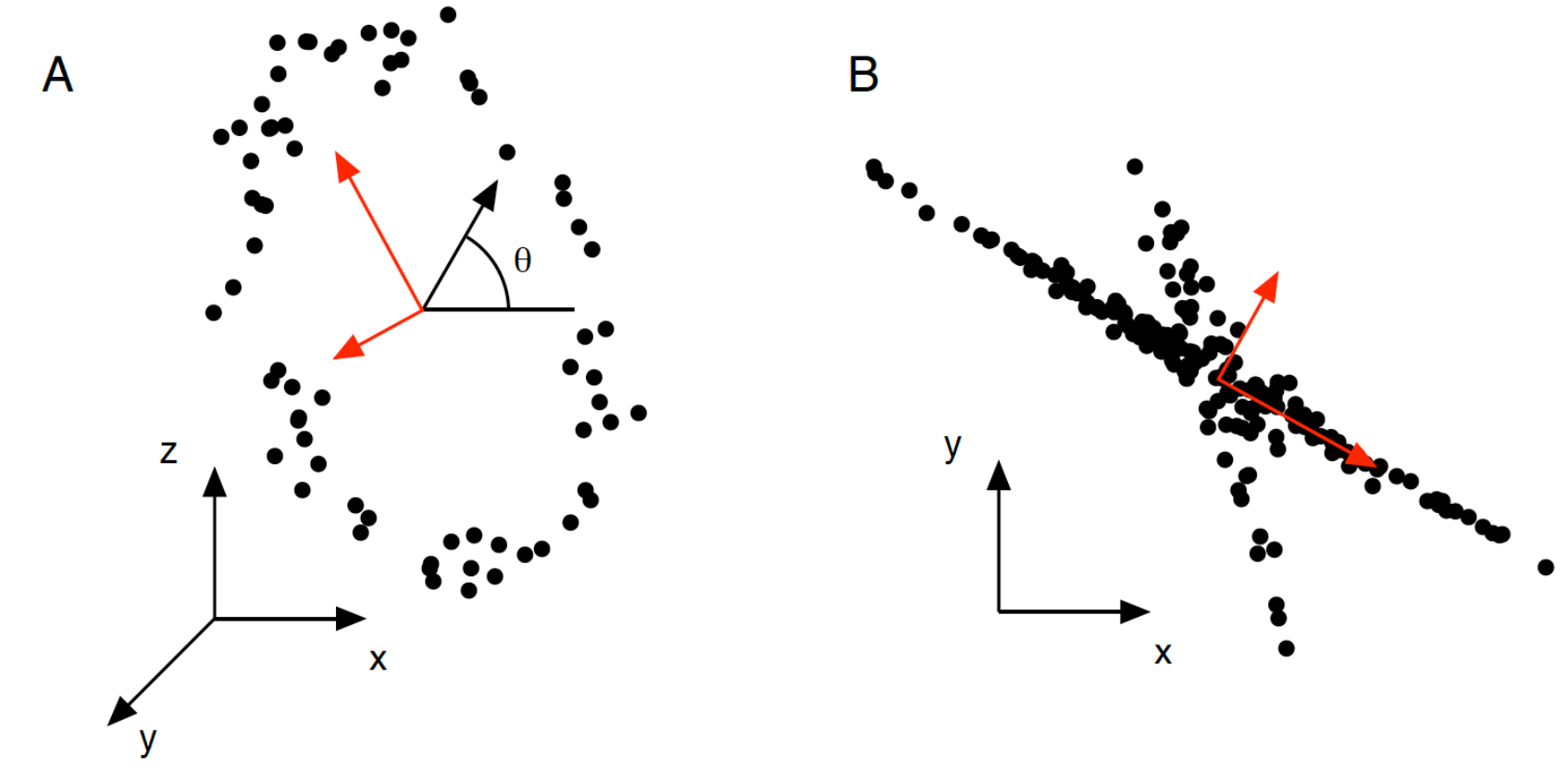
Correspondence analysis of the Gram-negative



Chang, J.-M. *et al.* Efficient and Interpretable Prediction of Protein Functional Classes by Correspondence Analysis and Compact Set elations. *Plos One* **8,** e75542 (2013).

Probabilistic Latent Semantic Analysis

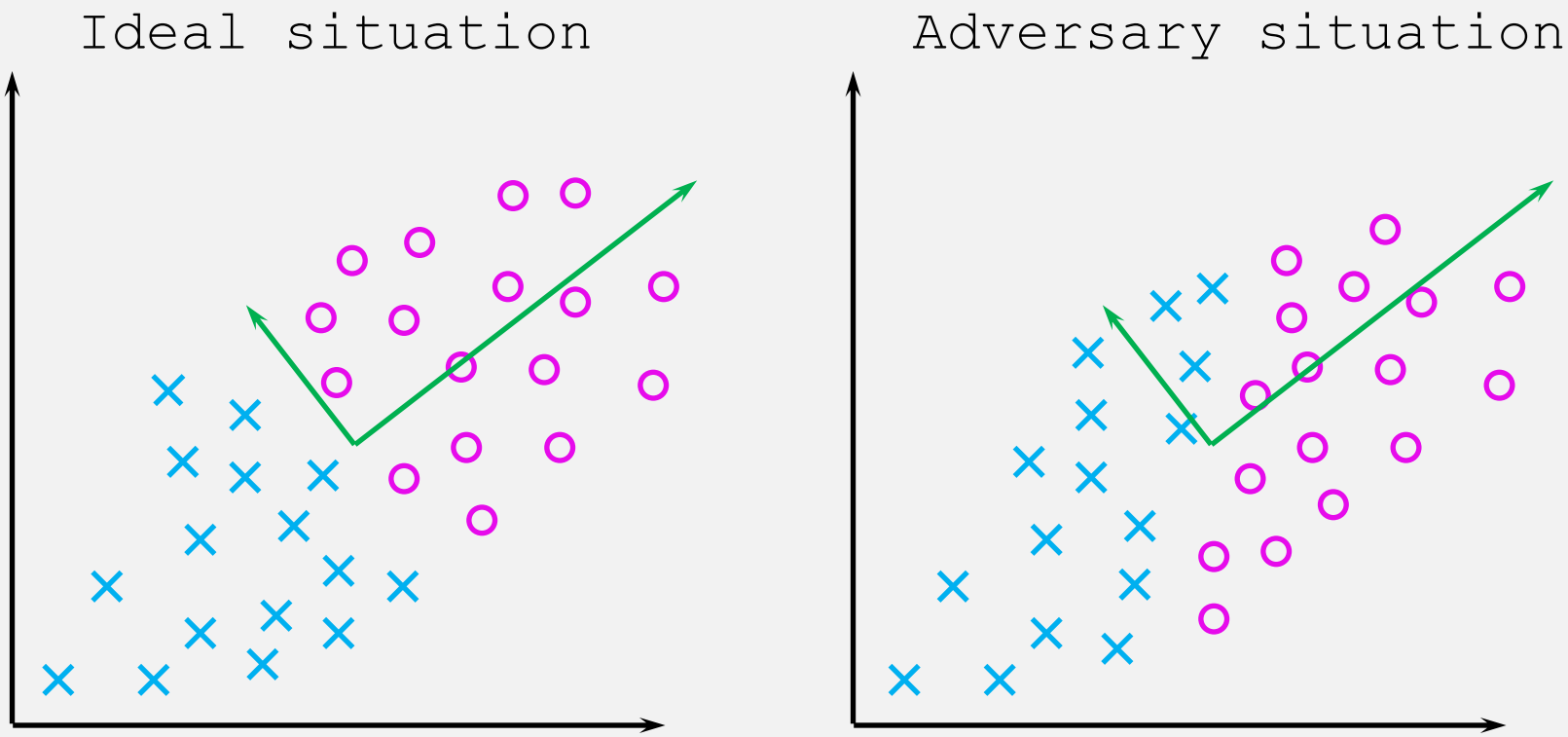
Example of when PCA fails (red lines)



Credit: A Tutorial on Principal Component Analysis, Jonathon Shlens

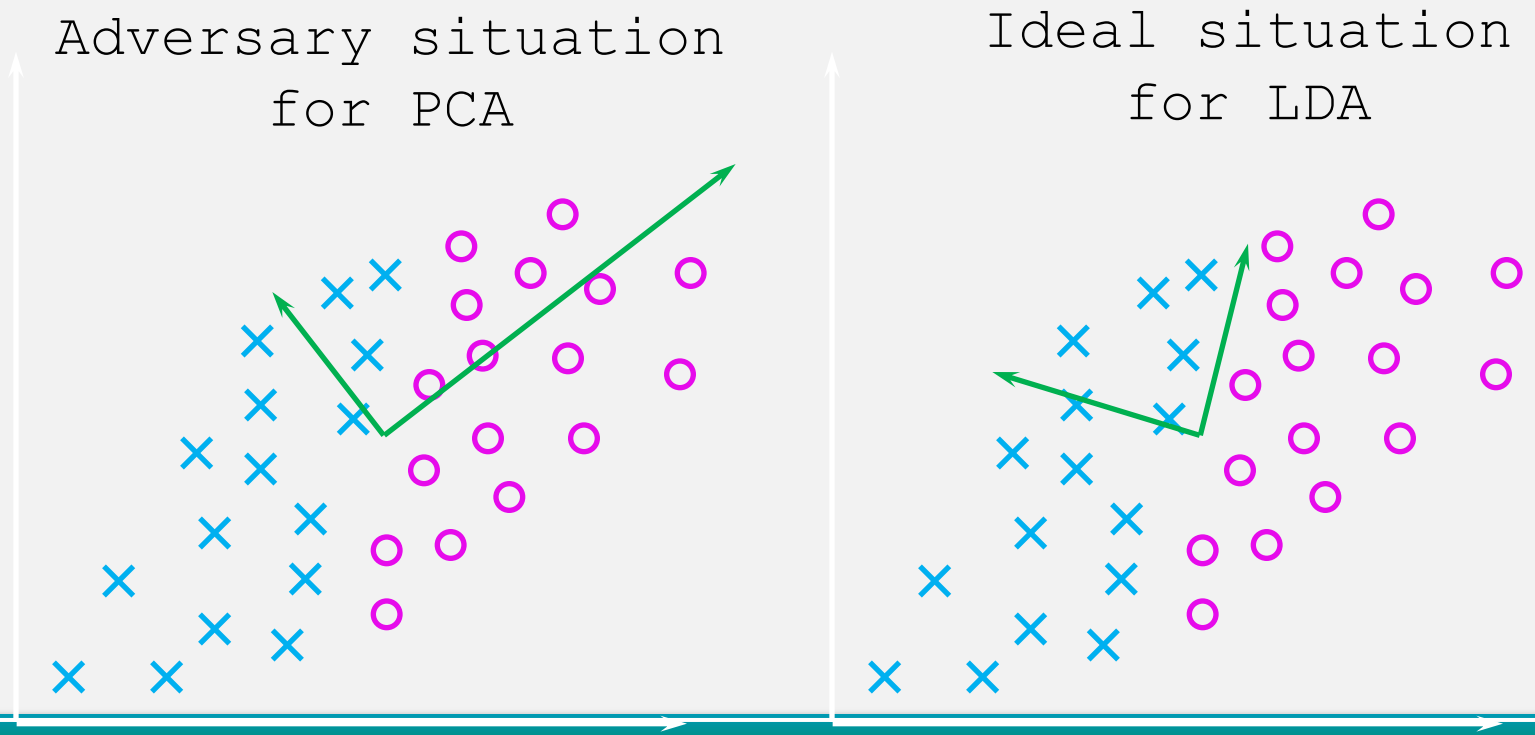
Weakness of PCA for Classification

Not designed for classification problem (with labeled training data)



Linear Discriminant Analysis

- LDA projection onto directions that can best separate data of different classes.



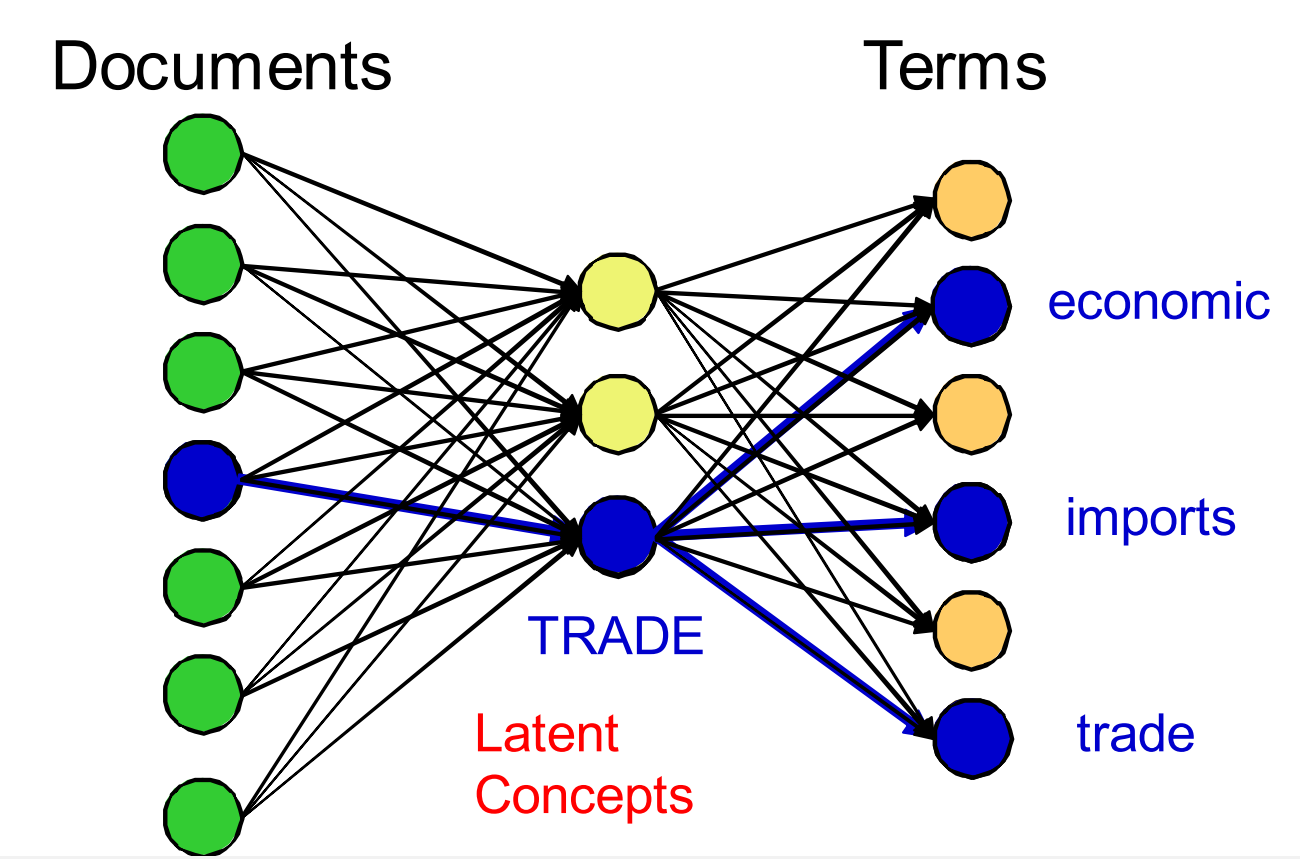
Disadvantages of SVD

- the resulting dimensions might be difficult to interpret.

  {(A0A), (A1A),…,(Z0Z)} -> {1.3x(A0A)-0.2x(B1B),…, (Z0Z)}

- the reconstruction may contain negative entries, which are inappropriate as a distance function for count vectors.

Feature Reduction - Probabilistic Latent Semantic Analysis (1/3)



Hofmann T. Unsupervised learning by probabilistic latent semantic analysis. Machine Learn 2001;42:177–196.

Feature Reduction - Probabilistic Latent Semantic Analysis (2/3)

- A joint probability between a term *w* and a document *d* can be modeled as:

$$P(w,d) = P(d)\sum_{z \in Z} P(w \mid z)P(z \mid d)$$

Latent variable z (*"small" #states*)

Concept expression probabilities

Document-specific mixing proportions

- The parameters could be estimated by *maximum-likelihood function* through *EM algorithm*.

PLSA model fitting

- Likelihood function

$$L = \sum_{w \in d} \sum_{d \in D} W(w, d) \log P(w, d)$$

Maximum Likelihood

- Probability vs Likelihood
  https://youtu.be/pYxNSUDSFH4
- Maximum Likelihood
  https://youtu.be/XepXtl9YKwc

PLSA model fitting

- *E*-step
  - the probability that a term *w* in a particular document *d* explained by the class corresponding to *z*

- *M*-step

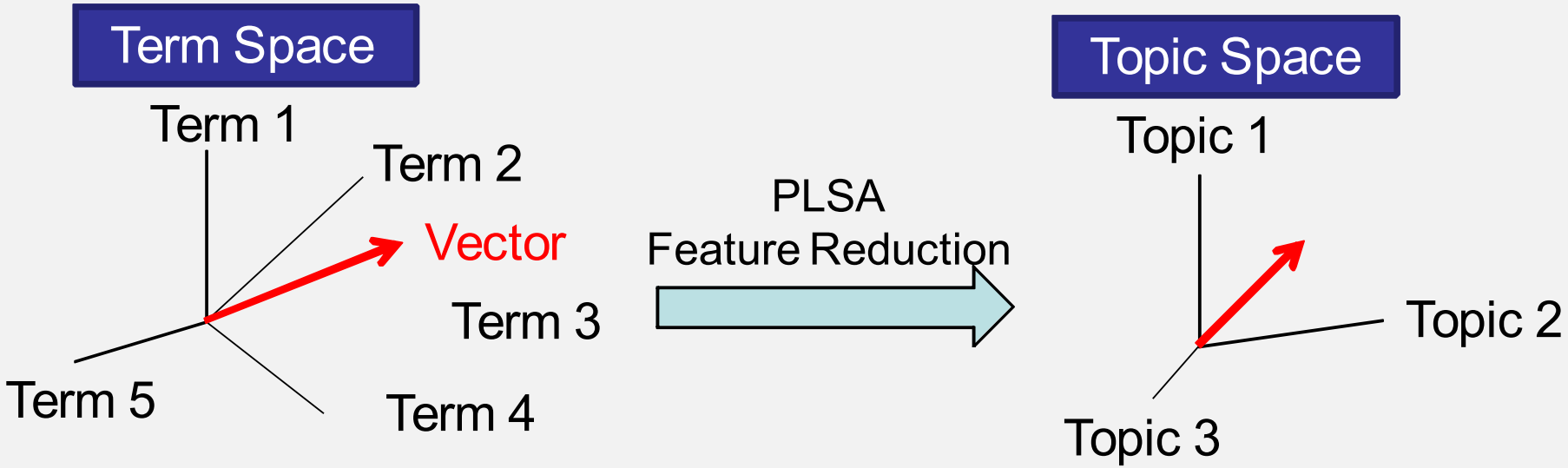
$$P(z|w,d) = \frac{P(d)P(z|d)P(w|z)}{P(d)\sum_{z'} P(w|z')P(z'|d)}$$

$$= \frac{P(w|z)P(z|d)}{\sum_{z'} P(w|z')P(z'|d)}$$

$$P(w \mid z) = \frac{\sum_d W(w,d)P(z|w,d)}{\sum_{w'}\sum_d W(w',d)P(z|w',d)}$$

$$P(z \mid d) = \frac{\sum_w W(w,d)P(z|w,d)}{\sum_{z'}\sum_w W(w,d)P(z'|w,d)}$$

Feature Reduction - Probabilistic Latent Semantic Analysis (3/3)



AITC
教育部人工智慧技術及應用人才培育計畫
Artificial Intelligence Talent Cultivation Program