One Week Workshop on Data Science Using R

Day 5: Unsupervised Learning for Data Science

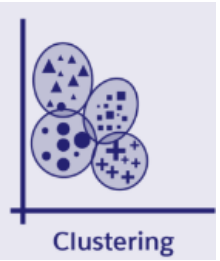
Presentation by
Suja A. Alex,
Assistant Professor,
Department of Information Technology,
St. Xavier's Catholic College of Engineering.

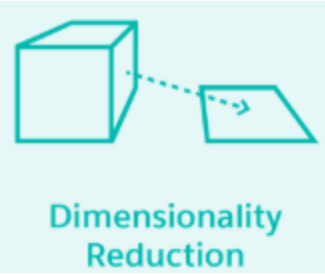
Outline

- **Introduction to Unsupervised Learning**
- **Cluster Analysis**
  - **Types of clustering**
  - **K-means clustering**
  - **Implementation of K-means in R**
- **Dimensionality Reduction**
  - **Curse of Dimensionality**
  - **Feature Selection vs. Feature Extraction**
  - **PCA Algorithm**
  - **Implementation of PCA in R**
- **Association Analysis**
  - **Association Rule Mining**
  - **Apriori Algorithm**
  - **Implementation of Apriori in R**

Unsupervised Learning

- Descriptive model explores the data
- Looking for structure or patterns or relation





- Unlabelled data
- No training set

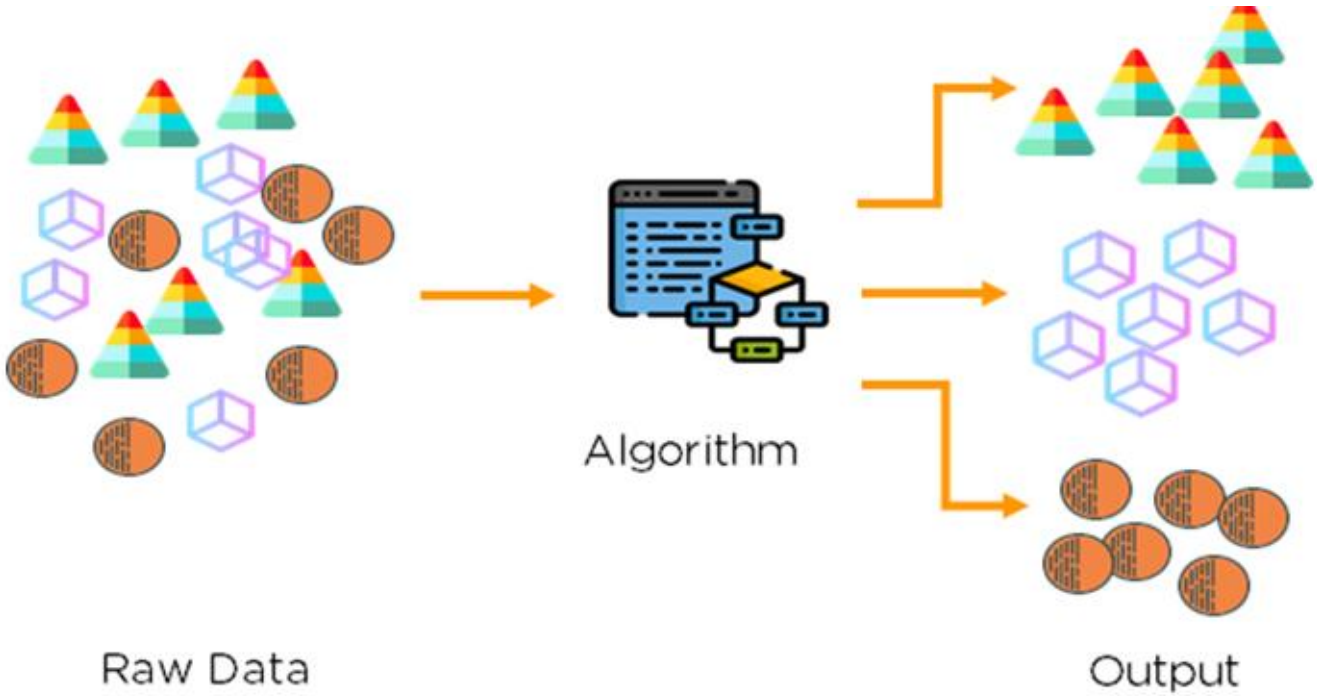
Techniques:

1. **Cluster Analysis**
2. **Dimensionality Reduction**
3. **Association Analysis**

1. Cluster Analysis

- Exploratory data analysis technique used to get an intuition about the structure of the data.
- Process of grouping objects into different sets called clusters.
- Similarity Measure → Euclidean Distance.

$$d_{euc}(x, y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$



Types of Clustering

1. Partitioning Clustering
2. Density-based Clusterir
3. Hierarchical Clustering
4. Grid-based Clustering
5. Model-based Clustering

History of K-means Clustering

1. Lloyd, S. P. (**1957**). “Least squares quantization in PCM”. Technical Note, Bell Laboratories. Published in 1982 in IEEE Transactions on Information Theory, **28**, 128--137.

2. Forgy, E. W. (**1965**). “Cluster analysis of multivariate data: efficiency vs interpretability of classifications”. Biometrics, 21, 768--769.

3. MacQueen, J. (**1967**). “Some methods for classification and analysis of multivariate observations”. In Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, eds L. M. Le Cam & J. Neyman, 1, pp.281--297. Berkeley, CA: University of California Press.

- Hartigan, J. A. and Wong, M. A. (**1979**). “Algorithm AS 136: A K-means clustering algorithm”. Applied Statistics, 28, 100--108. 10.2307/2346830.

K-means clustering

- It partitions a data set into **K clusters**
- Each cluster is associated with a **centroid.**
- Each point is assigned to the cluster with the **closest centroid.**
- Needs numeric, continuous data.

SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
5.1	3.5	1.4	0.2	Iris-setosa
4.9	3.0	1.4	0.2	Iris-setosa
4.7	3.2	1.3	0.2	Iris-setosa
4.6	3.1	1.5	0.2	Iris-setosa
5.0	3.6	1.4	0.2	Iris-setosa
5.4	3.9	1.7	0.4	Iris-setosa
4.6	3.4	1.4	0.3	Iris-setosa
5.0	3.4	1.5	0.2	Iris-setosa
4.4	2.9	1.4	0.2	Iris-setosa
4.9	3.1	1.5	0.1	Iris-setosa

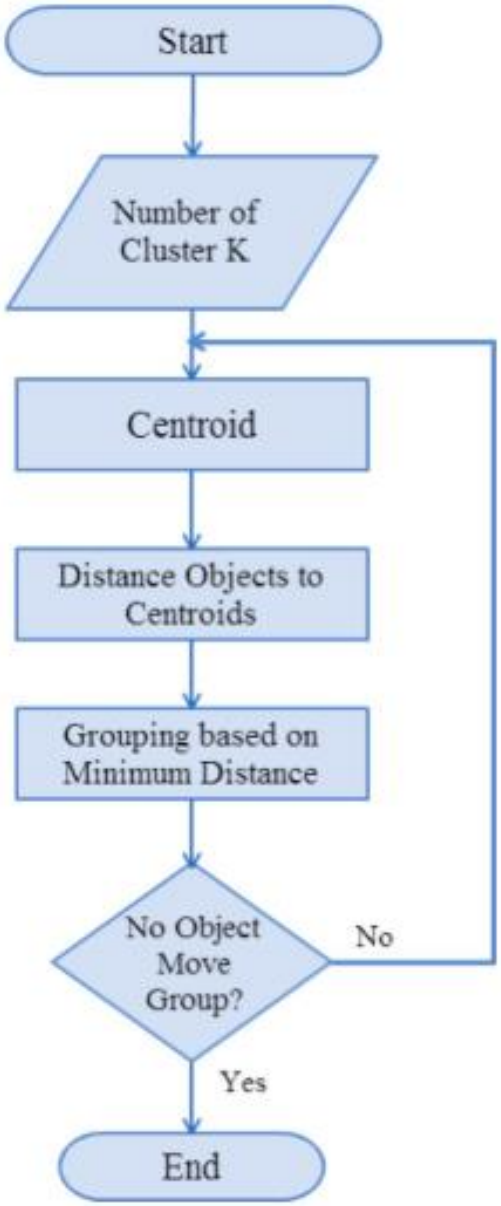
K-means Algorithm

Input : Numeric data set, K=2

Output : 2 Clusters

Steps

1. Initially, two centroids assigned randomly.
2. The Euclidean distance is used to find out which centroid is closest to each data point and the data points are assigned to the corresponding centroids.
3. Reposition the two centroids for optimization.
4. The process is iteratively repeated until our centroids become static.



Kmeans() in R

<u>Input parameters:</u>

- **x**: numeric matrix, numeric data frame or a numeric vector
- **centers**: K number of clusters
- **iter.max**: Default value is 10.
- **nstart**: Always > 1

Implementation of K-means in R

```
c <-kmeans(iris[,3:4],10)
c$cluster
c$centers
c$totss
c$withinss
c$tot.withinss
c$betweenss
c$size
c$iter
c$ifault
```

$$SS_{total} = \sum_{j=1}^{p} \sum_{i=1}^{n_j} (x_{ij} - \overline{x})^2$$

$$SS_{between} = \sum_{j=1}^{p} n_j (\overline{x}_j - \overline{x})^2$$

$$SS_{within} = \sum_{j=1}^{p} \sum_{i=1}^{n_j} (x_{ij} - \overline{x}_j)^2$$

Cluster	A vector of integers (from 1:k) indicating the cluster to which each point is allocated.
centers	A matrix of cluster centres.
totss	The total sum of squares.
withinss	Vector of within-cluster sum of squares, one component per cluster.
tot.withinss	Total within-cluster sum of squares, i.e. sum(withinss).
betweenss	The between-cluster sum of squares, i.e. totss-tot.withinss.
size	The number of points in each cluster.
iter	The number of (outer) iterations.
ifault	integer: indicator of a possible algorithm problem

Demonstration of K-means() in R

```
> c <-kmeans(iris[,3:4],10)
> c$cluster
  [1]  4  4  4  2  4  3  2  2  4  2  2  8  4  4  4  2  4  2  8  2  8  2  4  3  8  8  3  2  4  8  8  2  2  4  2  4  4
 [38]  4  4  2  4  4  4  3  3  2  8  4  2  4 10 10 10  1 10 10 10  6 10  1  6  1  1 10  6 10 10  1 10  1  5  1 10 10
 [75]  1 10 10  5 10  6  1  6  1  5 10 10 10 10  1  1 10 10  1  6  1  1  1  1  6  1  7  5  9  9  9  7 10  7  9  7  5
[112]  5  9  5  5  9  9  7  7  5  9  5  7  5  9  9  5  5  9  9  7  7  9  5  9  7  9  9  5  9  9  5  5  9  9  5  5  5
[149]  9  5
```

```
> c$centers
   Petal.Length Petal.Width
1      4.066667   1.2444444
2      1.481250   0.2500000
3      1.700000   0.4600000
4      1.314286   0.2047619
5      5.028571   1.8904762
6      3.414286   1.0571429
7      6.390000   2.1600000
8      1.662500   0.2125000
9      5.642857   2.0666667
10     4.586957   1.4391304
```

```
> c$size
 [1] 18 16  5 21 21  7 10  8 21 23
```

```
> c$ifault
[1] 0
```

```
> c$betweenss
[1] 543.0531
```

```
> c$tot.withinss
[1] 7.842189
```

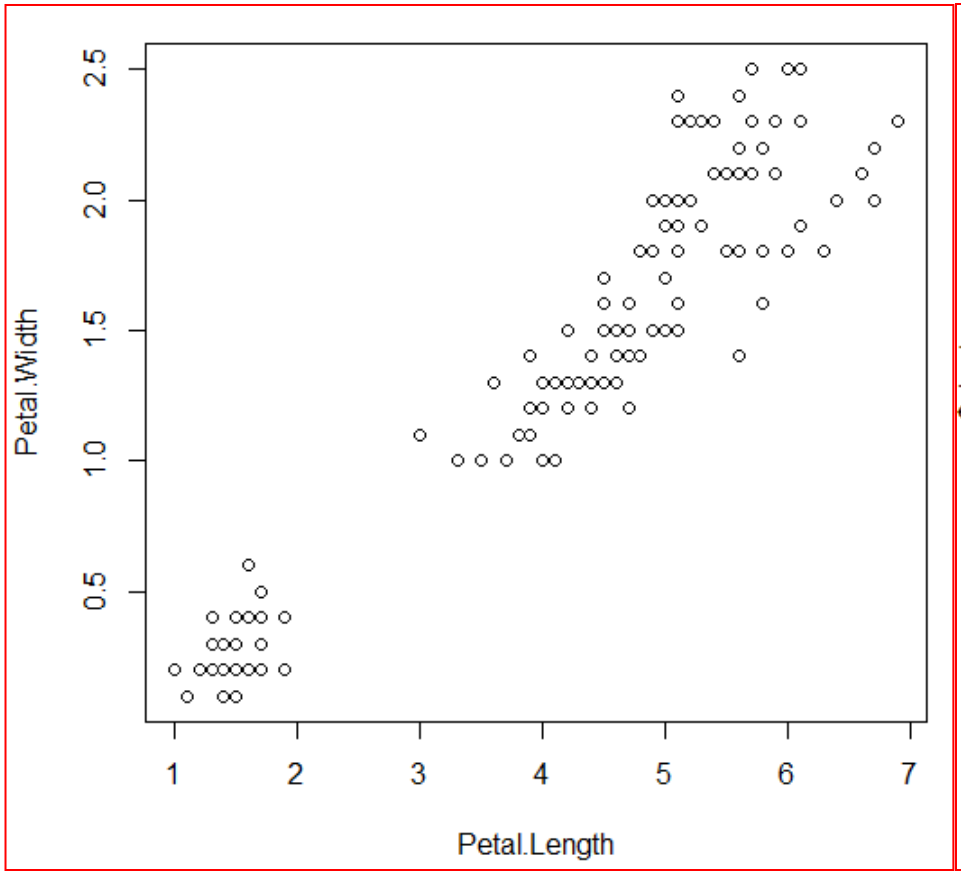
```
> c$totss
[1] 550.8953
```

```
> c$withinss
 [1] 0.6444444 0.1643750 0.0920000 0.3352381 1.5009524 0.4057143 1.4330000 0.0875000 2.3380952 0.8408696
```

Iris dataset

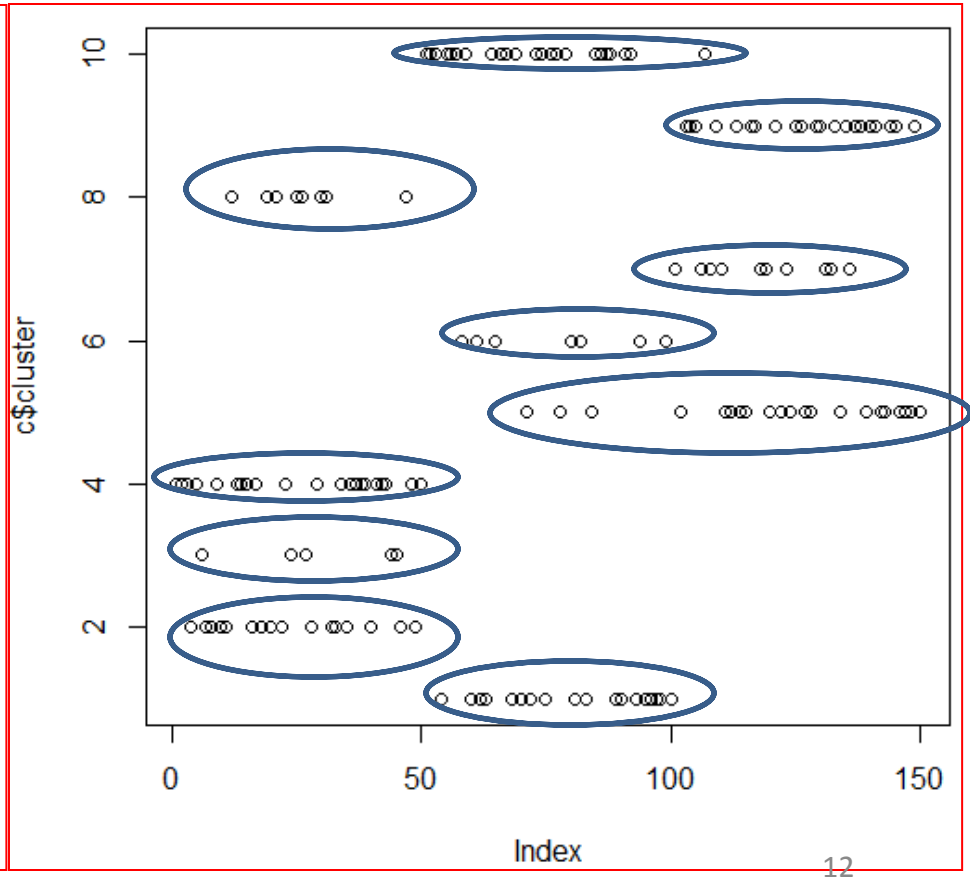
Before k-means Clustering

```
> plot(iris[,3:4])
```



After k-means Clustering (k=10)

```
> plot(c$cluster)
```



Elbow Method in K-means

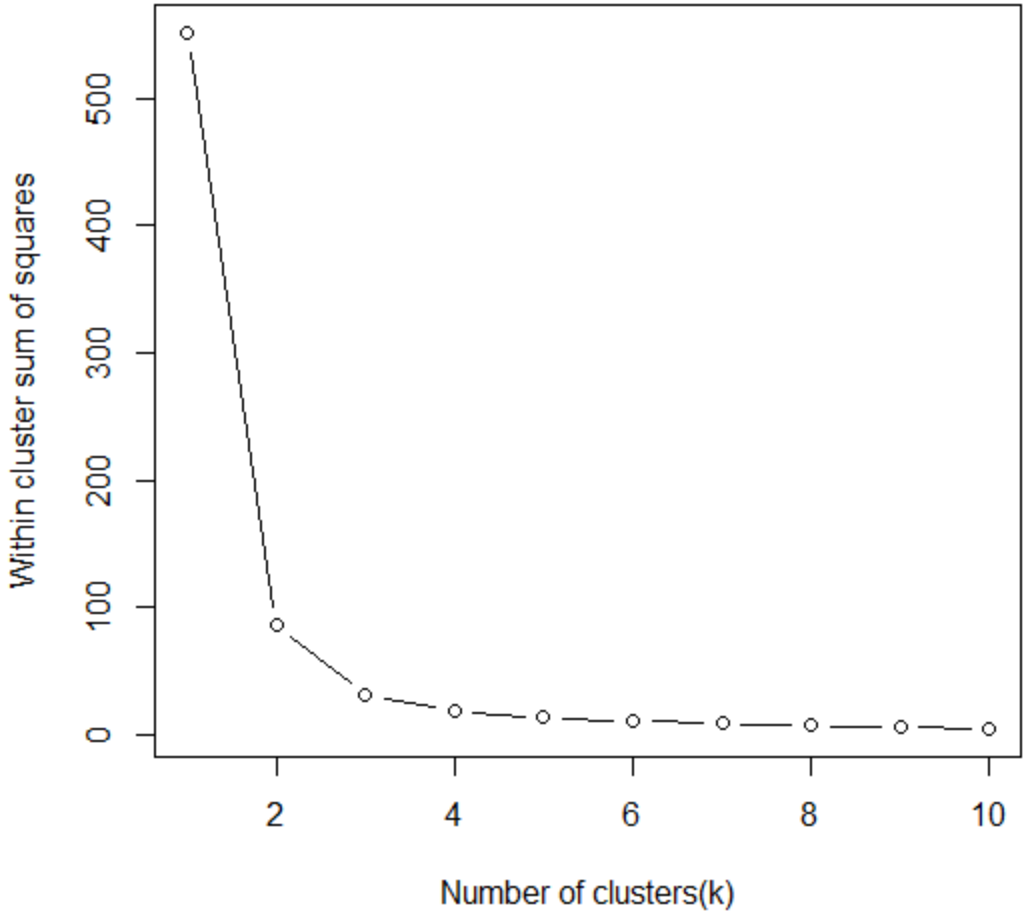
Elbow method → intra-cluster variation or total within-cluster sum of square is minimized.

```
k.max <- 10
wss<- sapply(1:k.max,function(k){kmeans(iris[,3:4],k,nstart =
                                20,iter.max = 20)$tot.withinss})
wss
plot(1:k.max,wss, type= "b", xlab = "Number of clusters(k)",
                                ylab = "Within cluster sum of squares")
```

Elbow method to find best value for K

Demonstration of Elbow curve in R

```
> k.max <- 10
> wss<- sapply(1:k.max,function(k){kmeans(iris[,3:4],k,nstart = 20,iter.max = 20)$tot.withinss})
> wss
 [1] 550.895333  86.390220  31.371359  19.465989  13.916909  11.025145
 [7]   9.206861   7.674414   6.456495   5.528149
>
> plot(1:k.max,wss, type= "b", xlab = "Number of clusters(k)", ylab = "Within cluster sum of squares")
```



Limitations of K-means Clustering

- ✓ **Works on numeric, continuous data.**

Solution: **If all of the attributes are categorical or mixed then use k-mode or k-prototype algorithms.**

- ✓ **Suitable for small dimensional dataset.**

Solution: **Apply dimensionality reduction technique, then use k-means algorithm.**

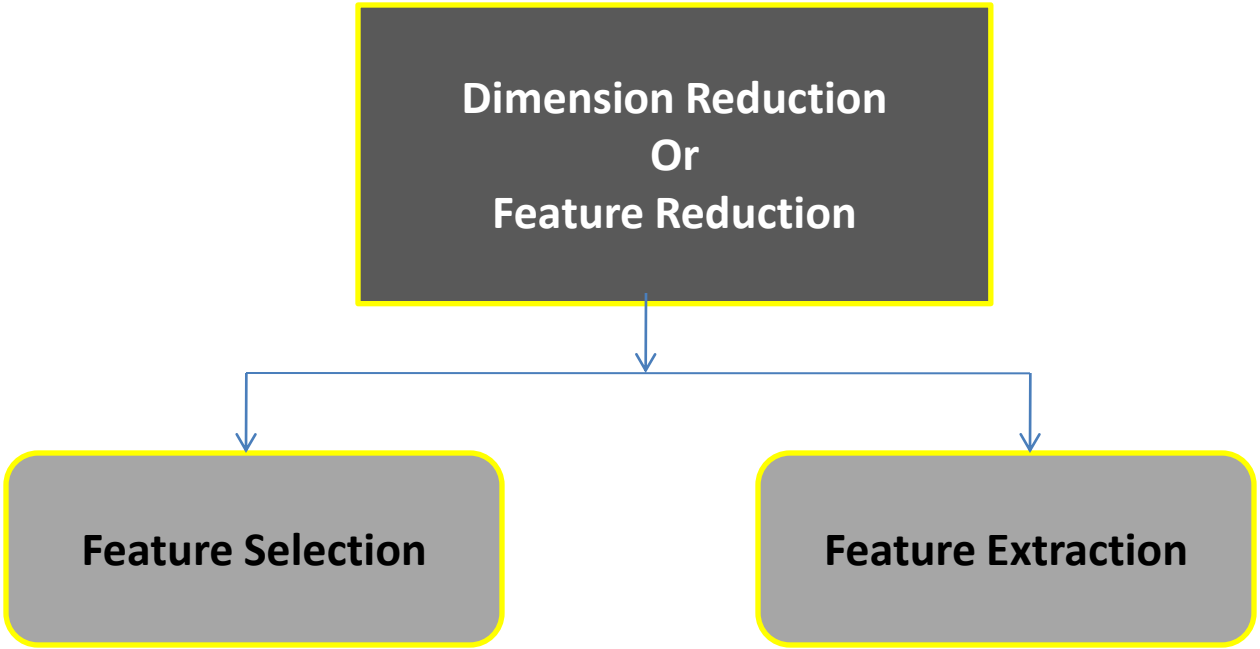
2. Dimensionality Reduction

- It is a process of converting a data set having vast dimensions into a data set with lesser dimensions.
- It ensures that the converted data set conveys similar information concisely.
- **How does dimensionality reduction improve performance?**

Curse of Dimensionality

Feature Selection vs Feature Extraction

- Feature selection → Finds relevant features
- Feature extraction → Finding new features after transforming the data from a high dimensional space to a lower dimensional space.

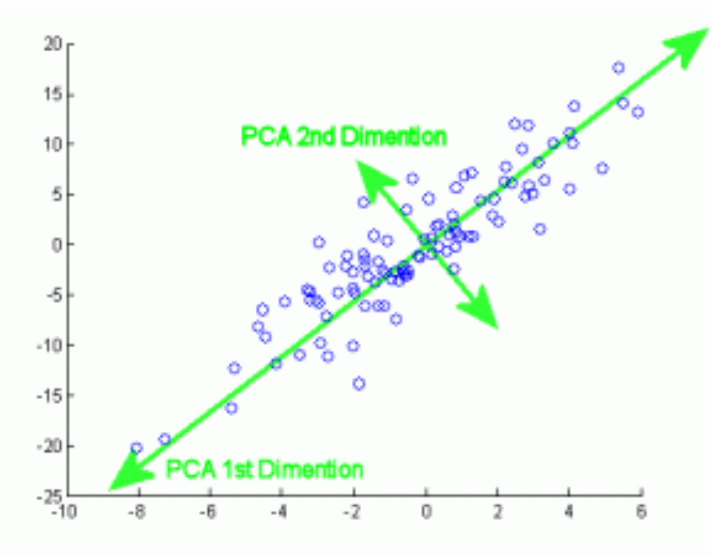


Principal Component Analysis

- Linear transformation that transforms data to a new coordinate system.
- Principal components are orthogonal.
- The first principal component accounts for most of the possible variation of original data.
- The second principal component does its best to capture the variance in the data.

How many principal components are generated?

- Number of principal components depends on number of features of a dataset.
- Example: There can be only two principal components for a two-dimensional data set.



PCA Algorithm

Step 1: Get data.

Step 2: Compute the mean vector (μ).

Step 3: Subtract mean from the given data.

Step 4: Calculate the covariance matrix.

Step 5: Calculate the eigen vectors and eigen values of the covariance matrix.

Step 6: Choosing components and forming a feature vector.

Step 7: Deriving the new data set.

Example

Consider the two dimensional patterns (2, 1), (3, 5), (4, 3), (5, 6), (6, 7), (7, 8).

Compute principal components using PCA Algorithm?

Solution

Step-1: Get data.

The given feature vectors are:

$x_1 = (2, 1)$

$x_2 = (3, 5)$

$x_3 = (4, 3)$

$x_4 = (5, 6)$

$x_5 = (6, 7)$

$x_6 = (7, 8)$

$$\begin{bmatrix} 2 \\ 1 \end{bmatrix} \begin{bmatrix} 3 \\ 5 \end{bmatrix} \begin{bmatrix} 4 \\ 3 \end{bmatrix} \begin{bmatrix} 5 \\ 6 \end{bmatrix} \begin{bmatrix} 6 \\ 7 \end{bmatrix} \begin{bmatrix} 7 \\ 8 \end{bmatrix}$$

Step-2: Calculate the mean vector (μ).

Mean vector (μ) = ((2 + 3 + 4 + 5 + 6 + 7) / 6, (1 + 5 + 3 + 6 + 7 + 8) / 6)

= (4.5, 5)

Thus,

$$\text{Mean vector } (\mu) = \begin{bmatrix} 4.5 \\ 5 \end{bmatrix}$$

Step-3: Subtract mean vector (μ) from the given feature vectors.

$x_1 - \mu = (2 - 4.5, 1 - 5) = (-2.5, -4)$

$x_2 - \mu = (3 - 4.5, 5 - 5) = (-1.5, 0)$

$x_3 - \mu = (4 - 4.5, 3 - 5) = (-0.5, -2)$

$x_4 - \mu = (5 - 4.5, 6 - 5) = (0.5, 1)$

$x_5 - \mu = (6 - 4.5, 7 - 5) = (1.5, 2)$

$x_6 - \mu = (7 - 4.5, 8 - 5) = (2.5, 3)$

Feature vectors (x_i) after subtracting mean vector (μ) are:

$$\begin{bmatrix} -2.5 \\ -4 \end{bmatrix} \begin{bmatrix} -1.5 \\ 0 \end{bmatrix} \begin{bmatrix} -0.5 \\ -2 \end{bmatrix} \begin{bmatrix} 0.5 \\ 1 \end{bmatrix} \begin{bmatrix} 1.5 \\ 2 \end{bmatrix} \begin{bmatrix} 2.5 \\ 3 \end{bmatrix}$$

Step-4: Calculate the covariance matrix.

Covariance matrix is given by:

$$\text{Covariance Matrix} = \frac{\Sigma (x_i - \mu)(x_i - \mu)^t}{n}$$

$$m_1 = (x_1 - \mu)(x_1 - \mu)^t = \begin{bmatrix} -2.5 \\ -4 \end{bmatrix} \begin{bmatrix} -2.5 & -4 \end{bmatrix} = \begin{bmatrix} 6.25 & 10 \\ 10 & 16 \end{bmatrix}$$

$$m_2 = (x_2 - \mu)(x_2 - \mu)^t = \begin{bmatrix} -1.5 \\ 0 \end{bmatrix} \begin{bmatrix} -1.5 & 0 \end{bmatrix} = \begin{bmatrix} 2.25 & 0 \\ 0 & 0 \end{bmatrix}$$

$$m_3 = (x_3 - \mu)(x_3 - \mu)^t = \begin{bmatrix} -0.5 \\ -2 \end{bmatrix} \begin{bmatrix} -0.5 & -2 \end{bmatrix} = \begin{bmatrix} 0.25 & 1 \\ 1 & 4 \end{bmatrix}$$

$$m_4 = (x_4 - \mu)(x_4 - \mu)^t = \begin{bmatrix} 0.5 \\ 1 \end{bmatrix} \begin{bmatrix} 0.5 & 1 \end{bmatrix} = \begin{bmatrix} 0.25 & 0.5 \\ 0.5 & 1 \end{bmatrix}$$

$$m_5 = (x_5 - \mu)(x_5 - \mu)^t = \begin{bmatrix} 1.5 \\ 2 \end{bmatrix} \begin{bmatrix} 1.5 & 2 \end{bmatrix} = \begin{bmatrix} 2.25 & 3 \\ 3 & 4 \end{bmatrix}$$

$$m_6 = (x_6 - \mu)(x_6 - \mu)^t = \begin{bmatrix} 2.5 \\ 3 \end{bmatrix} \begin{bmatrix} 2.5 & 3 \end{bmatrix} = \begin{bmatrix} 6.25 & 7.5 \\ 7.5 & 9 \end{bmatrix}$$

Covariance matrix = $(m_1 + m_2 + m_3 + m_4 + m_5 + m_6) / 6$

On adding the above matrices and dividing by 6, we get-

$$\text{Covariance Matrix} = \frac{1}{6}\begin{bmatrix} 17.5 & 22 \\ 22 & 34 \end{bmatrix}$$

$$\text{Covariance Matrix} = \begin{bmatrix} 2.92 & 3.67 \\ 3.67 & 5.67 \end{bmatrix}$$

Step-5: Calculate the eigen values and eigen vectors of the covariance matrix.

λ is an eigen value for a matrix M if it is a solution of the characteristic equation $|M - \lambda I| = 0$.

So, we have:

$$\begin{vmatrix} 2.92 & 3.67 \\ 3.67 & 5.67 \end{vmatrix} - \begin{vmatrix} \lambda & 0 \\ 0 & \lambda \end{vmatrix} = 0$$

$$\begin{vmatrix} 2.92 - \lambda & 3.67 \\ 3.67 & 5.67 - \lambda \end{vmatrix} = 0$$

We use the following equation to find the eigen vector-

$$\mathbf{MX = \lambda X}$$

where-

M = Covariance Matrix

X = Eigen vector

λ = Eigen value

Substituting the values in the above equation, we get-

$$\begin{bmatrix} 2.92 & 3.67 \\ 3.67 & 5.67 \end{bmatrix} \begin{bmatrix} X1 \\ X2 \end{bmatrix} = 8.22 \begin{bmatrix} X1 \\ X2 \end{bmatrix}$$

Solving these, we get-

$2.92X_1 + 3.67X_2 = 8.22X_1$

$3.67X_1 + 5.67X_2 = 8.22X_2$

On simplification, we get-

$5.3X_1 = 3.67X_2$ …….(1)

$3.67X_1 = 2.55X_2$ …….(2)

- From (1) and (2), $X_1 = 0.69X_2$

- From (2), the eigen vector is-

$$\text{Eigen Vector :} \quad \begin{bmatrix} X1 \\ X2 \end{bmatrix} = \begin{bmatrix} 2.55 \\ 3.67 \end{bmatrix}$$

- Thus, principal component for the given data set is-

$$\text{Principal Component :} \quad \begin{bmatrix} X1 \\ X2 \end{bmatrix} = \begin{bmatrix} 2.55 \\ 3.67 \end{bmatrix}$$

Implementation of PCA in R

```
head(iris)
d<-iris[,1:4]
head(d)
dim(d)
library(graphics)
pc<-princomp(d, cor=TRUE,score=TRUE)
summary(pc)
plot(pc)
plot(pc,type="l")
pc$loadings
pc$scores
biplot(pc)
```

Demonstration of PCA in R

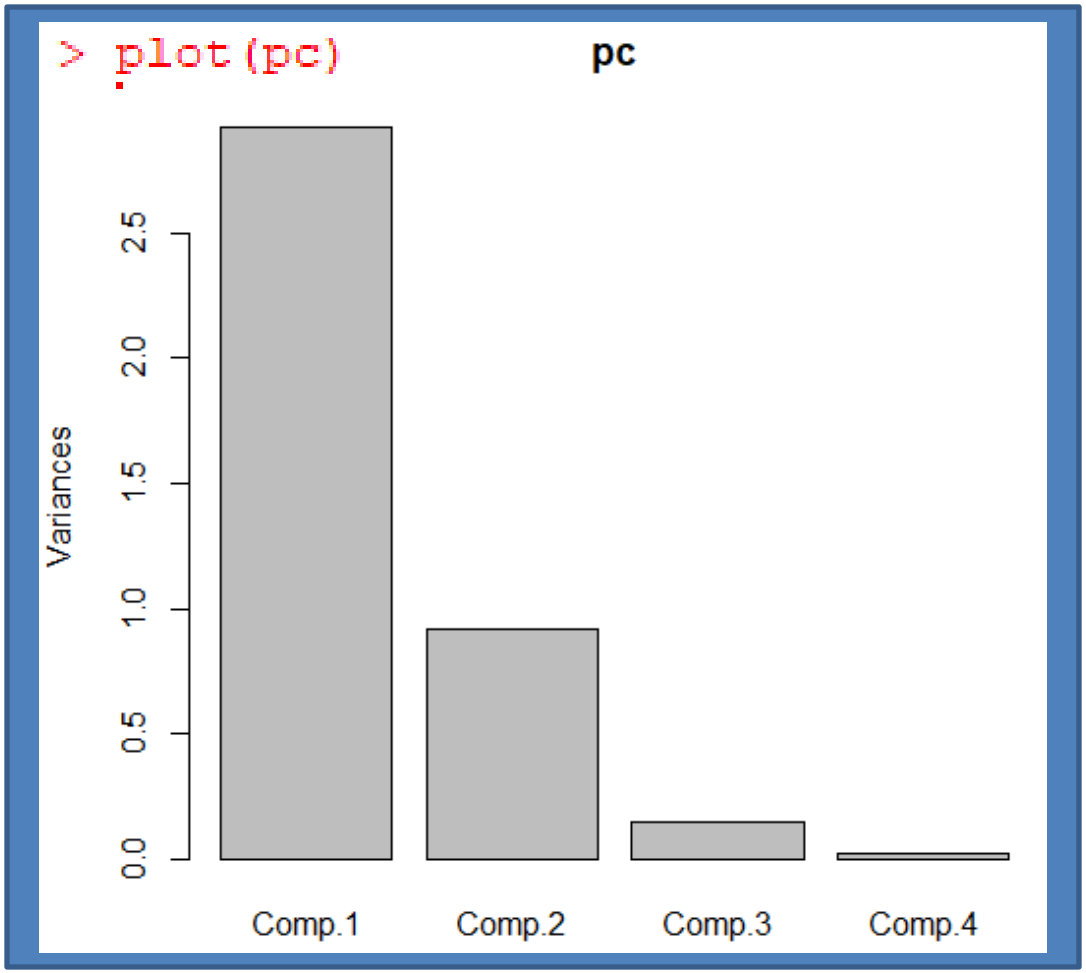
```
> head(iris)
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5    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

```
> d<-iris[,1:4]
> head(d)
  Sepal.Length Sepal.Width Petal.Length Petal.Width
1          5.1         3.5          1.4         0.2
2          4.9         3.0          1.4         0.2
3          4.7         3.2          1.3         0.2
4          4.6         3.1          1.5         0.2
5          5.0         3.6          1.4         0.2
6          5.4         3.9          1.7         0.4
> dim(d)
[1] 150   4
```

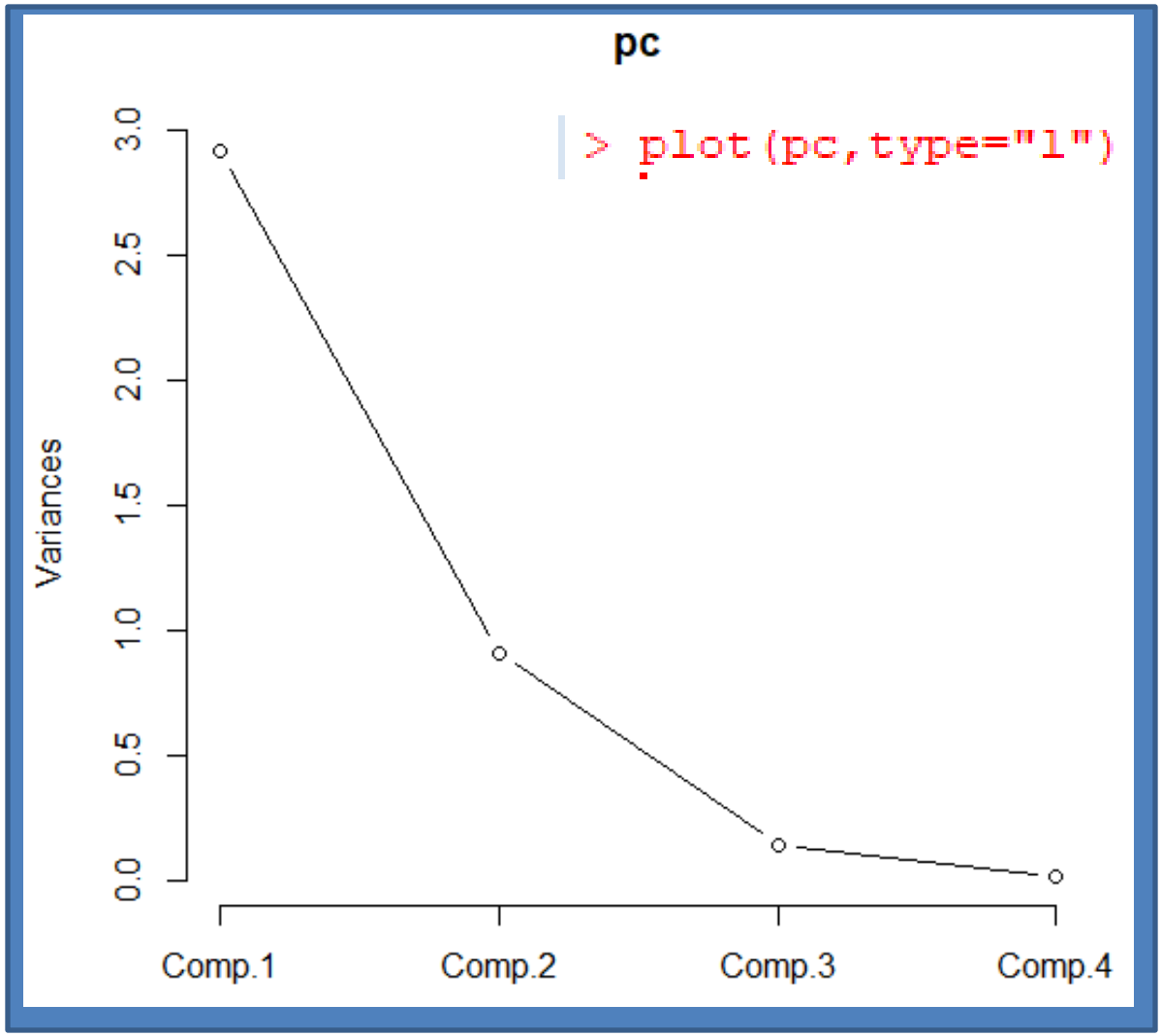
Demonstration of PCA in R

```
> library(graphics)
> pc<-princomp(d, cor=TRUE,score=TRUE)
> summary(pc)
Importance of components:
                          Comp.1    Comp.2     Comp.3      Comp.4
Standard deviation     1.7083611 0.9560494 0.38308860 0.143926497
Proportion of Variance 0.7296245 0.2285076 0.03668922 0.005178709
Cumulative Proportion  0.7296245 0.9581321 0.99482129 1.000000000
```

Demonstration of PCA in R



Demonstration of PCA in R



Demonstration of PCA in R

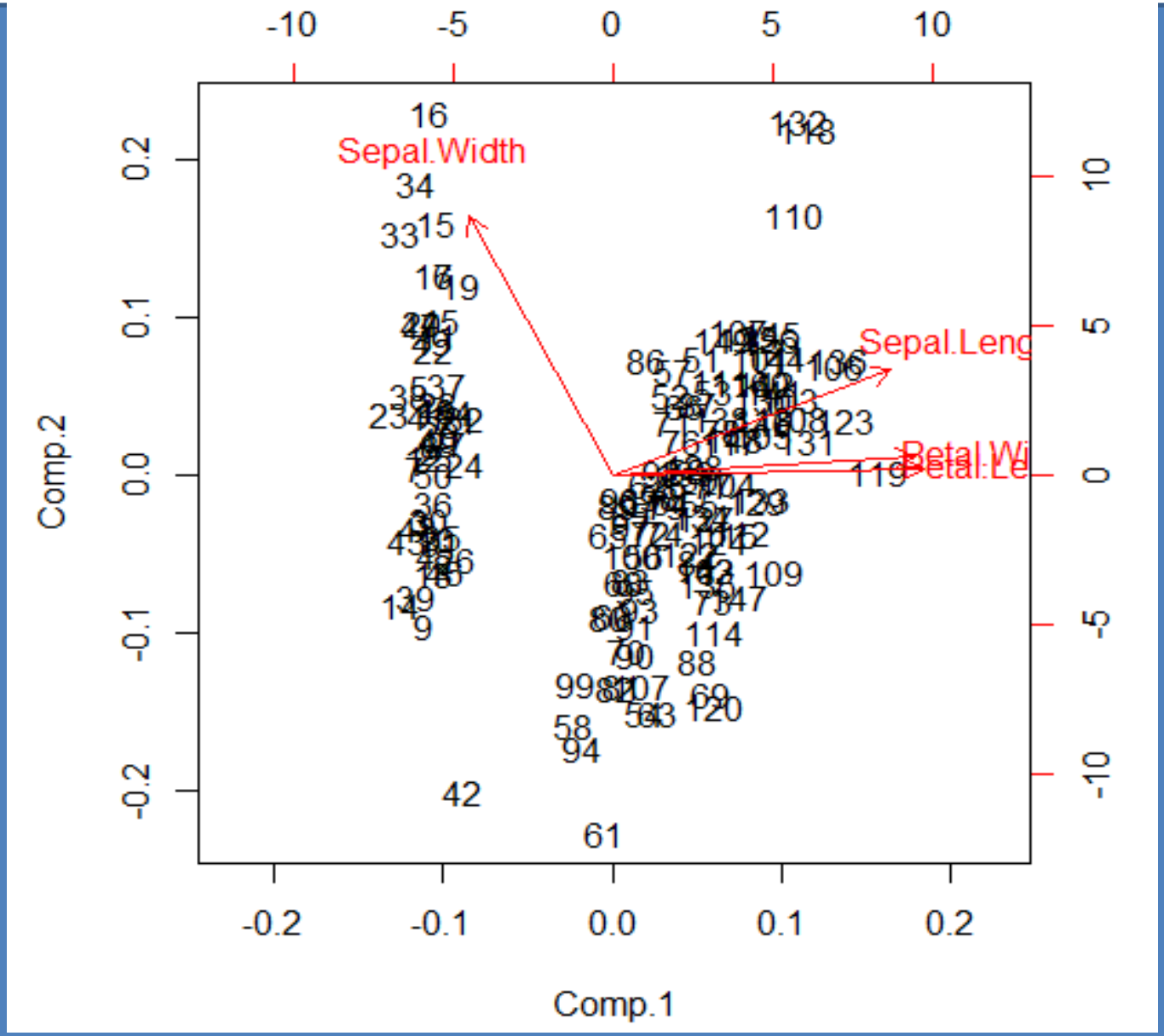
```
> pc$loadings

Loadings:
             Comp.1 Comp.2 Comp.3 Comp.4
Sepal.Length  0.521  0.377  0.720  0.261
Sepal.Width  -0.269  0.923 -0.244 -0.124
Petal.Length  0.580        -0.142 -0.801
Petal.Width   0.565        -0.634  0.524

               Comp.1 Comp.2 Comp.3 Comp.4
SS loadings      1.00   1.00   1.00   1.00
Proportion Var   0.25   0.25   0.25   0.25
Cumulative Var   0.25   0.50   0.75   1.00
```

Demonstration of PCA in R

```
> pc$scores
            Comp.1       Comp.2       Comp.3       Comp.4
 [1,] -2.26470281  0.480026597  0.127706022  0.024168204
 [2,] -2.08096115 -0.674133557  0.234608854  0.103006775
 [3,] -2.36422905 -0.341908024 -0.044201485  0.028377053
 [4,] -2.29938422 -0.597394508 -0.091290106 -0.065955560
 [5,] -2.38984217  0.646835383 -0.015738196 -0.035922813
 [6,] -2.07563095  1.489177523 -0.026968294  0.006608180
 [7,] -2.44402884  0.047644198 -0.335470401 -0.036775557
 [8,] -2.23284716  0.223148073  0.088695498 -0.024612096
 [9,] -2.33464048 -1.115327675 -0.145076864 -0.026859221
[10,] -2.18432817 -0.469013561  0.253765567 -0.039899288
[11,] -2.16631010  1.043690653  0.268681102  0.016731367
[12,] -2.32613087  0.133078335 -0.093759244 -0.133483413
[13,] -2.21845090 -0.728676165  0.230911237  0.002425038
[14,] -2.63310070 -0.961506729 -0.180796084 -0.019215534
[15,] -2.19874060  1.860057113  0.472900998  0.194731769
[16,] -2.26221453  2.686284485 -0.030526609  0.050533737
[17,] -2.20758770  1.483609363  0.005344094  0.188817432
[18,] -2.19034951  0.488838316  0.044215316  0.093090438
[19,] -1.89857200  1.405018794  0.374343275  0.061095967
[20,] -2.34336905  1.127849382 -0.132630467 -0.037756420
```

Demonstration of PCA in R

```
> biplot(pc)
```



Benefits of Dimensionality Reduction

- Data compression
- Reduces computation time
- Removes redundant features
- Improves the model performance

3. Association Analysis

Two steps:

Step 1: Finding all frequent itemsets whose supports are not less than minimum support threshold.

Step 2: From the frequent itemsets, generating association rules with confidence above minimum confidence threshold.

Apriori

- Apriori (Agrawal & Srikant,1994)
  - Candidate generation based frequent pattern
- Classic algorithm for association rule mining.
- Counts transactions to find frequent itemsets.
- Generates candidate itemsets by using downward closure property.

Downward closure property:

- Any subset of a frequent itemset must be frequent.

Example:

- If {bread,butter,jam} is frequent then {bread,butter} also frequent.

Association Rules

- Shows itemsets that occur together frequently.

Association rule form: A => B

A and B are itemsets

A : antecedent or LHS

B : consequent or RHS

- In the database, tuples having items in the left hand of the rule are also likely to having those items in the right hand.

Examples of Association rules:

- Bread => butter
- Computer => software

Best association rules

Measures of Interestingness:

$$
\begin{aligned}
\text{support}(A \Rightarrow B) &= \text{support}(A \cup B) = P(A \wedge B) \\
\text{confidence}(A \Rightarrow B) &= P(B|A) \\
&= \frac{P(A \wedge B)}{P(A)} \\
\text{lift}(A \Rightarrow B) &= \frac{\text{confidence}(A \Rightarrow B)}{P(B)} \\
&= \frac{P(A \wedge B)}{P(A)P(B)}
\end{aligned}
$$

Example

- Assume there are 100 students. 10 out of them know data mining techniques, 8 know R language and 6 know both of them.

  R => DM (If a student know R, the he/she knows data mining.)

Find support?

$$\text{support} = P(R \wedge DM) = 6/100 = 0.06$$

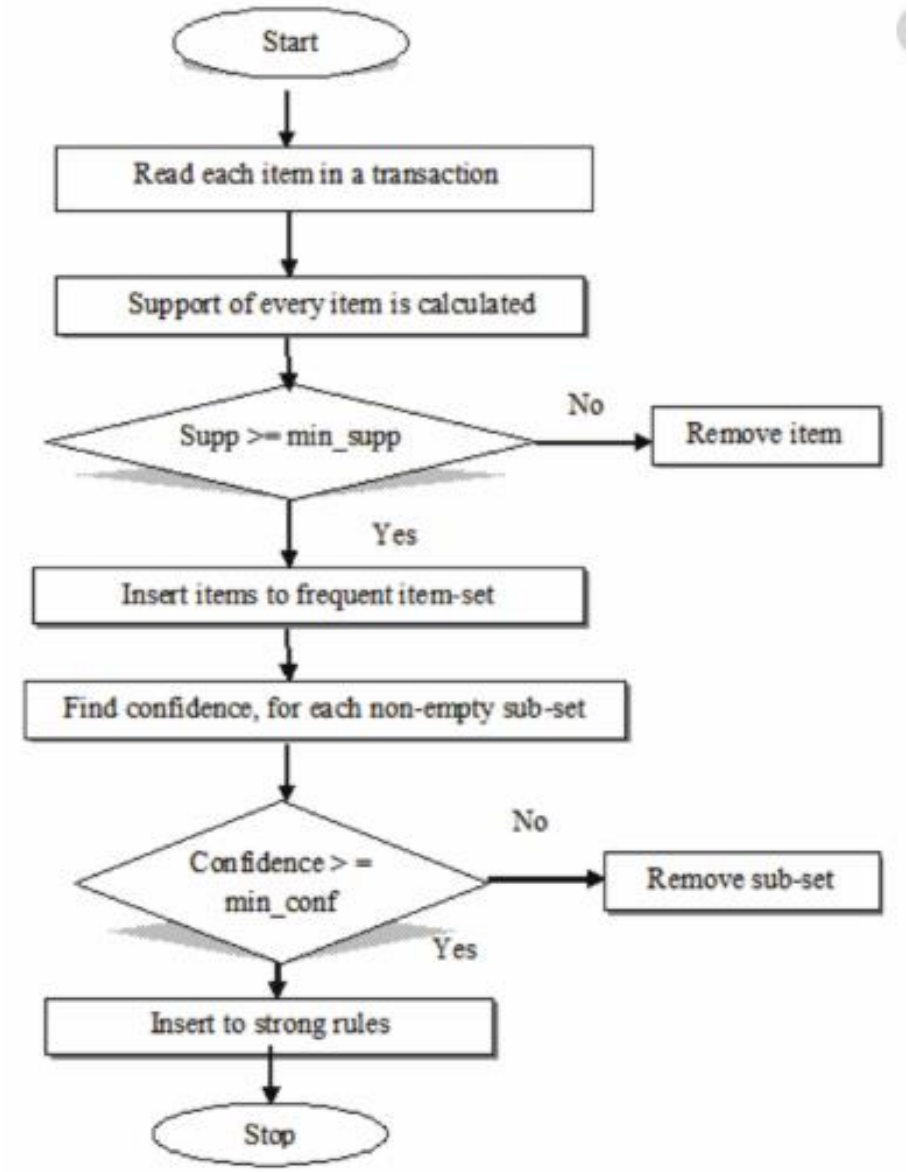
Find confidence?

$$\text{confidence} = \text{support} \ / \ P(R) = 0.06/0.08 = 0.75$$

Find lift?

$$\text{lift} = \text{confidence} \ / \ P(DM) = 0.75/0.1 = 7.5$$

Apriori Algorithm



Example

Dataset has 4 transactions with **support = 2**, find frequent itemset using apriori?

Tid	Items
c1	g1,g3,g4
c2	g2,g3,g5
c3	g1,g2,g3,g5
c4	g2,g5

T

Iteration 1

Itemset	Sup.
{g1}	0.5
{g2}	0.75
{g3}	0.75
{g4}	0.25
{g5}	0.75

Itemset
{g1}
{g2}
{g3}
{g5}

Iteration 2

Itemset	Sup.
{g1,g2}	0.25
{g1,g3}	0.5
{g1,g5}	0.25
{g2,g3}	0.5
{g2,g5}	0.75
{g3,g5}	0.5

Itemset
{g1,g3}
{g2,g3}
{g2,g5}
{g3,g5}

Iteration 3

Itemset	Sup.
{g1,g2,g3}	0.25
{g1,g2,g3,g5}	0.25
{g1,g3,g5}	0.25
{g2,g3,g5}	0.5

Final frequent itemsets

Itemset
{g2,g3,g5}

Implementation of Apriori in R

```
install.packages("arules")
library(arules)
data("Adult")
rules <- apriori(Adult, parameter = list(supp = 0.5, conf
    = 0.9, target = "rules"))
inspect(head(rules, by = "lift"))
```

Demonstration of Apriori in R

```
> install.packages("arules")
Installing package into 'C:/Users/USER/Documents/R/win-library/3.6'
(as 'lib' is unspecified)
--- Please select a CRAN mirror for use in this session ---
trying URL 'https://mirror.las.iastate.edu/CRAN/bin/windows/contrib/3.6/arules_1.6-6.zip'
Content type 'application/zip' length 2678893 bytes (2.6 MB)
downloaded 2.6 MB

package 'arules' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\USER\AppData\Local\Temp\RtmpaanZG2\downloaded_packages
```

```
> library(arules)
Loading required package: Matrix

Attaching package: 'arules'

The following objects are masked from 'package:base':

    abbreviate, write

Warning message:
package 'arules' was built under R version 3.6.3
```

Demonstration of Apriori in R

```
> data("Adult")
> rules <- apriori(Adult, parameter = list(supp = 0.5, conf = 0.9, target = "rules"))
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen maxlen target  ext
        0.9    0.1    1 none FALSE            TRUE       5     0.5      1     10  rules TRUE

Algorithmic control:
 filter tree heap memopt load sort verbose
    0.1 TRUE TRUE  FALSE TRUE    2    TRUE

Absolute minimum support count: 24421

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[115 item(s), 48842 transaction(s)] done [0.12s].
sorting and recoding items ... [9 item(s)] done [0.01s].
creating transaction tree ... done [0.06s].
checking subsets of size 1 2 3 4 done [0.00s].
writing ... [52 rule(s)] done [0.00s].
creating S4 object  ... done [0.01s].
```

Demonstration of Apriori in R

```
> inspect(head(rules, by = "lift"))
    lhs                                                          rhs                            support   confidence coverage  lift     count
[1] {sex=Male,native-country=United-States}                   => {race=White}                   0.5415421 0.9051090  0.5983170 1.058554 26450
[2] {sex=Male,capital-loss=None,native-country=United-States} => {race=White}                   0.5113632 0.9032585  0.5661316 1.056390 24976
[3] {race=White}                                              => {native-country=United-States} 0.7881127 0.9217231  0.8550428 1.027076 38493
[4] {race=White,capital-loss=None}                            => {native-country=United-States} 0.7490480 0.9205626  0.8136849 1.025783 36585
[5] {race=White,sex=Male}                                     => {native-country=United-States} 0.5415421 0.9204803  0.5883256 1.025691 26450
[6] {race=White,capital-gain=None}                            => {native-country=United-States} 0.7194628 0.9202807  0.7817862 1.025469 35140
```

Applications

- Market basket analysis
- Churn analysis and selective marketing
- Credit card risk analysis
- Stock market analysis
- Medical diagnosis

References

- Ronnie,Domingo,Jesus S.AguilarGene,Association analysis: A survey of frequent pattern mining from gene expression data, Briefings in Bioinformatics 11(2):210-24,October 2009
- https://www.coursera.org/lecture/machine-learning/unsupervised-learning-introduction-czmip
- https://onlinelibrary.wiley.com/doi/epdf/10.1002/widm.1062
- https://stackoverflow.com/questions/61736709/how-to-deal-with-categorical-data-in-k-means-clustering-method-when-we-have-mixe
- https://courses.packtpub.com/pages/free
- https://www.simplilearn.com/tutorials/machine-learning-tutorial/k-means-clustering-algorithm
- https://www.rdocumentation.org/packages/stats/versions/3.6.2/topics/kmeans
- https://www.slideshare.net/rdatamining/rdataminingslidesassociationrule miningwithr


Q&A


Thank You!