Universiteit Leiden

Computer Science and Science Based Business

Exploratory Data Analysis on Multivariate Data

Name	Kaihua Liu
Supervisors	Aske Plaat Eva Deckers Grada Degenaars Irene Martorelli Janne van Kollenburg

MASTER'S THESIS (PUBLIC VERSION)

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Contents

Abstract **5**

1 Introduction **6**
1.1 Background . . . 6
1.2 Exploratory Data Analysis . . . 6
1.3 Context and Research Questions . . . 7
1.4 Outline . . . 7

2 Datasets **8**
2.1 Product prototyping . . . 8
2.2 Participants . . . 8
2.3 Data Acquisition . . . 8
2.3.1 Quantitative Data . . . 8
2.3.2 Qualitative Data . . . 8
2.3.3 Raw Datasets . . . 8
2.4 Data Pre-processing . . . 8

3 Approaches **9**
3.1 Supervised Learning . . . 9
3.1.1 Linear Regression . . . 9
3.1.2 Classification Tree . . . 13
3.2 Unsupervised Learning . . . 17
3.2.1 K-means Clustering . . . 17
3.2.2 Principal Components Analysis . . . 19
3.2.3 Time Series Analysis . . . 22

4 Empirical studies **25**

5 Conclusions **26**

6 Reference **27**

Appendix **29**

List of Figures

1.1 Steps of *Exploratory Data Analysis*.[1] 7

3.1 The linear regression result of *faithful* is plotted as the red line. 12
3.2 A decision tree with *Root*, *Decision node*, *Leaf node* and *Subtree* 14
3.3 Decision tree of *iris* dataset. 16
3.4 *Iris* visualization. 18
3.5 Plot of the variances (Y-axis) associated with the principle components (X-axis). 20
3.6 A PCA *Biplot* of *Iris* dataset. 20
3.7 A plot of passenger loads time series *AirPassengers*. 22
3.8 Decomposition of time series *AirPassengers*. 22
3.9 A plot of a multivariate time series recorded by sensors. 23
3.10 Segmentation boundaries of a multivariate time series. 24

Listings

3.1 Simple Linear Regression of *faithful* dataset. 12
3.2 R.squared and r of *faithful* dataset. 12
3.3 Classification tree example 15
3.4 3D Scatter Plot example 17
3.5 K-means example 18
3.6 PCA example 19

Abstract

Exploratory Data Analysis enables people to uncover underlying structure and extract influencing variables from data, especially in the case of the lack of prior research. This thesis is based on the new project in Philips Healthcare, which aims to use sensors to monitor the procedure and gain insight into the collected data. The goal of this thesis is to apply *Exploratory Data Analysis* to the qualitative and quantitative dataset, to answer the research questions on the characteristics of the users, patterns of their behavior, and at the same time on the influence of environmental factors. Multiple data mining models have been applied in the thesis, including *Linear Regression*, *Classification Tree*, *K-means Clustering*, *Principal Components Analysis* and *Time Series Analysis*. Furthermore, the differences between the observations and the similarities between variables have been visualized, compared and explained. The algorithm developed to extract attributes and segment time series has been implemented. The insight also discloses the shortcomings and limitations, and thus this thesis can be used as a base for future development.

Chapter 1

Introduction

1.1 Background

This document is based on my graduation project at Philips Design Eindhoven and Leiden Institute of Advanced Computer Science. I joined Philips Design as an intern data analyst in April 2015, during my internship, I have learned domain knowledge in a business context, supported the team in the development of a new product prototype, applied data analysis techniques to real-world datasets, and delivered insights in visualization.

In the context of *Data Driven Design*[1], Philips expects data analysts to discover useful information, to understand the customers and to push forward the design process. However, little quantitative data has yet been collected or explored in related work. In such a condition, my work is exploratory rather than confirmatory.

1.2 Exploratory Data Analysis

With the fast development of sensor networks and computers, the amount of available data grows in an explosive way, and thus the problem of managing and understanding the data becomes more difficult and challenging. *Exploratory Data Analysis* [1] has been developed to enable people to make useful discoveries from data, especially in case of the lack of prior hypotheses or researches. The primary goal is to maximize insight into a data set [2]:

- uncover underlying structure;
- extract influencing variables;
- detect outliers and anomalies;
- test underlying assumptions;
- find good-fitting models;
- optimize parameters.

Steps of *Exploratory Data Analysis* are described in Figure 1.1. Generally, the process starts with defining the problem, before moving into the step of collecting data. When we make sure the quality of data is ready for further analysis by pre-processing, we can apply a variety of data mining techniques. The final step is data visualization, where the results are presented in visual plots and graphs.

[1] *Data Driven Business Innovation in Philips.* http://www.innovationservices.philips.com/news/data-driven-business-innovation

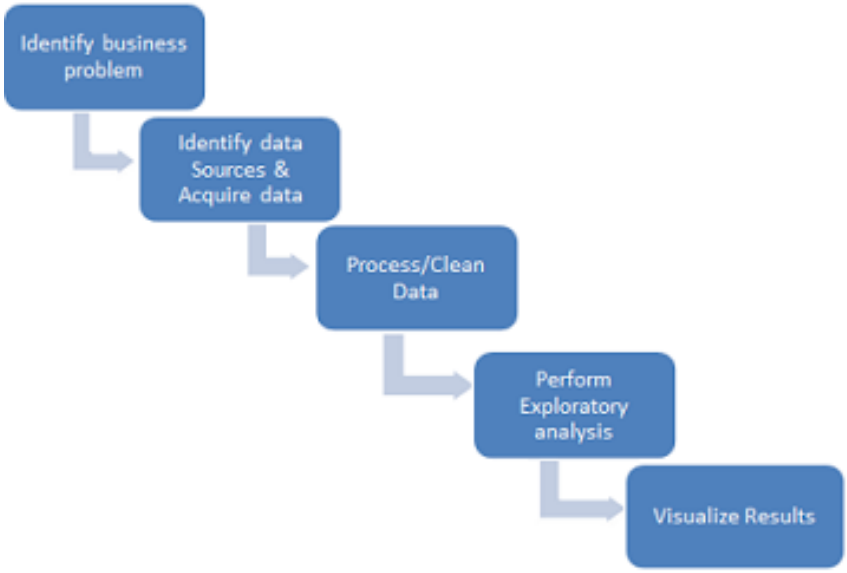


Figure 1.1: Steps of *Exploratory Data Analysis*.[2]

In this thesis, the following steps were taken in order to analyze the datasets from Philips.

1.3 Context and Research Questions

CONFIDENTIAL CONTENT

1.4 Outline

The outline of the thesis is presented as follows. In Chapter 2, we introduce the product prototypes, data sets collected from experiments, and how we pre-process them. Chapter 3 discusses the approaches including supervised/unsupervised learning techniques and time series analysis, which we use in the following section. Next, Chapter 4 describes the empirical process of selecting models, optimizing parameters and visualizing the results. In Chapter 5, we review the conclusions and discuss future work.

[2] *Data Analysis Steps.* http://www.dataperspective.info/2014/02/data-analysis-steps.html

Chapter 2

Datasets

In this Chapter, we introduce the process of data acquisition and data pre-processing. This chapter is divided into four sections. The first section starts with the overview of prototype sensors and the data flow. Next, the second section introduces the details of participant families, then the third section explains quantitative and qualitative variables that were collected from the prototypes, and the final section introduces the pre-processing process of the raw dataset.

2.1 Product prototyping

CONFIDENTIAL CONTENT

2.2 Participants

CONFIDENTIAL CONTENT

2.3 Data Acquisition

2.3.1 Quantitative Data

CONFIDENTIAL CONTENT

2.3.2 Qualitative Data

CONFIDENTIAL CONTENT

2.3.3 Raw Datasets

CONFIDENTIAL CONTENT

2.4 Data Pre-processing

CONFIDENTIAL CONTENT

Chapter 3

Approaches

In this chapter, we introduce the mining algorithms and the analysis methods used on the pre-processed data. This chapter is divided into three sections, the first one will discuss the supervised learning approach, followed by the unsupervised learning in the next section, and finally the last section will introduce and cover the multivariate time series analysis applied to our dataset.

3.1 Supervised Learning

Supervised learning is a mining task of gaining insight from labeled dataset. In supervised learning, variables can be split into two categories: predictor variables and target variable. A supervised learning algorithm takes the predictors as input and target as output, and then produces the corresponding function. The supervised learning approach is able to provide us with the insight about which predictor variables are the most influencing in regard to the corresponding target.

Supervised learning splits into two broad categories: *regression* and *classification*. Main difference between them lies in the type of output. In a regression problem, we try to analyze the results within a continuous output, in other words, we try to map the input variables to a continuous function. In a classification problem, instead, we try to analyze the results in a discrete output, meaning mapping the input variables into discrete categories.

3.1.1 Linear Regression

Definition

The goal of regression analysis is to determine the values of parameters for a function that cause the function to best fit a set of data observations. As following equation shows, regression is the process of estimating the value of a continuous target (y) as a function (F) of predictors (X = $\{x_1 , x_2 , ..., x_n\}$), a set of parameters ($\theta_1 , \theta_2 , ..., \theta_n$), and a measure of error (ε) [5].

$$y = F(X, \theta) + \varepsilon \tag{3.1}$$

The process of training a regression model involves finding the best parameter values for the function that minimize a measure of the error, e.g. *residual sum of squared error*(SS_{res}).

$$SS_{res} = \sum_{i=1}^{n} (y_i - f_i)^2 \tag{3.2}$$

There are different families of regression functions and different ways of measuring the error. In general, regression analysis could be divided into two main categories: *linear regression* and *non-linear regression*. For this work, we will focus on *linear regression*.

Given a data set $y_i, x_{i1}, ..., x_{ip}$ of n observations, a linear regression model assumes that the relationship between the target variable y_i and the predictor variables x_i is linear. This relationship is modeled through an error variable ε_i, which is unobserved random variable that adds noise to the linear relationship between the target variable and predictor variables.

$$y_i = \beta_1 x_{i1} + ... + \beta_p x_{ip} + \varepsilon_i, \qquad i = 1, ..., n \tag{3.3}$$

Often these n equations are written together in vector form as follows,

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon} \tag{3.4}$$

where $\boldsymbol{y}$ is called the *target variable*, $\boldsymbol{X}$ are called *predictor variables*, $\boldsymbol{\beta}$ is a *p-dimensional parameter vector*, and $\boldsymbol{\varepsilon}$ is called the *error term*. Elements in vector $\boldsymbol{\beta}$ are called *regression coefficients*, and they are interpreted as the partial derivatives of the target variable with respect to the predictor variables.

Coefficient of Determination

The *Coefficient of Determination* [10], or "r-squared" (denoted r^2), is a number that indicates the proportion of the variance in the target variable that is explained by the predictor variable.

Given a data set with n values $y_1, ..., y_n$, each value is associated with a modeled value $f_1, ..., f_n$. The μ_y is the mean of all y:

$$\mu_y = \frac{1}{n} \sum_{i=1}^{n} y_i \tag{3.5}$$

The *total sum of squares* is defined as:

$$SS_{tot} = \sum_{i} (y_i - \mu_y)^2, \tag{3.6}$$

And the *residual sum of squares* is defined as:

$$SS_{res} = \sum_{i} (y_i - f_i)^2 \tag{3.7}$$

The general definition of the Coefficient of Determination is,

$$r^2 = 1 - \frac{SS_{res}}{SS_{tot}} \tag{3.8}$$

- Since r^2 is a proportion, it is always a number between 0 and 1.
- If $r^2 = 1$, all of the data points fall perfectly on the regression line. The predictor x accounts for all of the variation in y.
- If $r^2 = 0$, the estimated regression line is perfectly horizontal. The predictor x accounts for none of the variation in target y.

Correlation Coefficient

Correlation Coefficient [4] (denoted r) measures the strength and the direction of a linear correlation between two variables X and Y, giving a value between $+1$ and -1 inclusive, where $+1$ is total positive correlation, 0 is no correlation, and -1 is total negative correlation. It is widely used as a measurement of the degree of linear correlation between two variables.

Correlation coefficient is the *covariance* of the two variables divided by the product of their *standard deviations*, which is is commonly represented as ρ.

$$\rho_{X,Y} = \frac{\operatorname{cov}(X,Y)}{\sigma_X \sigma_Y} \tag{3.9}$$

where $\operatorname{cov}(X,Y)$ is the *covariance*, σ_X is the *standard deviation* of X, and σ_Y is the *standard deviation* of Y.

$$\operatorname{cov}(X,Y) = \mathrm{E}[(X - \mu_X)(Y - \mu_Y)] \tag{3.10}$$

where μ_X is the *mean* of X, and E is the *expectation*.

- *Positive correlation*: If x and y have a strong positive linear correlation, r is close to $+1$. $+1$ indicates a perfect positive fit. Positive values indicate a relationship between x and y variables such that as values for x increases, values for y also increase.
- *Negative correlation*: If x and y have a strong negative linear correlation, r is close to -1. -1 indicates a perfect negative fit. Negative values indicate a relationship between x and y such that as values for x increase, values for y decrease.
- *No correlation*: If there is no linear correlation or a weak linear correlation, r is close to 0. A value near zero means that there is a random, nonlinear relationship between the two variables.

Applying Simple Linear Regression model in R

A simple linear regression model describes the relationship between two variables x and y, and can be expressed by the following equation. The numbers α and β are called parameters, and ε is the error term.

$$y = \alpha + \beta x + \varepsilon \tag{3.11}$$

Taking the dataset *faithful* as an example, the R code of fitting linear regression between the predictor variable *waiting* and the target variable *eruptions* is shown in Listing 3.1. The result is plotted in Figure 3.1.

```
# Fit linear model
fit <- lm(eruptions ~ waiting, data = faithful)
# Plot
plot(faithful$waiting, faithful$eruptions, xlab="waiting", ylab="eruptions", pch=19)
# Draw regression line
abline(fit, col="red")
```

Listing 3.1: Simple Linear Regression of *faithful* dataset.

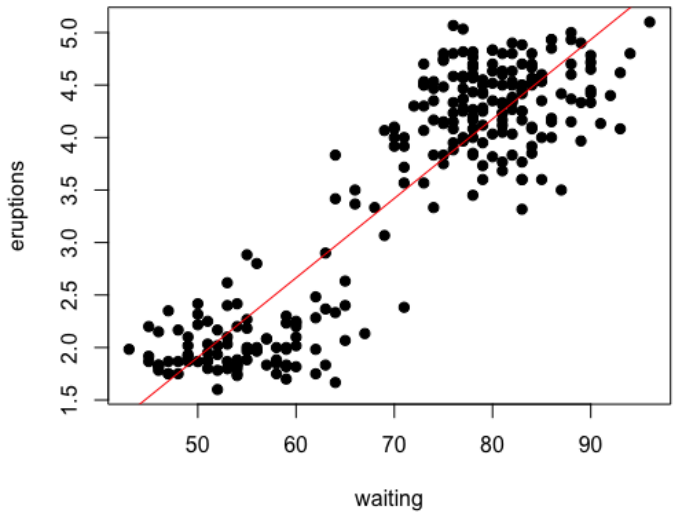


Figure 3.1: The linear regression result of *faithful* is plotted as the red line.

As seen in Listing 3.2, the *R-squared* is 0.8115 and we can say that 81% of the variation in the *eruptions* is explained by *waiting*. In *simple linear regression*, we could use the square root of R-squared as well as *cor* function to calculate *Correlation Coefficient*. The *Correlation Coefficient* r is 0.9008, which is close to 1 and we can say that there is a strong positive linear relationship between *eruptions* and *waiting*.

```
> rs <- summary(fit)$r.squared
> rs
[1] 0.81146

> r <- cor(faithful$eruptions, faithful$waiting)
> r
[1] 0.9008112

> r ^ 2
[1] 0.8114406
```

Listing 3.2: R.squared and r of *faithful* dataset.

3.1.2 Classification Tree

Definition

Classification is a data mining function that assigns items of a collection into categories. The main difference between classification and regression methods, is the target variable of classification is normally discrete variable. A model with a numerical target uses a regression algorithm, not a classification algorithm.

Generally, if there are n observations of variable Y that takes in total k values Y_1, Y_2, ..., Y_k, and p predictors X_1, ..., X_p, the goal of classification is to find a model for predicting the values of Y form new X values. The ideal solution is a partition of the X space into k disjoint sets A_1, A_2, ... A_k, such that the predicted value of Y is j if X belongs to A_j, for j = 1,2, .., k [6].

Steps

To get a classification tree, a common strategy is to grow the tree until each node contains a small number of instances then use pruning to remove nodes that do not provide additional information[7]. The steps are as follows:

1. **Building the tree**: using function *rpart* [11] to build a tree.

   Classification tree methods recursively partition the dataset one variable at a time. A tree can be grown by splitting the source set into subsets based on an attribute value. This process is repeated on each derived subset, in a recursive manner called *recursive partitioning*. The recursion is completed when the subset at a node has the same value of the target variable, or when splitting no longer adds value to the predictions. This process is called *top-down induction of decision* trees [8]. Pseudo-e for tree construction by exhaustive search are as follows,

   Algorithm 1: Tree construction by exhaustive search [6].

   (a) Start at the root node.
   (b) For each X, find the set S that minimizes the sum of the impurities of the two child nodes, and choose the split $X^* \in S^*$ that gives the minimum overall X and S.
   (c) If a stopping criterion is reached, exit. Otherwise, apply step 2 to each child node in turn.

   CART [9] follows this approach and is implemented in the *rpart* package [11], which we will use in the experiments.

2. **Examine results**: using function *printcp*, *plotcp*, $summary(fit)$ to display *cp* table and detailed results.

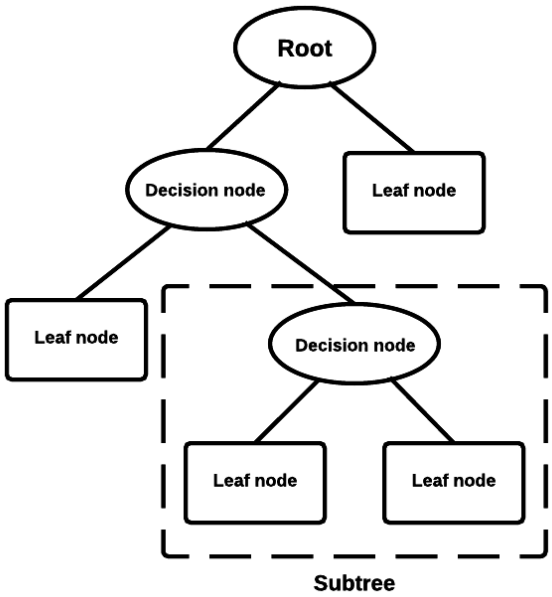


Figure 3.2: A decision tree with *Root*, *Decision node*, *Leaf node* and *Subtree*

3. **Prune the tree**: We prune the tree to avoid *over-fitting* of the data. In over-fitting, the model describes random error or noise instead of the underlying relationship. Over-fitting occurs when a model is excessively complex, such as having too many parameters relative to the number of observations.

   As shown in Figure 3.2,

   (a) *Root*: it represents entire population, which is the splitting start of the tree.
   (b) *Decision node*: a node that splits into further nodes; it is shown as a square.
   (c) *Leaf node*: a node that does not split; it is shown as a circle.
   (d) *Subtree*: a branch with all descendant nodes.

   In a classification tree, over-fitting may result in subtrees or leaves based on too few examples. Pruning of the tree is done by replacing a whole subtree by a leaf node. The replacement takes place if a decision rule establishes that the expected error rate in the subtree is greater than the single leaf node

   We start with *printcp* to examine the cross-validated error results, then select the complexity parameter associated with minimum cross-validated error $xerror$, and place it into the *prune* function. The following code fragment could automatically select the complexity parameter associated with the smallest cross-validation error [12].

```
fit$cptable[which.min(fit$cptable[,"xerror"]),"CP"]
```

4. **Visualize the tree**: using function $plot(fit)$, $text(fit)$ or other visualization packages to draw the tree. For this work $fancyRpartPlot$ from *rattle* package is applied since it provides better visual affects compared to default *plot*.

Applying classification tree in R

Taking the dataset *Iris* shown in Table 3.1 as an example, the goal of classification is to categorize the data points into *Species* with numeric variables as predictors. We apply *rpart* package to fit the classification model, and *rattle* package to visualize the result.

```
library(rpart)
library(rattle)
# load data
data(iris)
# fit model
fit <- rpart(Species~., data=iris)
# plot
fancyRpartPlot(fit)
```

Listing 3.3: Classification tree example

Sepal.Length	Sepal.Width	Petal.Length	Petal.Length	Species
5.1	3.5	1.4	0.2	setosa
5.0	3.3	1.4	0.4	setosa
7.0	3.2	4.7	1.3	versicolor
4.9	2.4	3.3	1.6	versicolor
5.8	2.7	5.1	2.0	virginica
7.1	3.0	5.9	2.3	virginica
...	...	...	...	...

Table 3.1: Part of *Iris* dataset.

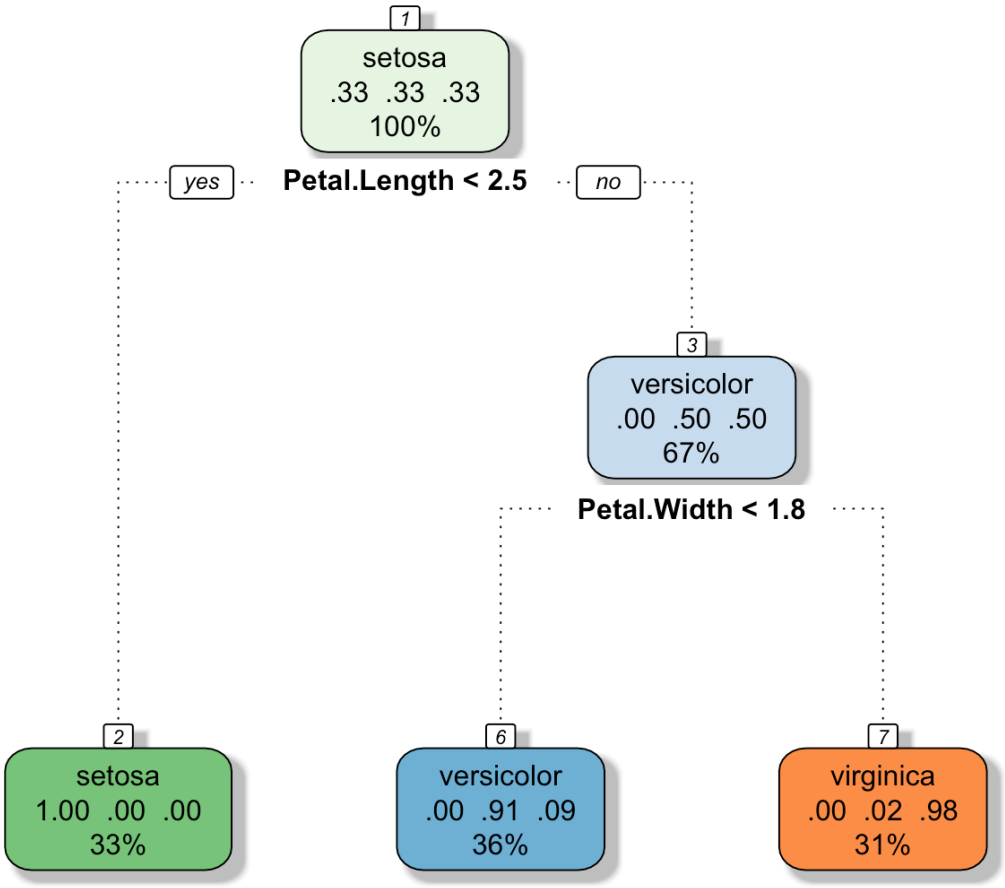


Figure 3.3: Decision tree of *iris* dataset.

From the classification tree in Figure 3.3, we could find that observations are partitioned into three target categories by two conditions, $petal.length < 2.5$ and $petal.width < 1.8$. Each node contains the percentage of components. For example, the orange node (*virginica*) shows the percents of each category in this node, which are 0% of *setosa*, 2% of *versicolor* and 98 of *virginica*. As listed in the bottom, 31% is the sample ratio out of the whole population. As seen in three leaf nodes at bottom, 100% of *setosa* (green), 91% of *versicolor* (blue) and 98% of *virginica* (orange) observations are correctly partitioned into corresponding group. This tree delivers a straightforward and compact insight of the classification result.

3.2 Unsupervised Learning

Compared to supervised learning, unsupervised learning on the other hand allows us to approach problems with little or no domain knowledge. We can derive structure from data without necessarily knowing the affects of the variables.

3.2.1 K-means Clustering

Clustering methods attempt to gain data insight by partitioning data points into disjoint groups such that data points belonging to same cluster are similar, while data points belonging to different clusters are dissimilar. One of the most popular and efficient clustering methods is the K-means method [13, 14]. K-means clustering is a widely used data clustering for unsupervised learning tasks.

Definition

$$arg\ min \sum_{i=1}^{k} \sum_{x \in S_i} \|x - \mu_i\|^2, \quad \mu_i = mean(S_i) \tag{3.12}$$

K-means clustering aims to partition m points into k clusters in which each points belongs to the cluster, such that the sum of squares from points to the assigned cluster centers is minimized. It uses k centroids of clusters to characterize the data.

3D Scatter plot in R

Taking the dataset $Iris$ as an example, firstly, we use the function *scatterplot3d* to have a direct "feel" about the dataset in three-dimensional space. The code snippet is shown in Listing 3.4 and the result is shown in Figure 3.4a.

```
# use library
library(scatterplot3d)
# define colors for each category
colors <- c("#999999", "#E69F00", "#56B4E9")
colors <- colors[as.numeric(iris$Species)]
# scatter plot
s3d <- scatterplot3d(iris[,1:3], pch = 16, color=colors)
```

Listing 3.4: 3D Scatter Plot example

Figure 3.4a shows the distribution of the dataset $Iris$ in the 3-dimensional space of $Sepal.Length$ as X, $Petal.Length$ as Y and $Sepal.Width$ as Z. The points are grouped in different colors by the categories.

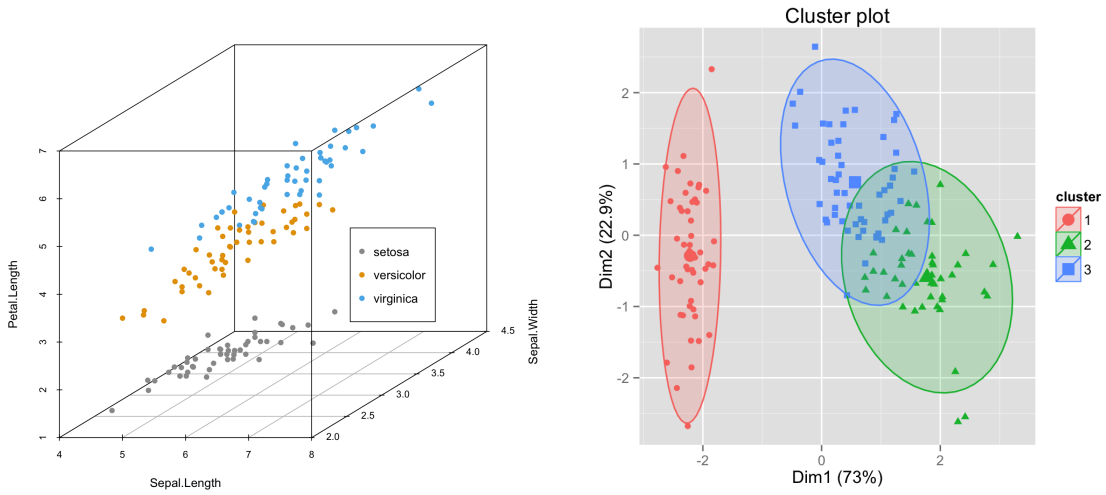
Apply K-means model in R

```
library(factoextra)
# Load the data
data(iris)
head(iris)
# Remove species column (5) and scale the data
iris.scaled <- scale(iris[, -5])
# K-means clustering
set.seed(123) # set seed
km.res <- kmeans(iris.scaled, 3, nstart = 25) # K = 3
# Visualize k-means clusters
fviz_cluster(km.res, data = iris.scaled, geom = "point",
             stand = FALSE, frame.type = "norm")
```

Listing 3.5: K-means example

The result of K-means clustering is plotted in Figure 3.4b. The red cluster locates in the left of the plot separately, while the blue and green clusters overlap each other.

Here we use the parameter $k = 3$, to cluster all points into 3 groups. The optimum choice of k is often ambiguous, depending on the distribution of data points and also the desired clustering resolution.



(a) 3D plotting of *Iris* dataset.

(b) K-means clustering of *Iris* dataset.

Figure 3.4: *Iris* visualization.

3.2.2 Principal Components Analysis

Definition

Principal Components Analysis (*PCA*) is a technique for simplifying a dataset. The aim of *PCA* is to reduce the dimensionality of multivariate data, while preserving as much of the relevant information as possible. It is a form of unsupervised learning since it relies entirely on the input data itself without reference to the corresponding target [16].

The goals of PCA are as follows [17],

1. extract the most important information from the data set.
2. compress the size of the data set by keeping only this important information.
3. simplify the description of the data set.
4. analyze the structure of the observations and the variables.

PCA transforms the data to a new coordinate system such that the new set of variables, the *Principal Components* (PC), are linear functions of the original variables. The greatest variance by any projection of the data comes to lie on the first coordinate, the second greatest variance on the second coordinate, and so on. This can be done by *Eigenvalue Decomposition* of a covariance matrix, or *Singular Value Decomposition* of a data matrix, usually after mean centering and normalizing the data matrix for each attribute [17]. In such a way, PCA projects data points into a low-dimensional space to find most influencing principal components.

Apply PCA model in R

Taking the dataset *Iris* as an example, the PCA code snippet is shown in Listing 3.6.

```
# Load data
data(iris)
# log transform
ir.log <- log(iris[, 1:4])
ir.species <- iris[, 5]

# apply PCA
ir.pca <- prcomp(ir.log,
                 center = TRUE,
                 scale. = TRUE)

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

# plot PCA biplot
library(ggbiplot)
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)
```

Listing 3.6: PCA example

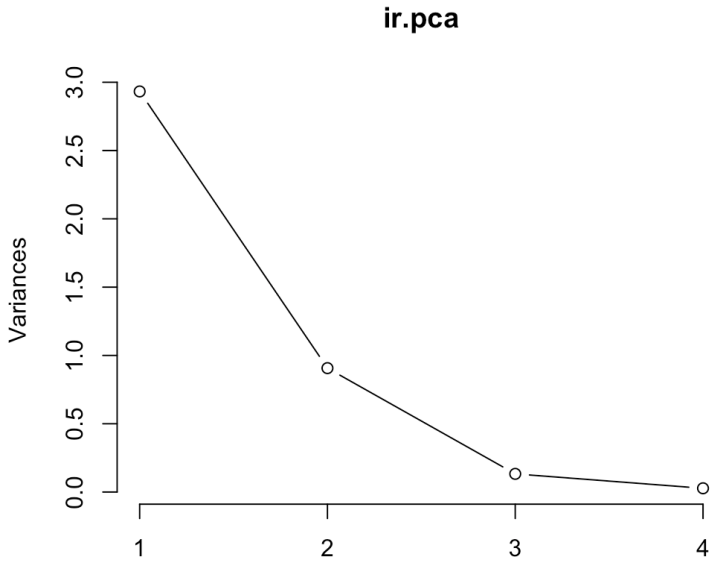


Figure 3.5: Plot of the variances (Y-axis) associated with the principle components (X-axis).

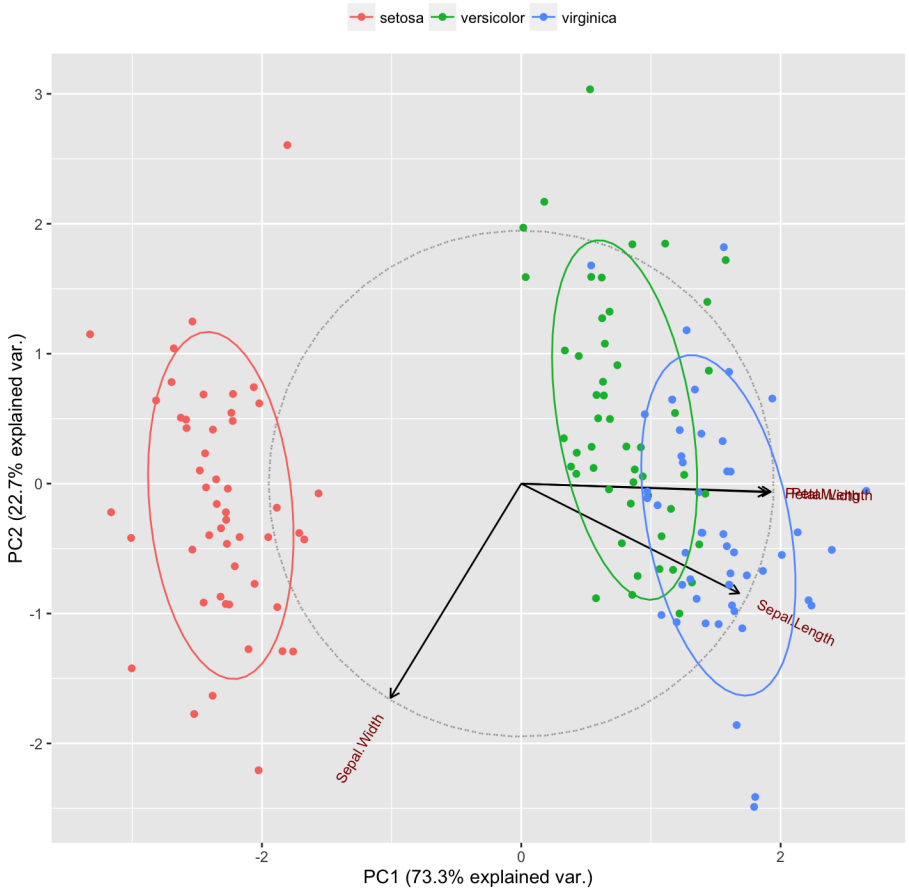


Figure 3.6: A PCA *Biplot* of *Iris* dataset.

The variance plot in Figure 3.5 is useful to decide how many PCs to retain for further analysis, for example, here we can see that the first 2 PCs explain most of the variability.

There are a variety of elements in a PCA *Biplot* as shown in Figure 3.6: axes, ellipses, arrows and a dashed circle.

The data points are colored by their groups. The x and y axes stand for two PCs. Each of them describes the proportion of explained variance, in this case *PC1* explains 73.3% of variance, and *PC2* explains 22.7%. In total, the first two PCs accounts for more than 95% of all variance. Each of the three *ellipses* covers corresponding group (red, blue and green) of data points. The arrows in the *Biplot* show how "positively" or "negatively" the original variables contribute to the PCs. The dotted circle is called a *correlation circle* [17] that help us to have a general idea of such contributions of original variables to each axis.

In our case, there are many different numeric variables which could reflect environmental factors and parenting styles of the feeds. If we take each of n variables as one dimension, then each of m feeding corresponds to one point located in this N-dimensional space like Figure 3.4a.

3.2.3 Time Series Analysis

A *Time Series* is a series of values of a quantity obtained at successive times with equal intervals between them. For example, Dow Jones Industrial Average, Gross Domestic Product, and unemployment rate are all *Time Series*. *Time Series Analysis* is the process of applying methods to analyze time series data in order to extract meaningful patterns and characteristics. Time series data have a natural temporal ordering. This makes *Time Series Analysis* distinct from the data analysis in which there is no natural ordering of the observations.

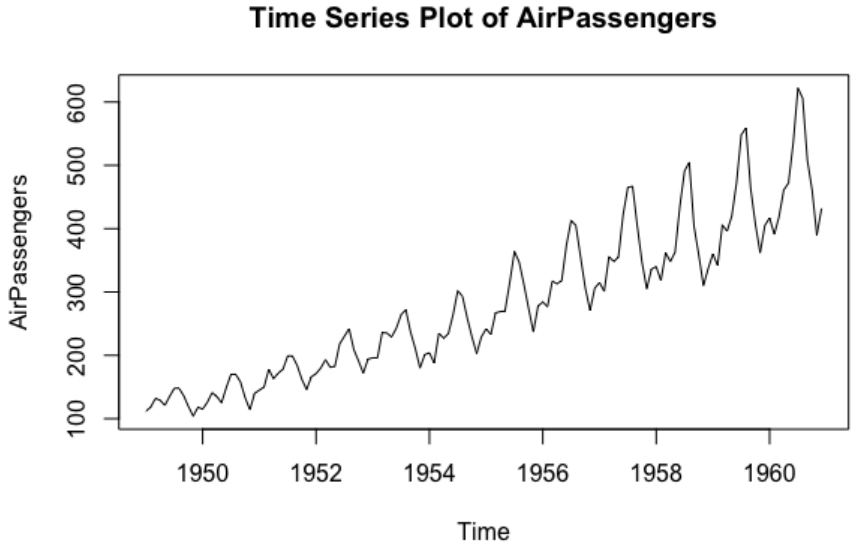


Figure 3.7: A plot of passenger loads time series *AirPassengers*.

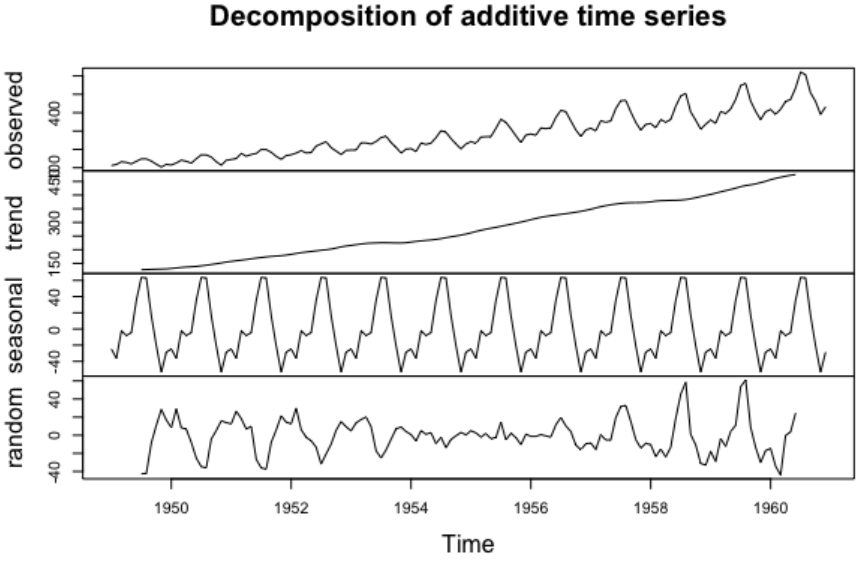


Figure 3.8: Decomposition of time series *AirPassengers*.

Figure 3.7 is a plot of time series dataset of airline passenger loads. As the plot shows, in a

long-time perspective the number of passengers rises steadily, while within a short interval it could fluctuate with peaks and valleys. Figure 3.8 shows the decomposition of this time series: the first row is the original time series, the second row is the *trend* component, the third shows *seasonal* factors, and the last row is the *remainingcomponent*. Time series analysis offers methods and algorithms to quantitatively extract such patterns, which are not easy to obtain without considering data sequences.

Plotting Multivariate Time Series

In the data flow of sensor networks, sensors record multiple types of variables as time goes by. Such data sequence can be extracted to a time series dataset with multiple variables, which is called *Multivariate Time Series*. As shown in Figure 3.9, it plots a multivariate time series dataset recorder by a sensor network. Each row stands for one variable of the time series.

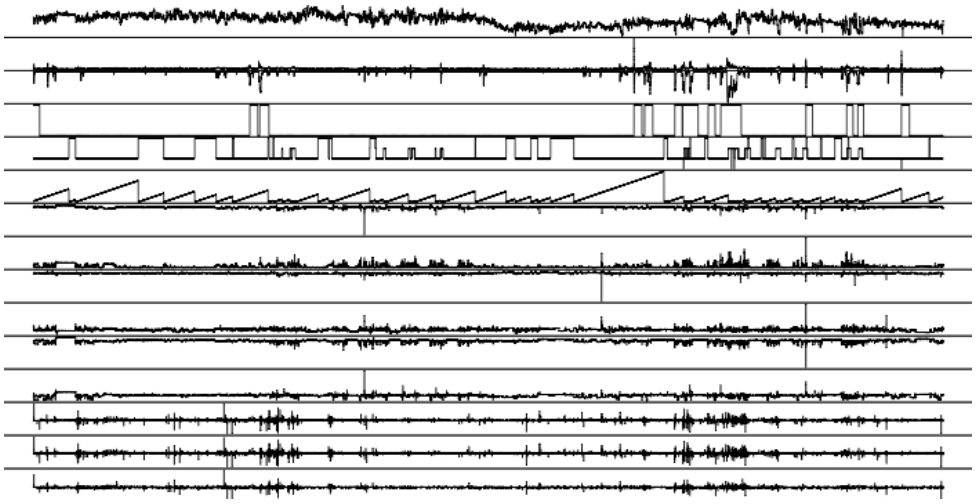


Figure 3.9: A plot of a multivariate time series recorded by sensors.

Visualizing the time series to get a "feel" for the data is usually the first and an important step to understand it.

Segmentation and Biclustering

A time series can be represented as a sequence of discrete segments of finite length. The aim of segmentation is to divide a time series into a sequence of discrete segments, in order to reveal the underlying properties of its source.

One of the most widely used approach to segment is to look for **change points** in the time-series. We may assign a segment boundary whenever there is a large jump in the average value of the signal. For example, the audio recording of a speech could be divided according to the pause between sentences or words.

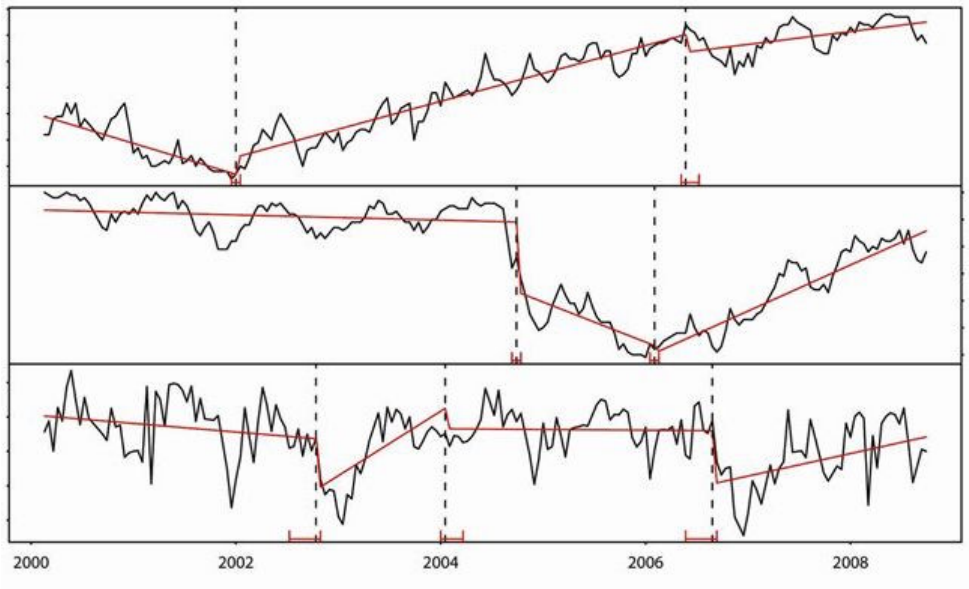


Figure 3.10: Segmentation boundaries of a multivariate time series.

An example of segmenting a time series [18] is shown in Figure 3.10. The change points of the trend component(red) are extracted, and the vertical dash lines show the boundaries of segments. As seen in this graph, this process of "cutting" the time series into small segments provides insight of trend analysis.

Generally, clustering aims to partition time series data, often the segments, into groups based on similarity or distance. For example, in the context of speak recognition, the similar segments could be clustered into one group for the same word.

Biclustering is an unsupervised approach that performs simultaneous clustering on the row and column dimensions of the data matrix [19]. Recently biclustering has been shown to be very affective in a variety of applications. A *bicluster* in Time Series is a subset of the variables exhibiting consistent patterns over a subset of the time sequence.

In our case, one complete feed contains many stages, these stages can be reflected as segments. We are interested in finding *biclusters* in the segments, and see how these biclusters reflect feeding patterns and parenting styles. The discussion is given in Chapter 4.

Chapter 4

Empirical studies

CONFIDENTIAL CONTENT

Chapter 5

Conclusions

This thesis is an overview of Exploratory Data Analysis of bottle feeding dataset from *Smart Bottle* prototypes by Philips Healthcare. We started from introducing the background and context of this thesis, and raised many research questions which is based on real world dataset and corresponds with the motivations.

The structure of this thesis follows the common steps of data analysis application, and the process reflects what we implemented in the *Smart Bottle* project. The iterative process started with the univariate analysis of quantitative data, such as feeding routine and bottle movement, then the scope was expanded to the union of qualitative and quantitative variables. The first approach in multivariate analysis is *supervised learning*, because the baby's reaction to the feed is generally reckoned to be the key element to reflect a "good" feed, to find the most influencing predictors is highly meaningful. Furthermore, *unsupervised learning* is introduced to study the inherent structure of the dataset, for example, clustering the feeds to see pattern groups, and biclustering the time series to find stages during the feeding duration. Not only the differences between the observations, but also the similarities between quantitative variables are studied.

The results from these approaches have been explained and compared, more importantly, the essence of the difference or the coincidence is studied. For example, application of *SD* and *IQR* in routine flexibility. To represent the data insight, a variety of visualization techniques have been applied: *histogram*, *scatter plot*, *heat map*, *decision tree*, *3D plot*, *clustering plot*, and*multivariate time series plot*. In terms of the improvement of the prototype, A new algorithm has been developed and implemented to detect feed breaks, and the questionnaire in the application has been enhanced with a new self-adaptive feature.

Chapter 6

Reference

[1] Tukey, John (1977), *Exploratory Data Analysis*, Addison-Wesley.

[2] *NIST/SEMATECH e-Handbook of Statistical Methods.* `http://www.itl.nist.gov/div898/handbook/`. 10/30, 2013.

[3] Pyle, D., 1999. *Data Preparation for Data Mining.* Morgan Kaufmann Publishers, Los Altos, California.

[4] Karl Pearson, 1895. *Notes on regression and inheritance in the case of two parents*, Proceedings of the Royal Society of London, 58 : 240242.

[5] Draper, N.R.; Smith, H. (1998). *Applied Regression Analysis (3rd ed.).* John Wiley. ISBN 0-471-17082-8.

[6] Loh, WeiYin. *Classification and regression trees.* Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 1.1 (2011): 14-23.

[7] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning.* Springer: 2001, pp. 269-272

[8] Quinlan, J. R., (1986). *Induction of Decision Trees.* Machine Learning 1: 81-106, Kluwer Academic Publishers

[9] Breiman L, Friedman JH, Olshen RA, Stone CJ. *Classification and Regression Trees.* CRC Press; 1984.

[10] Nagelkerke, Nico JD. *A note on a general definition of the coefficient of determination.* Biometrika 78.3 (1991): 691-692.

[11] Terry Therneau, Beth Atkinson and Brian Ripley (2015). *rpart: Recursive Partitioning and Regression Trees.* R package version 4.1-10. `https://CRAN.R-project.org/package=rpart`

[12] Everitt, B. S., and T. Hothorn. *HSAUR: a handbook of statistical analyses using R. R package. (2013).*

[13] Hartigan, J. A. and Wong, M. A. (1979). *A K-means clustering algorithm.* Applied Statistics 28, 100108.

[14] Jolliffe, I. (2002). *Principal component analysis.* Springer. 2nd edition.

[15] Sebastien Le, Julie Josse, Francois Husson (2008). *FactoMineR: An R Package for Multivariate Analysis.* Journal of Statistical Software,25(1), 1-18. 10.18637/jss.v025.i01

[16] JOLLIFFE, I.T., 2002. *Principal Component Analysis, second edition.* New York: Springer-Verlag New York, Inc.

[17] Abdi, Herv, and Lynne J. Williams. *Principal component analysis.* Wiley Interdisciplinary Reviews: Computational Statistics 2.4 (2010): 433-459.

[18] Tian, F., Fensholt, R., Verbesselt, J., Grogan, K., Horion, S., & Wang, Y. *Evaluating temporal consistency of long-term global NDVI datasets for trend analysis.* Remote Sensing of Environment, 2015, 163, 326340.

[19] Madeira, Sara C., and Arlindo L. Oliveira. *A linear time biclustering algorithm for time series gene expression data.* International Workshop on Algorithms in Bioinformatics. Springer Berlin Heidelberg, 2005.

[20] Benjamin Spock, *The Common Sense Book of Baby and Child Care*,1946

[21] Upton, Graham; Cook, Ian (1996). *Understanding Statistics.* Oxford University Press. p. 55. ISBN 0-19-914391-9.

[22] Cleveland, William S. *LOWESS: A program for smoothing scatterplots by robust locally weighted regression.* The American Statistician 35.1 (1981): 54.

[23] Jack Weiss (2010), `https://www.unc.edu/courses/2010spring/ecol/562/001/docs/lectures/lecture22.htm` Statistics for Environmental Science.

[24] Therneau, Terry M., and Elizabeth J. Atkinson. *"An introduction to recursive partitioning using the RPART routines."* (1997).

[25] Malika Charrad, Nadia Ghazzali, Veronique Boiteau, Azam Niknafs (2014). *NbClust: An R Package for Determining the Relevant Number of Clusters in a Data Set.* Journal of Statistical Software, 61(6), 1-36. `http://www.jstatsoft.org/v61/i06//`

[26] Ricardo Cachucho, Kaihua Liu, Siegfried Nijssen, Arno Knobbe (2016) *Bipeline: a Web-based Visualization Tool for Biclustering of Multivariate Time Series.* ECML-PKDD 2016 Demo Track. `http://www.ecmlpkdd2016.org/program.html`

[27] Cheng, Y. & Church, G.M. *Biclustering of Expression Data Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology*, 2000, 1, 93-103

[28] Prelic, A.; Bleuler, S.; Zimmermann, P.; Wil, A.; Buhlmann, P.; Gruissem, W.; Hennig, L.; Thiele, L. & Zitzler, E. *A Systematic Comparison and Evaluation of Biclustering Methods for Gene Expression Data Bioinformatics*, Oxford Univ Press, 2006, 22, 1122-1129

Appendix

CONFIDENTIAL CONTENT