**Name: Ashutosh Mishra**

**Roll No.: 14**

**Subject: Data Science**

|  |  |  |  |
| --- | --- | --- | --- |
| **Index** | | | |
| **Sr.No.** | **Date** | **Title** | **Page.**  **No.** |
| 1 |  | Practical of Simple/Multiple Linear Regression. | 3 |
| 2 |  | Practical of Time-series forecasting | 7 |
| 3 |  | Practical of Time Series Analysis. | 11 |
| 4 |  | Practical of k-means Clustering. | 14 |
| 5 |  | Practical of Logistics Regression. | 18 |
| 6 |  | Practical of Decision Tree. | 25 |
| 7 |  | Practical of Hypothesis Testing. | 28 |
| 8 |  | Practical of Analysis of Variance. | 31 |
| 9 |  | Practical of Principle Component Analysis. | 35 |

**Practical 1**

**Aim: Practical of Simple/Multiple Linear Regression.**

**Description:**

* A Supervised technique used for predicting the value when the data is continuous or real valued is known as regression. Regression consists variable know as Y or output variable and Independent variable known as X or input variable.
* Examples of the task in which the regression can be used are:
  + Predicting the house price.
  + Predicting age of person.
  + Predicting nationality of person.
  + Predicting stock price of the company etc.
* Some of the Important features of linear regression are as below:
  + Linear regression is a fast and easy to model technique and is mainly useful when relationship is modelled and is not extremely complex.
  + Linear regression is not feasible for fewer amounts of data.
  + Linear regression is very sensitive to outliers.
  + Regression is based on hypothesis and it can be linear, quadratic, polynomial, non-linear etc.
  + Regression is divided mainly into two types which are:
    - Simple Regression : It has only one feature and it is simple to use and develop. Simple regression is further classified into simple linear regression and simple non-linear regression.
    - Multiple Regression : It has two or more than features and it is little complex to develop with respect to simple regression.

**Source Code:**

#initializing height vector

height<-c(102,117,105,141,135,115,138,144,137,100,131,119,115,121,113)

#initializing weight vector

weight<-c(61,46,62,54,60,69,51,50,46,64,48,56,64,48,59)

#generate regression where height act as an linear predictor for weight

student<-lm(weight~height)

student

#making a prediction using above linear regression model

predict(student,data.frame(height=199),interval="confidence")

#plot regression model

plot(student)

**Output:**

**Console Output:**

student<-lm(weight~height)

> student

Call:

lm(formula = weight ~ height)

Coefficients:

(Intercept) height

93.5530 -0.3084

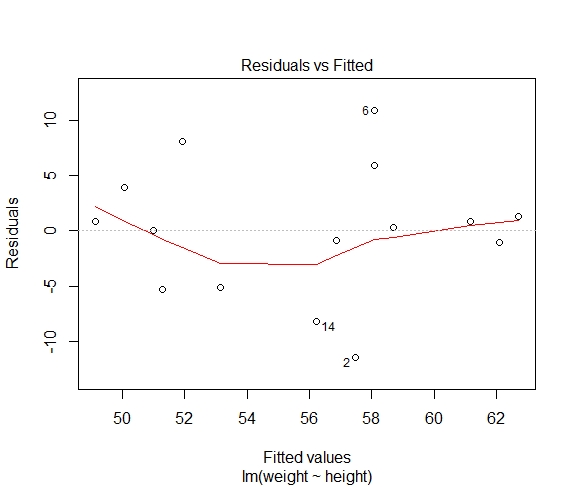
> #making a prediction using above linear regression model

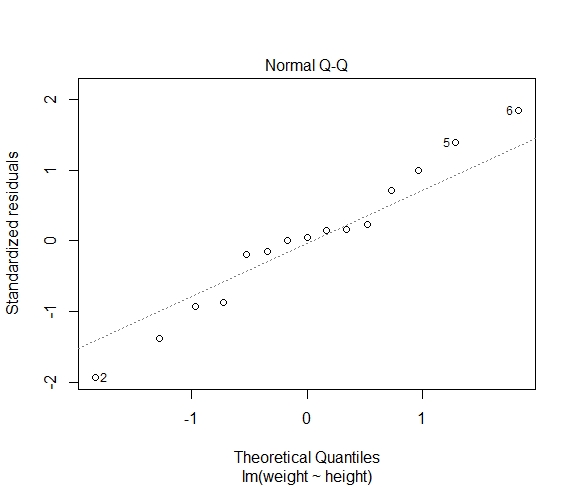
> predict(student,data.frame(height=199),interval="confidence")

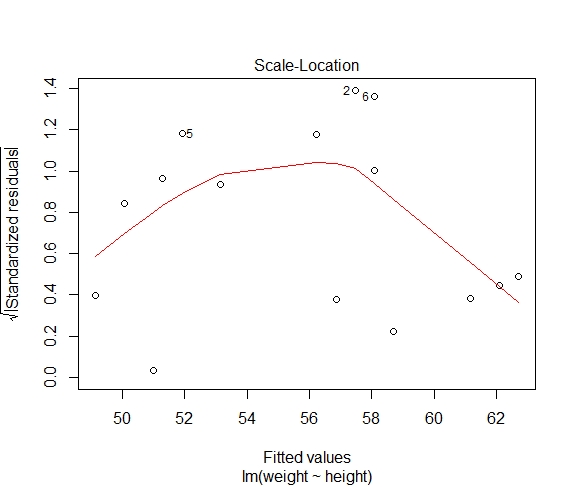
fit lwr upr

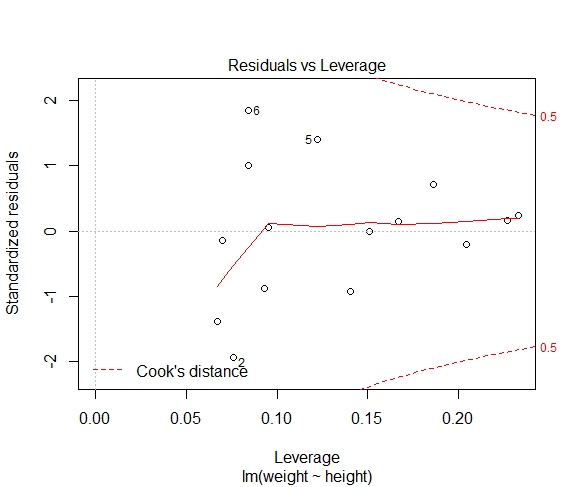
1 32.18165 13.07863 51.28468

**Plots:**

****

****

****

****

**Practical 2**

**Aim: Practical of Time-series forecasting.**

**Description:**

* Time series analysis consists of set of methods used to analyze various data facts or statistis from various characteristics of the data.
* Time series analysis is uscd for continuous data for example cconomic growth of an organization, share prices, sales temperature, weather etc.
* Time series analysis model has time factor "t" as an independent variable and the target is a dependentvariable denoted by Y The output from the time scries model is a predicted value of Y at the given titne t.Time series is the process of recording of the data at regular interval of time.
* Time Series Components**:** There are various time series components which are as follows:
  + Trend:
    - It is considered to behavior of the feature at a particular amount of time, it can be categorized as increasingtrend, decreasing trend or constant trend.
    - When the particular feature value increases in particular amount of time it is increasing trend, similarly if itdecreases it is decreasing trend and when it does not change over the period of time then it is constant trend.
  + Seasonality**:**
    - Seasonalitya pattern which repcats at the constant frcquency. For example here the demand for the umbrellas will be in rainy season only.
  + Cycles**:**
    - Cycles are type of seasonality pattern but is does not repeat at regular frequency. Cycle can be generallyconsidered as the task completion time.
    - For example, in iterative model of software engincering every iteration can have different time requirement,but the crvery task has to undergo all stages in a single iteration.
* Forecasting:
  + It is the processes of making prediction of the future based on the present and the past data most commonlyby analysis of trends. Prediction is similar term to the forecasting but not cxactly the same.

**Source Code:**

#loading dataset "AirPassengers

data(AirPassengers)

class(AirPassengers)

#returns starting interval

start(AirPassengers)

#returns end interval

end(AirPassengers)

#retunrs number of observations before pattern repeats

frequency(AirPassengers)

summary(AirPassengers)

#plot AirPassengers

plot(AirPassengers)

#adds a linear model to above plot

abline(reg=lm(AirPassengers~time(AirPassengers)))

#cycle(AirPassengers)

#computing mean for data subsets in AirPassengers and plotting it

plot(aggregate(AirPassengers,FUN=mean))

#generate a boxplot for AirPassengers data on monthly basis

boxplot(AirPassengers~cycle(AirPassengers))

**Output:**

**Console Output:**

> start(AirPassengers)

[1] 1949 1

>

> #returns end interval

> end(AirPassengers)

[1] 1960 12

>

> #retunrs number of observations before pattern repeats

> frequency(AirPassengers)

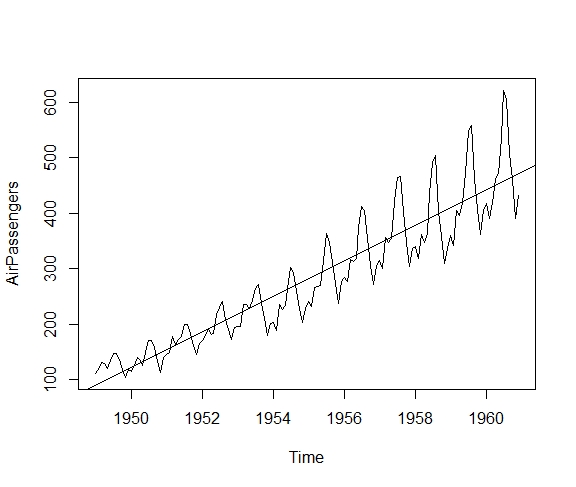
[1] 12

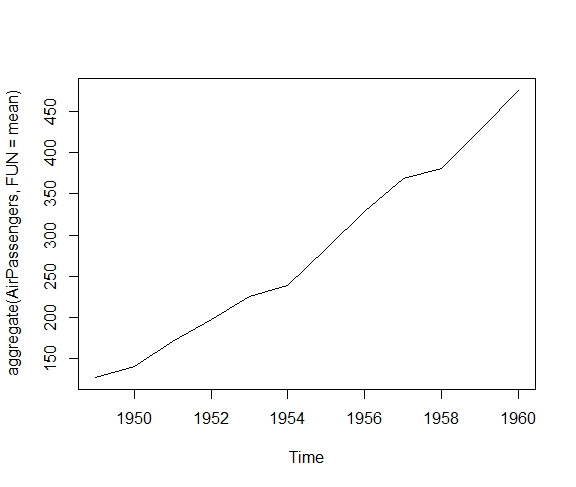
> summary(AirPassengers)

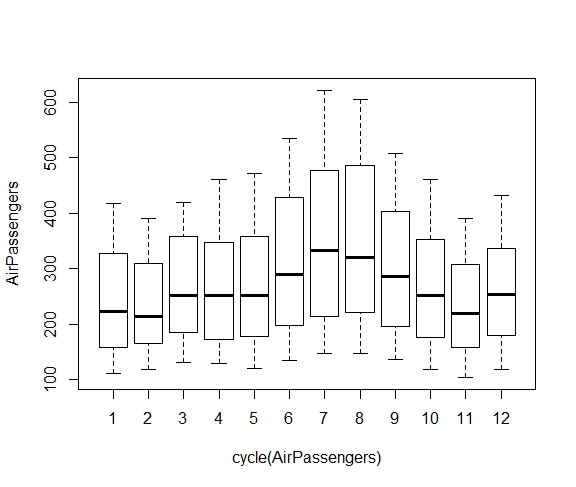
Min. 1st Qu. Median Mean 3rd Qu. Max.

104.0 180.0 265.5 280.3 360.5 622.0

**Plots:**

****

****

****

**Practical 3**

**Aim: Practical of Time Series Analysis.**

**Description:**

* Time series analysis consists of set of methods used to analyze various data facts or statistis from various characteristics of the data.
* Time series analysis is uscd for continuous data for example cconomic growth of an organization, share prices, sales temperature, weather etc.
* Time series analysis model has time factor "t" as an independent variable and the target is a dependentvariable denoted by Y The output from the time scries model is a predicted value of Y at the given titne t.Time series is the process of recording of the data at regular interval of time.
* Time Series Components**:** There are various time series components which are as follows:
  + Trend:
    - It is considered to behavior of the feature at a particular amount of time, it can be categorized as increasingtrend, decreasing trend or constant trend.
    - When the particular feature value increases in particular amount of time it is increasing trend, similarly if itdecreases it is decreasing trend and when it does not change over the period of time then it is constant trend.
  + Seasonality**:**
    - Seasonalitya pattern which repcats at the constant frcquency. For example here the demand for the umbrellas will be in rainy season only.
  + Cycles**:**
    - Cycles are type of seasonality pattern but is does not repeat at regular frequency. Cycle can be generallyconsidered as the task completion time.
    - For example, in iterative model of software engincering every iteration can have different time requirement,but the crvery task has to undergo all stages in a single iteration.
* Forecasting:
  + It is the processes of making prediction of the future based on the present and the past data most commonlyby analysis of trends. Prediction is similar term to the forecasting but not cxactly the same.

**Source Code:**

# Get the data points in form of a R vector.

rainfall <- c(799,1174.8,865.1,1334.6,635.4,918.5,685.5,998.6,784.2,985,882.8,1071)

# Convert it to a time series object.

rainfall.timeseries <- ts(rainfall,start = c(2012,1),frequency = 12)

# Print the timeseries data.

print(rainfall.timeseries)

# Give the chart file a name.

png(file = "rainfall.png")

# Plot a graph of the time series.

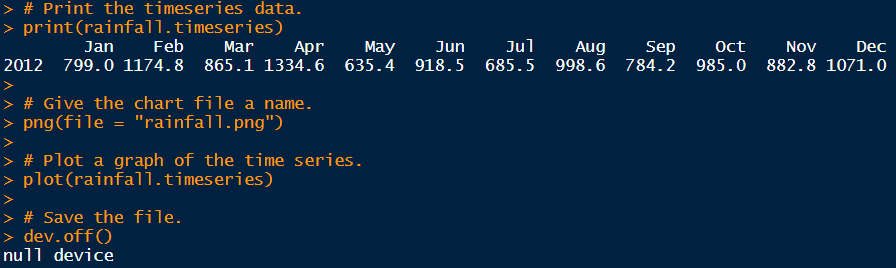
plot(rainfall.timeseries)

# Save the file.

dev.off()

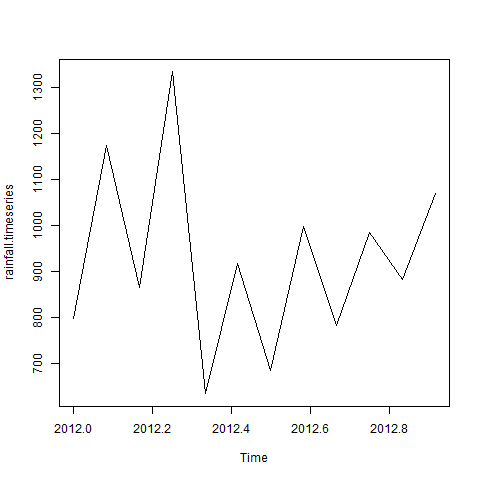
**Output:**

**Console Output:**



**Plots:**

**rainfall.png**

****

**Practical 4**

**Aim: Practical of k-means Clustering.**

**Description:**

* Some of the Key features of K-mneans are as follow:
  + K-means is an exploratory data analysis technique.
  + Implements nonhierarchical method of grouping objects logether.
  + K-means delermines the centroid using Euclidean method for distance calculation.
  + After calculating minimum distances groups of objects are created by considering the minimum distance.
  + K-means clustering is a type of unsupervised learning, which is used when you have unlabeled data (i.e.,data without defined categories or groups).
  + The goal of this algonthm is to find groups in the data, with the number of groups represented by thevariable K. The algorithm works iteratively to assign each data point to one of K groups based on thefeatures that are provided.
  + Data pointsare clustered based on feature similarity. The results of the K-means clustering algorithm are:
    - The centroids of the K clusters, which can be used to label new data
    - Labels for the training data (each data point is assigned to a single cluster)Rather than defining groups before looking at the data, clustering allows you to find and analyze thegroups that have formed organically. The "Choosing K" section below describes how the number ofgroups can be determined.
  + Each centroid of a cluster is a collection of feature values which define the resulting groups. Examining thecentroid feature weights can be used to qualitatively interpret what kind of group each cluster represents.
  + This introduction to the K-means clustering algorithm covers:
    - Common busincss cases where K-means is used.
    - The steps involved in running the algorithm.
    - A Python example using delivery fleet data.

**Source Code:**

#loading dataset "iris"

data(iris)

#returns the variables names in dataset iris

names(iris)

#stores the subset of iris dataset without Species variable in symbol new\_data

new\_data<-subset(iris,select=c(-Species))

#forming clusters in dataset new\_data using kmeans() where 3 incates number cluster to be formed

cl<-kmeans(new\_data,3)

#assigning new\_data values to data

data<-new\_data

#apply function to all the elements of data and store the resulting vector in wss

wss<-sapply(1:15,function(k){kmeans(data,k)$tot.withinss})

#plot values in wss vector

plot(1:15,wss,type="b",pch=19,frame=FALSE,xlab="Number of clusters K",ylab="Total within-clusters sums of squares")

#importing cluster package

library(cluster)

#plotting clusters computed by kmeans function

clusplot(new\_data,cl$cluster,color=TRUE,shade=TRUE,labels=2,lines=0)

# cl$cluster

# cl$centers

#hierarchical cluster analysis for dissimilarity structure of 3rd and 4th variable of iris dataset

clusters<-hclust(dist(iris[,3:4]))

#plotting Cluster Dendrogram

plot(clusters,cex=0.65)

#Dividing Dendrogram into 3 grounps

clusterCut<-cutree(clusters,3)

#printing the values of dendrogram groups w.r.t Scpecies

table(clusterCut,iris$Species)

**Output:**

**Console Output:**

names(iris)

[1] "Sepal.Length" "Sepal.Width" "Petal.Length" "Petal.Width" "Species"

table(clusterCut,iris$Species)

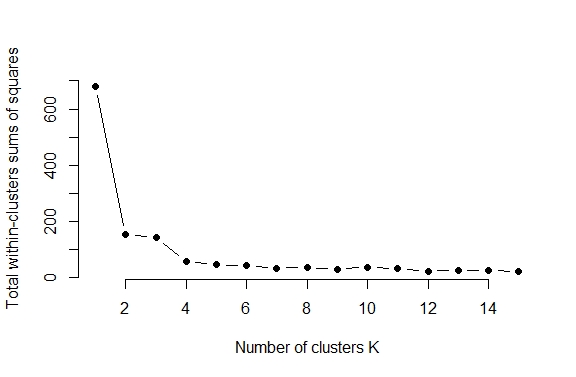
clusterCut setosa versicolor virginica

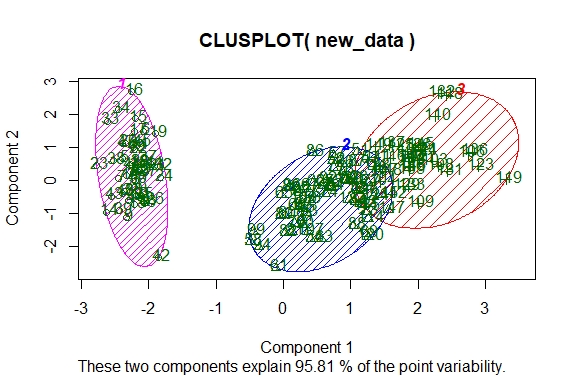
1 50 0 0

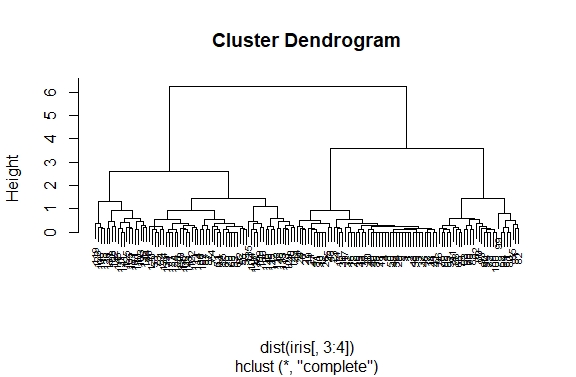
2 0 21 50

3 0 29 0

**Plots:**

****

****

****

**Practical 5**

**Aim: Practical of Logistics Regression.**

**Description:**

* Logistic Regression was used in the biological sciences in early twentieth century. It was then used in many social science applications. Logistic Regression is used when the dependent variable(target) is categorical.
* For example,
  + To predict whether an email is spam (1) or (0)
  + Whether the tumor is malignant (1) or not (0)
* Consider a scenario where we need to classify whether an email is spam or not. If we use linear regression for this problem, there is a need for setting up a threshold based on which classification can be done.
* Say if the actual class is malignant, predicted continuous value 0.4 and the threshold value is 0.5, the data point will be classified as not malignant which can lead to serious consequence in real time.
* From this example, it can be inferred that linear regression is not suitable for classification problem. Linear regression is unbounded, and this brings logistic regression into picture. Their value strictly ranges from 0 to 1.

**Source Code:**

#install ISLR package(Introduction to Statistical Learning with Applications in R)

install.packages("ISLR")

# Call the Library ISLR and data file smarket

library(ISLR)

# List the variable names

names(Smarket)

# Dimensions and summary stats for continuous variables

dim(Smarket)

summary(Smarket)

# Scatterplot Matrix

pairs(Smarket)

# To know more info about the data:

`?`(Smarket)

# Correlation matrix

cor(Smarket[, -9])

# Attach column names

attach(Smarket)

# plot multiple graphs in single plot row wise

par(mfrow=c(1,1))

# plot Volume

plot(Volume)

# Fitting GLM model

glm.fits=glm(Direction~Lag1+Lag2+Lag3+Lag4+Lag5+Volume,data=Smarket,family=binomial)

summary(glm.fits)

# Coefficients from the GLM fit

coef(glm.fits)

summary(glm.fits)$coef

# Only The 4th column in Coefficients, prob using z test

summary(glm.fits)$coef[,4]

# Generating probabilities using predict function and Type=response.

glm.probs=predict(glm.fits,type="response")

# Probabilities 1 through 10 are printed and the direction of prediction is up=1.

glm.probs[1:10]

contrasts(Direction)

# Using the same predict function on all 1250 observations, we are creating the direction of the

# market as Up if the prob exceeds 0.5 and downotherwise.

# The table of predicted vs. direction is printed too.

glm.pred=rep("Down",1250)

glm.pred[glm.probs>.5]="Up"

glm.probs[1:10]

glm.pred[1:10]

table(glm.pred,Direction)

(507+145)/1250

# Following divides the total true prediction by total to get

# the portion of correct predictions in this table.

mean(glm.pred==Direction)

# The vector train pertains to all observationd prior to year 2005.

train=(Year<2005)

Smarket.2005=Smarket[!train,]

dim(Smarket.2005) # Contains 252 observations and 9 variables

Direction.2005=Direction[!train]

#GLM Logistic regression for the training dataset using all 6 predictors.

glm.fits=glm(Direction~Lag1+Lag2+Lag3+Lag4+Lag5+Volume,data=Smarket,family=binomial,subset=train)

summary(glm.fits)

glm.probs=predict(glm.fits,Smarket.2005,type="response")

#direction calculation and mean direction of training data

glm.pred=rep("Down",252)

glm.pred[glm.probs>.5]="Up"

# Table of actual vs. predicted directions

table(glm.pred,Direction.2005)

mean(glm.pred==Direction.2005)

mean(glm.pred!=Direction.2005)

# GLM Logistic regression model using only lag1 & lag2 as predictors and the training dataset,

# generating proabilities, direction and mean direction

glm.fits=glm(Direction~Lag1+Lag2,data=Smarket,family=binomial,subset=train)

glm.probs=predict(glm.fits,Smarket.2005,type="response")

# Dividing correct predictions by total to get the correct portion predicted

mean(glm.pred==Direction.2005)

(106+35)/252

106/(106+35)

76/(36+76)

**Output:**

**Console Output:**

> names(Smarket)

[1] "Year" "Lag1" "Lag2" "Lag3" "Lag4" "Lag5" "Volume" "Today"

[9] "Direction"

> dim(Smarket)

[1] 1250 9

> summary(Smarket)

Year Lag1 Lag2 Lag3 Lag4

Min. :2001 Min. :-4.922000 Min. :-4.922000 Min. :-4.922000 Min. :-4.922000

1st Qu.:2002 1st Qu.:-0.639500 1st Qu.:-0.639500 1st Qu.:-0.640000 1st Qu.:-0.640000

Median :2003 Median : 0.039000 Median : 0.039000 Median : 0.038500 Median : 0.038500

Mean :2003 Mean : 0.003834 Mean : 0.003919 Mean : 0.001716 Mean : 0.001636

3rd Qu.:2004 3rd Qu.: 0.596750 3rd Qu.: 0.596750 3rd Qu.: 0.596750 3rd Qu.: 0.596750

Max. :2005 Max. : 5.733000 Max. : 5.733000 Max. : 5.733000 Max. : 5.733000

Lag5 Volume Today Direction

Min. :-4.92200 Min. :0.3561 Min. :-4.922000 Down:602

1st Qu.:-0.64000 1st Qu.:1.2574 1st Qu.:-0.639500 Up :648

Median : 0.03850 Median :1.4229 Median : 0.038500

Mean : 0.00561 Mean :1.4783 Mean : 0.003138

3rd Qu.: 0.59700 3rd Qu.:1.6417 3rd Qu.: 0.596750

Max. : 5.73300 Max. :3.1525 Max. : 5.733000

> # Correlation matrix

> cor(Smarket[, -9])

Year Lag1 Lag2 Lag3 Lag4 Lag5 Volume Today

Year 1.00000000 0.029699649 0.030596422 0.033194581 0.035688718 0.029787995 0.53900647 0.030095229

Lag1 0.02969965 1.000000000 -0.026294328 -0.010803402 -0.002985911 -0.005674606 0.04090991 -0.026155045

Lag2 0.03059642 -0.026294328 1.000000000 -0.025896670 -0.010853533 -0.003557949 -0.04338321 -0.010250033

Lag3 0.03319458 -0.010803402 -0.025896670 1.000000000 -0.024051036 -0.018808338 -0.04182369 -0.002447647

Lag4 0.03568872 -0.002985911 -0.010853533 -0.024051036 1.000000000 -0.027083641 -0.04841425 -0.006899527

Lag5 0.02978799 -0.005674606 -0.003557949 -0.018808338 -0.027083641 1.000000000 -0.02200231 -0.034860083

Volume 0.53900647 0.040909908 -0.043383215 -0.041823686 -0.048414246 -0.022002315 1.00000000 0.014591823

Today 0.03009523 -0.026155045 -0.010250033 -0.002447647 -0.006899527 -0.034860083 0.01459182 1.000000000

> glm.fits=glm(Direction~Lag1+Lag2+Lag3+Lag4+Lag5+Volume,data=Smarket,family=binomial)

> summary(glm.fits)

Call:

glm(formula = Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 +

Volume, family = binomial, data = Smarket)

Deviance Residuals:

Min 1Q Median 3Q Max

-1.446 -1.203 1.065 1.145 1.326

Coefficients:

Estimate Std. Error z value Pr(>|z|)

(Intercept) -0.126000 0.240736 -0.523 0.601

Lag1 -0.073074 0.050167 -1.457 0.145

Lag2 -0.042301 0.050086 -0.845 0.398

Lag3 0.011085 0.049939 0.222 0.824

Lag4 0.009359 0.049974 0.187 0.851

Lag5 0.010313 0.049511 0.208 0.835

Volume 0.135441 0.158360 0.855 0.392

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 1731.2 on 1249 degrees of freedom

Residual deviance: 1727.6 on 1243 degrees of freedom

AIC: 1741.6

Number of Fisher Scoring iterations: 3

> coef(glm.fits)

(Intercept) Lag1 Lag2 Lag3 Lag4 Lag5 Volume

-0.126000257 -0.073073746 -0.042301344 0.011085108 0.009358938 0.010313068 0.135440659

> summary(glm.fits)$coef

Estimate Std. Error z value Pr(>|z|)

(Intercept) -0.126000257 0.24073574 -0.5233966 0.6006983

Lag1 -0.073073746 0.05016739 -1.4565986 0.1452272

Lag2 -0.042301344 0.05008605 -0.8445733 0.3983491

Lag3 0.011085108 0.04993854 0.2219750 0.8243333

Lag4 0.009358938 0.04997413 0.1872757 0.8514445

Lag5 0.010313068 0.04951146 0.2082966 0.8349974

Volume 0.135440659 0.15835970 0.8552723 0.3924004

> summary(glm.fits)$coef[,4]

(Intercept) Lag1 Lag2 Lag3 Lag4 Lag5 Volume

0.6006983 0.1452272 0.3983491 0.8243333 0.8514445 0.8349974 0.3924004

> glm.probs=predict(glm.fits,type="response")

> glm.probs[1:10]

1 2 3 4 5 6 7 8 9 10

0.5070841 0.4814679 0.4811388 0.5152224 0.5107812 0.5069565 0.4926509 0.5092292 0.5176135 0.4888378

> contrasts(Direction)

Up

Down 0

Up 1

> glm.pred=rep("Down",1250)

> glm.pred[glm.probs>.5]="Up"

> glm.probs[1:10]

1 2 3 4 5 6 7 8 9 10

0.5070841 0.4814679 0.4811388 0.5152224 0.5107812 0.5069565 0.4926509 0.5092292 0.5176135 0.4888378

> glm.pred[1:10]

[1] "Up" "Down" "Down" "Up" "Up" "Up" "Down" "Up" "Up" "Down"

> table(glm.pred,Direction)

Direction

glm.pred Down Up

Down 145 141

Up 457 507

> (507+145)/1250

[1] 0.5216

> mean(glm.pred==Direction)

[1] 0.5216

> train=(Year<2005)

> Smarket.2005=Smarket[!train,]

> dim(Smarket.2005) # Contains 252 observations and 9 variables

[1] 252 9

> Direction.2005=Direction[!train]

> glm.fits=glm(Direction~Lag1+Lag2+Lag3+Lag4+Lag5+Volume,data=Smarket,family=binomial,subset=train)

> summary(glm.fits)

Call:

glm(formula = Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 +

Volume, family = binomial, data = Smarket, subset = train)

Deviance Residuals:

Min 1Q Median 3Q Max

-1.302 -1.190 1.079 1.160 1.350

Coefficients:

Estimate Std. Error z value Pr(>|z|)

(Intercept) 0.191213 0.333690 0.573 0.567

Lag1 -0.054178 0.051785 -1.046 0.295

Lag2 -0.045805 0.051797 -0.884 0.377

Lag3 0.007200 0.051644 0.139 0.889

Lag4 0.006441 0.051706 0.125 0.901

Lag5 -0.004223 0.051138 -0.083 0.934

Volume -0.116257 0.239618 -0.485 0.628

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 1383.3 on 997 degrees of freedom

Residual deviance: 1381.1 on 991 degrees of freedom

AIC: 1395.1

Number of Fisher Scoring iterations: 3

> glm.probs=predict(glm.fits,Smarket.2005,type="response")

> glm.pred=rep("Down",252)

> glm.pred[glm.probs>.5]="Up"

> table(glm.pred,Direction.2005)

Direction.2005

glm.pred Down Up

Down 77 97

Up 34 44

> mean(glm.pred==Direction.2005)

[1] 0.4801587

> mean(glm.pred!=Direction.2005)

[1] 0.5198413

> glm.fits=glm(Direction~Lag1+Lag2,data=Smarket,family=binomial,subset=train)

> glm.probs=predict(glm.fits,Smarket.2005,type="response")

> mean(glm.pred==Direction.2005)

[1] 0.4801587

> (106+35)/252

[1] 0.5595238

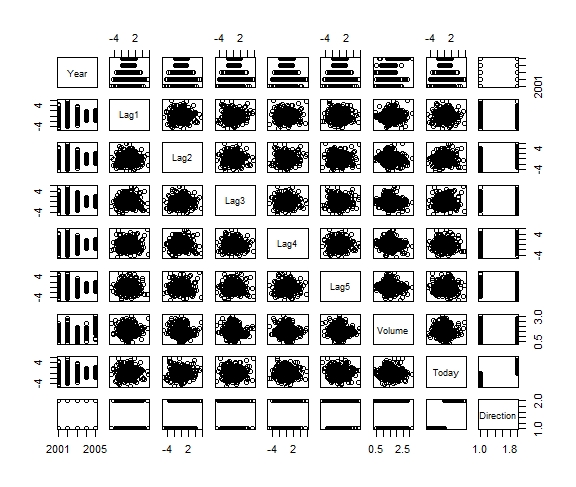
> 106/(106+35)

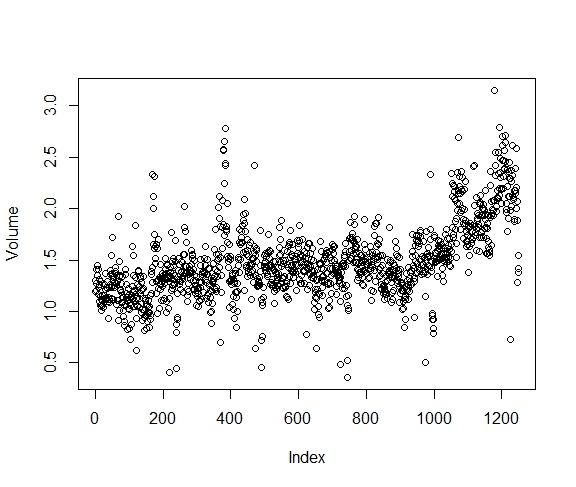
[1] 0.751773

> 76/(36+76)

[1] 0.6785714

**Plots:**

****

****

**Practical 6**

**Aim: Practical of Decision Tree.**

**Description:**

* A Decision Tree is an algorithm used for supervised learning problems such as classification or regression. A decision tree or a classification tree is a tree in which cach internal (nonleaf) node is labeled with an input feature.
* The arcs coming from a node labeled with a feature are labeled with cach of the possible values of the feature. Each leaf of the tree is labeled with a class or a probability distribution over the classes.
* A tree can be "learned" by splitting the source set into subsets based on attribute value test. This process is repeated on cach derived subset in a recursive manner called recursive partitioning.
* The recursion is completed when the subset at a node has all the same value of the target variable, or when splitting no longer adds value to the predictions.
* This process of top-down induction of decision trees is an example of a greedy algorithm, and it is the most common strategy for learning decision trees. Decision trees used in data mining are of two main types
  + Classification tree : when the response is a nominal variable, for example if an email is spam or not.
  + Regression tree : when the predicted outcome can be considered a real number (e.g. the salary of a worker).
* Decision trees are a simple method, and as such has some problems. One of this issues is the high variance in the resulting models that decision trees produce. In order to alleviate this problem, ensemble methods of decision trees were developed. There are two groups of ensemble methods currently used extensively
  + Bagging decision trees : These trees are used to build multiple decision trees by repeatedly resampling training data with replacement, and voting the trees for a consensus prediction. This algorithm has been called random forest.
  + Boosting decision trees : Gradient boosting combines weak learners; in this case, decision trees into a single strong learner, in an iterative fashion. It fits a weak tree to the data and iteratively keeps fitting weak learners in order to correct the error of the previous model.

**Source Code:**

# Load the party package. It will automatically load other

# dependent packages.

install.packages("party")

library(party)

# Create the input data frame.

input.dat <- readingSkills[c(1:105),]

# Give the chart file a name.

png(file = "decision\_tree.png")

# Create the tree.

output.tree <- ctree(

nativeSpeaker ~ age + shoeSize + score,

data = input.dat)

# Plot the tree.

plot(output.tree)

# Save the file.

dev.off()

**Output:**

**Console Output:**

> # Load the party package. It will automatically load other

> # dependent packages.

> install.packages("party")

WARNING: Rtools is required to build R packages but is not currently installed. Please download and install the appropriate version of Rtools before proceeding:

https://cran.rstudio.com/bin/windows/Rtools/

Installing package into ‘C:/Users/VRUTIKA/Documents/R/win-library/3.6’

(as ‘lib’ is unspecified)

trying URL 'https://cran.rstudio.com/bin/windows/contrib/3.6/party\_1.3-3.zip'

Content type 'application/zip' length 901670 bytes (880 KB)

downloaded 880 KB

package ‘party’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in

C:\Users\VRUTIKA\AppData\Local\Temp\Rtmp88ZbXh\downloaded\_packages

> library(party)

Loading required package: grid

Loading required package: mvtnorm

Loading required package: modeltools

Loading required package: stats4

Loading required package: strucchange

Loading required package: zoo

Attaching package: ‘zoo’

The following objects are masked from ‘package:base’:

as.Date, as.Date.numeric

Loading required package: sandwich

Warning messages:

1: package ‘party’ was built under R version 3.6.2

2: package ‘strucchange’ was built under R version 3.6.2

3: package ‘zoo’ was built under R version 3.6.2

4: package ‘sandwich’ was built under R version 3.6.2

>

> # Create the input data frame.

> input.dat <- readingSkills[c(1:105),]

>

> # Give the chart file a name.

> png(file = "decision\_tree.png")

>

> # Create the tree.

> output.tree <- ctree(nativeSpeaker ~ age + shoeSize + score, data = input.dat)

>

> # Plot the tree.

> plot(output.tree)

>

> # Save the file.

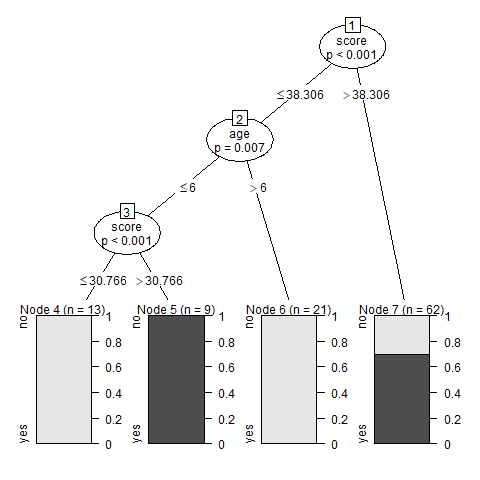
> dev.off()

null device

1

**Plots:**

**decision\_tree.png**

****

**Practical 7**

**Aim: Practical of Hypothesis Testing.**

**Description:**

* Hypothesis Tests, or Statistical Hypothesis Testing, is a technique used to compare two datasets, or a sample from a dataset. It is a statistical inference method so, in the end of the test, you'll draw a conclusion - you'll infer something- about the characteristics of what you're comparing.
* A statistical hypothesis is an assumption made by the researcher about the population of data collected for any experiment. It is not mandatory for this assumption to be true every time. Hypothesis testing is, in a way, the formal way of validating the hypothesis made by the researcher.
* In order to validate a hypothesis, it will consider the entire population into account. However, this is not possible practically. Thus, to validate a hypothesis, it will use random samples from a population. On the basis of the result from testing over the sample data, it either selects or rejects the hypothesis
* Statistical Hypothesis can be categorized into 2 types as below:
* Null Hypothesis : Hypothesis lests are used to test the validity of a claim that is made about a population. This claim that's on trial, in essence, is called the null hypothesis. The null hypothesis testing is denoted by H0.
* Alternative Hypothesis : The alternative hypothesis is the one you would believe if the null hypothesis is concluded to be untrue. The evidence in the trial is your data and the statistics that go along with it. The alternative hypothesis testing is denoted by H1 or Ha.

**Source Code:**

# generate a sequence(1,2,3...19,20)

dataf<-seq(1,20,by=1)

dataf

#Calculate mean

mean(dataf)

#Calculate Standard Deviation

sd(dataf)

# x = dataf

#a (non-empty) numeric vector of data values.

#alternative a character string specifying the alternative hypothesis, must be one of "two.sided" (default),

#"greater" or "less". You can specify just the initial letter.

#mu = 10

#a number indicating the true value of the mean (or difference in means if you are

#performing a two sample test).

a<-t.test(dataf,alternate="two.sided",mu=10,conf.int=0.95)

a

# print p-value and statistics for test

a$p.value

a$statistic

# t-test formula

(10.5-10)/(sd(dataf)/sqrt(length(dataf)))

length(dataf)=1

length(dataf)

dataf

dataf<-seq(1,20,by=1)

length(dataf)-1

**Output:**

**Console Output:**

> data("warpbreaks")

> head(warpbreaks)

breaks wool tension

1 26 A L

2 30 A L

3 54 A L

4 25 A L

5 70 A L

6 52 A L

> summary(warpbreaks)

breaks wool tension

Min. :10.00 A:27 L:18

1st Qu.:18.25 B:27 M:18

Median :26.00 H:18

Mean :28.15

3rd Qu.:34.00

Max. :70.00

>

> # Fit an analysis of variance model by a call to lm for each stratum.

> #formula = breaks~wool+tension

> #A formula specifying the model.

> #data = warpbreaks

> #A data frame in which the variables specified in the formula will be found.

> Model\_1<-aov(breaks~wool+tension,data=warpbreaks)

> summary(Model\_1)

Df Sum Sq Mean Sq F value Pr(>F)

wool 1 451 450.7 3.339 0.07361 .

tension 2 2034 1017.1 7.537 0.00138 \*\*

Residuals 50 6748 135.0

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

>

> plot(Model\_1)

Hit <Return> to see next plot: Model\_2<-aov(breaks~wool+tension+wool:tension,data=warpbreaks)

Hit <Return> to see next plot: summary(Model\_2)

Hit <Return> to see next plot: plot(Model\_2)

Hit <Return> to see next plot: train=(Year<2005)

> # generate a sequence(1,2,3...19,20)

> dataf<-seq(1,20,by=1)

> dataf

[1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20

>

> #Calculate mean

> mean(dataf)

[1] 10.5

>

> #Calculate Standard Deviation

> sd(dataf)

[1] 5.91608

>

> # x = dataf

> #a (non-empty) numeric vector of data values.

> #alternative a character string specifying the alternative hypothesis, must be one of "two.sided" (default),

> #"greater" or "less". You can specify just the initial letter.

> #mu = 10

> #a number indicating the true value of the mean (or difference in means if you are

> #performing a two sample test).

>

> a<-t.test(dataf,alternate="two.sided",mu=10,conf.int=0.95)

> a

One Sample t-test

data: dataf

t = 0.37796, df = 19, p-value = 0.7096

alternative hypothesis: true mean is not equal to 10

95 percent confidence interval:

7.731189 13.268811

sample estimates:

mean of x

10.5

>

> # print p-value and statistics for test

> a$p.value

[1] 0.7096465

> a$statistic

t

0.3779645

>

> # t-test formula

> (10.5-10)/(sd(dataf)/sqrt(length(dataf)))

[1] 0.3779645

>

> length(dataf)=1

> length(dataf)

[1] 1

> dataf

[1] 1

> dataf<-seq(1,20,by=1)

> length(dataf)-1

[1] 19

**Practical 8**

**Aim: Practical of Analysis of Variance.**

**Description:**

* It is property of different predictive models with a lower bias for parameter estimation have a higher variance for given dataset and vice versa.
* Bias: It is an error from the erroneous assumptions made during the learning of an algorithm. Higher bias can lead to missing of the relevant data of feature needed for the targeted value 1 other words it leads to underfitting.
* Variance : It is an crror form the sensitivity of an algorithm where small fluctuation of samples in thetraining set can lead to an error. High variance in an algorithm can lead to generation of random noise in thetraining data and can deviate the output.
* In other words it leads to overfitting.
* Bias-Variance trade off is generally faced in supervised nlgorithms due to which the accuracy andgeneralization both cannot be ndopted in the model.
* Any model can be bad or not optimal because of two main reasons:
  + It is not accurate.
  + It does not match the data well.
* The reason for the first is bias and other is variance. If models are made complex then it leads toImprovement in the bias but such models are very costly which leads to higher variance, whercas when themodel is made more specific to the data then the variance will be reduced but on the other hand it leads tohigher bias.
* Variance is the amount that the estimate of the target functions which will change if different training datawas used. Algorithm should have some variance.
* Low variance provides small changes to tbe estimate of the target functions with the changes to thc trainingdataset. High variance provides large changcs to the estimale of the target function with changes to thetraining set. Whenever the model is choosing with low complexity and low variance automatically the highvariance is introduced.

**Source Code:**

# loading warpbreaks

data("warpbreaks")

head(warpbreaks)

summary(warpbreaks)

# Fit an analysis of variance model by a call to lm for each stratum.

#formula = breaks~wool+tension

#A formula specifying the model.

#data = warpbreaks

#A data frame in which the variables specified in the formula will be found.

Model\_1<-aov(breaks~wool+tension,data=warpbreaks)

summary(Model\_1)

plot(Model\_1)

#formula = breaks~wool+tension+wool:tension

Model\_2<-aov(breaks~wool+tension+wool:tension,data=warpbreaks)

summary(Model\_2)

plot(Model\_2)

**Output:**

**Console Output:**

> # loading warpbreaks

> data("warpbreaks")

> head(warpbreaks)

breaks wool tension

1 26 A L

2 30 A L

3 54 A L

4 25 A L

5 70 A L

6 52 A L

> summary(warpbreaks)

breaks wool tension

Min. :10.00 A:27 L:18

1st Qu.:18.25 B:27 M:18

Median :26.00 H:18

Mean :28.15

3rd Qu.:34.00

Max. :70.00

>

> # Fit an analysis of variance model by a call to lm for each stratum.

> #formula = breaks~wool+tension

> #A formula specifying the model.

> #data = warpbreaks

> #A data frame in which the variables specified in the formula will be found.

> Model\_1<-aov(breaks~wool+tension,data=warpbreaks)

> summary(Model\_1)

Df Sum Sq Mean Sq F value Pr(>F)

wool 1 451 450.7 3.339 0.07361 .

tension 2 2034 1017.1 7.537 0.00138 \*\*

Residuals 50 6748 135.0

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

>

> plot(Model\_1)

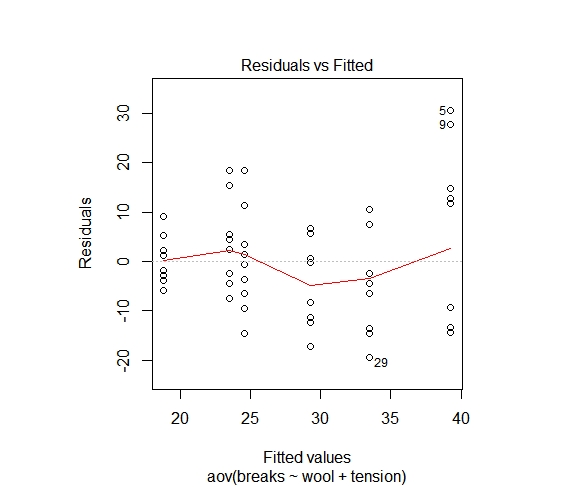
Hit <Return> to see next plot: Model\_2<-aov(breaks~wool+tension+wool:tension,data=warpbreaks)

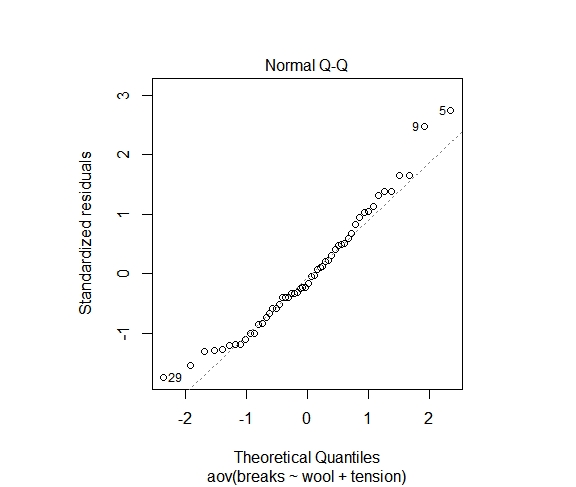
Hit <Return> to see next plot: summary(Model\_2)

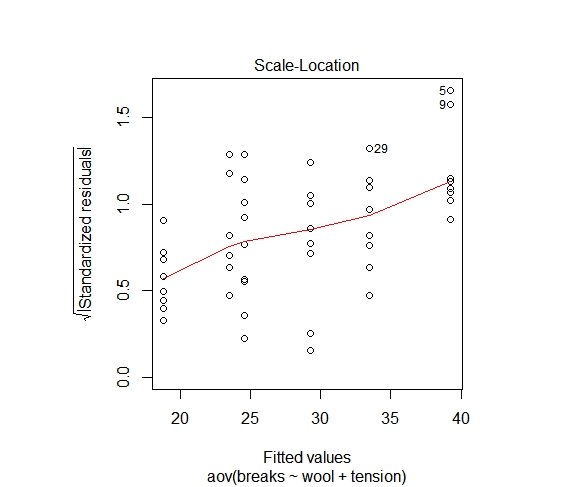
Hit <Return> to see next plot: plot(Model\_2)

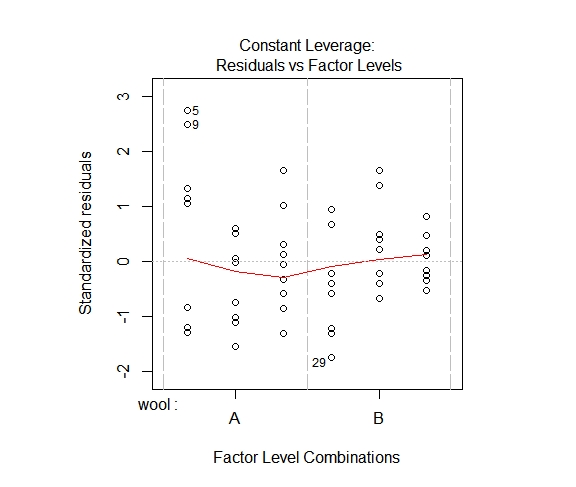
Hit <Return> to see next plot: train=(Year<2005)

**Plots:**





****

****

**Practical 9**

**Aim: Practical of Principal Component Analysis.**

**Description:**

* A Principal Component Analysis (PCA) can be considered as a rotation of the axes of the original variable coordinate system to new orthogonal axes, called as the principal axes, such that the new axes coincide with directions of maximum variation of the original observations.
* PCA is also called as Karhunen-Loeve or K-L Method. This is method of dimensionality reduction searches for k n-dimensional orthogonal vectors that can best used to represent the dala. The original data are thus projected onto a much smaller space which results to dimensionality reduction.
* Basic Procedure Followed in PCA
  + Firstly the input data is normalized so that all the features fall into the similar range. This step is performed in order to normalize the large and small domain values.
  + PCA then computes K ortho normal vectors which provide a basis for normalized input data. These are unit vectors and are perpendicular lo each other's. These vectors are also referred as the Principal Components.
  + The Principal Components are then stored in order of decreasing strength.
  + Principal components provide new set of axes.
  + As the components are stored in the decreasing order of strength, the size of the data can be reduced by climinating the weaker components.

**Source Code:**

# loading dataset iris

data("iris")

head(iris)

library()

# to find principal component

mypr<-prcomp(iris[,-5],scale=T)

# to understand use of scale

plot(iris$Sepal.Length,iris$Sepal.Width)

plot(scale(iris$Sepal.Length),scale(iris$Sepal.Width))

mypr

summary(mypr)

plot(mypr,type="l")

# plot the biplot showing first two PC’s and the original feature vectors in this

#2D space i.e original feature vectors as linear combination of first two PC’s

biplot(mypr,scale=0)

# extract pc scores

str(mypr)

mypr$x

iris2<-cbind(mypr$x[,1:2])

head(iris2)

# find co-relations(x=iris[,5], y=iris[,1:2])

cor(iris[,-5],iris2[,1:2])

# install pls package

install.packages("pls")

library(pls)

names(iris)

**Output:**

**Console Output:**

> data("iris")

> 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

> mypr<-prcomp(iris[,-5],scale=T)

>

> # to understand use of scale

> plot(iris$Sepal.Length,iris$Sepal.Width)

> plot(scale(iris$Sepal.Length),scale(iris$Sepal.Width))

> mypr

Standard deviations (1, .., p=4):

[1] 1.7083611 0.9560494 0.3830886 0.1439265

Rotation (n x k) = (4 x 4):

PC1 PC2 PC3 PC4

Sepal.Length 0.5210659 -0.37741762 0.7195664 0.2612863

Sepal.Width -0.2693474 -0.92329566 -0.2443818 -0.1235096

Petal.Length 0.5804131 -0.02449161 -0.1421264 -0.8014492

Petal.Width 0.5648565 -0.06694199 -0.6342727 0.5235971

> summary(mypr)

Importance of components:

PC1 PC2 PC3 PC4

Standard deviation 1.7084 0.9560 0.38309 0.14393

Proportion of Variance 0.7296 0.2285 0.03669 0.00518

Cumulative Proportion 0.7296 0.9581 0.99482 1.00000

> plot(mypr,type="l")

> str(mypr)

List of 5

$ sdev : num [1:4] 1.708 0.956 0.383 0.144

$ rotation: num [1:4, 1:4] 0.521 -0.269 0.58 0.565 -0.377 ...

..- attr(\*, "dimnames")=List of 2

.. ..$ : chr [1:4] "Sepal.Length" "Sepal.Width" "Petal.Length" "Petal.Width"

.. ..$ : chr [1:4] "PC1" "PC2" "PC3" "PC4"

$ center : Named num [1:4] 5.84 3.06 3.76 1.2

..- attr(\*, "names")= chr [1:4] "Sepal.Length" "Sepal.Width" "Petal.Length" "Petal.Width"

$ scale : Named num [1:4] 0.828 0.436 1.765 0.762

..- attr(\*, "names")= chr [1:4] "Sepal.Length" "Sepal.Width" "Petal.Length" "Petal.Width"

$ x : num [1:150, 1:4] -2.26 -2.07 -2.36 -2.29 -2.38 ...

..- attr(\*, "dimnames")=List of 2

.. ..$ : NULL

.. ..$ : chr [1:4] "PC1" "PC2" "PC3" "PC4"

- attr(\*, "class")= chr "prcomp"

> iris2<-cbind(mypr$x[,1:2])

> head(iris2)

PC1 PC2

[1,] -2.257141 -0.4784238

[2,] -2.074013 0.6718827

[3,] -2.356335 0.3407664

[4,] -2.291707 0.5953999

[5,] -2.381863 -0.6446757

[6,] -2.068701 -1.4842053

>

> # find co-relations(x=iris[,5], y=iris[,1:2])

> cor(iris[,-5],iris2[,1:2])

PC1 PC2

Sepal.Length 0.8901688 -0.36082989

Sepal.Width -0.4601427 -0.88271627

Petal.Length 0.9915552 -0.02341519

Petal.Width 0.9649790 -0.06399985

>

> # install pls package

> install.packages("pls")

WARNING: Rtools is required to build R packages but is not currently installed. Please download and install the appropriate version of Rtools before proceeding:

https://cran.rstudio.com/bin/windows/Rtools/

Installing package into ‘C:/Users/VRUTIKA/Documents/R/win-library/3.6’

(as ‘lib’ is unspecified)

trying URL 'https://cran.rstudio.com/bin/windows/contrib/3.6/pls\_2.7-2.zip'

Content type 'application/zip' length 1230513 bytes (1.2 MB)

downloaded 1.2 MB

package ‘pls’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in

C:\Users\VRUTIKA\AppData\Local\Temp\RtmpwLgCMC\downloaded\_packages

> library(pls)

Attaching package: ‘pls’

The following object is masked from ‘package:stats’:

loadings

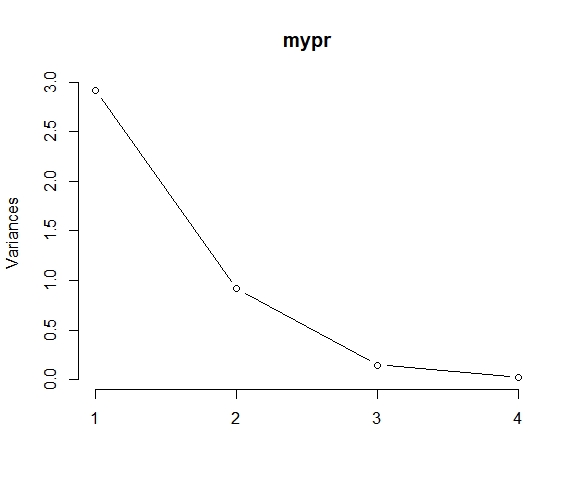
Warning message:

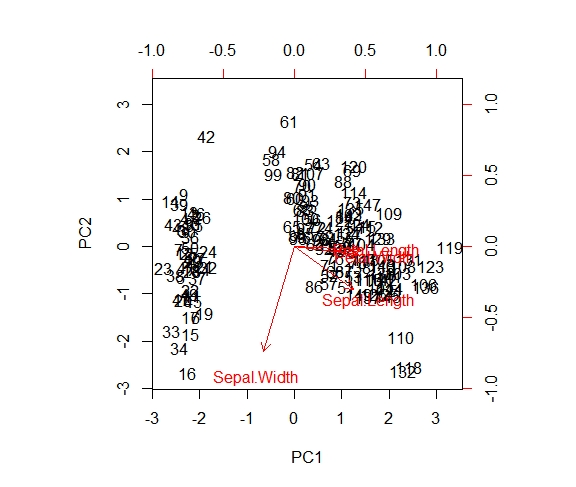
package ‘pls’ was built under R version 3.6.2

> names(iris)

[1] "Sepal.Length" "Sepal.Width" "Petal.Length" "Petal.Width" "Species"

**Plots:**

****

****