**#SIMPLE LINEAR REGRESSION**

wc.at <- read.csv(“file.choose()”)

install.packages("lattice")

library("lattice")

??lattice

View(wc\_at)

summary(wc\_at)

dotchart(wc\_at$Waist)

dotchart(wc\_at$Waist, main="Dot Plot of Waist Circumferences")

plot(Waist,AT)

dotchart(wc\_at$AT, main="Dot Plot of Adipose Tissue Areas")

boxplot(wc\_at$Waist,col="dodgerblue4")

boxplot(wc\_at$AT,col="red", horizontal = T)

attach(wc\_at)

plot(Waist,AT,main = "scatter plot for SLR")

?plot

attach(wc\_at)

# Correlation coefficient

cor(Waist, AT)

#Build Linear regression

reg <- lm(AT~Waist, data=wc\_at)

summary(reg)

confint(reg,level=0.95)

predict(reg,interval="predict")

reg\_log <- lm(AT ~ log(Waist))

summary(reg\_log)

confint(reg\_log, level=0.95)

predict(reg\_log, interval="predict")

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**#MULTIPLE LINEAR REGRESSION**

Cars <- read.csv(file.choose())

View(Cars)

attach(Cars)

cor(HP,VOL)

cor(HP,SP)

pairs(Cars)

cor(Cars)

install.packages("corpcor")

library(corpcor)

cor2pcor(cor(Cars))

model.car <- lm(MPG~VOL+HP+SP+WT)

summary(model.car)

model.carV<-lm(MPG~VOL)

summary(model.carV)

model.carW<-lm(MPG~WT)

summary(model.carW)

model.carVW<-lm(MPG~VOL+WT)

summary(model.carVW)

pairs(Cars,upper.panel = panel.cor,main="Scatter plot matrix with Correlation coefficients")

influence.measures(model.car)

library(car)

influenceIndexPlot(model.car,id.n=5)

influencePlot(model.car,id.n=5)

model.car1<-lm(MPG~VOL+SP+HP+WT,data=Cars[-77,])

summary(model.car1)

model.car2<-lm(MPG~VOL+SP+HP+WT,data=Cars[-c(71,77),])

summary(model.car2)

model.car3<-lm(MPG~VOL+SP+HP+WT,data=Cars[-c(71,77,79),])

summary(model.car3)

vif(model.car)

finalmodel<-lm(MPG~VOL+SP+HP)

summary(finalmodel)

finalmodel1 <- lm(MPG~SP+HP+WT)

summary(finalmodel1)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**#LOGISTIC REGRESSION**

claimants <- read.csv(file.choose())

View(claimants)

attach(claimants)

fit<-lm(ATTORNEY~factor(CLMSEX)+factor(CLMINSUR)+factor(SEATBELT)+CLMAGE+LOSS)

summary(fit)

logit<- glm(ATTORNEY~factor(CLMSEX)+factor(CLMINSUR)+factor(SEATBELT)+CLMAGE+LOSS,family=binomial,data = claimants)

summary(logit)

exp(coef(logit))

prob <- predict(logit,claimants)

prob

confusion<-table(prob>0.5,claimants)

confusion

Accuracy<-sum(diag(confusion)/sum(confusion))

Accuracy

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**#HIERARCHIAL CLUSTERING**

library(readxl)

input <- read\_excel("G:\\Hierarchical Clustering\\University\_Clustering.xlsx")

View(input)

attach(input)

mydata<-input[,c(3,4:8)]

View(mydata)

normalized\_data<-scale(mydata)

View(normalized\_data)

d<-dist(normalized\_data,method="euclidean")

summary(d)

d

?hclust

fit<-hclust(d,method="complete")

fit

?hclust

plot(fit)

plot(fit,hang=-1)

groups<-cutree(fit,k=3)

groups

rect.hclust(fit,k=3,border="red")

membership<-as.matrix(groups)

View(membership)

final1<-data.frame(input,membership)

View(final1)

library(xlsx)

write.xlsx(final1,file="jan1st.xlsx")

getwd()

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**#K-MEANS CLUSTERING**

install.packages("plyr")

library(plyr)

View(input)

mydata<-input[1:25,c(2,3:8)]

View(mydata)

View(normalized\_data)

km <- kmeans(normalized\_data,4)

str(km)

install.packages("animation")

library(animation)

km <- kmeans.ani(normalized\_data, 4)

km$centers

final2<- data.frame(mydata, km$cluster) # append cluster membership

final2

final3 <- final2[,c(ncol(final2),1:(ncol(final2)-1))]

final3

library(xlsx)

write.xlsx(final1,file="final1.xlsx")

getwd()

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**#PRINCIPAL COMPONENT ANALYSIS**

mydata<-read\_csv("E:/Confidential/R Codes/PCA/Universities.csv")

View(mydata)

help(princomp)

data <- mydata[-1]

attach(data)

cor(data)

pcaObj<-princomp(mydata[-1], cor = TRUE, scores = TRUE, covmat = NULL)

summary(pcaObj)

loadings(pcaObj)

plot(pcaObj)

biplot(pcaObj)

pcaObj$scores[,1:3]

mydata<-cbind(mydata,pcaObj$scores[,1:3])

View(mydata)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**#KNN(K-NEAREST NEIGHBOUR)**

View(wbcd)

wbcd <- wbcd[-1]

View(wbcd)

table(wbcd$diagnosis)

wbcd$diagnosis <- factor(wbcd$diagnosis, levels = c("B", "M"),

labels = c("Benign", "Malignant"))

View(wbcd)

round(prop.table(table(wbcd$diagnosis)) \* 100, digits = 1)

View(wbcd)

summary(wbcd)

summary(wbcd[,c("radius\_mean", "area\_mean", "smoothness\_mean")])

str(wbcd)

normalize <- function(x) {

return ((x - min(x)) / (max(x) - min(x))) }

wbcd\_n <- as.data.frame(lapply(wbcd[2:31], normalize))

View(wbcd\_n)

summary(wbcd\_n)

wbcd\_train <- wbcd\_n[1:469, ]

View(wbcd\_test)

wbcd\_test <- wbcd\_n[470:569, ]

wbcd\_train\_labels <- wbcd[1:469, 1]

View(wbcd\_train\_labels)

wbcd\_test\_labels <- wbcd[470:569, 1]

library(class)

wbcd\_train\_labels <- wbcd\_train\_labels[["diagnosis"]]

wbcd\_test\_labels <- wbcd\_test\_labels[["diagnosis"]]

wbcd\_test\_pred <- knn(train = wbcd\_train, test = wbcd\_test,

cl = wbcd\_train\_labels, k=23)

library(gmodels)

CrossTable(x = wbcd\_test\_labels, y = wbcd\_test\_pred,

prop.chisq=FALSE)

wbcd\_test\_pred <- knn(train = wbcd\_train, test = wbcd\_test, cl = wbcd\_train\_labels, k=2)

CrossTable(x = wbcd\_test\_labels, y = wbcd\_test\_pred, prop.chisq=FALSE)

wbcd\_test\_pred <- knn(train = wbcd\_train, test = wbcd\_test, cl = wbcd\_train\_labels, k=21)

CrossTable(x = wbcd\_test\_labels, y = wbcd\_test\_pred, prop.chisq=FALSE)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**#SVM(SIMPLE VECTOR MACHINE)**

data()

data("iris")

View(iris)

attach(iris)

install.packages('e1071', dependencies=TRUE)

library(e1071)

final\_svm <- svm(Species ~ ., data=iris)

summary(final\_svm)

pred <- predict(final\_svm,iris)

tab <-table(predicted = pred,Actual = iris$Species)

sum(diag(tab))/sum(tab)

View(iris)

svm\_linear <- svm(Species ~ ., data=iris,kernel = "linear")

summary(svm\_linear)

pred <- predict(svm\_linear,iris)

tab <-table(predicted = pred,Actual = iris$Species)

sum(diag(tab))/sum(tab)

library(kernlab)

svm\_nonlinear <- ksvm(Species ~ ., data = iris, kernel = "rbfdot1")

prediction <- predict(svm\_nonlinear,iris)

table(prediction)

prop.table(table(prediction))

tab <-table(prediction = prediction,Actual = iris$Species)

tab

sum(diag(tab))/sum(tab)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**#NAVIES BAYES**

library(mlbench)

data("HouseVotes84")

?? Housevotes84

View(HouseVotes84)

help("HouseVotes84")

summary(HouseVotes84)

barplot(table(as.factor(HouseVotes84[,1]),as.factor(HouseVotes84[,2])),legend=c("democrat","republic"))

plot(as.factor(HouseVotes84[HouseVotes84$Class=="republican",2]))

plot(as.factor(HouseVotes84[HouseVotes84$Class=="democrat",2]))

str(HouseVotes84)

set.seed(3)

train<-order(runif(290))

test<--train

View(train)

View(test)

training<-HouseVotes84[train,]

View(training)

testing<-HouseVotes84[test,]

View(training)

View(testing)

library(e1071)

model<-naiveBayes(training$Class~.,data=training)

model

pred<-predict(model,newdata = testing[,-1])

mean(pred==testing[,1])

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**#ASSOCIATION RULES(APRIORI ALGORITHIM)**

install.packages("arules")

library(arules)

install.packages("arulesViz") # extension to arules

library(arulesViz)

?arules

data()

data(Groceries)

View(Groceries)

summary(Groceries)

itemFrequencyPlot(Groceries)

itemFrequencyPlot(Groceries, topN = 10)

inspect(head(Groceries, 2))

rules\_fileName <- apriori(Groceries,parameter=list(support=0.001, confidence = 1))

rules\_fileName

inspect(head(sort(rules\_fileName, by = "confidence"), 3))

inspect(sort(rules\_fileName, by="lift"))

inspect(sort(rules\_fileName, by="lift")[1:10])

library(xlsx)

write(rules\_fileName,file="newrules.csv",sep=",",row.names = FALSE)

getwd()

install.packages("arulesViz")

plot(rules\_fileName)

plot(head(rules\_fileName))

head(quality(rules\_fileName))

plot((rules\_fileName), method="grouped")

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**#DECISION TREE**

install.packages("C50")

install.packages("tree")

library(C50)

data()

data("iris")

View(iris)

?iris

iris\_setosa<-iris[iris$Species=="setosa",] # 50

iris\_versicolor <- iris[iris$Species=="versicolor",] # 50

iris\_virginica <- iris[iris$Species=="virginica",] # 50

iris\_train <- rbind(iris\_setosa[1:25,],iris\_versicolor[1:25,],iris\_virginica[1:25,])

iris\_test <- rbind(iris\_setosa[26:50,],iris\_versicolor[26:50,],iris\_virginica[26:50,])

View(iris\_train)

View(iris\_test)

irisc5.0\_train <- C5.0(iris\_train[,-5],iris\_train$Species)

plot(irisc5.0\_train) # Tree graph

plot(irisc5.0\_train,type="simple")

mean(iris\_train$Species==predict(irisc5.0\_train,iris\_train)) # 97.33% Accuracy

predc5.0\_test <- predict(irisc5.0\_train,newdata=iris\_test) # predicting on test data

mean(predc5.0\_test==iris\_test$Species) # 94.66% accuracy

library(gmodels)

CrossTable(iris\_test$Species,predc5.0\_test)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**#RANDOM FOREST**

library(randomForest)

data(iris)

View(iris)

iris\_setosa<-iris[iris$Species=="setosa",] # 50

iris\_versicolor <- iris[iris$Species=="versicolor",] # 50

iris\_virginica <- iris[iris$Species=="virginica",] # 50

iris\_train <- rbind(iris\_setosa[1:25,],iris\_versicolor[1:25,],iris\_virginica[1:25,])

iris\_test <- rbind(iris\_setosa[26:50,],iris\_versicolor[26:50,],iris\_virginica[26:50,])

model1 <- randomForest(Species~.,data=iris\_train)

model1

model2 <- randomForest(Species~.,data=iris\_train, ntree = 100, importance = TRUE)

model2

importance(model2)

varImpPlot(model2)

plot(round(importance(fit.forest)))

?randomeforest

table(predict(model1),iris\_train$Species)

pred\_test <- predict(model1,newdata=iris\_test)

table(pred\_test,iris\_test$Species)

CM <- table(pred\_test,iris\_test$Species)

CM

accuracy <- (sum(diag(CM)))/sum(CM)

accuracy

pred\_test <- predict(model2,newdata=iris\_test)

table(pred\_test,iris\_test$Species)

CM <- table(pred\_test,iris\_test$Species)

CM

accuracy <- (sum(diag(CM)))/sum(CM)

accuracy