1. **Pca**

data("iris")

head(iris)

summary(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")

biplot(mypr,scale=0)

"extract pc scores"

str(mypr)

mypr$x

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

head(iris2)

cor(iris[,-5],iris2[,6:7])

"End of prog"

install.packages("pls")

library(pls)

names(iris)

pcmodel<-pcr(Sepal.Length~Species+Sepal.Width+Petal.Length+Petal.Width,ncomp=3,data=iris,scale=T)

iris$pred<-predict(pcmodel,iris,ncomp = 2)

head(iris)

1. **"k-means clustering "**

data("iris")

names(iris)

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

new\_data

cl<-kmeans(new\_data,3)

cl

data <- new\_data

wss <- sapply(1:15,

function(k){kmeans(data, k )$tot.withinss})

wss

plot(1:15, wss,

type="b", pch = 19, frame = FALSE,

xlab="Number of clusters K",

ylab="Total within-clusters sum of squares")

install.packages("cluster")

library(cluster)

clusplot(new\_data, cl$cluster, color=TRUE, shade=TRUE,

labels=2, lines=0)

cl$cluster

cl$centers

1. **"agglomarative clustering "**

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

plot(clusters)

clusterCut <- cutree(clusters, 3)

table(clusterCut, iris$Species)

ggplot(iris, aes(Petal.Length, Petal.Width, color = iris$Species)) +

geom\_point(alpha = 0.4, size = 3.5) + geom\_point(col = clusterCut) +

scale\_color\_manual(values = c('black', 'red', 'green'))

clusters <- hclust(dist(iris[, 3:4]), method = 'average')

clusterCut1 <- cutree(clusters, 3)

table(clusterCut1, iris$Species)

plot(clusters)

ggplot(iris, aes(Petal.Length, Petal.Width, color = iris$Species)) +

geom\_point(alpha = 0.4, size = 3.5) + geom\_point(col = clusterCut1) +

scale\_color\_manual(values = c('black', 'red', 'green'))

1. **//hypothesis//**

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

dataf

mean(dataf)

sd(dataf)

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

a

1. **//time series//**

data("AirPassengers")

class(AirPassengers)

start(AirPassengers)

end(AirPassengers)

frequency(AirPassengers)

summary(AirPassengers)

plot(AirPassengers)

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

cycle(AirPassengers)

plot(aggregate(AirPassengers,FUN=mean))

boxplot(AirPassengers~cycle(AirPassengers))

1. **Linear regression**

height<-c(102,117,105,141,135,115,138)

weight<-c(61,46,62,54,60,69,51)

student<-lm(weight~height)

student

predict (student,data.frame(height=119),interval="confidence)

plot(student)

ftest<-read.csv(file.choose(),sep=",",header=T)

var.test(ftest$time\_g1,ftest$time\_g2,alternative = "two.sided")

**"one way anova"**

data1<-read.csv(file.choose(),sep = ",",header = T)

names(data1)

summary(data1)

head(data1)

anv<-aov(formula = satindex~dept,data=data1)

summary(anv)

1. **"two way anova"**

data2<-read.csv(file.choose(),sep=",",header = T)

names(data2)

summary(data2)

anv1<-aov(formula = satindex~ dept+exp+dept\*exp,data = data2)

summary(anv1)

**Practical No :**

**Aim: Implement regression in R**

**Code:**

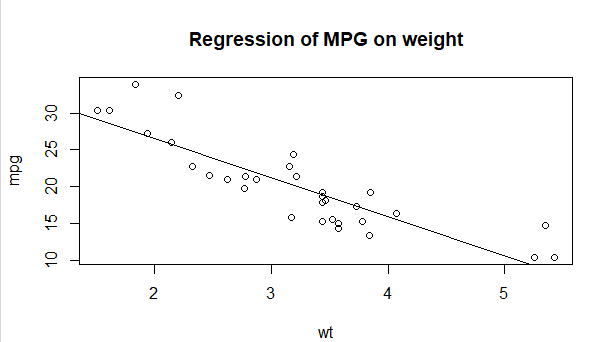
attach(mtcars)

plot(wt,mpg)

abline(lm(mpg~wt))

title("Regression of MPG on weight")

**Output:**

****

**Practical No:**

**Aim:**

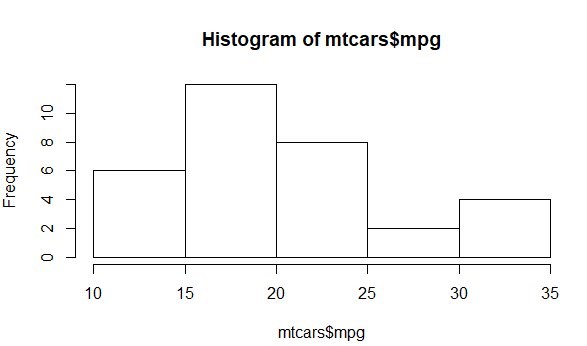
**Code**

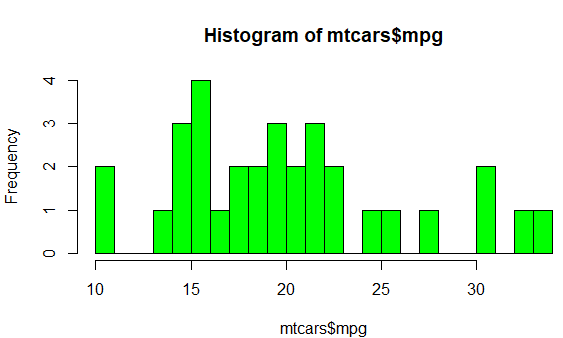
Histogram

hist(mtcars$mpg)

hist(mtcars$mpg, breaks = 20, col = "green")

**Output:**

****

****

**DotChart**

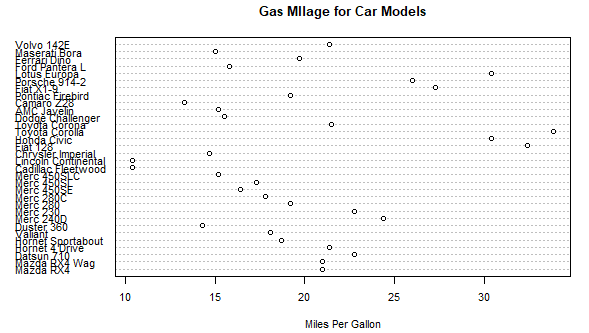
**Code:**

dotchart(mtcars$mpg,labels = row.names(mtcars),cex = .7,

main="Gas MIlage for Car Models",

xlab = "Miles Per Gallon")

**Output:**

****

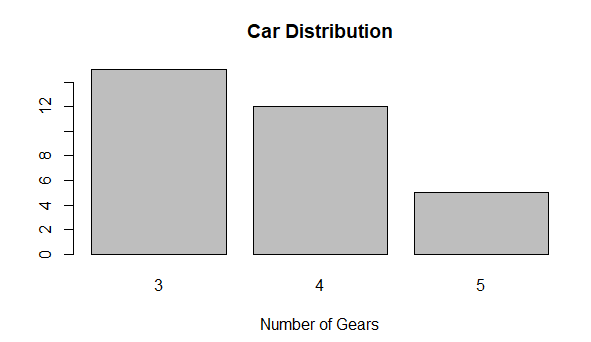
**Barplot**

**Code**

counts<-table(mtcars$gear)

barplot(counts, main="Car Distribution",xlab = "Number of Gears")

**output:**

****

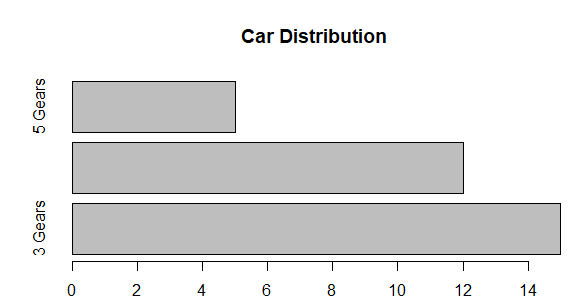
**Code:**

counts<-table(mtcars$gear)

barplot(counts, main="Car Distribution", horiz = TRUE,

names.arg = c("3 Gears","4 Gears", "5 Gears"))

output:



Code:

counts<-table(mtcars$vs,mtcars$gear)

barplot(counts,main = "Car distribution by Gears and VS",

xlab = "Number of gears", col =c("darkblue","red"),

legend= rownames(counts))

output:

