Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING(SCOPE)

WINTER SEMESETER 2021-22

LAB-1

NAME : RISHABH SINGH

REG NO: 19BCE1500

COURSE: ESSENTIAL OF DATA ANALYTIC(CSE3506)

FACULTY: DR.LASKHMI PATHI JAKKAMPUTI

SLOT: L21+L22

Tasks for Week-1: Regression

Understand the following operations/functions on random dataset and perform similar operations on mtcars and 'data.csv' dataset based on given instructions.

Aim: To develop linear regression model for the given data using R programming and to verify the null hypothesis

Algorithm:

1. Set the current working directory.
2. Loat all the required library using library() function.
3. Load the dataset using read.csv () function.
4. Using sample () function split the data for train and testing. (We are taking the 75% data for training purpose and 25% for testing purpose).
5. Plot the weight(wt) vs Mileage(mpg).
6. Find the correlation between wt and mpg using cor.test() function.
7. Prepare the model using lm(mpg~wt,data) function.
8. Print the summary of the model using summary(lmodel) function.
9. Now predict the value for testing data.
10. Using MAE() function find the mean absolute error in the predicted and the original values.

Statistic Case 1: mtcars

Residuals:

Min	**1Q**	**Median**	**3Q**	**Max**
-4.6037	-2.6129	-0.1983	1.3715	6.5714

Coefficients:	**Estimate**	**Std. Error**	**t value**	**Pr(>|t|)**
(Intercept)	38.2943	2.2919	16.71	5.50e-14
wt	-5.6437	0.7171	-7.87	7.73e-08

Residual standard error: 3.336 on 22 degrees of freedom

Multiple R-squared: 0.7379, **Adjusted R-squared:** 0.726

F-statistic: 61.94 on 1 and 22 DF**,** **p-value:** 7.733e-08

Statistic Case 2: data.csv

```
Residuals:
    Min       1Q    Median     3Q      Max
  -30.307 -13.598  1.082    13.168   28.924


Coefficients:
                     Estimate Std. Error     t value      Pr(>|t|)
(Intercept) 170.562451    2.772873           61.511       <2e-16
Weight        -0.004918    0.025536           -0.193        0.847
```

Residual standard error: 16.22 on 373 degrees of freedom

Multiple R-squared: 9.944e-05, **Adjusted R-squared:** -0.002581

F-statistic: 0.03709 on 1 and 373 DF, **p-value:** 0.8474

Inference:

CASE 1: mtcars

The p-value(7.733e-08) is less than the 0.05 which means the model is accepted.

CASE 2: data.csv

The p-value(0.8474) is greater than 0.05 which means the model is rejected.

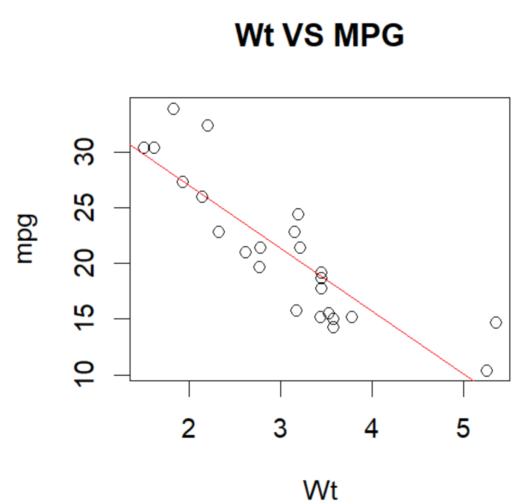
Program:

1.Mtcars:

```
rm(list=ls())
library(dplyr)
library(Metrics)

## 75% of the sample size
```

```r
smp_size <- floor(0.75 * nrow(mtcars))
#setting  the seed to make your partition reproducible
set.seed(123)
train_ind <- sample(seq_len(nrow(mtcars)), size = smp_size)
train <- mtcars[train_ind, ]
test <- mtcars[-train_ind, ]

correlation<-cor.test(train$wt,train$mpg)

print(correlation)

plot(train$wt,train$mpg,xlab = "Wt",ylab = "mpg",main="Wt VS MPG")
##Linear model

lmodel<-lm(mpg~wt,data=train)
abline(lmodel,col="red")

print(lmodel)
summary(lmodel)

predicted<-predict(lmodel,data=test)
mae(test$mpg,predicted)
```

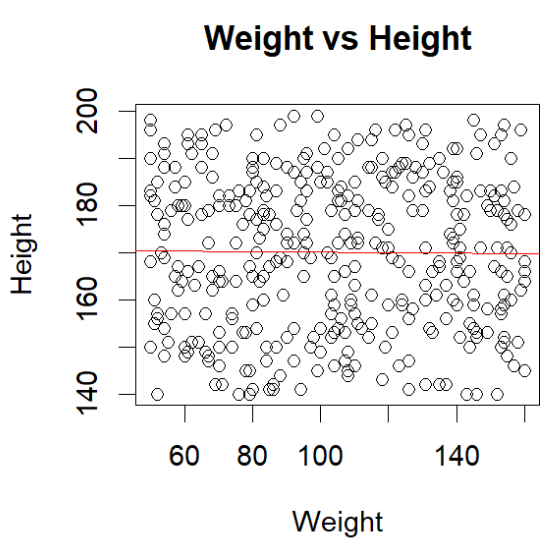
OUTPUT:



2.data.csv:

```
rm(list=ls())
library(dplyr)

setwd("D:/6th_Semester/Data_Analytics_Lab/Lab 1")
data<-read.csv('data.csv')
## 75% of the sample size
smp_size <- floor(0.75 * nrow(data))
#setting  the seed to make your partition reproducible
set.seed(123)
train_ind <- sample(seq_len(nrow(data)), size = smp_size)
train <- data[train_ind, ]
test <- data[-train_ind, ]
correlation<-cor.test(train$Height,train$Weight)

print(correlation)

plot(train$Weight,train$Height,xlab = "Weight",ylab =
"Height",main="Weight vs Height")
##Linear model
lmodel<-lm(Height~Weight,data=train)
abline(lmodel,col="red")
summary(lmodel)
predicted<-predict(lmodel,data=test)
mae(test$Height,predicted)
```

OUTPUT:



Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING(SCOPE)

WINTER SEMESETER 2021-22

LAB-2

NAME : RISHABH SINGH

REG NO: 19BCE1500

COURSE: ESSENTIAL OF DATA ANALYTIC(CSE3506)

FACULTY: DR.LASKHMI PATHI JAKKAMPUTI

SLOT: L21+L22

Tasks for Week-2: Forecasting

Understand the following operations/functions on random dataset and perform similar operations on gold and gdp dataset based on given instructions.

Aim: To develop a forecasting model that forecasts the value 24 units ahead of time

Algorithm:

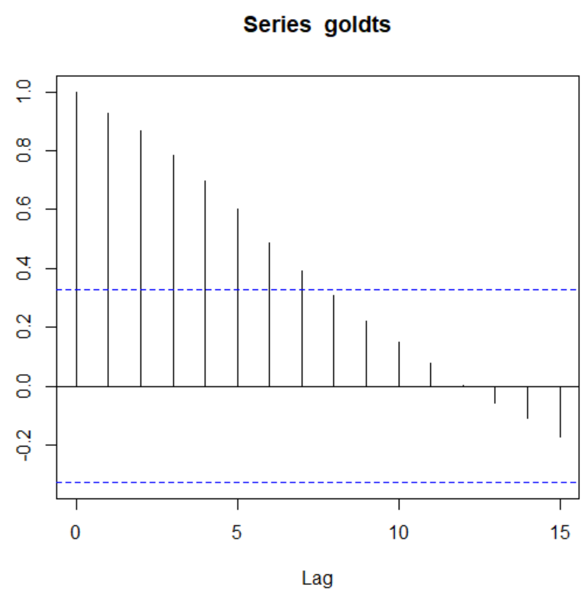
1. Set the current working directory.
2. Loat all the required library using library() function.
3. Load the dataset using read.csv () function.
4. Convert the dataset into timeseries data using the ‘ts’ function.
5. Using acf and pacf check if the data is stationary visually, if the readings are below the blue line then the data is stationary
6. Using adf.test check if the data is stationary using p-value.
7. Using auto ARIMA, find out which model is the best.
8. Pass the best model to the forecast function so that the forecasting is done with 95% confidence for the next 24 units of data.

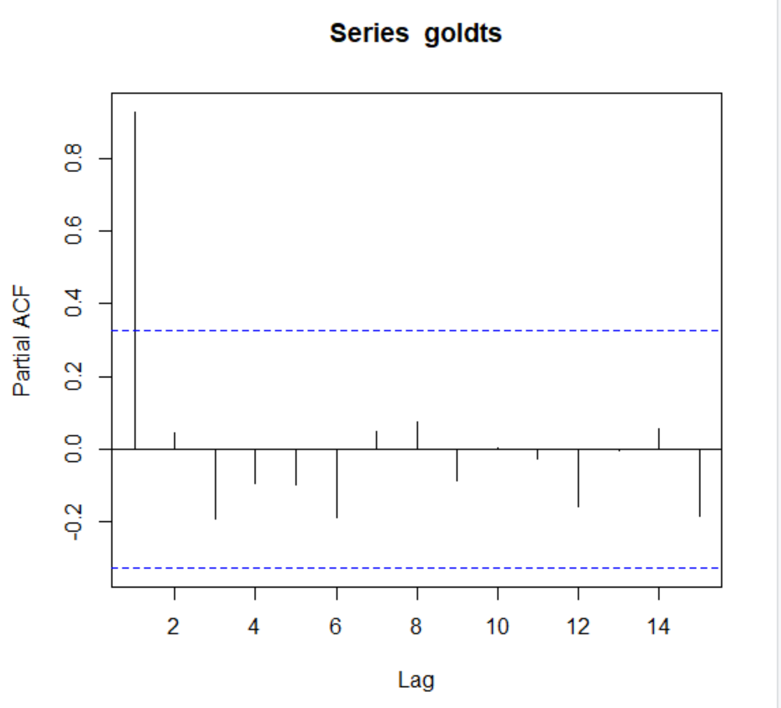
Inference :

1.Gold

The given data is not stationary. We can see this using p value and acf and pacf graph.

p-value = 0.4359





Best Arima Model

Auto ARIMA

```
ARIMA(2,1,2) with drift        : Inf

 ARIMA(0,1,0) with drift        : 457.5809

 ARIMA(1,1,0) with drift        : 459.3633

 ARIMA(0,1,1) with drift        : 459.385

 ARIMA(0,1,0)                    : 459.9305

 ARIMA(1,1,1) with drift        : 461.3121

Best model: ARIMA(0,1,0) with drift
```

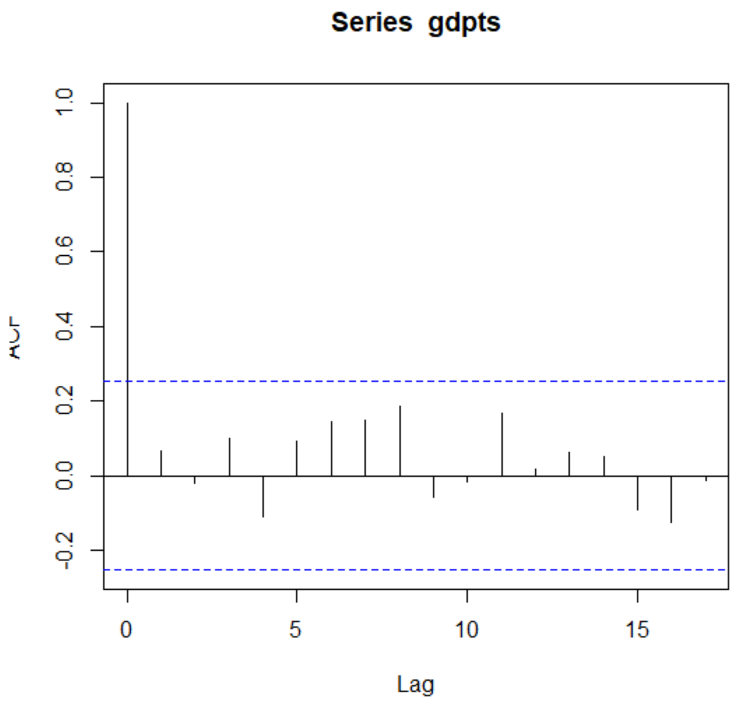
Accuracy of the Model

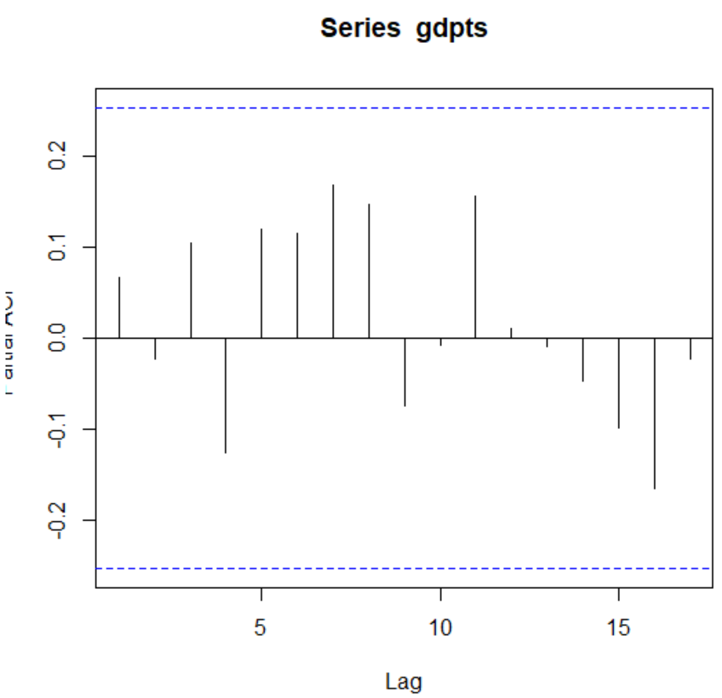
```
             ME     RMSE      MAE        MPE     MAPE      MASE
Training set 0.08218409 155.5098 116.6965 -0.1799051 2.960037 0.9286895
             ACF1
Training set -0.07882193
```

2.GDP.csv

The given data is stationary. We can see this using p value and acf and pacf graph.

p-value – 0.01





Best Model

Auto ARIMA

```
ARIMA(2,1,2) with drift         : Inf
 ARIMA(0,1,0) with drift        : 341.4397
 ARIMA(1,1,0) with drift        : 332.4653
 ARIMA(0,1,1) with drift        : Inf
 ARIMA(0,1,0)                   : 339.554
 ARIMA(2,1,0) with drift        : 326.0715
 ARIMA(3,1,0) with drift        : 327.9755
 ARIMA(2,1,1) with drift        : Inf
 ARIMA(1,1,1) with drift        : Inf
 ARIMA(3,1,1) with drift        : Inf
 ARIMA(2,1,0)                   : 324.2097
 ARIMA(1,1,0)                   : 330.5929
```

```
ARIMA(3,1,0)            : 326.1139

ARIMA(2,1,1)            : 317.8228

ARIMA(1,1,1)            : 316.651

ARIMA(0,1,1)            : 314.6516

ARIMA(0,1,2)            : 316.6508

ARIMA(1,1,2)            : 316.6275



Best model: ARIMA(0,1,1)
```

Accuracy of the Model

	ME	RMSE	MAE	MPE	MAPE	MASE	ACF1
Training set	0.2704179	3.287709	2.345416	121.6616	161.0542	0.7720211	-0.02667223

Program :

1.GOLD:

```
rm(list=ls())
setwd("D:/6th_Semester/Data_Analytics_Lab/Lab 2")
gold <- read.csv("gold.csv")
library(forecast)
library(tseries)
view(gold)
goldts<-ts(gold$Price, start = min(gold$Month), end = max(gold$Month),
frequency = 1)
class(goldts)
plot(goldts)
acf(goldts)
pacf(goldts)
adf.test(goldts) # stationary only if p value <0.05
# To make it stationary, differentiate
goldmodel=auto.arima(goldts, ic='aic', trace = TRUE)
```

```
goldf=forecast(goldmodel, level=c(95), h=24)
goldf
plot(goldf)
accuracy(goldmodel)
```

2.GDP:

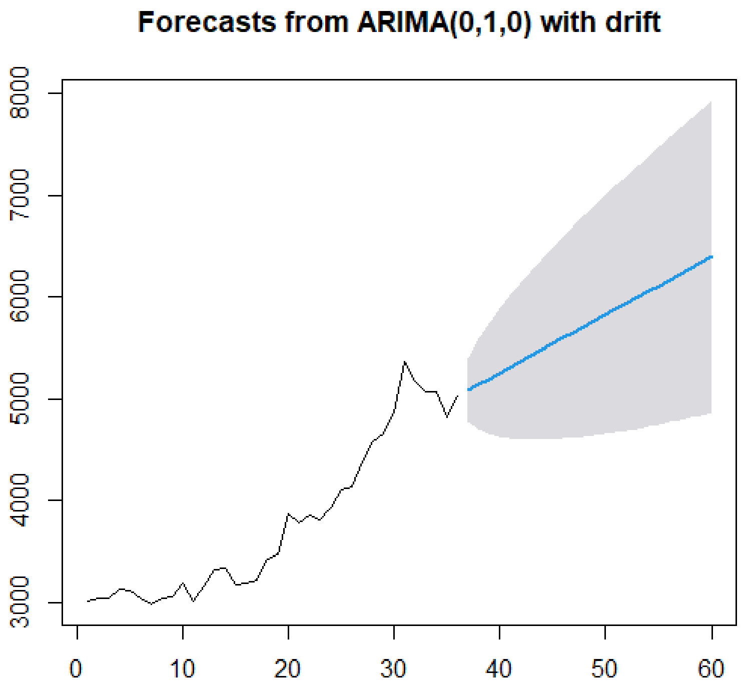
```
rm(list=ls())
setwd("D:/6th_Semester/Data_Analytics_Lab/Lab 2")
gdp <- read.csv("gdp.csv")
library(forecast)
library(tseries)
view(gdp)
gdpts<-ts(gdp$GDP_gr, start = min(gdp$Year), end = max(gdp$Year),
frequency = 1)
class(gdpts)
plot(gdpts)
acf(gdpts)
pacf(gdpts)
adf.test(gdpts) # stationary only if p value <0.05
# To make it stationary, differentiate
gdpmodel=auto.arima(gdpts, ic='aic', trace = TRUE)
gdpf=forecast(gdpmodel, level=c(95), h=24)
gdpf
plot(gdpf,col = 'red')
accuracy(gdpmodel)
```

Result:

1.Forecast for Gold

	Point Forecast	Lo 95	Hi 95
37	5081.371	4767.741	5395.001
38	5138.743	4695.203	5582.283
39	5196.114	4652.891	5739.338
40	5253.486	4626.226	5880.746
41	5310.857	4609.559	6012.155
42	5368.229	4599.995	6136.462

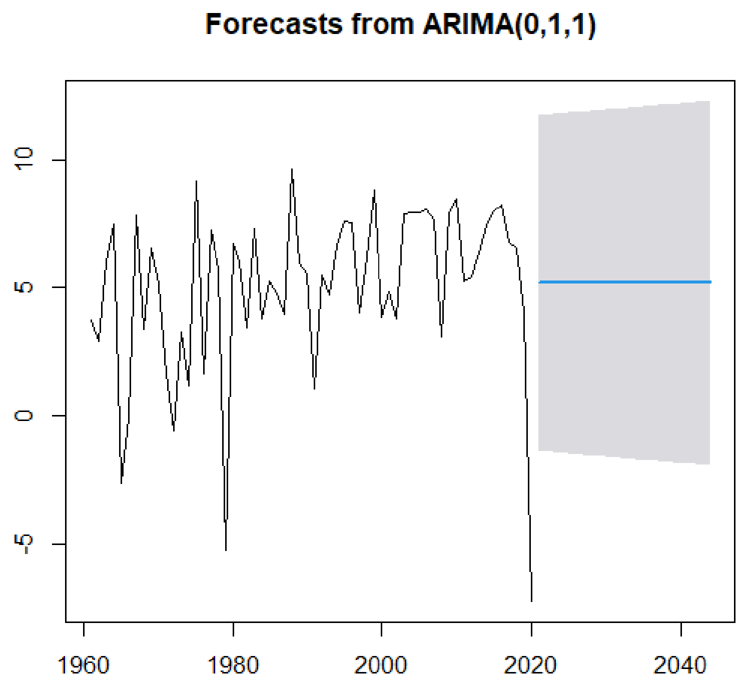
43	5425.600	4595.813	6255.387
44	5482.971	4595.892	6370.051
45	5540.343	4599.453	6481.233
46	5597.714	4605.929	6589.500
47	5655.086	4614.892	6695.279
48	5712.457	4626.011	6798.904
49	5769.829	4639.019	6900.638
50	5827.200	4653.704	7000.696
51	5884.571	4669.887	7099.255
52	5941.943	4687.423	7196.463
53	5999.314	4706.184	7292.444
54	6056.686	4726.066	7387.305
55	6114.057	4746.975	7481.139
56	6171.429	4768.832	7574.025
57	6228.800	4791.566	7666.034
58	6286.171	4815.116	7757.227
59	6343.543	4839.426	7847.660
60	6400.914	4864.447	7937.382



2.Forecast for GDP

	Point Forecast	Lo 95	Hi 95
2021	5.177274	-1.376684	11.73123
2022	5.177274	-1.401989	11.75654
2023	5.177274	-1.427197	11.78174
2024	5.177274	-1.452309	11.80686
2025	5.177274	-1.477327	11.83187
2026	5.177274	-1.502250	11.85680
2027	5.177274	-1.527082	11.88163
2028	5.177274	-1.551821	11.90637
2029	5.177274	-1.576470	11.93102
2030	5.177274	-1.601029	11.95558
2031	5.177274	-1.625500	11.98005
2032	5.177274	-1.649882	12.00443

2033	5.177274	-1.674178	12.02873
2034	5.177274	-1.698389	12.05294
2035	5.177274	-1.722514	12.07706
2036	5.177274	-1.746555	12.10110
2037	5.177274	-1.770513	12.12506
2038	5.177274	-1.794389	12.14894
2039	5.177274	-1.818183	12.17273
2040	5.177274	-1.841896	12.19644
2041	5.177274	-1.865530	12.22008
2042	5.177274	-1.889085	12.24363
2043	5.177274	-1.912561	12.26711
2044	5.177274	-1.935960	12.29051



Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING(SCOPE)

WINTER SEMESETER 2021-22

LAB-3

NAME : RISHABH SINGH

REG NO: 19BCE1500

COURSE: ESSENTIAL OF DATA ANALYTIC(CSE3506)

FACULTY: DR.LASKHMI PATHI JAKKAMPUTI

SLOT: L21+L22

Tasks for Week-3: Regression and Forecasting on Weather Data

Perform multi-regression and forecasting on weather related dataset "weatherHistory2016.csv"

Aim: To develop a multi-regression and forecasting model for the given data using R programming and to predict the temperature.

Algorithm:

Step 1: Import the dataset and load the dplyr, forecast and tseries library.

Step 2: Take random samples from the dataset and store it in trail.

Step 3: Create Time series of weather data frame starting from 1st Jan 2016 to 31st Dec 2016 iterating with a frequency of 24.

Step 4: Check the correlation between dependent and independent variables and select the significant values.

Step 5: Train a multiple regression model using lm function for dependent and independent variables(cor>0.5).

Step 6: generate the summary and analyze the F-statistics

Step 7: If p-value<0.05, then the model is accepted otherwise it is not.

Step 8: Plot the time series.

Step 9: Check whether the time series is stationary or not using the Auto Correlation Function, Partial Auto Correlation Function and Augmented Dickey-Fuller Test.

Step 10: Apply Auto Arima to find the best model.

Step 11: Using this model, forecast the temperature for the next month.

Step 12: Plot the forecasted values with low and high range.

Step 13: Measure the accuracy of the model and check the acceptance.

Statistics/Result :

1. Correlation values of the three significant attributes.

a) Apparent Temperature

```
        Pearson's product-moment correlation

data:  a$Temperature..C. and a$Apparent.Temperature..C.
t = 136.24, df = 198, p-value < 2.2e-16
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.9930099 0.9959955
sample estimates:
      cor 
0.9947087 
```

b) Humidity

```
        Pearson's product-moment correlation

data:  a$Temperature..C. and a$Humidity
t = -12.017, df = 198, p-value < 2.2e-16
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 -0.7230228 -0.5612574
sample estimates:
       cor 
-0.6494278 
```

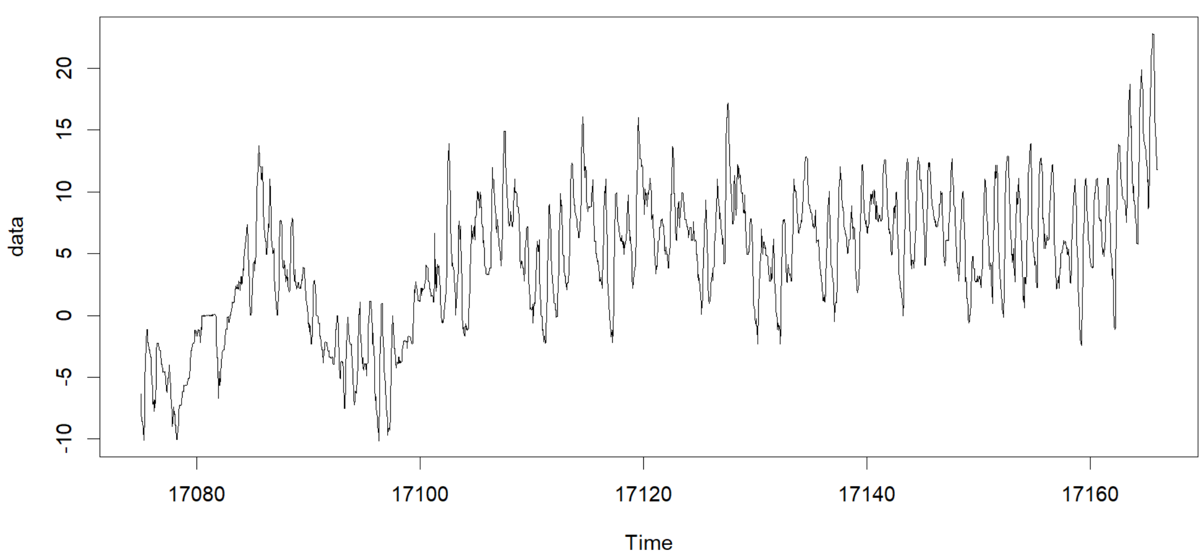
c) Visibility

```
> cor.test(a$Temperature..C.,a$Visibility..km.)

        Pearson's product-moment correlation

data:  a$Temperature..C. and a$Visibility..km.
t = 7.933, df = 198, p-value = 1.545e-13
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.3781281 0.5896680
sample estimates:
      cor 
0.4911049 
```

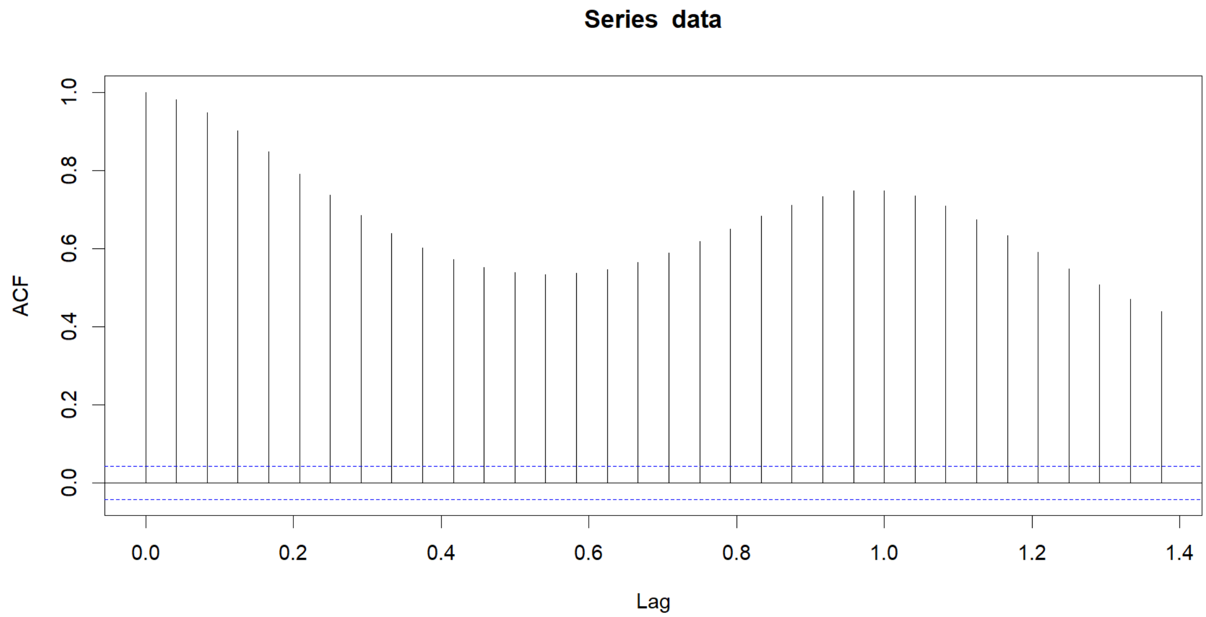
2. F-statistics and Summary of the multiple regression model

```
Call:
lm(formula = a$Temperature..C. ~ a$Apparent.Temperature..C. +
    a$Humidity + a$Visibility..km.)

Residuals:
    Min      1Q  Median      3Q     Max
-3.2769 -0.5062  0.1693  0.6139  2.1758

Coefficients:
                            Estimate Std. Error t value Pr(>|t|)
(Intercept)                 4.624238   0.431299  10.722   <2e-16 ***
a$Apparent.Temperature..C.  0.855291   0.007776 109.993   <2e-16 ***
a$Humidity                 -2.828868   0.425982  -6.641    3e-10 ***
a$Visibility..km.           0.018346   0.014927   1.229    0.221
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.8869 on 196 degrees of freedom
Multiple R-squared:  0.9916,    Adjusted R-squared:  0.9915
F-statistic:  7739 on 3 and 196 DF,  p-value: < 2.2e-16
```

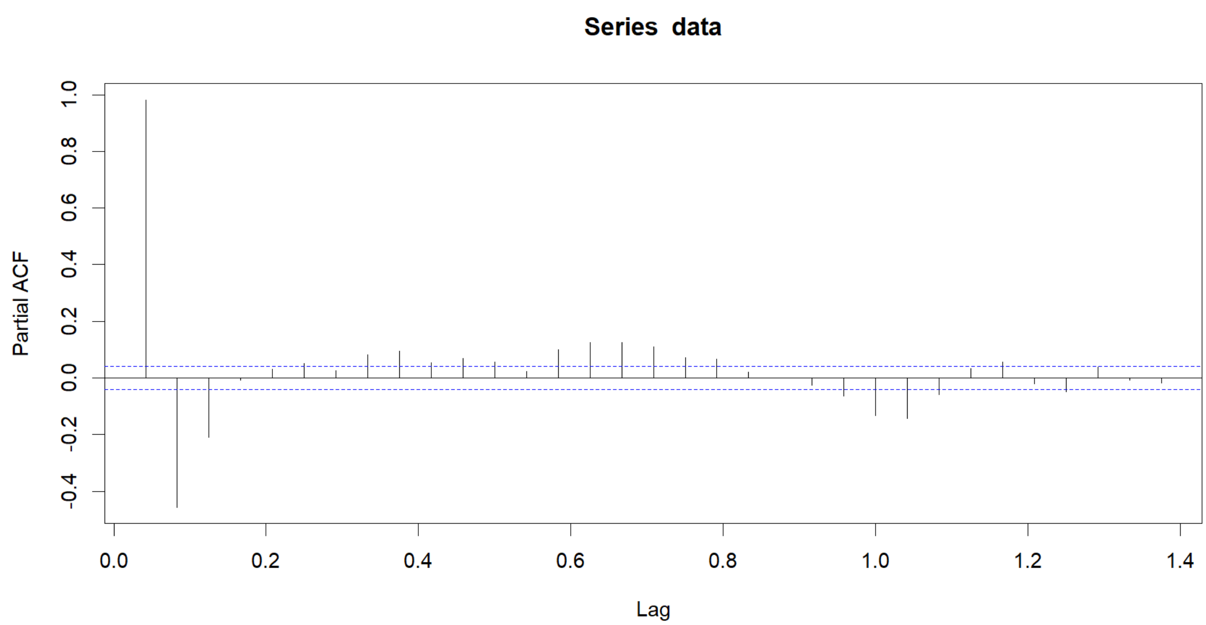
3. Plotting the data time series:



4. Autocorrelation of time series: acf(data)



5. Partial acf(gdpts)



6. adf test

```
> adf.test(data)

        Augmented Dickey-Fuller Test

data:  data
Dickey-Fuller = -6.287, Lag order = 12, p-value = 0.01
alternative hypothesis: stationary
```

7. Auto Arima model

```
>model=auto.arima(data,ic="aic",trace=TRUE)

 Fitting models using approximations to speed things up...

 ARIMA(2,0,2)(1,1,1)[24] with drift         : Inf

 ARIMA(0,0,0)(0,1,0)[24] with drift         : 11466.33

 ARIMA(1,0,0)(1,1,0)[24] with drift         : 5676.337

 ARIMA(0,0,1)(0,1,1)[24] with drift         : 8977.075

 ARIMA(0,0,0)(0,1,0)[24]                    : 11473.89

 ARIMA(1,0,0)(0,1,0)[24] with drift         : 6252.305

 ARIMA(1,0,0)(2,1,0)[24] with drift         : 5438.054

 ARIMA(1,0,0)(2,1,1)[24] with drift         : Inf

 ARIMA(1,0,0)(1,1,1)[24] with drift         : Inf

 ARIMA(0,0,0)(2,1,0)[24] with drift         : 11281.55

 ARIMA(2,0,0)(2,1,0)[24] with drift         : 5374.887

 ARIMA(2,0,0)(1,1,0)[24] with drift         : 5600.859

 ARIMA(2,0,0)(2,1,1)[24] with drift         : Inf

 ARIMA(2,0,0)(1,1,1)[24] with drift         : Inf

 ARIMA(3,0,0)(2,1,0)[24] with drift         : 5331.394

 ARIMA(3,0,0)(1,1,0)[24] with drift         : 5559.53

 ARIMA(3,0,0)(2,1,1)[24] with drift         : Inf
```

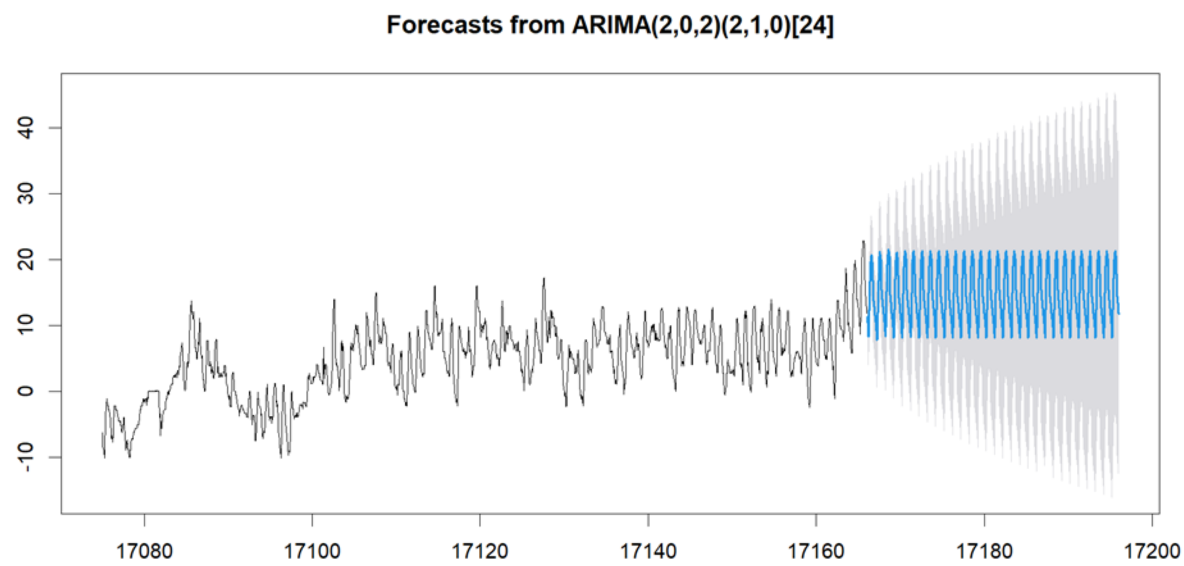
```
ARIMA(3,0,0)(1,1,1)[24] with drift         : Inf
ARIMA(4,0,0)(2,1,0)[24] with drift         : 5332.032
ARIMA(3,0,1)(2,1,0)[24] with drift         : 5331.313
ARIMA(3,0,1)(1,1,0)[24] with drift         : 5558.243
ARIMA(3,0,1)(2,1,1)[24] with drift         : Inf
ARIMA(3,0,1)(1,1,1)[24] with drift         : Inf
ARIMA(2,0,1)(2,1,0)[24] with drift         : 5340.401
ARIMA(4,0,1)(2,1,0)[24] with drift         : 5334.033
ARIMA(3,0,2)(2,1,0)[24] with drift         : 5332.077
ARIMA(2,0,2)(2,1,0)[24] with drift         : 5330.361
ARIMA(2,0,2)(1,1,0)[24] with drift         : 5556.545
ARIMA(2,0,2)(2,1,1)[24] with drift         : Inf
ARIMA(1,0,2)(2,1,0)[24] with drift         : 5343.612
ARIMA(2,0,3)(2,1,0)[24] with drift         : 5331.938
ARIMA(1,0,1)(2,1,0)[24] with drift         : 5390.12
ARIMA(1,0,3)(2,1,0)[24] with drift         : 5332.634
ARIMA(3,0,3)(2,1,0)[24] with drift         : 5334.228
ARIMA(2,0,2)(2,1,0)[24]                    : 5329.467
ARIMA(2,0,2)(1,1,0)[24]                    : 5555.177
ARIMA(2,0,2)(2,1,1)[24]                    : Inf
ARIMA(2,0,2)(1,1,1)[24]                    : Inf
ARIMA(1,0,2)(2,1,0)[24]                    : 5342.563
ARIMA(2,0,1)(2,1,0)[24]                    : 5339.546
ARIMA(3,0,2)(2,1,0)[24]                    : 5331.22
ARIMA(2,0,3)(2,1,0)[24]                    : 5331.029
ARIMA(1,0,1)(2,1,0)[24]                    : 5388.923
ARIMA(1,0,3)(2,1,0)[24]                    : 5331.69
```

```
 ARIMA(3,0,1)(2,1,0)[24]                    : 5330.489
 ARIMA(3,0,3)(2,1,0)[24]                    : Inf

Now re-fitting the best model(s) without approximations...

 ARIMA(2,0,2)(2,1,0)[24]                    : 5384.374

 Best model: ARIMA(2,0,2)(2,1,0)[24]
```

8. Forecasting temperature for next 1 day

	Point Forecast	Lo 95	Hi 95
17166.04	11.248803	9.608901	12.88871
17166.08	10.731641	8.285073	13.17821
17166.12	10.136301	6.972405	13.30020
17166.17	9.559795	5.800959	13.31863
17166.21	8.281384	4.035497	12.52727
17166.25	8.287412	3.641851	12.93297
17166.29	10.885030	5.909181	15.86088
17166.33	13.721924	8.470936	18.97291
17166.38	15.721102	10.239178	21.20303
17166.42	17.608383	11.931330	23.28544
17166.46	19.327787	13.484923	25.17065
17166.50	19.884300	13.899869	25.86873
17166.54	20.491771	14.385993	26.59755
17166.58	20.578257	14.368124	26.78839
17166.62	19.558943	13.258821	25.85906
17166.67	19.521777	13.143878	25.89968
17166.71	18.510491	12.065241	24.95574
17166.75	15.073873	8.570209	21.57754
17166.79	13.978204	7.423805	20.53260
17166.83	13.325428	6.726914	19.92394
17166.88	13.118927	6.482017	19.75584
17166.92	12.649515	5.979158	19.31987
17166.96	11.722233	5.022720	18.42175
17167.00	11.328230	4.603285	18.05318

9. Plotting the forecast



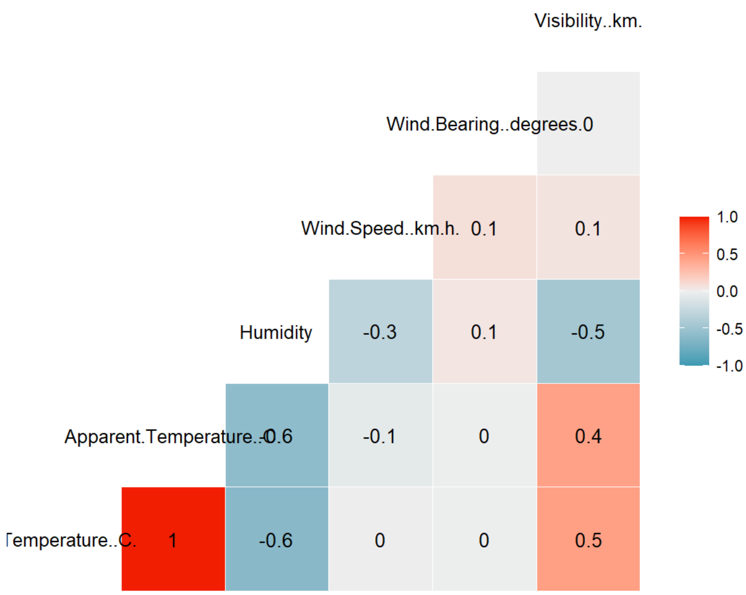
10. Calculating Accuracy of the Model

```
> accuracy(model)
                     ME      RMSE       MAE MPE MAPE      MASE
Training set 0.01742321 0.8309363 0.6157592 NaN  Inf 0.2180051
                     ACF1
Training set 0.0008098431
```

Inference:

a.From Multiple regression Model:

Multivariate Regression The best model was made after considering 3 variable which were highly correlated to the dependent variable and those variables were Apparent.Temperature (0.9931), Humidity(-0.67) and Visibility(0.519).



As seen in the summary of the model, the p-value in the F-Statistics is less than 0.05, hence the model is accepted. The correlation test of all the independent attributes with the dependent variable was conducted and the significant attributes have been displayed and considered for multiple regression model.

b. From Forecasting Model:

Since, the p-value in Augmented Dickey-Fuller Test is found to be 0.01, being less than 0.05, the **time series is stationary.** The Best ARIMA model found was **model: ARIMA(2,0,2)(2,1,0)[24] .** The forecast has been plotted with its low and high range. The forecasted values have been displayed for the next month. The **mean error** in the auto Arima model was found to be **0.01742321**.

Program:

1.Mulitple Linear Regression Model:

```
rm(list=ls())
setwd("D:/6th_Semester/Data_Analytics_Lab/Lab 3")
df=read.csv("weatherHistory2016.csv")
head(df)
library(dplyr)
library(tidyr)
library(GGally)
a=sample_n(df,200)
a %>% drop_na()
a <- a[,c(4:9)]
head(a)
cor.test(a$Temperature..C.,a$Apparent.Temperature..C.)
cor.test(a$Temperature..C.,a$Humidity)
cor.test(a$Temperature..C.,a$Wind.Speed..km.h.)
cor.test(a$Temperature..C.,a$Wind.Bearing..degrees.)
cor.test(a$Temperature..C.,a$Pressure..millibars.)
cor.test(a$Temperature..C.,a$Visibility..km.)
ggcorr(a, label = TRUE)
lmodel=lm(a$Temperature..C.~a$Apparent.Temperature..C.+a$Humidity+a$Visibilit
y..km.)
summary(lmodel)
```

2.Time Series Forecasting:

```
rm(list=ls())
setwd("D:/6th_Semester/Data_Analytics_Lab/Lab 3")
df=read.csv("weatherHistory2016.csv")
library(forecast)
library(tseries)
data<-ts(df$Temperature..C.,start = as.Date("2016-10-01"),end =
as.Date("2016-12-31"),frequency = 24)
plot(data)
acf(data)
pacf(data)
adf.test(data)
model=auto.arima(data,ic="aic",trace=TRUE)
forecastedVal=forecast(model,level=c(95),h=24)
print(forecastedVal)
plot(forecastedVal)
accuracy(model)
```

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING(SCOPE)

WINTER SEMESETER 2021-22

LAB-4

NAME : RISHABH SINGH

REG NO: 19BCE1500

COURSE: ESSENTIAL OF DATA ANALYTIC(CSE3506)

FACULTY: DR.LASKHMI PATHI JAKKAMPUTI

SLOT: L21+L22

Tasks for Week-4: Analysis of Variance (ANOVA)

Perform ANOVA test and determine the statistical differences between the means of individual groups given in the data.

Aim: To perform an ANOVA test and determine the statistical differences between the means of individual groups given in the data.

Step 1: Load the dplyr library and import the dataset.

Step 2: Using the group_by function, group the data based on color.

Step 3: Apply ANOVA using response with respect to color and generate summary.

Step 4: If the Pr(>F) -value is less than 0.05, then perform the Tukey HSD test.

Step 5: If the pair's p-adjusted value is less than 0.05, they're significantly different; otherwise, they're not.

Statistics

1) Applying group by

```
group_by(data,color) %>% summarise(count = n(),mean = mean(response, na.rm=TRUE))
```

color <chr>	count <int>	mean <dbl>
blue	24	10.632083
green	24	8.530417
red	24	2.491667

3 rows

2) Summary of ANOVA

```
ANOVA <- aov(response~color, data = data)

summary(ANOVA)
```

```
             Df Sum Sq Mean Sq F value   Pr(>F)    
color         2  857.2   428.6   14.81 4.44e-06 ***
Residuals    69 1996.4    28.9                     
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

3) Conducting Tukey HSD Test

```
TukeyHSD(ANOVA)
```

```
  Tukey multiple comparisons of means
    95% family-wise confidence level

Fit: aov(formula = response ~ color, data = data)

$color
                diff        lwr       upr     p adj
green-blue -2.101667  -5.821045  1.617711 0.3709119
red-blue   -8.140417 -11.859795 -4.421039 0.0000049
red-green  -6.038750  -9.758128 -2.319372 0.0006628
```

Inference:

As seen in the summary of ANOVA, the profit value (Pr(>F)) is less than 0.05, hence the null hypothesis is rejected and the Tukey HSD test is required.

As seen in the Tukey HSD test results,

a) green and blue are not significantly different since p adj is more than 0.05.
b) red and blue are significantly different since p adj is less than 0.05.
c) green and red are significantly different since p adj is less than 0.05.

PROGRAM:

```
rm(list=ls())
data <- read.csv("color-anova-example.csv")
library(dplyr) # To group the data
group_by(data,color) %>% summarise(count = n(),mean = mean(response, na.rm =
TRUE))
# ANOVA
ANOVA <- aov(response~color, data = data)
summary(ANOVA)
TukeyHSD(ANOVA)
```

Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING(SCOPE)

WINTER SEMESETER 2021-22

LAB-5

NAME : RISHABH SINGH

REG NO: 19BCE1500

COURSE: ESSENTIAL OF DATA ANALYTIC(CSE3506)

FACULTY: DR.LASKHMI PATHI JAKKAMPUTI

SLOT: L21+L22

Tasks for Week-5: Logistic Regression

Understand the following operations/functions to perform logistic Regression and perform similar operations on the 'Social_Network_Ads' dataset based on given instructions.

Aim: To perform logistic Regression and perform similar operations on the 'Social_Network_Ads' dataset.

Algorithm:

1. Import the dataset and load the caTools library.
2. Split the data using split function into test and train data in a ratio=0.8.
3. Convert the purchased and Gender variable to categorical variable using as.factor.
4. Apply the generalized linear model using glm command for the dependent and independent variables and print the summary.
5. Using the trained model, predict the output for the test data and observe the accuracy and plot the graphs.
6. Generate and Display the confusion matrix

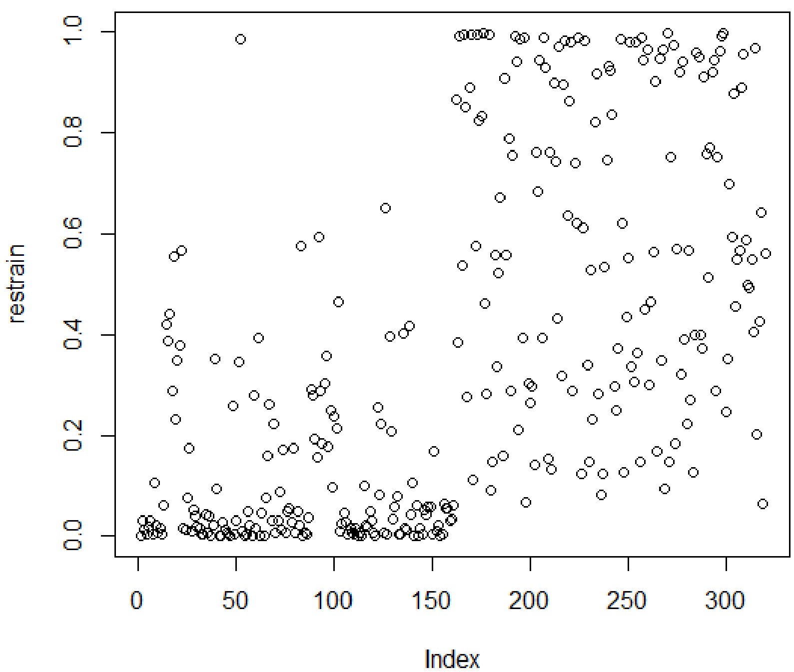
Statistics

1) Summary of the applied model

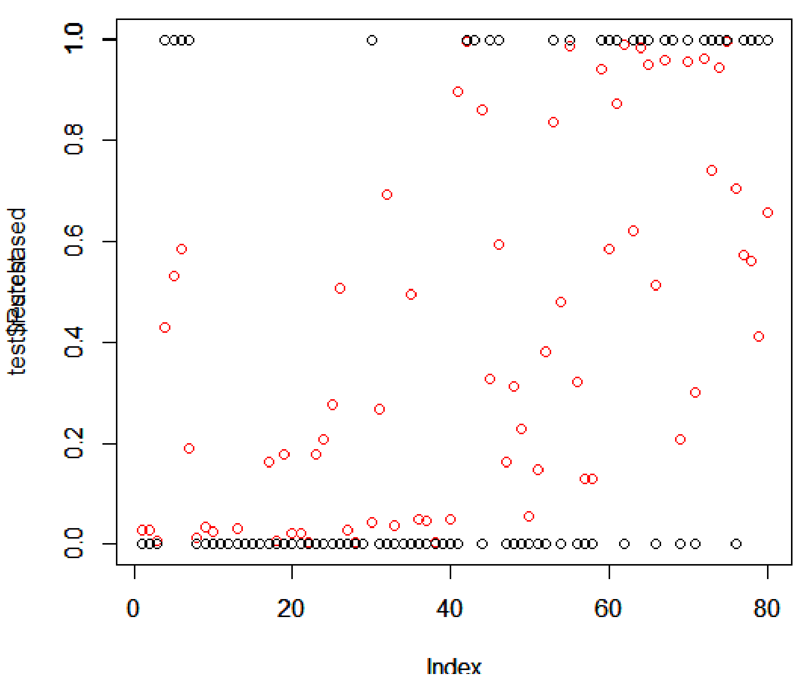
```
Call:
glm(formula = Purchased ~ Age + Gender + EstimatedSalary, family = "binomial",
    data = train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-3.0223  -0.4848  -0.1298   0.3453   1.8205

Coefficients:
                  Estimate Std. Error z value Pr(>|z|)
(Intercept)     -1.372e+01  1.647e+00  -8.328  < 2e-16 ***
Age              2.534e-01  3.208e-02   7.899 2.81e-15 ***
GenderMale       5.388e-01  3.520e-01   1.531    0.126
EstimatedSalary  3.952e-05  6.308e-06   6.266 3.71e-10 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 417.96  on 319  degrees of freedom
Residual deviance: 213.93  on 316  degrees of freedom
AIC: 221.93

Number of Fisher Scoring iterations: 6
```

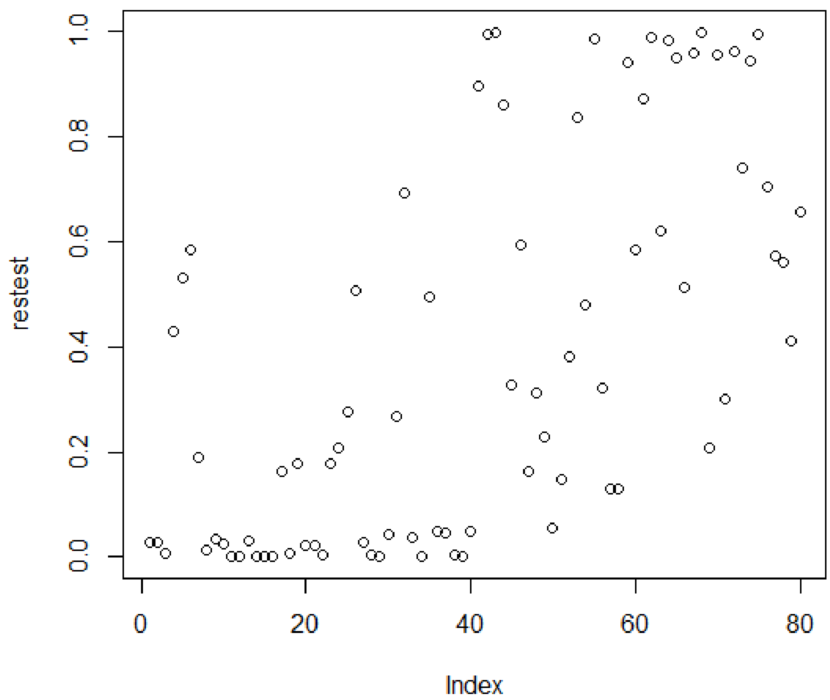
```
> cfmatrix
   pred
Act FALSE TRUE
  0    45    7
  1     5   23
> Acc=(cfmatrix[[1,1]]+cfmatrix[[2,2]])/sum(cfmatrix)
> Acc
[1] 0.85
>
```

2) Graph of predicted train data



3) Graph of Predicted Test





Inference:

1) As seen in the summary of glm model, the profit value (Pr(>|z|)) is less than 0.05 for all variables apart from GenderMale. Hence, all but one are accepted in the model.

2) The accuracy of the trained model is observed to be 0.85 i.e., 85%.

3) There are a total of 80 objectives as seen from the confusion matrix.

4) The graph shows the actual and predicted values of the trained model for the train and test data.

Program:

```r
rm(list=ls())
setwd("C:\\Users\\risha\\Desktop\\6thSemester\\EDA\\Lab\\05")
mydata<-read.csv("Social_Network_Ads.csv")
library(caTools)
splitd<-sample.split(mydata,SplitRatio = 0.8)
train=subset(mydata,splitd=="TRUE")
test=subset(mydata,splitd=="FALSE")
train
mydata$Gender<-as.factor(mydata$Gender)
mydata$Purchased<-as.factor(mydata$Purchased)
mymodel <- glm(Purchased ~ Age+Gender+EstimatedSalary, data=train,
               family='binomial')
summary(mymodel)
restrain<-predict(mymodel,train,type='response')
plot(restrain)
restest<-predict(mymodel,test,type='response')
plot(restest,col='red')
par(new=TRUE)
plot(test$Purchased)
cfmatrix<-table(Act=test$Purchased, pred=restest>0.5)
cfmatrix
Acc=(cfmatrix[[1,1]]+cfmatrix[[2,2]])/sum(cfmatrix)
Acc
plot(restest)
```

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING(SCOPE)

WINTER SEMESETER 2021-22

Lab -6

NAME : RISHABH SINGH

REG NO: 19BCE1500

COURSE: ESSENTIAL OF DATA ANALYTIC(CSE3506)

FACULTY: DR.LASKHMI PATHI JAKKAMPUTI

SLOT: L21-L22

Tasks for Week-6: K-NN Algorithm

Aim: Understand the following operations/functions on to perform K- NN algorithm and perform similar operations on 'wdbc' dataset based on given instructions.

Algorithm:

1. Removing all the values from the global environment
2. Because our dataset isn't in a csv format, we're going to use a file.choose() is used to select a dataset for prediction.
3. Use the view() function to see the dataset.
4. The Mynorm function was built to normalise the values that separate each other.
5. Combining each value with its min value, dividing the difference between the max and min values.
6. Make a new dataframe called mydata and put all of the normalised values in it. except the first column, which is a category, in that new dataframe data.
7. For comparing the original dataset and normalized dataset take 2 to 5 columns of both data set and apply summary() function to find summary.
8. Divide the first 400 values into a train dataset and the remaining 169 values into a passenger dataset. from mydata's test dataset (normalized dataset). Use the library() function to import a class.
9. Perform the knn algorithm and save all predicted values to the pred variable.
10. Using the expected data from the pred variable, create a confusion matrix.In the first dataset, there are 401 to 569 rows.
11. Find the accuracy of the data by adding the [1,1] element and [2,2] element and dividing its summation with the whole sum.

Inference: The accuracy of the model is 97%. so, we can say that the model is best fit model.

Confusion matrix:

Pred	B	M
B	128	3
M	2	36

Accuracy: 0.9704142

Program:

```
rm(list=ls())
setwd("D:\6th_Semester\Data_Analytics_Lab\New folder")
wdbc<-read.table(file.choose(),sep=',')
view(wdbc)
wdbc<-wdbc[,-1]
mynorm<-function(x){((x-min(x))/(max(x)-min(x)))}
mydata<-as.data.frame(lapply(wdbc[,-1], mynorm))
summary(wdbc[,2:5])
summary(mydata[,1:4])
train<-mydata[1:400,]
test<-mydata[401:569,]
library(class)
pred<-knn(train,test,wdbc[1:400,1],k=21)
cf<-table(pred,wdbc[401:569,1])
cf
acc=(cf[[1,1]]+cf[[2,2]])/sum(cf)
acc
```

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING(SCOPE)

WINTER SEMESETER 2021-22

Lab -7

NAME : RISHABH SINGH

REG NO: 19BCE1500

COURSE: ESSENTIAL OF DATA ANALYTIC(CSE3506)

FACULTY: DR.LASKHMI PATHI JAKKAMPUTI

SLOT: L21-L22

Tasks for Week-7 : Partition Based clustering

Aim: Understand the following operations/functions on 'iris' data and perform similar operations on 'USArrests' dataset based on given instructions.

Algorithm:

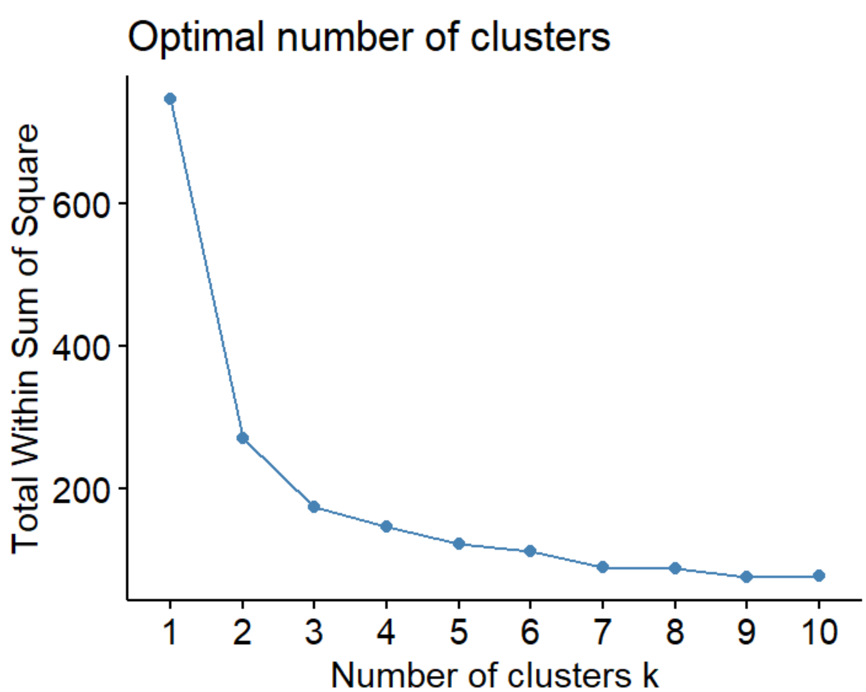
1. Remove all the values from the global environment.
2. Set the working directory to the dataset where we store by using setwd().
3. To see the dataset use view() function.
4. By using scale function. We scale the data and store it in another variable.
5. Using kmeans function we find the kmeans clustering with 2 center at first it can be of any centers and store the result in fit
6. By using fit$cluster we can find the cluster values.
7. By using fit$size we can find the size of each cluster.
8. By using fit$withnss we can find with in cluster sum of squares for each cluster.
9. By using fit$tot.withnss we can find with in cluster sum of squares with respective to all clusters.
10. Create the no of iterations we need to find the perfect cluster and size of wcss and the nclust list.
11. To find the best no of center from 1 to 15 we create a for loop.
    a. find the kmeans cluster with each center value in for loop
    b. put to the total with in cluster sum of squares for each iteration in wcss
    c. put the size of cluster in nclust.
12. plot the graph between the no of center and the wcss values for each center. the place where we find the bend that is our no of cluster should be taken.
13. In other way we can use factoextra libaray.

14. Using fviz_nbclust function we can find the graph
15. Using fviz_cluster function we can find the clusters
16. Call cluster library
17. We use pam function to find the k medoid clusters and store the values in fitm.
18. By using the fitm$medoid we can find no of medoid.
19. Using fviz_cluster function we can find the medoids.

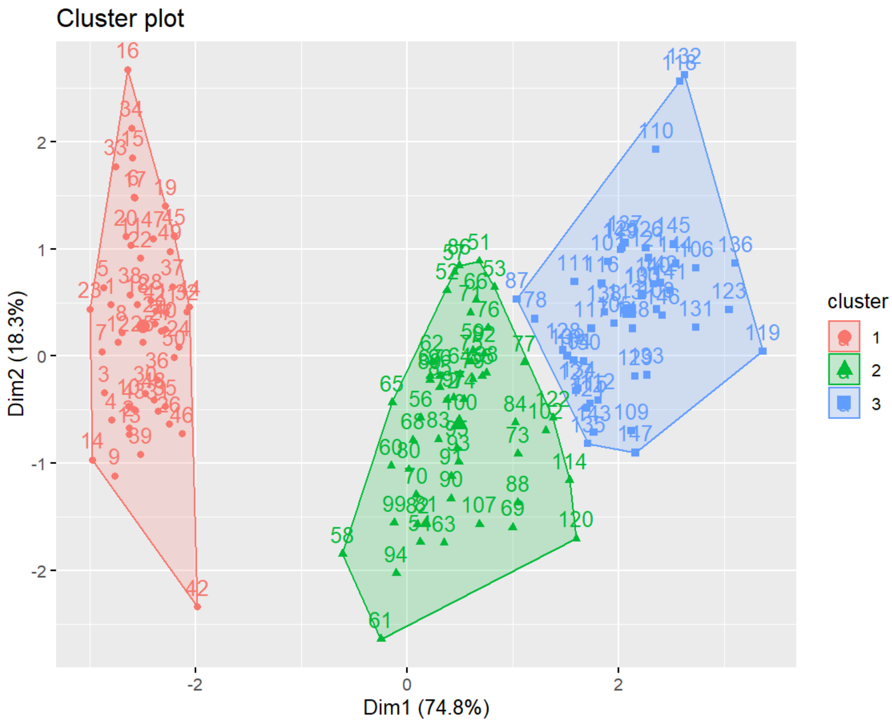
Result

1.Dataset: iris.csv

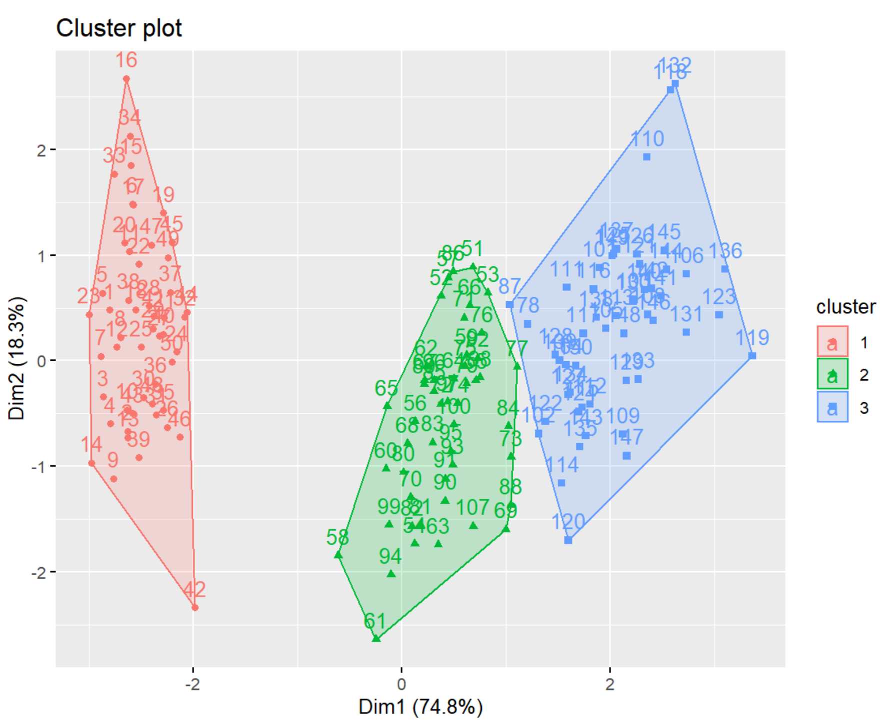
1.Optimal number of clusters:



2.K-means centers:

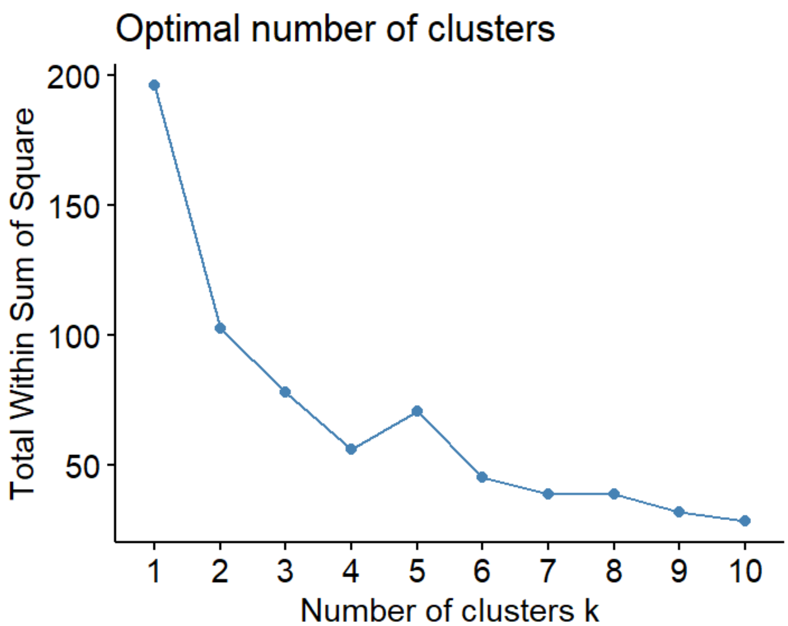


3.K-medoid centers:

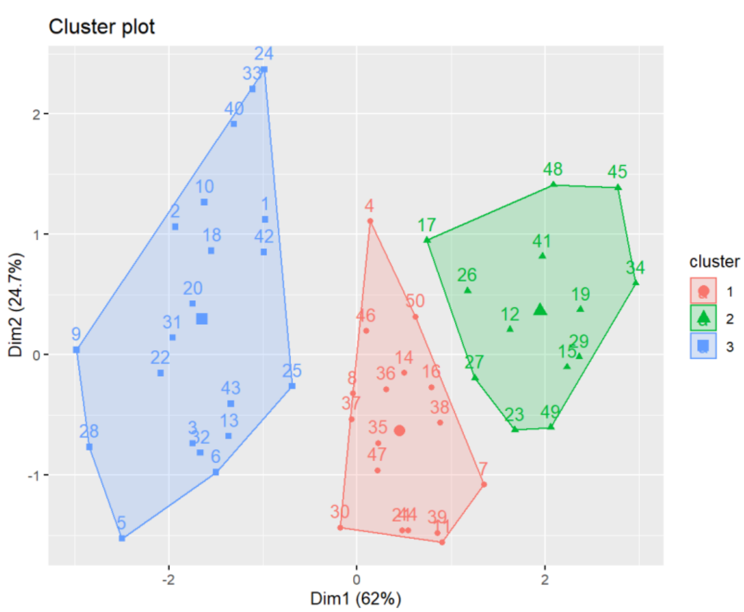


2.Dataset: USArrest.csv

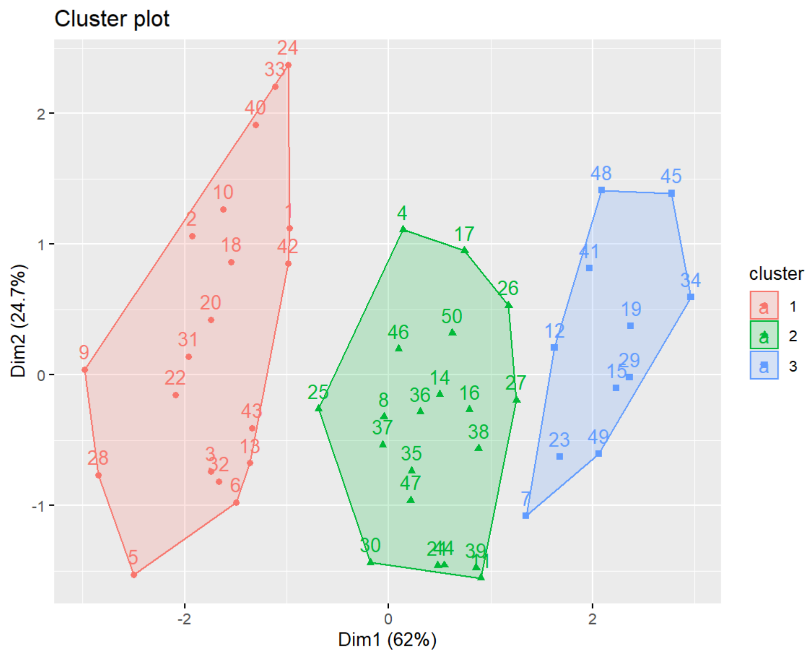
1.Optimal number of clusters:



2.K-means centers:



3.K-medoid centers:



Statistics

Dataset: iris.csv

K-means centers

x	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width
-1.15087068	-1.01119138	0.85041372	-1.3006301	-1.2507035
0.07534946	0.03881135	-0.73324663	0.3059615	0.2137533
1.13936197	1.03196952	-0.07784286	1.0386287	1.0894947

K-medoid centers

x	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width
-1.07030973	-0.7769106	0.7861738	-1.3357516	-1.3110521
-0.08056095	0.3099591	-0.5903951	0.1370873	0.1320673
0.95522266	0.7930124	-0.1315388	0.9868021	0.7880307

Dataset: USArrest.csv

K-means centers

Murder	Assault	UrbanPop	Rape
-0.4469795	-0.3465138	0.4788049	-0.2571398
-0.9615407	-1.1066010	-0.9301069	-0.9667633
1.0049340	1.0138274	0.1975853	0.8469650

K-medoid centers

Murder	Assault	UrbanPop	Rape
0.8292944	1.3708088	0.3081225	1.1603196
-0.2727580	-0.2371077	0.1699510	-0.1315342
-1.2829727	-1.3770485	-0.5899924	-1.0603878

Program:

Dataset: iris.csv:

```
rm(list=ls())
setwd("D:/6th_Semester/Data_Analytics_Lab/lab 7")
data1<-read.csv("iris.csv")
View(data1)
df<-scale(data1)
fit<-kmeans(df,centers=2)
fit$cluster
fit$size
fit$withinss
fit$tot.withinss
Kmax<-15
wcss<-rep(NA,Kmax)
nClust<- list()
for(i in 1:Kmax){
  fit<-kmeans(df,i)
  wcss[i]<-fit$tot.withinss
  nClust[[i]]<-fit$size
}
plot(1:Kmax,wcss,type="b",pch=19)
fit<-kmeans(df,centers=3)
fit$cluster
fit$size
fit$center
library(factoextra)
fviz_nbclust(df, kmeans, method = "wss")
fviz_cluster(fit, data1)
library(cluster)
fitm <- pam(df, 3, metric = "manhattan")
fitm
fitm$medoids
fviz_cluster(fitm, data1)
```

Dataset: USArrest.csv:

```r
rm(list=ls())
setwd("D:/6th_Semester/Data_Analytics_Lab/lab 7")
data2<-read.csv("USArrests.csv")
view(data2)
data2<-data2[,-1]
df1<-scale(data2)
fit1<-kmeans(df1,centers=2)
fit1$cluster
fit1$size
fit1$withinss
fit1$tot.withinss
Kmax1<-15
wcss1<-rep(NA,Kmax1)
nClust1<- list()
for(i in 1:Kmax1){
  fit1<-kmeans(df1,i)
  wcss1[i]<-fit1$tot.withinss
  nClust1[[i]]<-fit1$size
}
plot(1:Kmax1,wcss1,type="b",pch=19)
fit1<-kmeans(df1,centers=3)
fit1$cluster
fit1$size
fit1$center
library(factoextra)
fviz_nbclust(df1, kmeans, method = "wss")
fviz_cluster(fit1, data2)
library(cluster)
fitm1 <- pam(df1, 3, metric = "manhattan")
fitm1
fitm1$medoids
fviz_cluster(fitm1, data2)
```

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING(SCOPE)

WINTER SEMESETER 2021-22

LAB-8

NAME : **RISHABH SINGH**

REG NO: **19BCE1500**

COURSE: **ESSENTIAL OF DATA ANALYTIC(CSE3506)**

FACULTY: **DR.LASKHMI PATHI JAKKAMPUTI**

SLOT: **L21+L22**

Tasks for Week-8: Hierarchical Clustering

Aim: To understand the following operations/functions on ‘USArrests’ data and perform similar operations on ‘iris’ dataset based on given instructions.

Algorithm:

1. Removing all the values from the global environment
2. Set the working directory to the dataset where we store by using setwd().
3. To see the dataset use view() function.
4. By using scale function, we scale the data and store it in another variable.
5. Using dist function we find the Euclidean distances for the scaled data.
6. By using the Euclidean distances and hclust function we can create and then plot the hierarchical
clustering dendogram.
7. By using cutree we divide the elements of the dendogram into k number of clusters (k=4 in our
case).
8. Then, by using rect.hclust function we can divide the dendogram into k clusters (k=4 in our case) i.e. create k rectangular divisions/borders in the dendogram

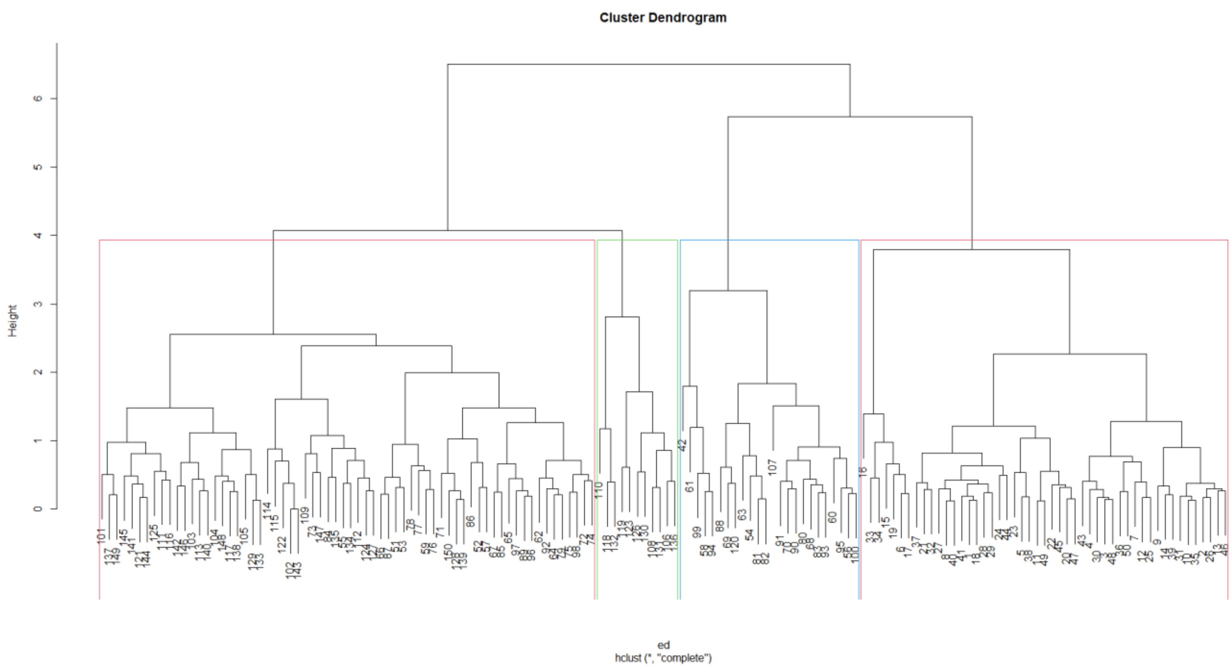
Result

<u>Dataset: iris.csv</u>

Cluster:

```
> cluster
  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42 
  1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   2 
 43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84 
  1   1   1   1   1   1   1   1   3   3   3   2   3   2   3   2   3   2   2   3   2   3   3   3   3   2   2   2   3   3   3   3   3   3   3   3   3   2   2   2   2   3 
 85  86  87  88  89  90  91  92  93  94  95  96  97  98  99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 
  3   3   3   2   3   2   2   3   2   2   2   3   3   3   2   2   3   3   3   3   3   4   2   4   3   4   3   3   3   3   3   3   3   4   4   2   3   3   4   3   3   4 
127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150 
  3   3   3   4   4   4   3   3   3   4   3   3   3   3   3   3   3   3   3   3   3   3   3   3 
```

Cluster Dendogram

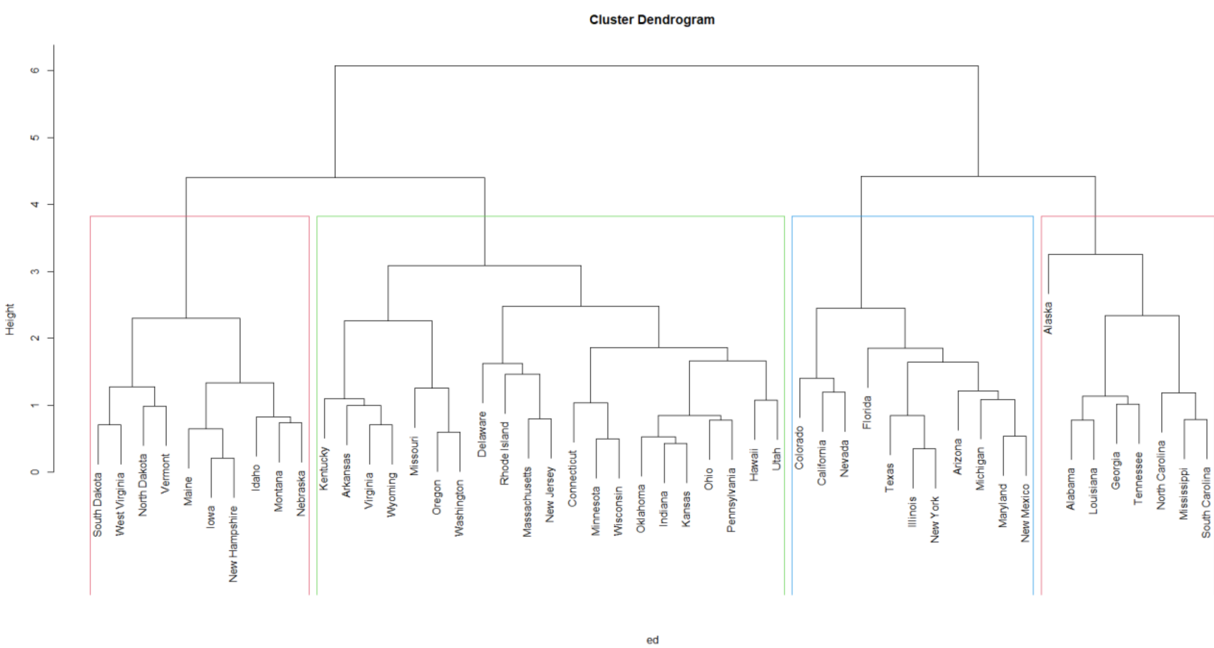


Dataset: USArrest.csv

Cluster:

```
> cluster
       Alabama         Alaska        Arizona       Arkansas     California       Colorado    Connecticut       Delaware        Florida        Georgia         Hawaii 
             1              1              2              3              2              2              3              3              2              1              3 
         Idaho       Illinois        Indiana           Iowa         Kansas       Kentucky      Louisiana          Maine       Maryland  Massachusetts       Michigan 
             4              2              3              4              3              3              1              4              2              3              2 
     Minnesota    Mississippi       Missouri        Montana       Nebraska         Nevada  New Hampshire     New Jersey     New Mexico       New York North Carolina 
             3              1              3              4              4              2              4              3              2              2              1 
  North Dakota           Ohio       Oklahoma         Oregon   Pennsylvania   Rhode Island South Carolina   South Dakota      Tennessee          Texas           Utah 
             4              3              3              3              3              3              1              4              1              2              3 
       Vermont       Virginia     Washington  West Virginia      Wisconsin        Wyoming 
             4              3              3              4              3              3 
```

Cluster Dendogram



Program:

Dataset: iris.csv

```
rm(list=ls())
setwd("D:\\EDA\\Lab\\08")
data <- read.csv("iris.csv",row.names=1)
View(data)
df <- scale(data)
View(df)
ed <- dist(df, method = 'euclidean')
hierClust <- hclust(ed, method = 'complete')
plot(hierClust)
cluster <- cutree(hierClust, k = 4)
cluster
rect.hclust(hierClust, k = 4, border = 2:4)
```

Dataset: USArrest.csv

```
rm(list=ls())
setwd("D:\\EDA\\Lab\\08")
data <- read.csv("USArrests.csv",row.names=1)
View(data)
df <- scale(data)
View(df)
ed <- dist(df, method = 'euclidean')
hierClust <- hclust(ed, method = 'complete')
plot(hierClust)
cluster <- cutree(hierClust, k = 4)
cluster
rect.hclust(hierClust, k = 4, border = 2:4)
```

Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING(SCOPE)

WINTER SEMESETER 2021-22

Lab -9

NAME : RISHABH SINGH

REG NO: 19BCE1500

COURSE: ESSENTIAL OF DATA ANALYTIC(CSE3506)

FACULTY: DR.LASKHMI PATHI JAKKAMPUTI

SLOT: L21-L22

Week 9 Task

AIM: To calculate the value of a and b for y=ax+b using gradient descent method.

ALGORITHM:

Step 1: Initialize the weights (a & b) with random values and calculate the loss function.

Step 2: Calculate the gradient. This helps us move the values of a & b in the direction in which loss function is minimized.

Step 3: Adjust the weights with the gradients to reach the optimal values where loss function is minimized.

Step 4: Use the new weights for prediction and to calculate the new loss function.

Step 5: Repeat steps 2 and 3 till further adjustments to weights don't significantly reduce the Error.

STATISTICS:

Values using Gradient Descent method:

FIELD	VALUE
Optimum Slope	-5.33401243341807
Optimum Intercept	37.2487084651956
Number of iterations	580
Loss function	0.00411973531571587

Values using Linear Regression:

FIELD	VALUE
(Intercept)	37.285
Slope	-5.344

RESULT:

We can see that slope and intercept we have calculated using gradient descent method is almost same as values of Linear Regression. Therefore, we have calculated the values of m and c for y=mx+c.

INFERENCE:

Hence, we have obtained the optimal value of the weights **m** and **c**.

CODE:

```
rm(list=ls())
gd<-function(x,y,m,c,alpha,conv_thr,iter){
  iterations=0
  Lf=0
  while(iterations<=iter){
    y_pred=m*x+c
    Lf_new=0.5*(sum(y_pred-y)^2)
    m=m-alpha*sum((y_pred-y)*x)
    c=c-alpha*sum(y_pred-y)
    if(abs(Lf-Lf_new)<conv_thr){
      break;
    }
    Lf=Lf_new
    iterations=iterations+1
  }
  return(paste('Optimum Slope',m,"Optimum Intercept",c,"Number of
iterations",iterations,"Loss function",Lf))
}
data<-mtcars
gd(data$wt,data$mpg,32,-0.2,0.005,0.0001,10000)
reg<-lm(data$mpg~data$wt)
reg
```

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING(SCOPE)

WINTER SEMESTER 2021-22

Lab -10

NAME: RISHABH SINGH

REG NO: 19BCE1500

COURSE: ESSENTIALS OF DATA ANALYTIC(CSE3506)

FACULTY: DR.LASKHMI PATHI JAKKAMPUTI

SLOT: L21-L22

Tasks for Week-10: Gradient Descent with Momentum Optimizer

Aim – Apply multiple regression on mtcars dataset using momentum optimized gradient descent.

Algorithm:

1. Select columns for multiple regression
2. Give learn rate, gamma(momentum) and max iterations in function
3. Pick values for m1, m2 & c.
4. Initialize values for nu_m1, nu_m2 and nu_c to be 0.
5. Initialize iteration=0
6. If iteration<max_iteration
    a. Calculate y_pred
    b. Calculate loss function
    c. Update nu_m1,nu_m2 and nu_c using the below formula:
        i. Nu_m1=gamma*nu_m1+alpha*sum((y_pred-y)*x1)
        ii. Nu_m2=gamma*nu_m2+alpha*sum((y_pred-y)*x2)
        iii. Nu_c=gamma*nu_c+alpha*sum(y_pred-y)
    d. Update m1,m2,c &Lf
    e. Print intercept, slope and loss function
7. Repeat step 5 continuously.
8. Use lm function to check for linear model.

Statistics-

1. Momentum optimizer

C	**37.2272414172067**
M1	**-3.87782187933926**
M2	**-0.0317729604979703**

2. Using lm function Multilinear regression

C	**37.22727**
M1	**-3.87783**
M2	**-0.03177**

```
> mgd(data$wt,data$hp,data$mpg,-0.2,-0.2,32,0.000002,0.98,50000)
[1] "Optimal intercept: 37.2272414172067 Optimal slope: -3.87782187933926 -0.0317729604979703
 Loss function 97.5238773718236"
```

Using lm() function multilinear regression:

```
> summary(model)

Call:
lm(formula = data$mpg ~ data$hp + data$wt)

Residuals:
   Min     1Q Median     3Q    Max 
-3.941 -1.600 -0.182  1.050  5.854 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 37.22727    1.59879  23.285  < 2e-16 ***
data$hp     -0.03177    0.00903  -3.519  0.00145 ** 
data$wt     -3.87783    0.63273  -6.129 1.12e-06 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2.593 on 29 degrees of freedom
Multiple R-squared:  0.8268,	Adjusted R-squared:  0.8148 
F-statistic: 69.21 on 2 and 29 DF,  p-value: 9.109e-12
```

Inferences:

1. In momentum gradient, descent loss function is not important but in gradient descent, loss function is important as it is used for convergence
2. If we put gamma as 0 the model behaves like gradient descent.
3. We can decrease the learning rate or increase the number of iterations to increase the accuracy

CODE:

```
#Momentum based Gradient Descent
mgd=function(x1,x2,y,m1,m2,c,alpha,gamma,iter){
  iterations=0
  #Lf=0
  nu_m1=0
  nu_m2=0
  nu_c=0
  while(iterations<=iter){
    y_pred=m1*x1+m2*x2+c
    Lf_new=0.5*sum((y_pred-y)^2)
    nu_m1=gamma*nu_m1+alpha*sum((y_pred-y)*x1)
    nu_m2=gamma*nu_m2+alpha*sum((y_pred-y)*x2)
    nu_c=gamma*nu_c+alpha*sum(y_pred-y)
    m1=m1-nu_m1
    m2=m2-nu_m2

    c=c-nu_c
    Lf=Lf_new
    iterations=iterations+1
  }
  paste("Optimal intercept:",c,"Optimal slope:",m1,m2,"Loss function",Lf)
}
data=mtcars
mgd(data$wt,data$hp,data$mpg,-0.2,-0.2,32,0.000002,0.98,50000)
model=lm(data$mpg~data$hp+data$wt)
summary(model)
```