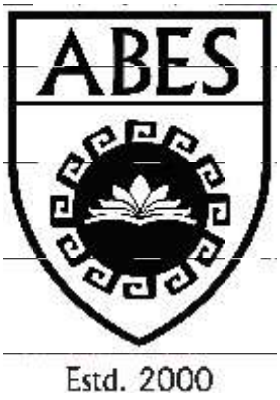
Practical File

Lab Name … **Data Analytics Lab…….** ***Lab Code……KIT-651……***

.



Name……Harshul Gupta…….

Adm.No…2018BCI1055…….. Univ. Roll No…1803211023……..…..

Course ……B.Tech…….……Branch……CEIT……..……..……..

Sem……6th…….…… Section……A……..……..……..

ABES
Engineering College
(College Code-032)
NAAC Accredited, NBA Accredited Branches (CSE, ECE, EN & IT)

19th Km. Stone, NH-09, Ghaziabad - 201009 (UP), India
Phone : 0120-7135112, 9999889341 Fax : 0120-7135115 Website : www.abes.ac.in, Email: info@abes.ac.in

ABES Engineering College, Ghaziabad

Student Name:..... Harshul Gupta
Roll No:..... 1803211023

I N D E X

Exp. No	Name of the Experiment	Date of Practical	Assessment by Faculty Member					
		Schedule / Actual	Pre-Lab Writing work * MM:05	Implementation/ Active Participation MM:10	Graph, Results/ Output, Calc. MM: 05	In Time submission of Lab reports/ Viva Voce MM:05	Total MM:25	Signature with Date
1	Understanding of statistics on dataset	4/13/2021						
2	Handling data files in R	20/04/2021						
3	Handling Multi-dimensional Array	22/05/2021						
4	Perform mean, median, mode and standard deviation	13/04/2021						
5	Perform data processing operations	27/04/2021						
6	Perform Dimensionality Reduction	4/5/2021						
7	Perform Simple Linear Regression	24/05/2021						

Teacher's Remarks (if any): **Average Marks:**

Name & Sign. Of Faculty members(s) with date **Name & Sign. Of Lab In charge** **HOD**

Note:
1. *Pre-Lab writing work should include problem statement, objective , Algorithum (if applicable)and Methodology.
2. Faculty members will check pre Lab writing work & Lab work readings/output in each class and sign with date.
3. Please use pen with blue colour Ink only.

Roll No: 1803211023 Date 13/04/2021 Page No. 1

Practical Name: To get the input from user & perform numerical op (Max, Min, Avg, Sum, SQRT, Round) using in R. Practical No. 1

Objective → Understanding of statistics on dataset.

Algorithm →

1. Take the array of number from user
2. find minimum, maximum, average, sum and square root of each number.
3. Use min(), max(), avg(), sum(), sqrt(), round() functions
4. Display output.

Code →

```
x ← strsplit(readline(prompt="Enter the list of numbers:"), "")
x ← as.double(x[[1]])
print(x)
print(c("Max:", max(x)))
print(c("Min:", min(x)))
print(c("Sum:", sum(x)))
print(c("Average:", mean(x)))
print(c("Round:", round(x)))
print(c("Square Root:", sqrt(x)))
```

Output →

```
Enter the list of numbers: 5 4.8 55 12.6
5.0 4.8 55.0 12.6
Max: 55
Min: 4.8
Sum: 77.4
Average: 19.35
Round: 5 5 55 13
Squareroot: 2.23 2.19 7.41 3.54
```

Roll No: 18032110023 Date 20/04/2021 Page No.

Practical Name: To perform data import/export (.csv, .xls, .txt) Practical No. 2

Objective → Handling the data files in R programming.

Algorithm →

1. Explore the UCI machine learning repository.
2. Download the dataset in .csv, .xls and .txt file format.
3. Use read() functions to import the dataset in R.
4. Save the file in data frame and print the header using header() function.

Code →

```
data1 ← read.csv("E:\\sample.csv")
data2 ← read.table("E:\\sample.csv", sep="",
                   dec=".")
data3 ← read.xlsx("E:\\sample.xlsx",
                  sheetName="Sheet1")
head(data1)
head(data2)
head(data3)
v ← c(1,2,3,4,5)
write.csv(v, file="E:\\test.csv")
write.table(v, file="E:\\test.txt", append=F,
            sep="", dec=".", row.names=T,
            col.names=T)
write.xlsx(v, file="E:\\test.xlsx",
           sheetName="Sheet1", col.names=T,
```

Roll No: 1803211023 Date 20/04/2021 Page No. .....

ctical Name : To perform data import/export Practical No. 2

row.names = T, append = F)

Output →

	X
1	1
2	2
3	3
4	4
5	5

	X
1	1
2	2
3	3
4	4
5	5

	X
1	1
2	2
3	3
4	4
5	5

Roll No: 1803211023 Date 22/05/2021 Page No.

ical Name : To get input matrix from user and perform Practical No. 3

Matrix addition, subtraction, multiplication, inverse, transpose and division

Objective → Handling of multidimensional array

Algorithm →

1. Take the input from the user in 2-D array of size (3 x 3).
2. Perform addition, subtraction, multiplication, inverse, transpose and division operation in R.
3. Display the output

Code →

```
print("Enter the elements of first matrix :")
d1 ← scan(nmax=9)
print("Enter the elements of second matrix :")
d2 ← scan(nmax=9)
rname = c("row1", "row2", "row3")
cname = c("col1", "col2", "col3")
M1 ← matrix(d1, nrow=3, byrow=TRUE,
            dimnames=list(rname, cname))
print(M1)
M2 ← matrix(d2, nrow=3, byrow=TRUE,
            dimnames=list(rname, cname))
print(M2)
```

```
print("Addition of two matrix:")
print(M1+M2)
print("Subtraction of two matrix:")
print(M1-M2)
print("Multiplication of two matrix:")
print(M1%*%M2)
print("Inverse of matrix:")
print(Solve(M1))
print("Transpose of matrix:")
print(t(M1))
print("Division of two matrix:")
print(M1/M2)
```

Output ⇒

Enter the elements of first matrix:

20 30 23
14 21 28
18 19 17

Enter the elements of second matrix:

15 12 28
19 25 30
21 23 11

Addition of two matrix:

	col1	col2	col3
row1	35	42	51
row2	33	46	58
row3	39	42	28

Roll No: 1803211023 Date 22/05/2021 Page No.

al Name :

Practical No. 3

Subtraction of two matrix:

	col1	col2	col3
row1	5	18	-5
row2	-5	-4	-2
row3	-3	-4	6

Multiplication of two matrix:

	col1	col2	col3
row1	1353	1514	1713
row2	1197	1337	1330
row3	988	1087	1261

Inverse of matrix:

	col1	col2	col3
row1	-0.09	-0.03	1.87
row2	0.13	-0.03	-1.25
row3	-0.05	0.08	-3.73

Transpose of matrix:

	row1	row2	row3
col1	20	14	18
col2	30	21	19
col3	23	28	17

Division of matrix:

	col1	col2	col3
row1	1.33	2.50	0.82
row2	0.73	0.84	0.93
row3	0.85	0.82	1.54

Roll No: 1803211023 Date 13/04/2021 Page No.

Practical Name: To perform statistical operations (Mean, Median, Mode and Standard Deviation) using R Practical No. 4

Objective → To find out the statistics of the dataset.

Algorithm →

1. Apply Mean, Median and Standard Deviation formulas to find statistics of given dataset.
2. Apply mean(), median(), mode() and sd() function.
3. Find all the statistics parameter on data frame calculated
4. Display the output.

Code →

```
data1 ← read.csv("E:\\Sample.csv")
head(data1)
mode ← function(v) {
        uniqv ← unique(v)
        uniqv[which.max(tabulate(match(v, uniqv)))]
      }
print(c("Mean:", mean(data1[["x"]])))
print(c("Median:", median(data1[["x"]])))
print(c("Mode:", mode(data1[["x"]])))
print(c("Standard deviation", sd(data1[["x"]])))
```

Roll No: 1803211023 Date 13/04/2021 Page No. .....

Practical Name: To perform Statistical operations Practical No. 4

Output :

	X
1	23
2	56
3	78
4	23
5	56
6	79

Mean : 53.875

Median : 56

Mode : 23

Standard deviation : 28.3167

Roll No: 1803211023 Date 27/04/2021 Page No. .....

Practical Name : To perform data pre-processing operations Practical No. 5

(i) Handling missing data (ii) Min-Max normalization

Objective → Data Cleaning processing

Algorithm →

1. Find missing values in data using is.na() function.
2. Fill missing values using mean/median/mode
3. Normalize the data using min-max normalization.

Code →

```
v ← c(3, NA, 5, 6, 8, NA, 9, 12, 10, NA, 21, 25)
a ← v
b ← v
print(is.na(v))
a[is.na(a)] ← mean(a, na.rm=T)
b[is.na(b)] ← median(b, na.rm=T)
print("Using mean")
print(a)
print("Using median")
print(b)
d ← a
for(i in 1:length(a))
    d[i] = (d[i] - min(a)) / (max(a) - min(a))
print("Normalised data")
print(d)
```

Roll No: 1803211023 Date 27/04/2021 Page No. .....

al Name : ..... Practical No. 5

Output :-

FALSE TRUE FALSE FALSE FALSE TRUE FALSE FALSE FALSE
TRUE FALSE FALSE

Using mean

3 11 5 6 8 11 9 12 10 11 21 25

Using median

3 9 5 6 8 9 9 12 10 9 21 25

Normalised data

0.00 0.36 0.09 0.13 0.22 0.36 0.27
0.40 0.31 0.36 0.81 1.00

Roll No: 180321023 Date 05/04/2021 Page No.

Practical Name: To perform dimensionality reduction operation using PCA Practical No. 6

Objective → To get the precise and important data.

Algorithm →

1. Pre-process the dataset
2. Built pcachart function for EDA on variance.
3. Standardize the data by using scale and apply "prcomp" function.
4. Choose the principal components with highest variance
5. Visualize Data in new reduced dimension

Code →

```
data1 ← read.csv("E:\\sample.csv", header=F)
x ← as.matrix(data1)
print("Original Data")
print(x)
a ← scale(x)
c ← prcomp(a)$rotation[,1:2]
print("Required Principal Components")
print(c)
y ← x %*% c
print("Reduced Dimension Data")
print(y)
```

Roll No: 1803211023 Date 05/04/2021 Page No. .....

tical Name : ..... Practical No. 6

Output →

Original Matrix

V_1	V_2	V_3
90	60	90
90	90	30
60	60	60
60	60	90
30	30	30

Required Principal components

PC1	PC2
-0.7071	0.0000
-0.6666	0.3333
-0.2357	-0.9428

Reduced Dimension Data

PC1	PC2
-124.85	-64.85
-130.71	1.71
-96.56	-36.56
-103.63	-64.85
-48.28	-18.28

Roll No: B03211023 Date 24/05/2021 Page No.

al Name : To perform Simple linear regression with R Practical No. 7

Objective → Performing the regression technique on the given dataset

Algorithm →

(1) Load the data

(2) Make sure the data meet the assumptions:
- Normality
- Correlation
- Linearity

(3) The formula for linear regression is:

$y = a_0 + a_1 x$

y → predicted value

a_0 → the value of y when x is 0

a_1 → regression coefficient

x → independent variable

(4) Display the regression line

Code →

```
M2 ← matrix(c(95,85,85,95,80,70,70,70,65,60,70),
            nrow=5, byrow=TRUE)
mean_x = mean(M2[,1])
mean_y = mean(M2[,2])
print("Mean of x:")
print(mean_x)
print("Mean of y:")
print(mean_y)
```

```
aln =0
ald = 0
for (i in 1: length (M2 [,1])){
   aln = aln + ( (M2 [,1][i] - mean_x) * (M2 [,2][i] - mean_y))
   ald = ald + (M2 [,1][i] - mean_x)**2
}
al = aln/ald
a0 = mean_y - (a1 * mean_x)
print ("Regression line:")
print (cat ("y = ", a0, "+", a1, "x"))
```

Output :

```
Mean of x:
78
Mean of y:
77
Regression line:
y = 26.78082 + 0.6438356 x
```

Roll No: 1803211023 Date ..... Page No. .....

Practical Name: To perform K-Means clustering operation on iris data set Practical No. 8

Objective → Perform the clustering technique to categorize iris data in groups.

Algorithm →

1. Load the iris dataset
2. Specify the number of clusters (K) to be created
3. Select randomly K-objects from the dataset as the initial cluster centers or means.
4. Assign each observation to their closest centroid, based on the Euclidean distance between the object and the centroid.
5. For each of the K clusters update the cluster centroid by calculating the new mean values of all the datapoints in the cluster.
6. Iteratively minimize the total within sum of square i.e. iterate steps 3 & 4 until cluster can't be updated further.
7. Print the clusters formed.

Code →

```
library(datasets)
df ← iris
set.seed(25)
nrows ← sample(nrow(df))
df ← df[nrows, ]
```

Punch here

Roll No: 1803211023 Date ..... Page No. .....

Practical Name : K-Mean Clustering Practical No. 8

```
print("Head of Iris Dataset:")
head(df)
df <- df[, 2:4]

n <- nrow(df)
cols <- names(df)
df["Cluster"] <- NA
df["Dist"] <- 999
m <- length(cols)
n_c <- 3
s <- sample(1:n, n_c)
smdf <- df[0,]
for(i in s){
    smdf <- rbind(smdf, df[i,])
}

s <- nrow(smdf)
smdf <- smdf[, -5]
smdf <- smdf[, -5]
print("Centroids selected:")
smdf

a <- FALSE
while(a == FALSE){
    for(i in 1:n){
        for(j in 1:s){
            dist <- 0
            for(k in 1:m){
                dist <- dist + abs(df[i,k] - smdf[j,k])
            }
```

```
if (df[i,m+2] > dist){
    df[i,m+1] <- j
    df[i,m+2] <- dist
}
}
}
[illegible]
sm <- aggregate(x = df[,1:5],
                by = list(df$cluster),
                FUN = mean)
sm <- sm[,-1]
a <- identical(sm, smdf)
smdf <- sm
}
print("No Number of clusters found:")
df %>% count(Cluster)
print("Final Centroids")
print(smdf)
```

Output =>

Head of iris Dataset :

Sepal.length	Sepal.width	Petal.length	Petal.width	Species
6.1	2.6	5.6	1.4	Virginica
5.1	3.3	1.7	0.5	Setosa
5.3	3.7	1.5	0.2	Setosa
6.1	2.8	4.0	1.3	versicolor
6.5	3.0	5.8	2.2	virginica

Roll No: 1803241023 Date ..... Page No. .....

Practical Name: K-Means Clustering Practical No. 8

Centroids selected:

Sepal.length	Sepal.width	Petal.length	Petal.width
7.2	3.6	6.1	2.5
6.4	2.8	5.6	2.1
7.2	3.0	5.8	1.6

Number of clusters found:

Cluster	n
1	12
2	88
3	50

Final centroids:

Sepal.length	Sepal.width	Petal.length	Petal.width
7.291667	3.208333	6.25833	2.033333
5.388636	3.132955	2.798864	0.805681 8
6.296000	2.888000	4.846000	1.644000

Punch here

Roll No: 1803211029 Date ..... Page No. .....

Practical Name: To perform market basket analysis using Association Rules (Apriori) Practical No. 10

Objective → Perform the association rule for analysis

Algorithm →

1. Start with itemsets containing just a single item, such as {apple} & {pear}.
2. Determine the support for itemsets. Keep the itemsets that meet your minimum support threshold, and remove itemsets that do not.
3. Using the itemsets you have kept from Step 2, generate all the possible itemset configuration.
4. Repeat Step 1 & 2 until there are no newer itemsets

Code →

```
df ← read.csv(file = "C:/Users/purchase.csv", header=TRUE)
head(df)
a ← unlist(strsplit(df[1,2], split=",",""))
for (j in 2:nrow(df)) {
  s ← unlist(strsplit(df[j,2], split=",",""))
  a ← unlist(union(a,s))
}
for (i in 1:length(a)){
  df[a[i]] ← NA
}
```

Roll No: 1803211023 Date ..... Page No. .....

Practical Name : Association Rules (Apriori) Practical No. 10

```
for(i in 1: nrow(df)){
  s <- unlist(strsplit(df[1,2], split = ","))
  for(j in 1:length(a)){
    if(a[j] %in% s){
      df[i, a[j]] <- 1
    }
    else{
      df[i,a[j]] <- 0
    }
  }
sdf <- df[, 3:ncol(df)]
k <- 0
m <- length(a)
b <- a
while(k < m){
  for(j in 1:length(a)){
    s <- length(b)
    for(i in 1:s){
      if(colSums(data.frame(sdf[b[i]]*sdf[a[j]]))<2)
          ||(grepl(a[j], b[i]))){
        }
      else{
        b <- append(b, paste(b[i], a[j], sep=" "))
        sdf[paste(b[i], b[j], sep=" ")] = sdf[b[i]] *
                                          sdf[b[j]]
      }}
    k <- k+1
  }
}
```

Roll No: 1803211023 Date ..... Page No. .....

Practical Name: Association Rules (Apriori) Practical No. 10

```
~~for (i in 1:length)~~
print("Rules formed:")
  for (i in 1:length(B)){
      p <- colSums(as(B[i],))
      if (p >= 2){
          print(paste(B[i], "→", p))
      }
  }
```

Output →

TID	Items
100	1, 3, 4
200	2, 3, 5
300	1, 2, 3, 5
400	2, 5

Rules formed:

1 → 2
3 → 3
2 → 3
5 → 3
31 → 2
23 → 2
53 ← 2
52 → 3
235 → 2

Punch here