Department of Computer Science and Engineering

KPR Institute of Engineering and Technology

KPRIET LEARN BEYOND

M.E. – COMPUTER SCIENCE AND ENGINEERING

LABORATORY RECORD

P21CS204 DATA ANALYTICS LABORATORY

(Regulation 2021)

JUNE 2022

KPR INSTITUTE OF ENGINEERING AND TECHNOLOGY

(Autonomous)

COIMBATORE – 641 407

LABORATORY RECORD

Name : ……………………………………………………………….

Roll Number : ……………………………………………………………….

Subject Code & Title: ……………………………………………………………….

Department : ……………………………………………………………….

Year & Semester : ……………………………………………………………….

This is the certified record of work done by…………………………………………….

Register Number…………………………………………………………

Faculty In- Charge **Head of the Department**

Place:

Date:

He/ She has submitted the record for the End Semester Practical Examination held on ……………………

Internal Examiner **External Examiner**

INDEX

Ex. No.	Date	Experiment Name	Page No.	Marks	Signature
1					
2					
3					
4					
5					
6					
7					
8					
			Total Marks:		

Vision of the Institution

To become a premier institute of academic excellence by imparting technical, intellectual and professional skills to students for meeting the diverse needs of the industry, society, the nation and the world at large.

Mission of the Institution

1. Commitment to offer value-based education and enhancement of practical skills
2. Continuous assessment of teaching and learning process through scholarly activities
3. Enriching research and innovative activities in collaboration with industry and institute of repute
4. Ensuring the academic process to uphold culture, ethics and social responsibility

Vision of the Department

To foster the students by providing learner centric teaching environment, continuous learning, research and development to become thriving professionals and entrepreneurs to excel in the field of computer science and contribute to the society.

Mission of the Department

1. Providing value-based education and contented learning experience to the students
2. Educating the students with the state of art technologies and cultivating their proficiency in analytical and designing skills
3. Enabling the students to achieve a successful career in Computer Science and Engineering or related fields to meet the changing needs of various stakeholders
4. Guiding the students in research by nurturing their interest in continuous learning towards serving the society and the country.

Program Educational Objectives (PEOs)

- Graduates of mechanical engineering will have a successful professional career in their respective and/or related field of engineering to meet the changing needs of various stakeholders.
- Graduates with passion towards research and professional development will involve in continuing education by pursuing advanced professional studies and/or participate in professional activities.
- Graduates will act as a responsible leader or team member with good communication, environment safety, behavioral attitude and professional ethics for the economic development of the organization as well as the society

Program Outcomes (POs)

Engineering Graduates will be able to:

1. **Engineering knowledge**: Apply the knowledge of mathematics, science, engineering Fundamentals and an engineering specialization to the solution of complex engineering problems.

2. **Problem analysis**: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

3. **Design/development of solutions**: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

4. **Conduct investigations of complex problems**: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

5. **Modern tool usage**: Create, select, and apply appropriate techniques, resources, and modern

engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

9. **Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

11. **Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

12. **Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change

Program Specific Outcomes (PSOs)

A graduate of Mechanical Engineering will be able to

1. **PSO 1**: Apply mechanical engineering principles to design, develop and implement advanced machine/mechanical systems or process for better performance and less human effort.

2. **PSO 2**: Ensure quality by applying quality tools, maintenance principles and managerial skills to comprehend the mechanical engineering processes, products and services.

RUBRICS FOR ASSESSMENT

Criteria		Excellent (4 Marks)	Good (3 Marks)	Adequate (2 Marks)	Inadequate (1 Mark)
A. Preparation & Observation B	**Criterion #1** Ability to setup and conduct experiments	Able to develop contingency or alternative plans and anticipate problems during experiment.	Able to develop contingency or alternative plans.	Able to use theoretical framework, measurement techniques, testing apparatus or model.	Unable to identify theoretical framework, measurement techniques, testing apparatus or model.
	Criterion #2 Ability to take measurements / readings and present data	Able to formulate, controls and evaluate alternatives of the experiment. Able to evaluate data and relate to engineering phenomena for decision-making.	Able to evaluate data and relate to engineering phenomena for decision-making.	Able to apply constraint and assumption into the experimental design. Able to conduct experiment correctly and collect data.	Unable to discuss experimental processes and protocols
B. Results & Interpretation	**Criterion #3** Ability to analyze the data theoretically and logically to conclude experimental results	Able to combine /organize more than one set of data, interpret data and make meaningful conclusion.	Able to evaluate or compare data and make meaningful conclusion	Able to select and use and apply appropriate techniques or methods to analyse the data.	Unable to select and describe the techniques or methods of analyzing the data.
	Criterion #4 Ability to interpret and discuss any discrepancies between theoretical and experimental results	Able to verify and/or validate several sets of data and relates to engineering phenomena for decision making.	Able to verify and/or validate data and relate to engineering phenomena for decision making.	Able to identify and verify how results relate/differ from theory or previous results	Unable to identify how results relate/differ from theory or previous results.
C. Viva Voce	**Criterion #5** Demonstrate the ability to respond effectively to questions	Able to listen carefully and respond to questions appropriately; is able to explain and interpret results to the teacher	Able to listen carefully and respond to questions appropriately	Misunderstand the questions and does not respond appropriately to the teacher, or has some trouble in answering questions	Unable to listen carefully to questions and does not provide an appropriate answer, or is unable to answer questions

LIST OF EXPERIMENTS

S.NO	NAME OF THE EXPERIMENTS	PAGE NO
01	Install, Configure and run Hadoop and HDFS.	10
02	Implement word count / frequency programs using MapReduce.	14
03	Implement an MR program that processes a weather dataset in R.	19
04	Implement Linear and Logistic Regression.	24
05	Implement SVM / Decision Tree classification techniques.	27
06	Implement Clustering techniques.	29
07	Visualize data using any plotting framework.	32
08	Implement an application that stores big data in Hbase / MongoDB / Pig using Hadoop / R.	36

Prepared by **Approved by**

Dr.N.SARANYA, AP (SI.G) **HoD/ CSE**

CYCLE OF EXPERIMENTS

CYCLE – I

S.NO	NAME OF THE EXPERIMENTS
01	Install, Configure and run Hadoop and HDFS.
02	Implement word count / frequency programs using MapReduce.
03	Implement an MR program that processes a weather dataset in R.
04	Implement Linear and Logistic Regression.

CYCLE – II

S.NO	NAME OF THE EXPERIMENTS
05	Implement SVM / Decision Tree classification techniques.
06	Implement Clustering techniques.
07	Visualize data using any plotting framework.
08	Implement an application that stores big data in Hbase / MongoDB / Pig using Hadoop / R.

EX.NO: 01	INSTALL, CONFIGURE AND RUN HADOOP AND HDFS
DATE:	

AIM:

To install, Configure and Run and HDFS.

PROCEDURE:

Apparatus: -

Hardware: N/A

Software : Sun's JDK 1.6.0

Theory: -

Prerequisites

Hadoop is written in Java, so you will need to have Java installed on your machine, version 6 or later. Sun's JDK is the one most widely used with Hadoop, although others have been reported to work.

Hadoop runs on Unix and on Windows. Linux is the only supported production platform, but other flavors of Unix (including Mac OS X) can be used to run Hadoop for development. Windows is only supported as a development platform, and additionally requires Cygwin to run. During the Cygwin installation process, you should include the openness package if you plan to run Hadoop in pseudo-distributed mode (see following explanation).

Procedure:

Installation

- Start by deciding which user you'd like to run Hadoop as. For trying out Hadoop or developing Hadoop programs, it is simplest to run Hadoop on a single machine using your own user account.
- Download a stable release, which is packaged as a gzipped tar file, from the Apache Hadoop releases page (http://hadoop.apache.org/core/releases.html) and unpack it somewhere on your file system:

```
% tar xzf hadoop-x.y.z.tar.gz
```

- Before you can run Hadoop, you need to tell it where Java is located on your system. If you have the JAVA_HOME environment variable set to point to a suitable Java installation, that will be used, and you don't have to configure anything further. Otherwise, you can set the Java installation
- Hadoop uses by editing conf/hadoop-env.sh, and specifying the JAVA_HOME variable.

  For example, on my Mac I changed the line to read:

```
export JAVA_HOME=/System/Library/Frameworks/JavaVM.framework/Versions/1.6.0/Home
```

to point to version 1.6.0 of Java. On Ubuntu, the equivalent line is:

```
export JAVA_HOME=/usr/lib/jvm/java-6-sun
```

- It's very convenient to create an environment variable that points to the Hadoop installation directory (HADOOP_INSTALL, say) and to put the Hadoop binary directory on your command-line path. For example:

```
% export HADOOP_INSTALL=/home/tom/hadoop-x.y.z
% export PATH=$PATH:$HADOOP_INSTALL/bin
```

- Check that Hadoop runs by typing:

```
% hadoop version
Hadoop 0.20.0
Subversion    https://svn.apache.org/repos/asf/hadoop/core/branches/branch-0.20
-r 763504
Compiled by ndaley on Thu Apr 9 05:18:40 UTC 2009
```

Configuration

- Each component in Hadoop is configured using an XML file. Core properties go in coresite.xml, HDFS properties go in hdfs-site.xml, and MapReduce properties go in mapred site.xml. These files are all located in the conf subdirectory.

Note

In earlier versions of Hadoop, there was a single site configuration file for the Core, HDFS, and MapReduce components, called hadoop-site.xml. From release 0.20.0 onward this file has been split into three: one for each component. The property names have not changed, just the configuration file they have to go in. You can see the default settings for all the properties that are governed by these configuration files by looking in the docs directory of your Hadoop installation for HTML files called core-default.html, hdfs-default.html, and mapred- default.html.

Hadoop can be run in one of three modes:

- **Standalone (or local) mode**

There are no daemons running and everything runs in a single JVM. Standalone mode is suitable for running MapReduce programs during development, since it is easy to test and debug them.

- **Pseudo-distributed mode**

The Hadoop daemons run on the local machine, thus simulating a cluster on a small scale.

➢ **Fully distributed mode**

The Hadoop daemons run on a cluster of machines.

- To run Hadoop in a particular mode, you need to do two things: set the appropriate properties, and start the Hadoop daemons. Table A.1, “Key configuration properties for different modes” shows the minimal set of properties to configure each mode. In standalone mode, the local filesystem and the local MapReduce job runner are used, while in the distributed modes the HDFS and MapReduce daemons are started.

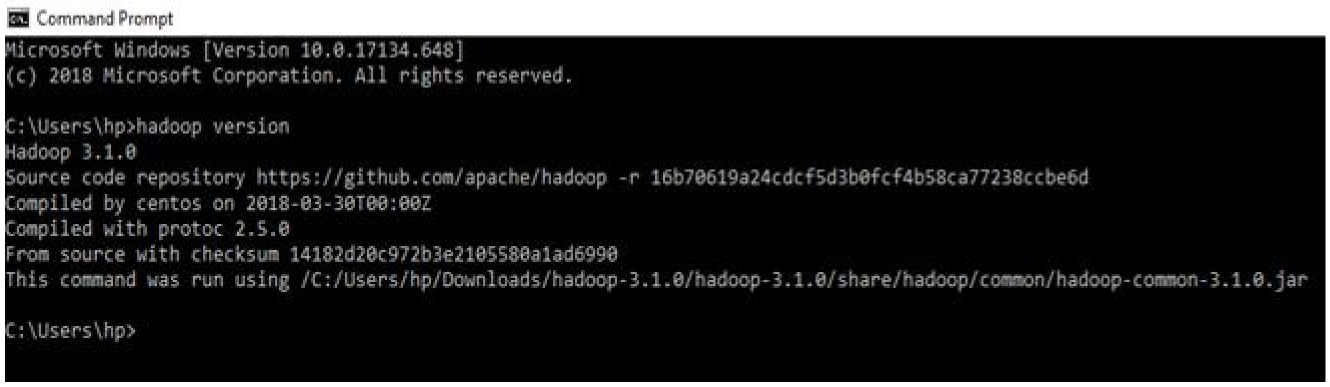
Table A.1. Key configuration properties for different modes

Component	**Property**	**Standalone**	**Pseudo-distributed**	**Fully distributed**
Core	fs.default.name	file:/// (default)	hdfs://localhost/	hdfs://*namenode*/
HDFS	dfs.replication	N/A	1	3 (default)
MapReduce	mapred.job.tracker	local (default)	localhost:8021	*jobtracker*:8021

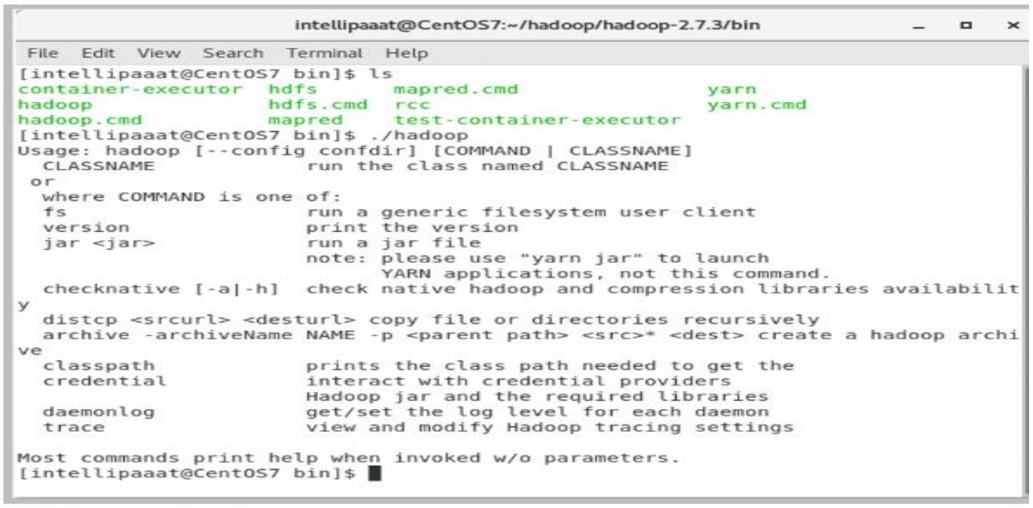
You can read more about configuration in the section called “Hadoop Configuration”.

Standalone Mode

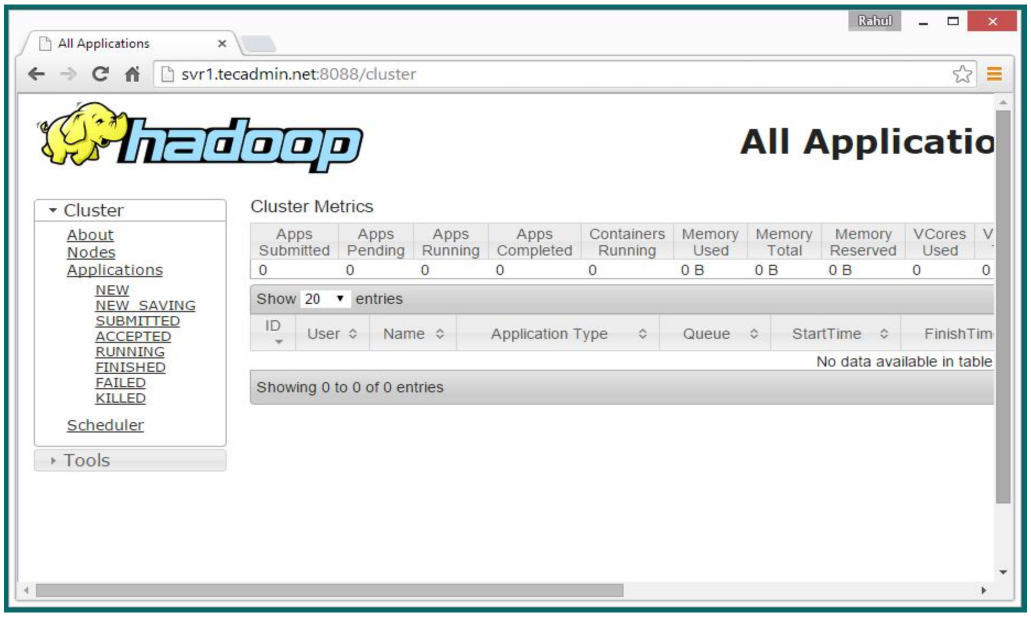
In standalone mode, there is no further action to take, since the default properties are set for standalone mode, and there are no daemons to run



```
Microsoft Windows [Version 10.0.17134.648]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\hp>hadoop version
Hadoop 3.1.0
Source code repository https://github.com/apache/hadoop -r 16b70619a24cdcf5d3b0fcf4b58ca77238ccbe6d
Compiled by centos on 2018-03-30T00:00Z
Compiled with protoc 2.5.0
From source with checksum 14182d20c972b3e2105580a1ad6990
This command was run using /C:/Users/hp/Downloads/hadoop-3.1.0/hadoop-3.1.0/share/hadoop/common/hadoop-common-3.1.0.jar

C:\Users\hp>
```


intellipaaat@CentOS7:~/hadoop/hadoop-2.7.3/bin
File Edit View Search Terminal Help

```
[intellipaaat@CentOS7 bin]$ ls
container-executor  hdfs      mapred.cmd               yarn
hadoop              hdfs.cmd  rcc                      yarn.cmd
hadoop.cmd          mapred    test-container-executor
[intellipaaat@CentOS7 bin]$ ./hadoop
Usage: hadoop [--config confdir] [COMMAND | CLASSNAME]
  CLASSNAME            run the class named CLASSNAME
 or
  where COMMAND is one of:
  fs                   run a generic filesystem user client
  version              print the version
  jar <jar>            run a jar file
                       note: please use "yarn jar" to launch
                             YARN applications, not this command.
  checknative [-a|-h]  check native hadoop and compression libraries availabilit
y
  distcp <srcurl> <desturl> copy file or directories recursively
  archive -archiveName NAME -p <parent path> <src>* <dest> create a hadoop archi
ve
  classpath            prints the class path needed to get the
  credential           interact with credential providers
                       Hadoop jar and the required libraries
  daemonlog            get/set the log level for each daemon
  trace                view and modify Hadoop tracing settings

Most commands print help when invoked w/o parameters.
[intellipaaat@CentOS7 bin]$
```



Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	VCores Used	V
0	0	0	0	0	0 B	0 B	0 B	0	0

Show 20 entries

ID	User	Name	Application Type	Queue	StartTime	FinishTim
No data available in table						

Showing 0 to 0 of 0 entries

RESULT :

Thus, the Hadoop is installed and configured.

EX.NO: 02	IMPLEMENT WORD COUNT USING MAPREDUCE
DATE:	

AIM :

To implement word count program using MapReduce.

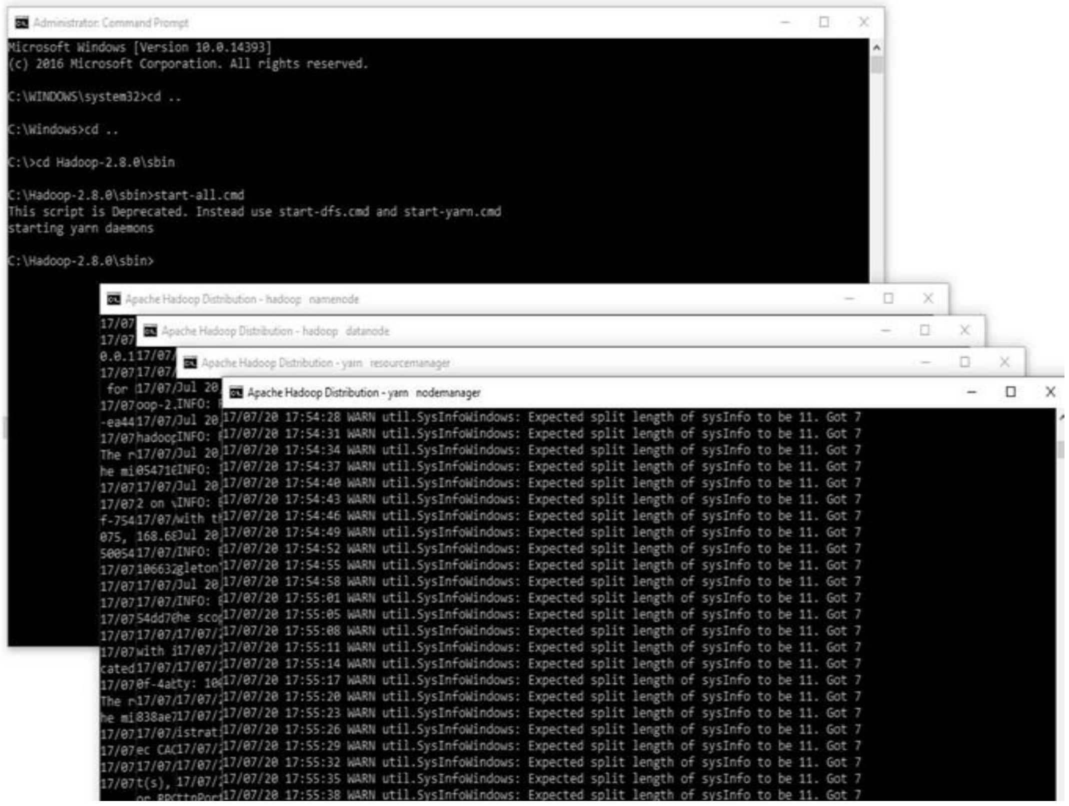
PROCEDURE :

1. Download MapReduceClient.jar from github.
2. Download Input_file.txt
3. Place both files in “C:/”

Hadoop Operation:

1. Open cmd in Administrative mode and move to “C:/Hadoop-2.8.0/sbin” and start cluster

   Start-all.cmd


Administrator: Command Prompt
Apache Hadoop Distribution - hadoop namenode
Apache Hadoop Distribution - hadoop datanode
Apache Hadoop Distribution - yarn resourcemanager
Apache Hadoop Distribution - yarn nodemanager

```
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>cd ..

C:\Windows>cd ..

C:\>cd Hadoop-2.8.0\sbin

C:\Hadoop-2.8.0\sbin>start-all.cmd
This script is Deprecated. Instead use start-dfs.cmd and start-yarn.cmd
starting yarn daemons

C:\Hadoop-2.8.0\sbin>
```

```
17/07/20 17:54:28 WARN util.SysInfoWindows: Expected split length of sysInfo to be 11. Got 7
17/07/20 17:54:31 WARN util.SysInfoWindows: Expected split length of sysInfo to be 11. Got 7
17/07/20 17:54:34 WARN util.SysInfoWindows: Expected split length of sysInfo to be 11. Got 7
17/07/20 17:54:37 WARN util.SysInfoWindows: Expected split length of sysInfo to be 11. Got 7
17/07/20 17:54:40 WARN util.SysInfoWindows: Expected split length of sysInfo to be 11. Got 7
17/07/20 17:54:43 WARN util.SysInfoWindows: Expected split length of sysInfo to be 11. Got 7
17/07/20 17:54:46 WARN util.SysInfoWindows: Expected split length of sysInfo to be 11. Got 7
17/07/20 17:54:49 WARN util.SysInfoWindows: Expected split length of sysInfo to be 11. Got 7
17/07/20 17:54:52 WARN util.SysInfoWindows: Expected split length of sysInfo to be 11. Got 7
17/07/20 17:54:55 WARN util.SysInfoWindows: Expected split length of sysInfo to be 11. Got 7
17/07/20 17:54:58 WARN util.SysInfoWindows: Expected split length of sysInfo to be 11. Got 7
17/07/20 17:55:01 WARN util.SysInfoWindows: Expected split length of sysInfo to be 11. Got 7
17/07/20 17:55:05 WARN util.SysInfoWindows: Expected split length of sysInfo to be 11. Got 7
17/07/20 17:55:08 WARN util.SysInfoWindows: Expected split length of sysInfo to be 11. Got 7
17/07/20 17:55:11 WARN util.SysInfoWindows: Expected split length of sysInfo to be 11. Got 7
17/07/20 17:55:14 WARN util.SysInfoWindows: Expected split length of sysInfo to be 11. Got 7
17/07/20 17:55:17 WARN util.SysInfoWindows: Expected split length of sysInfo to be 11. Got 7
17/07/20 17:55:20 WARN util.SysInfoWindows: Expected split length of sysInfo to be 11. Got 7
17/07/20 17:55:23 WARN util.SysInfoWindows: Expected split length of sysInfo to be 11. Got 7
17/07/20 17:55:26 WARN util.SysInfoWindows: Expected split length of sysInfo to be 11. Got 7
17/07/20 17:55:29 WARN util.SysInfoWindows: Expected split length of sysInfo to be 11. Got 7
17/07/20 17:55:32 WARN util.SysInfoWindows: Expected split length of sysInfo to be 11. Got 7
17/07/20 17:55:35 WARN util.SysInfoWindows: Expected split length of sysInfo to be 11. Got 7
17/07/20 17:55:38 WARN util.SysInfoWindows: Expected split length of sysInfo to be 11. Got 7
```

2. Create an input directory in HDFS.

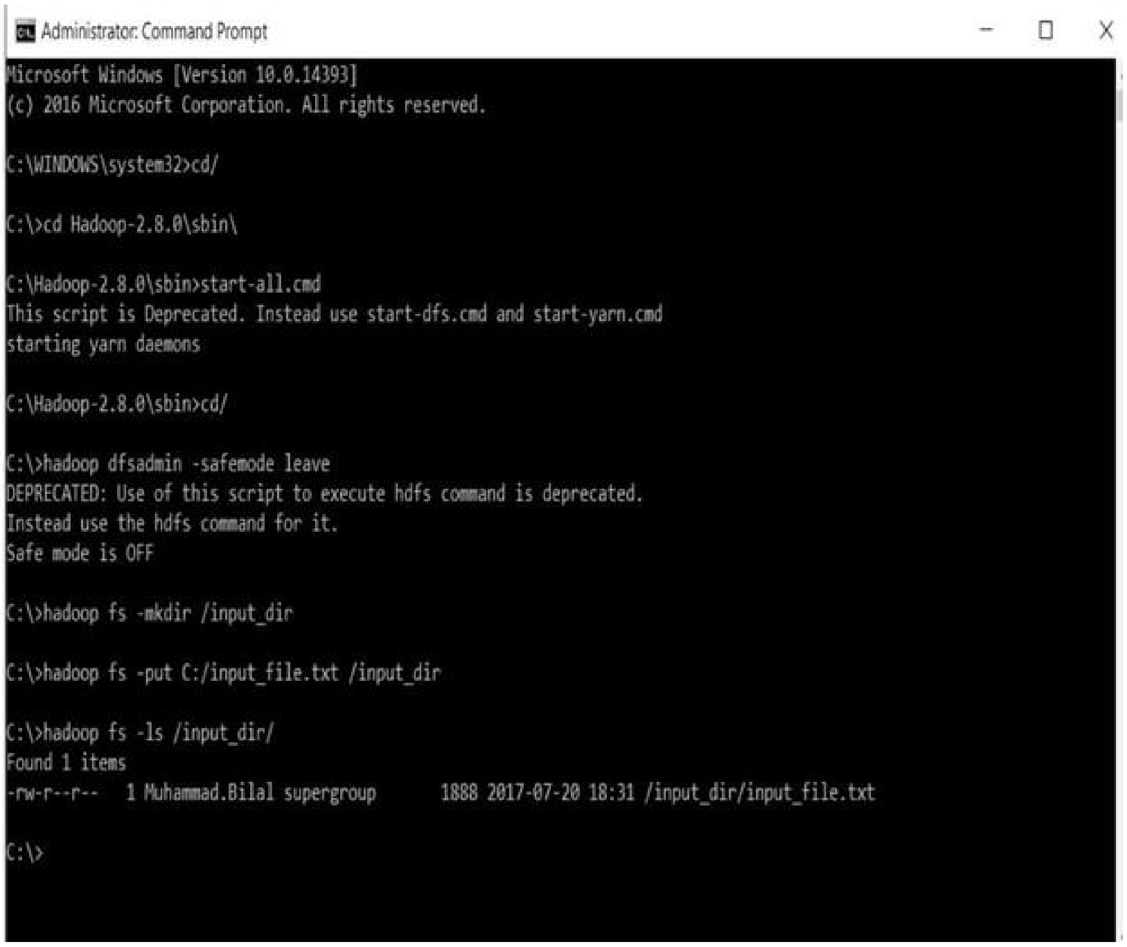
hadoop fs-mkdir/input_dir

3. Copy the input text file named input_file.txt in the directory (input_dir) of HDFS.

hadoop fs -put C:/input_file.txt/input_dir

4. Verify input_file.txt available in HDFS input directory(input_dir).

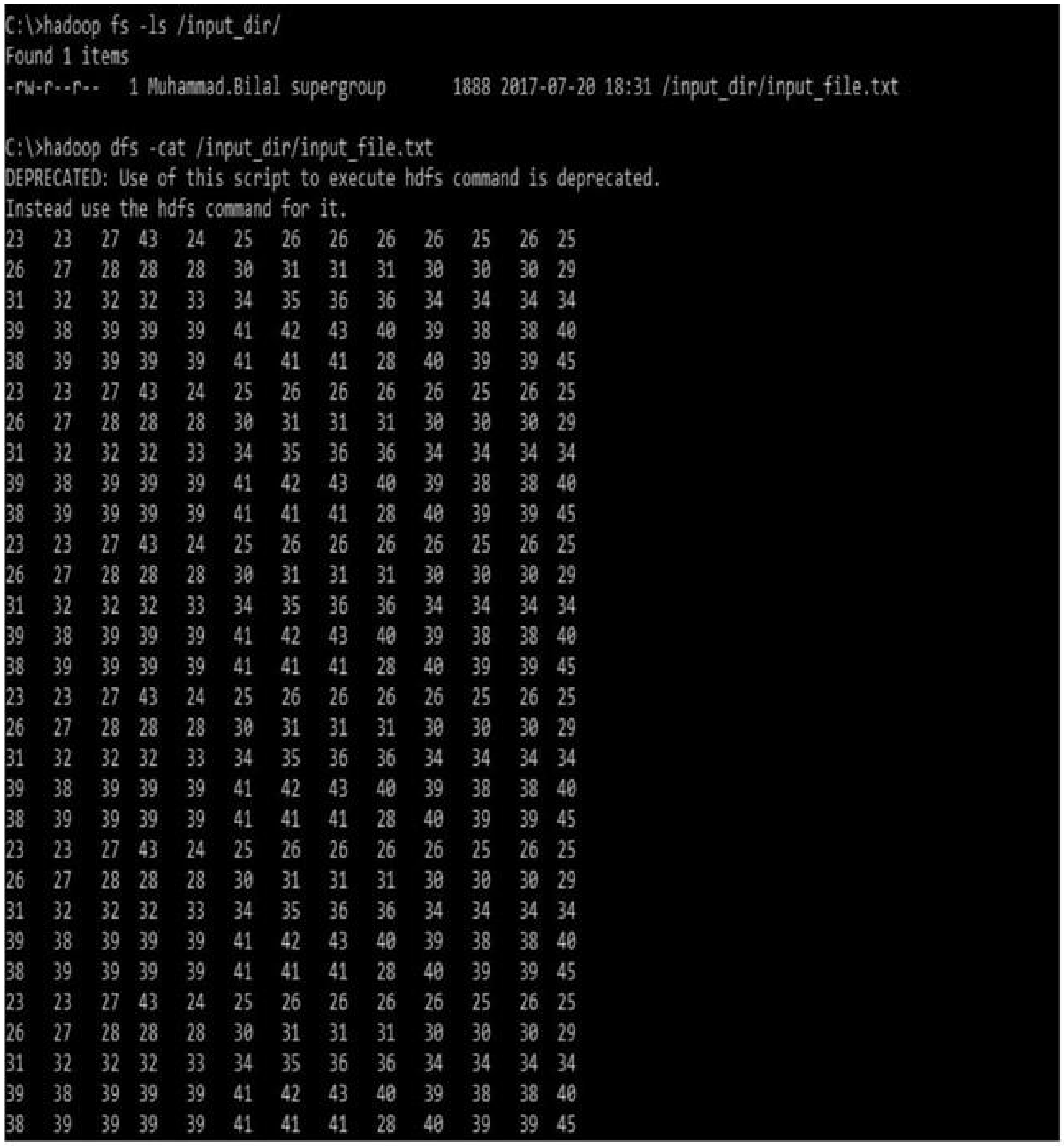
hadoop fs -ls /input_dir/



```
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>cd/

C:\>cd Hadoop-2.8.0\sbin\

C:\Hadoop-2.8.0\sbin>start-all.cmd
This script is Deprecated. Instead use start-dfs.cmd and start-yarn.cmd
starting yarn daemons

C:\Hadoop-2.8.0\sbin>cd/

C:\>hadoop dfsadmin -safemode leave
DEPRECATED: Use of this script to execute hdfs command is deprecated.
Instead use the hdfs command for it.
Safe mode is OFF

C:\>hadoop fs -mkdir /input_dir

C:\>hadoop fs -put C:/input_file.txt /input_dir

C:\>hadoop fs -ls /input_dir/
Found 1 items
-rw-r--r--   1 Muhammad.Bilal supergroup       1888 2017-07-20 18:31 /input_dir/input_file.txt

C:\>
```

5. Verify content of the copied file.

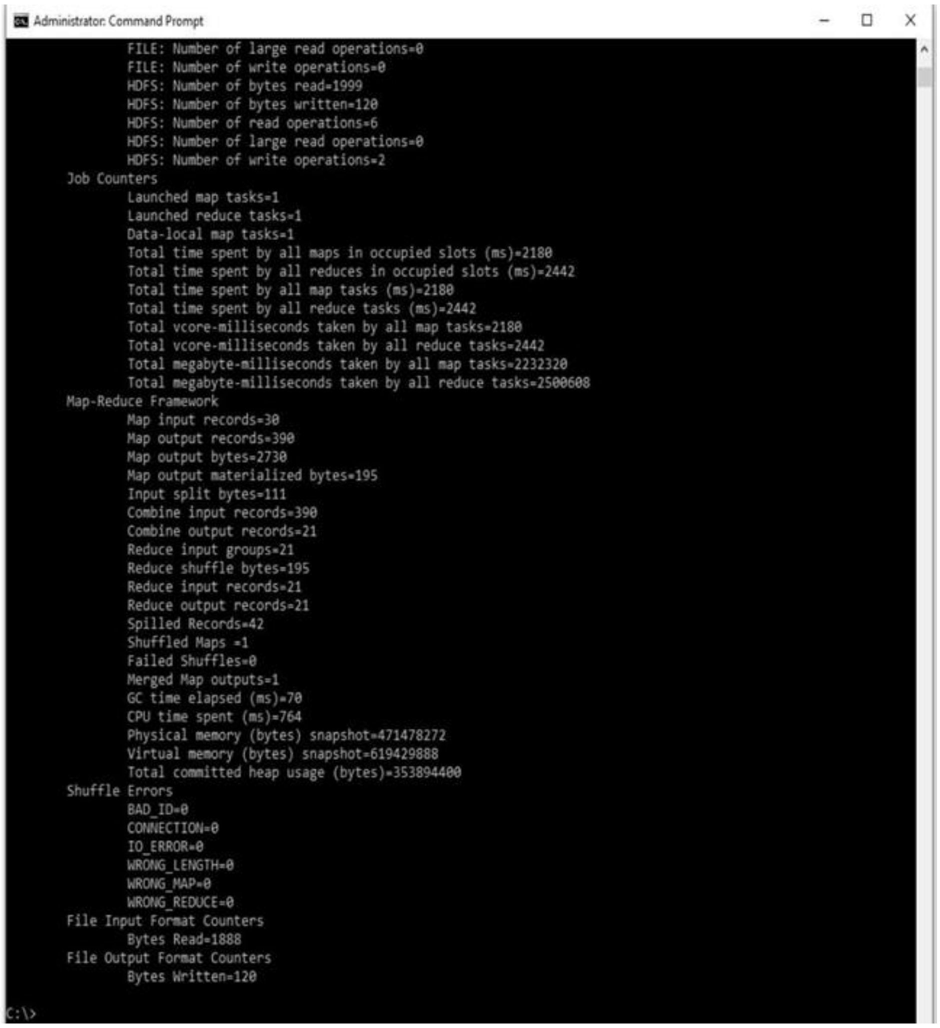
hadoop dfs -cat /input_dir/input_file.txt



```
C:\>hadoop fs -ls /input_dir/
Found 1 items
-rw-r--r--   1 Muhammad.Bilal supergroup       1888 2017-07-20 18:31 /input_dir/input_file.txt

C:\>hadoop dfs -cat /input_dir/input_file.txt
DEPRECATED: Use of this script to execute hdfs command is deprecated.
Instead use the hdfs command for it.
23   23   27  43   24   25   26   26   26   26   25   26  25
26   27   28  28   28   30   31   31   31   30   30   30  29
31   32   32  32   33   34   35   36   36   34   34   34  34
39   38   39  39   39   41   42   43   40   39   38   38  40
38   39   39  39   39   41   41   41   28   40   39   39  45
23   23   27  43   24   25   26   26   26   26   25   26  25
26   27   28  28   28   30   31   31   31   30   30   30  29
31   32   32  32   33   34   35   36   36   34   34   34  34
39   38   39  39   39   41   42   43   40   39   38   38  40
38   39   39  39   39   41   41   41   28   40   39   39  45
23   23   27  43   24   25   26   26   26   26   25   26  25
26   27   28  28   28   30   31   31   31   30   30   30  29
31   32   32  32   33   34   35   36   36   34   34   34  34
39   38   39  39   39   41   42   43   40   39   38   38  40
38   39   39  39   39   41   41   41   28   40   39   39  45
23   23   27  43   24   25   26   26   26   26   25   26  25
26   27   28  28   28   30   31   31   31   30   30   30  29
31   32   32  32   33   34   35   36   36   34   34   34  34
39   38   39  39   39   41   42   43   40   39   38   38  40
38   39   39  39   39   41   41   41   28   40   39   39  45
23   23   27  43   24   25   26   26   26   26   25   26  25
26   27   28  28   28   30   31   31   31   30   30   30  29
31   32   32  32   33   34   35   36   36   34   34   34  34
39   38   39  39   39   41   42   43   40   39   38   38  40
38   39   39  39   39   41   41   41   28   40   39   39  45
23   23   27  43   24   25   26   26   26   26   25   26  25
26   27   28  28   28   30   31   31   31   30   30   30  29
31   32   32  32   33   34   35   36   36   34   34   34  34
39   38   39  39   39   41   42   43   40   39   38   38  40
38   39   39  39   39   41   41   41   28   40   39   39  45
```

6. Run MapReduceClient.jar and also provide input and out directories.

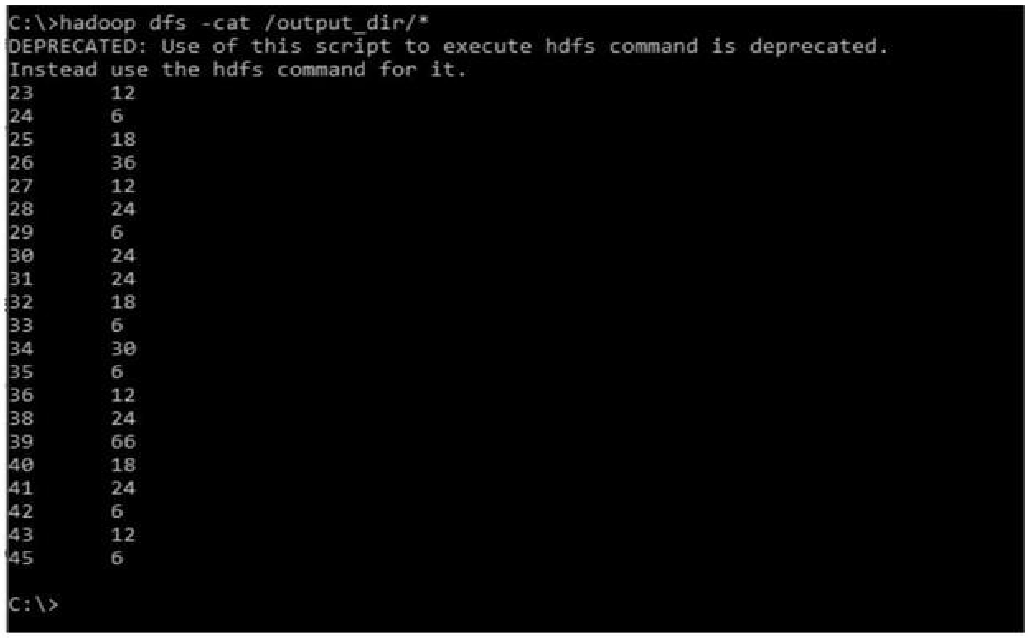
hadoop jar C:/MapReduceClient.jar wordcount /input_dir /output_dir



```
Administrator: Command Prompt
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=1999
                HDFS: Number of bytes written=120
                HDFS: Number of read operations=6
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
        Job Counters
                Launched map tasks=1
                Launched reduce tasks=1
                Data-local map tasks=1
                Total time spent by all maps in occupied slots (ms)=2180
                Total time spent by all reduces in occupied slots (ms)=2442
                Total time spent by all map tasks (ms)=2180
                Total time spent by all reduce tasks (ms)=2442
                Total vcore-milliseconds taken by all map tasks=2180
                Total vcore-milliseconds taken by all reduce tasks=2442
                Total megabyte-milliseconds taken by all map tasks=2232320
                Total megabyte-milliseconds taken by all reduce tasks=2500608
        Map-Reduce Framework
                Map input records=30
                Map output records=390
                Map output bytes=2730
                Map output materialized bytes=195
                Input split bytes=111
                Combine input records=390
                Combine output records=21
                Reduce input groups=21
                Reduce shuffle bytes=195
                Reduce input records=21
                Reduce output records=21
                Spilled Records=42
                Shuffled Maps =1
                Failed Shuffles=0
                Merged Map outputs=1
                GC time elapsed (ms)=70
                CPU time spent (ms)=764
                Physical memory (bytes) snapshot=471478272
                Virtual memory (bytes) snapshot=619429888
                Total committed heap usage (bytes)=353894400
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=1888
        File Output Format Counters
                Bytes Written=120

C:\>
```

7. Verify the content for generated output file.

hadoop dfs -cat /output_dir/*



```
C:\>hadoop dfs -cat /output_dir/*
DEPRECATED: Use of this script to execute hdfs command is deprecated.
Instead use the hdfs command for it.
23      12
24      6
25      18
26      36
27      12
28      24
29      6
30      24
31      24
32      18
33      6
34      30
35      6
36      12
38      24
39      66
40      18
41      24
42      6
43      12
45      6

C:\>
```

Some Other useful commands:

8. To leave Safe mode.

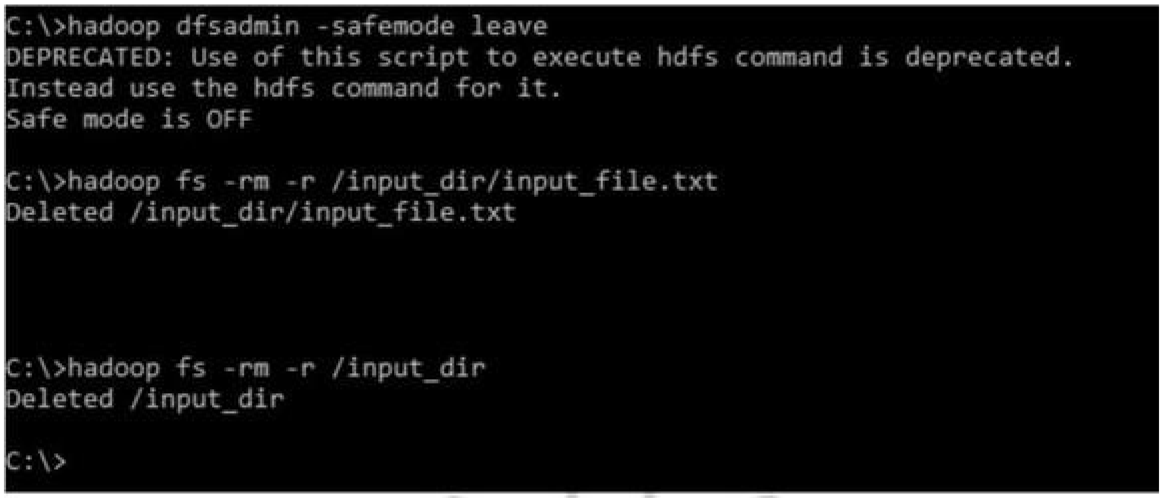
    hadoop dfsadmin -safemode leave

9. To delete file from HDFS directory.

    hadoop fs -rm -r /input_dir/input_file.txt

10. To delete directory from HDFS directory.

    hadoop fs -rm -r /input_dir



```
C:\>hadoop dfsadmin -safemode leave
DEPRECATED: Use of this script to execute hdfs command is deprecated.
Instead use the hdfs command for it.
Safe mode is OFF

C:\>hadoop fs -rm -r /input_dir/input_file.txt
Deleted /input_dir/input_file.txt



C:\>hadoop fs -rm -r /input_dir
Deleted /input_dir

C:\>
```

__**RESULT :**__

Thus, the word count program is implemented in Hadoop using MapReduce.

EX.NO: 03	**IMPLEMENT AN MR PROGRAM THAT PROCESSES A WEATHER DATASET IN R**
DATE:	

AIM :

To write a MapReduce Program to process a Weather Dataset using R.

PROGRAM :

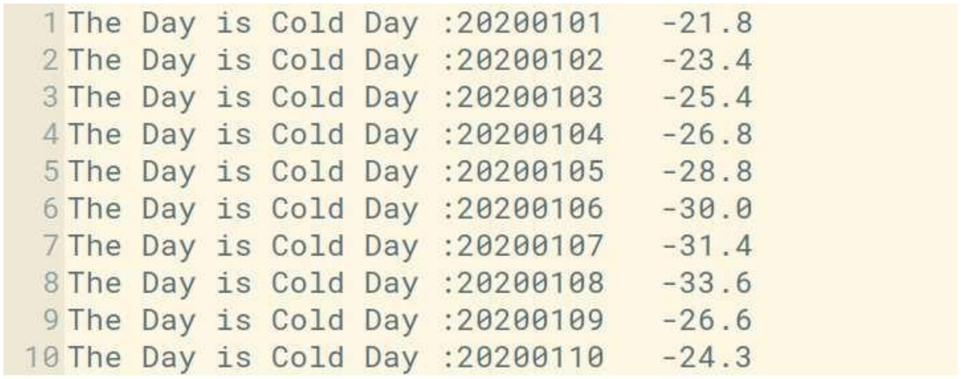
```
import java.io.IOException;
import java.util.Iterator;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.conf.Configuration;
public class MyMaxMin {
        public static class MaxTemperatureMapper extends
                        Mapper<LongWritable, Text, Text, Text> {
        // the data in our data set with
        // this value is inconsistent data
        public static final int MISSING = 9999;

@Override
        public void map(LongWritable arg0, Text Value, Context context)
                        throws IOException, InterruptedException {
```

```
        String line = Value.toString();
            // Check for the empty line
            if (!(line.length() == 0)) {

                // from character 6 to 14 we have
                // the date in our dataset
                String date = line.substring(6, 14);

                // similarly we have taken the maximum
                // temperature from 39 to 45 characters
                float temp_Max = Float.parseFloat(line.substring(39, 45).trim());

                // similarly we have taken the minimum
                // temperature from 47 to 53 characters

                float temp_Min = Float.parseFloat(line.substring(47, 53).trim());

                // if maximum temperature is
                // greater than 30, it is a hot day
                if (temp_Max > 30.0) {

                      // Hot day
                      context.write(new Text("The Day is Hot Day :" + date),
                                                            new
Text(String.valueOf(temp_Max)));
                       }
                // if the minimum temperature is
                // less than 15, it is a cold day
                if (temp_Min < 15) {

                     // Cold day
                     context.write(new Text("The Day is Cold Day :" + date),
                             new Text(String.valueOf(temp_Min)));
                   }
              }
            }
}

public static class MaxTemperatureReducer extends
            Reducer<Text, Text, Text, Text> {

      public void reduce(Text Key, Iterator<Text> Values, Context context)
                        throws IOException, InterruptedException {

            // putting all the values in
            // temperature variable of type String
            String temperature = Values.next().toString();
            context.write(Key, new Text(temperature));
      }

}
public static void main(String[] args) throws Exception {

       // reads the default configuration of the
       // cluster from the configuration XML files
       Configuration conf = new Configuration();

       // Initializing the job with the
       // default configuration of the cluster
       Job = new Job(conf, "weather example");

       // Assigning the driver class name
       job.setJarByClass(MyMaxMin.class);

       // Key type coming out of mapper
       job.setMapOutputKeyClass(Text.class);

       // value type coming out of mapper
       job.setMapOutputValueClass(Text.class);
      // Defining the mapper class name
```

```
  job.setMapperClass(MaxTemperatureMapper.class);

  // Defining the reducer class name
  job.setReducerClass(MaxTemperatureReducer.class);

  // Defining input Format class which is
// responsible to parse the dataset
// into a key value pair
job.setInputFormatClass(TextInputFormat.class);

// Defining output Format class which is
// responsible to parse the dataset
// into a key value pair
job.setOutputFormatClass(TextOutputFormat.class);

// setting the second argument
// as a path in a path variable
Path OutputPath = new Path(args[1]);

// Configuring the input path
// from the filesystem into the job
FileInputFormat.addInputPath(job, new Path(args[0]));

// Configuring the output path from
// the filesystem into the job
FileOutputFormat.setOutputPath(job, new Path(args[1]));

// deleting the context path automatically
// from hdfs so that we don't have
// to delete it explicitly
OutputPath.getFileSystem(conf).delete(OutputPath);

// exiting the job only if the
// flag value becomes false
System.exit(job.waitForCompletion(true) ? 0 : 1);
```

```
    }
}
```

OUTPUT :



```
The Day is Cold Day :20200101   -21.8
The Day is Cold Day :20200102   -23.4
The Day is Cold Day :20200103   -25.4
The Day is Cold Day :20200104   -26.8
The Day is Cold Day :20200105   -28.8
The Day is Cold Day :20200106   -30.0
The Day is Cold Day :20200107   -31.4
The Day is Cold Day :20200108   -33.6
The Day is Cold Day :20200109   -26.6
The Day is Cold Day :20200110   -24.3
```

RESULT:

Thus the MapReduce Program to process a Weather Dataset using R had been implemented.

EX.NO: 04	IMPLEMENT LINEAR AND LOGISTIC REGRESSION
DATE:	

AIM :

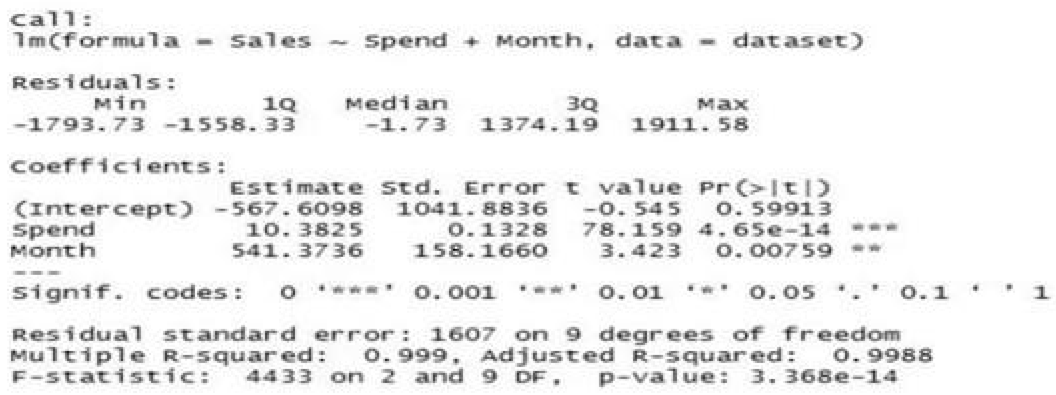
To implement Linear and Logistic Regression using R.

PROGRAM :

SIMPLE LINEAR REGRESSION:

```
dataset=read.csv(“data-marketing-budget.csv”,header=T)
colClasses=c(“numeric”,”numeric”,”numeric”))
head(dataset,5)
simple.fit=lm(Sales-Spend,data=dataset)
summary(simple.fit)
```

OUTPUT:

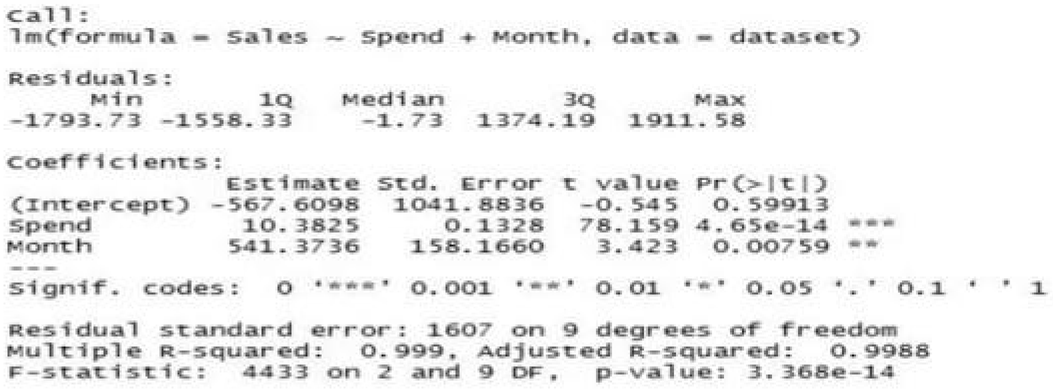


```
Call:
lm(formula = Sales ~ Spend + Month, data = dataset)

Residuals:
     Min       1Q   Median       3Q      Max
-1793.73 -1558.33    -1.73  1374.19  1911.58

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -567.6098  1041.8836  -0.545  0.59913
Spend         10.3825     0.1328  78.159 4.65e-14 ***
Month        541.3736   158.1660   3.423  0.00759 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1607 on 9 degrees of freedom
Multiple R-squared:  0.999, Adjusted R-squared:  0.9988
F-statistic:  4433 on 2 and 9 DF,  p-value: 3.368e-14
```

MULTIPLE LINEAR REGRESSION:

```
multi.fit= lm(Sales-Spend+Month,data=dataset)
summary(multi.fit)
```

OUTPUT:

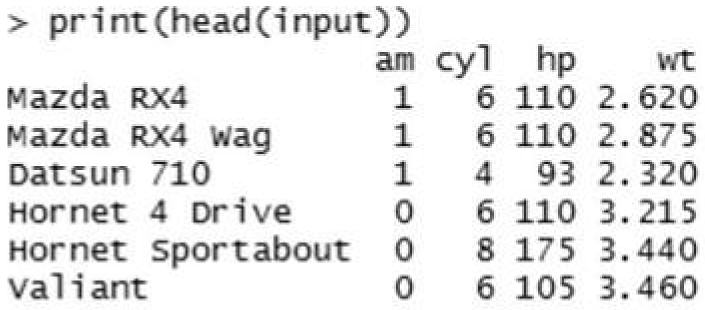


```
Call:
lm(formula = Sales ~ Spend + Month, data = dataset)

Residuals:
     Min       1Q   Median       3Q      Max
-1793.73 -1558.33    -1.73  1374.19  1911.58

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -567.6098  1041.8836  -0.545  0.59913
Spend         10.3825     0.1328  78.159 4.65e-14 ***
Month        541.3736   158.1660   3.423  0.00759 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1607 on 9 degrees of freedom
Multiple R-squared:  0.999, Adjusted R-squared:  0.9988
F-statistic:  4433 on 2 and 9 DF,  p-value: 3.368e-14
```

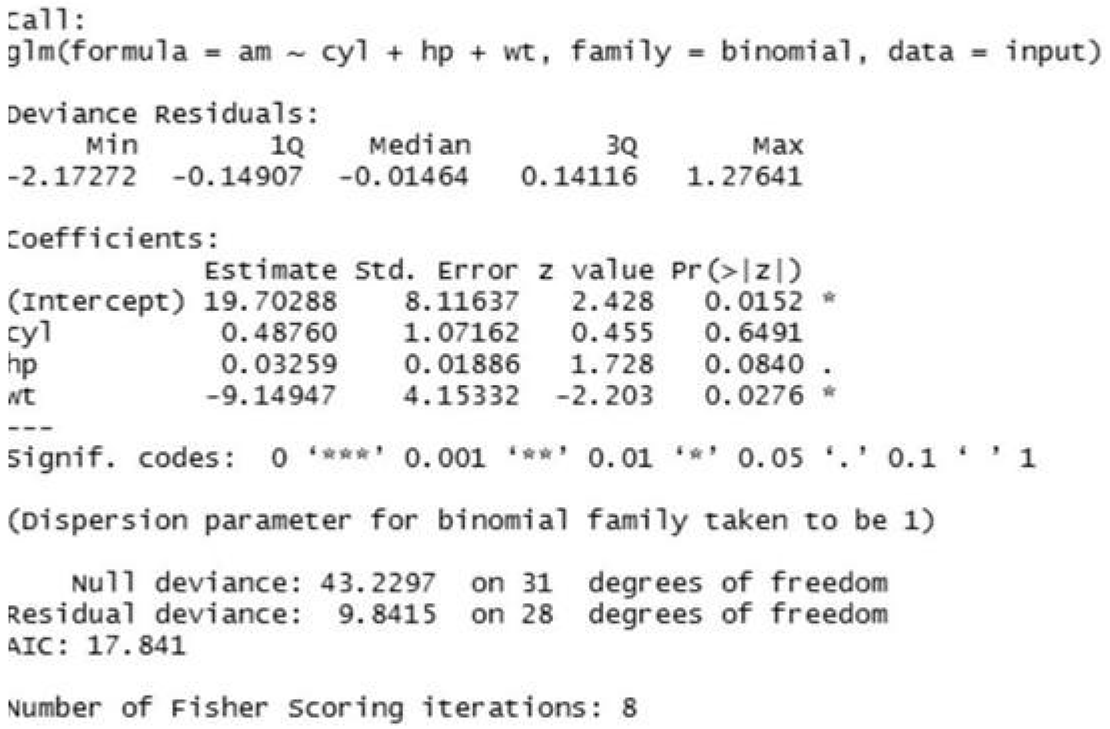
LOGISTIC REGRESSION:

```
input<-mtcars[,c("am","cyl","hp","wt")]
print(head(input))
input<-mtcars[,c("am","cyl","hp","wt")]
am.data=glm(formula=am+cyl+hp+wt, data=input, family=binomial)
print(summary(data))
```

OUTPUT:



```
> print(head(input))
                  am cyl  hp    wt
Mazda RX4          1   6 110 2.620
Mazda RX4 Wag      1   6 110 2.875
Datsun 710         1   4  93 2.320
Hornet 4 Drive     0   6 110 3.215
Hornet Sportabout  0   8 175 3.440
Valiant            0   6 105 3.460
```



```
Call:
glm(formula = am ~ cyl + hp + wt, family = binomial, data = input)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-2.17272  -0.14907  -0.01464   0.14116   1.27641

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) 19.70288    8.11637   2.428   0.0152 *
cyl          0.48760    1.07162   0.455   0.6491
hp           0.03259    0.01886   1.728   0.0840 .
wt          -9.14947    4.15332  -2.203   0.0276 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 43.2297  on 31  degrees of freedom
Residual deviance:  9.8415  on 28  degrees of freedom
AIC: 17.841

Number of Fisher Scoring iterations: 8
```

<u>RESULT :</u>

Thus the Linear and Logistic regression is implemented using R.

EX.NO: 05	**IMPLEMENT SVM CLASSIFICATION TECHNIQUES**
DATE:	

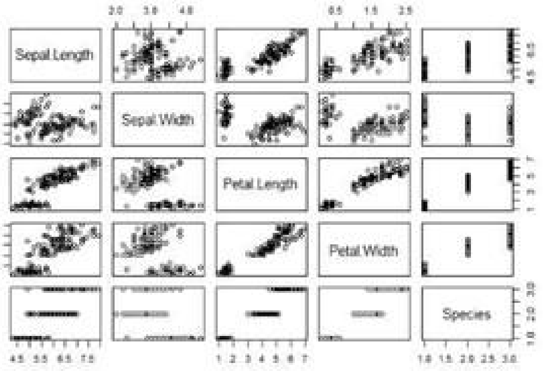
AIM :

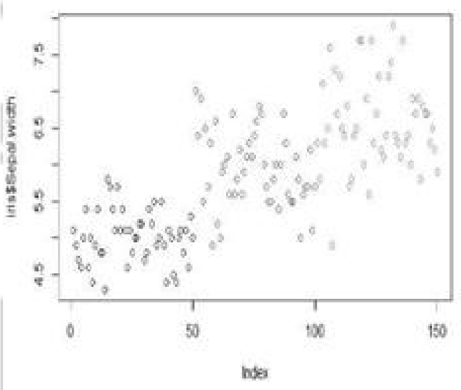
To implement SVM (Support Vector Machine) to find optimum hyper plane (Line in 2D, 3D hyper plane) which maximize the margin between two classes.

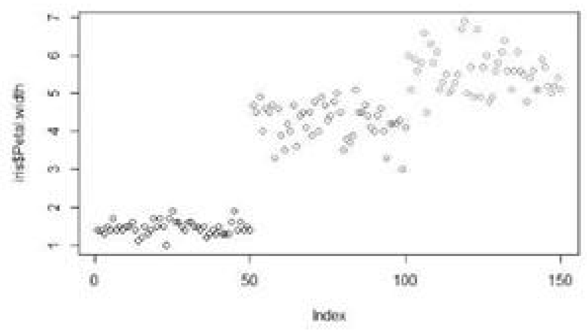
PROGRAM :

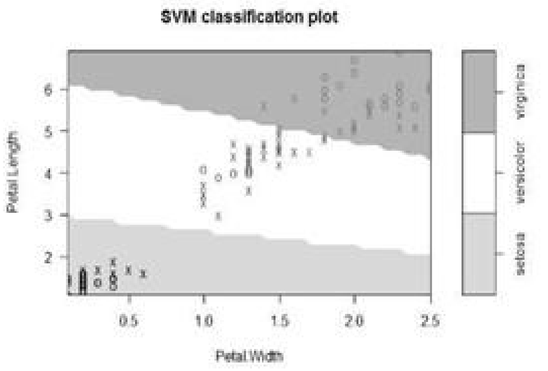
```
library(e1071)
plot(iris)
iris
plot(iris$Sepal.Length,iris$Sepal.width, col=iris$Species)
plot(iris$Sepal.Length,iris$Sepal.width, col=iris$Species)
s<-sample(150,100)
col<-c("Petal.Length","Petal.Width","Species")
iris_train<-iris[s,col]
iris_test<-iris[-s,col]
svmfit<-svm(Species~.,data=iris_train,kernel="linear",cost=.1,scale=FALSE)
print(svmfit)
plot(svmfit,iris_train[,col])
tuned<-tune(svm,Species~.,data=iris_train,kernel="linear",ranges=list(cost=c(0.001,0.01,.1,1,10,100)))
summary(tuned)
p<-predict(svmfit,iris_test[,col],type="class")
plot(p)
table(p,iris_test[,3])
mean(p==iris_test[,3])
```

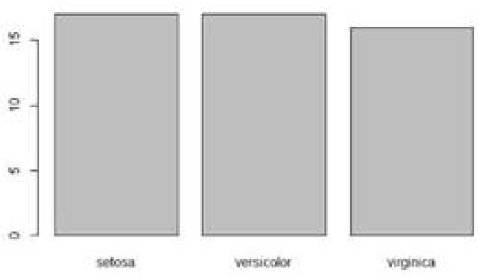
OUTPUT :











RESULT :

Thus SVM (Support Vector Machine) to find optimum hyper plane (Line in 2D, 3D hyper plane) which maximize the margin between two classes is implemented.

EX.NO: 06	**IMPLEMENT CLUSTERING TECHNIQUES**
DATE:	

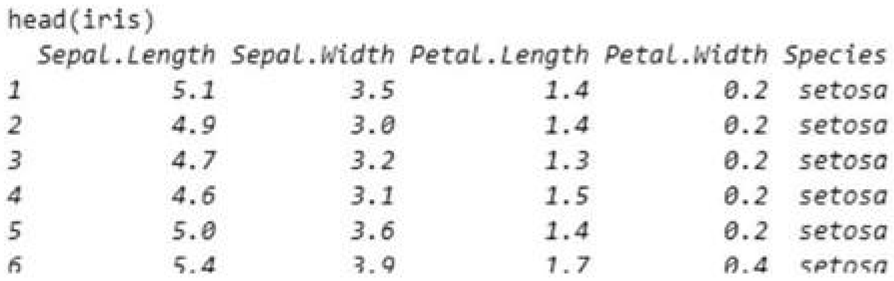
AIM :

To implement Clustering Techniques.

PROGRAM :

```
library(datasets)
head(iris)
library(ggplot2)
ggplot(iris, aes(Petal.Length, Petal.Width, color=Species)) + geom_point()
set.seed(20)
irisCluster<-kmeans(iris[,3:4],3,nstart=20)
irisCluster
table(irisCluster$cluster,iris$Species)
```

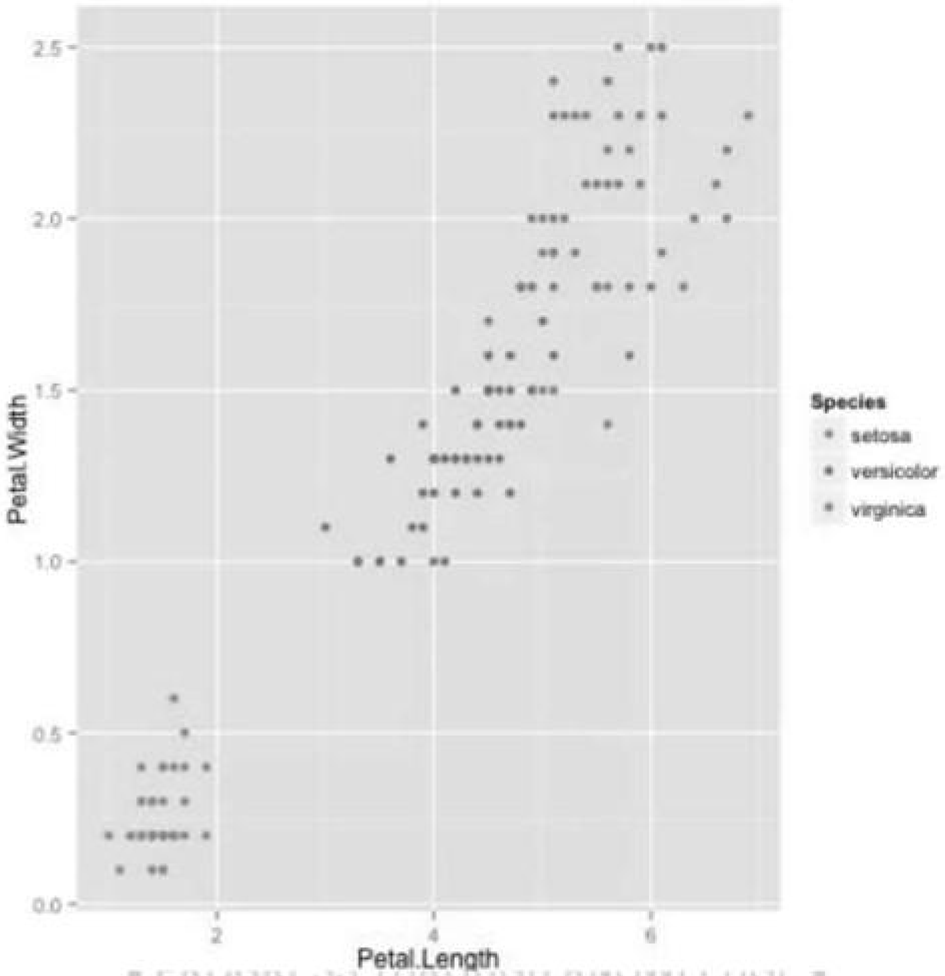
OUTPUT:

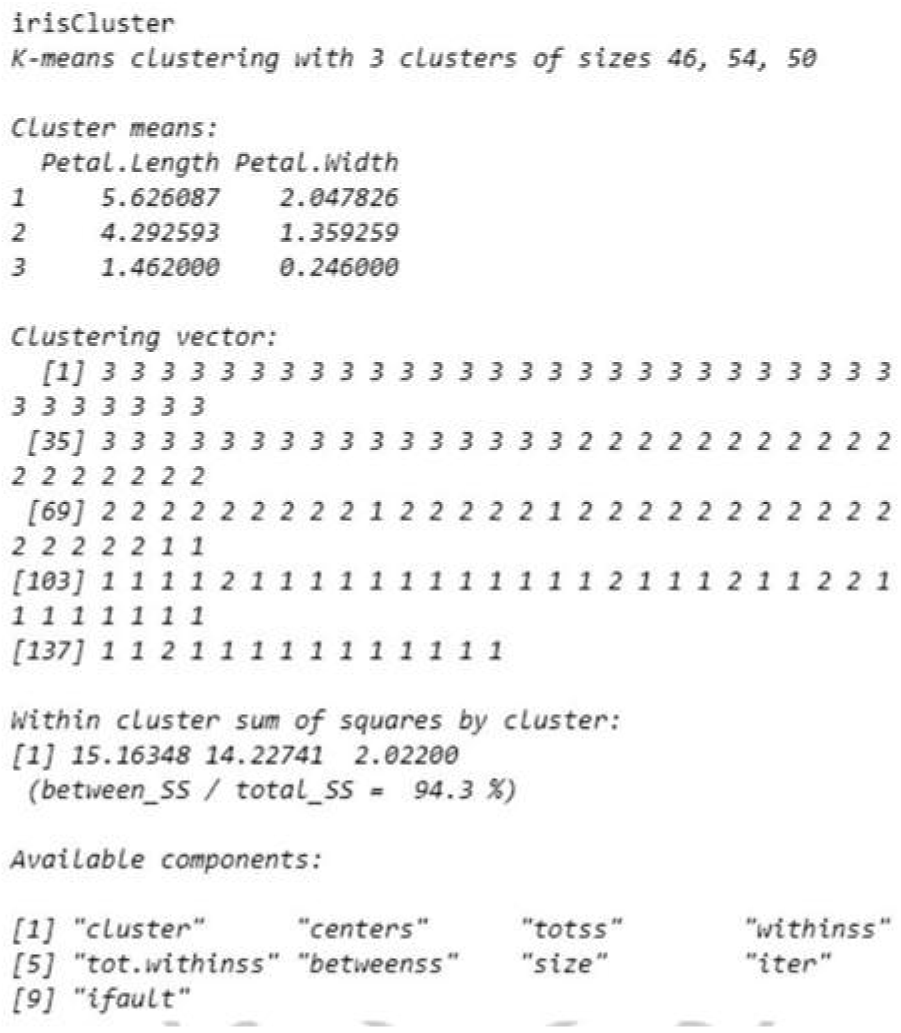


```
head(iris)
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5         0.2  setosa
5          5.0         3.6          1.4         0.2  setosa
6          5.4         3.9          1.7         0.4  setosa
```

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

Here is the plot:





```
irisCluster
K-means clustering with 3 clusters of sizes 46, 54, 50

Cluster means:
  Petal.Length Petal.Width
1     5.626087    2.047826
2     4.292593    1.359259
3     1.462000    0.246000

Clustering vector:
  [1] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
3 3 3 3 3 3 3
 [35] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2 2 2 2
 [69] 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2 2 1 1
[103] 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 2 1 1 2 2 1
1 1 1 1 1 1 1
[137] 1 1 2 1 1 1 1 1 1 1 1 1 1 1

Within cluster sum of squares by cluster:
[1] 15.16348 14.22741  2.02200
 (between_SS / total_SS =  94.3 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"
[5] "tot.withinss" "betweenss"    "size"         "iter"
[9] "ifault"
```

RESULT :

Thus the clustering techniques had been implemented and the output is verified.

EX.NO: 07	VISUALIZE DATA USING ANY PLOTTING FRAMEWORK
DATE:	

AIM :

To implement Data visualization is to provide an efficient graphical display for summarizing and reasoning about quantitative information.

PROCEDURE :

1.Histogram:

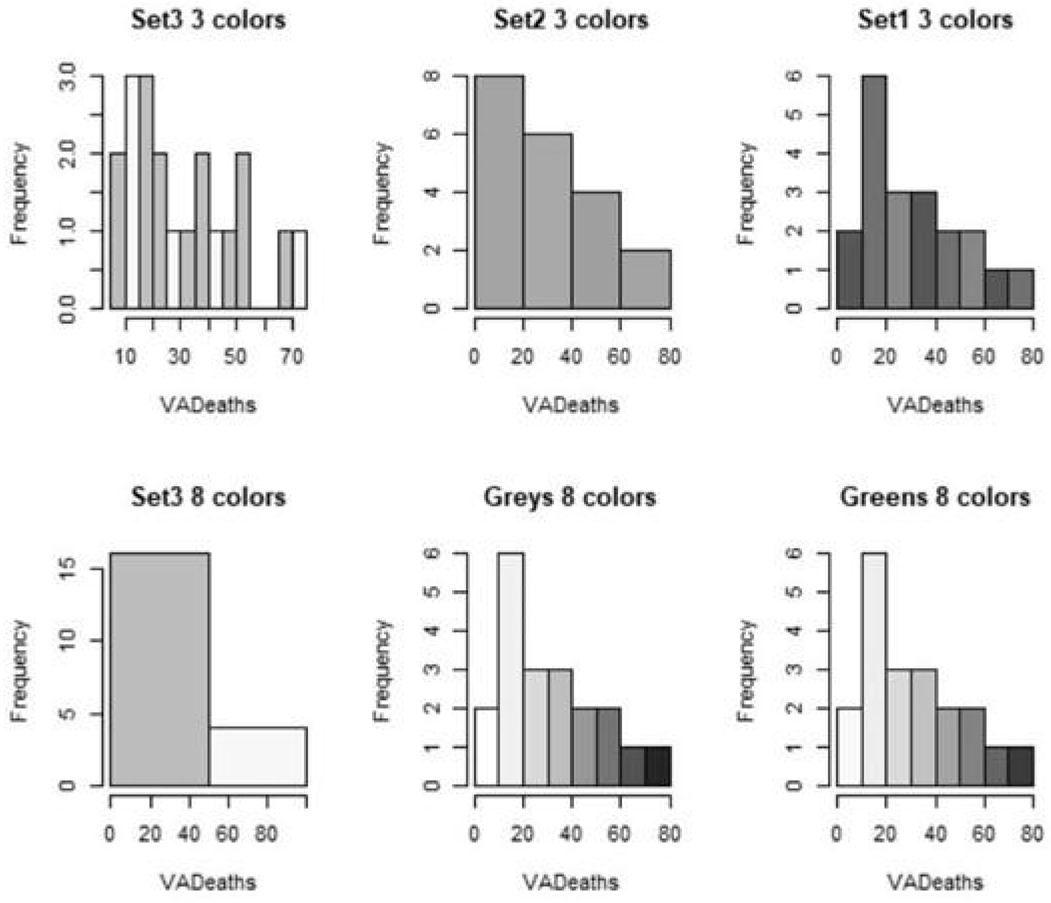
Histogram is basically a plot that breaks the data into bins or breaks and shows frequency distribution of these bins. You can change the breaks also and see the effect it has on data visualization in terms of understandability.

Note : We have used par (mfrow=c(2,5)) command to fit multiple graphs in same page for sake of clarity

Program:

```
library(RColorBrewer)
data(VADeaths)
par(mfrow=c(2,3))
hist(VADeaths,breaks=10,col=brewer.pal(3,”Set3”),main=”Set3 3 colors”)
hist(VADeaths,breaks=3,col=brewer.pal(3,”Set2”),main=”Set2 3 colors”)
hist(VADeaths,breaks=7,col=brewer.pal(3,”Set1”),main=”Set1 3 colors”)
hist(VADeaths,breaks=2,col=brewer.pal(8,”Set3”),main=”Set3 8 colors”)
hist(VADeaths,col=brewer.pal(8,”Greys”),main=”Greys 8 colors”)
hist(VADeaths,col=brewer.pal(8,”Greens”),main=”Greens 8 colors”)
```

OUTPUT:



2.Correlogram:

Correlated data is best visualized through corrplot. The 2D format is similar to a heat map, but it highlights statistics that are directly related.

Most correlograms highlight the amount of correlation between datasets at various points in time. Comparing sales data between different months or years is a basic example.

Program:

```
#data(“mtears”)
corr_matrix <- cor(mtears)
#with circles
corrplot(corr_matrix)
#with numbers and lower
corrplot(corr_matrix, method=’number’,type=”lower”)
```

OUTPUT:

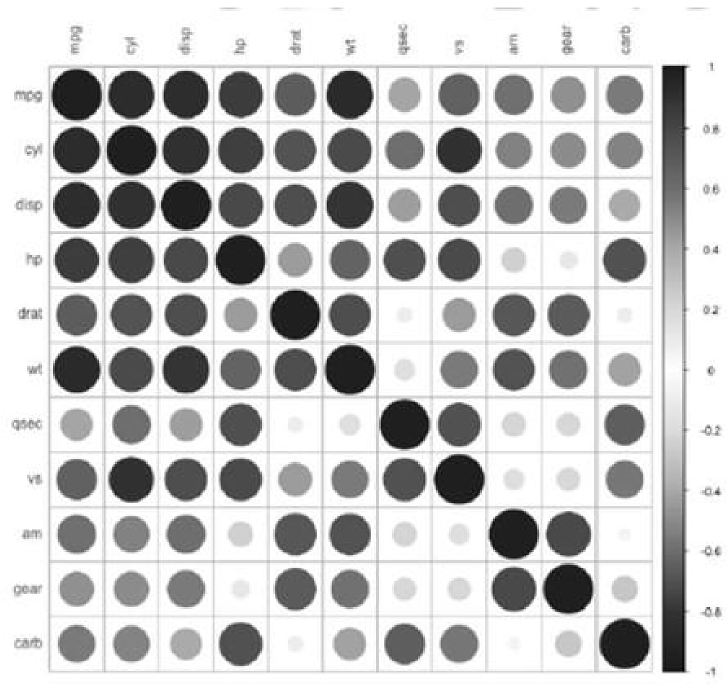


Fig 6. Correlogram with circles (courtesy of Abdul Majed Raja)

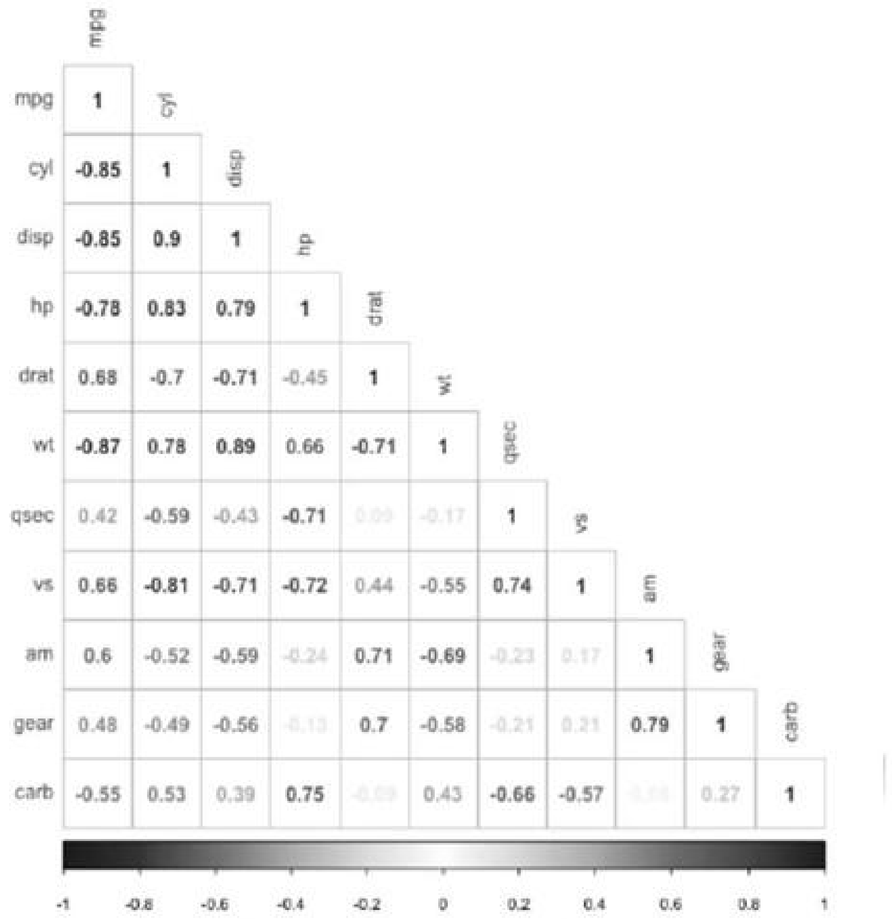


Fig 7. Correlogram with numbers (courtesy of Abdul Majed Raja)

RESULT :

The Histogram and Correlogram is implemented to visualize data by providing an efficient graphical display for summarizing and reasoning about quantitative information.

EX.NO: 08	**IMPLEMENT AN APPLICATION THAT STORES BIG DATA IN HBASE / MONGODB / PIG USING HADOOP / R**
DATE:	

AIM :

To Build an application to store big data in MongoDB

PROCEDURE :

MongoDB with R:

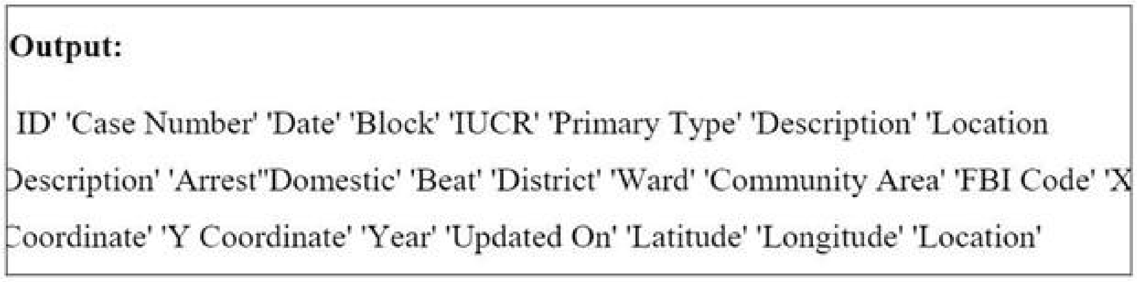
1. To use MongoDB with R, first, we have to download and install MongoDB. Next, start MongoDB. We can start MongoDB like so:

   mongodb

2. Inserting data

   Let's insert the crimes data from data.gov to MongoDB. The dataset reflects reported incidents of crime (with the exception of murders where data exists for each victim) that occurred in the city of Chicago since 2001.

```
library(ggplot2)
library(dplyr)
library(maps)
library(ggmap)
library(mongolite)
library(lubridate)
library(gridExtra)
crimes=data.table::fread("Crimes_2001_to_present.csv")
names(crimes)
```



Output:

```
ID' 'Case Number' 'Date' 'Block' 'IUCR' 'Primary Type' 'Description' 'Location
escription' 'Arrest''Domestic' 'Beat' 'District' 'Ward' 'Community Area' 'FBI Code' 'X
oordinate' 'Y Coordinate' 'Year' 'Updated On' 'Latitude' 'Longitude' 'Location'
```

3. Let's remove spaces in the column names to avoid any problems when we query it from MongoDB.

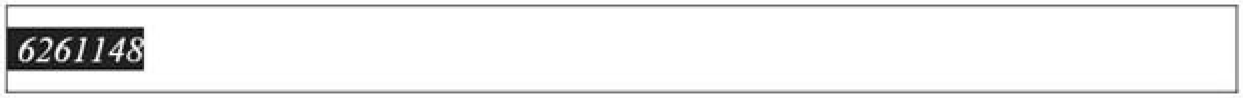
```
names(crimes)=gsub(“ “,” “,names(crimes))
names(crimes)
```

4. Let’s use the insert function from the mongolite package to insert rows to a collection in MongoDB. Let’s create a database called Chicago and call the collection crimes.

```
my_collection=mongo(collection=”crimes”,db=”Chicago”)
my_collection$insert(crimes)
```

5. Let’s check if we have inserted the “crimes” data.

```
my_collection$count()
```



```
6261148
```

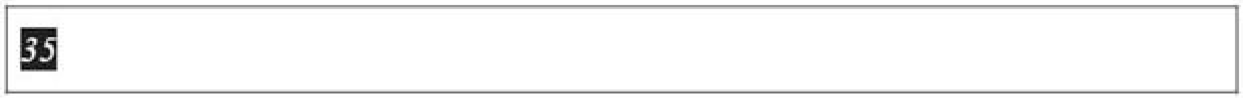
We see that the collection has 6261148 records.

6. First, let’s look what the data looks like by displaying one record:

```
my_collection$iterate()$one()
```

7. How many distinct “Primary Type” do we have?

```
Length(my_collection$distinct(“PrimaryType”))
```

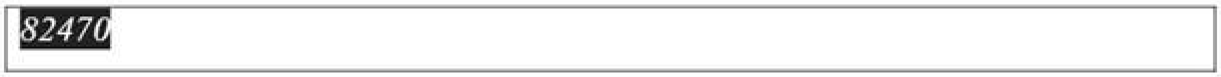


```
35
```

As shown above, there are 35 different crime primary types in the database. We will see the patterns of the most common crime types below.

8. Now, let’s see how many domestic assaults there are in the collection.

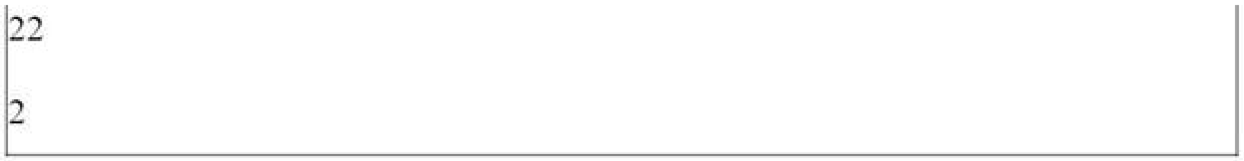
```
my_collection$count(‘{’’PrimaryType”:”ASSAULT”,”Domestic”:”true”}’)
```



```
82470
```

9. To get the filtered data and we can also retrieve only the columns of interest.

```
query1=my_collection$find('{"PrimaryType":"ASSAULT","Domestic":"true"}')
query2=my_collection$find('{"PrimaryType":"ASSAULT","Domestic":"true"}')
fields=('{"_id":0,"PrimaryType":1,"Domestic":1}')
ncol(query1)
ncol(query2)
```



```
22

2
```

RESULT:

Thus, the application had been built to store big data in MongoDB.