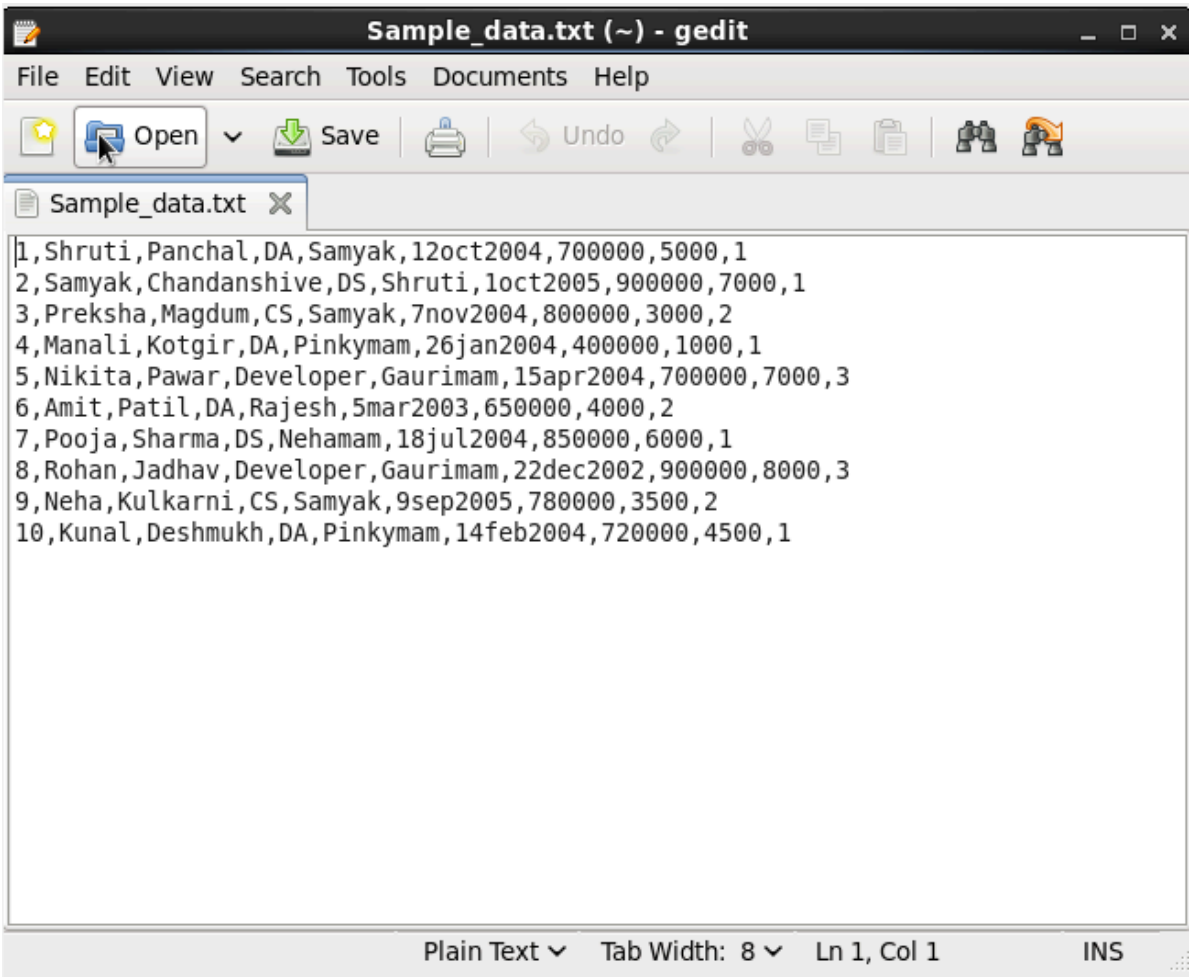
Index

No	Title	Date	Signature
1	Implement word count/frequency programmes using MapReduce.	12/02/2026	
2	Implement a MapReduce program that processes a weather dataset. with 1) pig 2) spark		
3	Implement an application that stores big data in MongoDB and manipulate it using Python		
4	Implement Apriori algorithm for market basket analysis.		
5	Configure the Hive and implement the application in Hive.		
6	Implement Agglomerative Hierarchy clustering algorithm A) Single Linkage Method B) Complete Linkage Method C) Average Linkage Method D) Ward Linkage Method		
7	Implement DBSCAN Density based clustering algorithm.		
8	Implement a program in python to demonstrate how to compute tf-idf values of words from a corpus.	9/03/2026	
9	Implement SVM classification techniques.		
10	Create an ARIMA Model for Time Series Forecasting in Python		

Practical No: 1
Aim: Implement word count/frequency programmes using MapReduce.

Create a text file with name sampledata.txt in cloudera(/home/cloudera/Sample_data.txt):



Move file in hdfs in :

hdfs dfs -put /home/cloudera/Sample_data.txt /user/cloudera/input/

- input is a **folder inside HDFS**, not your normal Linux folder.
- It exists inside **HDFS**, not inside /home.
- Check if input folder exists: hdfs dfs -ls /user/cloudera
-

Then start Pig:

```
[cloudera@quickstart ~]$ pig
log4j:WARN No appenders could be found for logger (org.apache.hadoop.util.Shell)
.
log4j:WARN Please initialize the log4j system properly.
log4j:WARN See http://logging.apache.org/log4j/1.2/faq.html#noconfig for more in
fo.
2026-02-11 02:14:02,713 [main] INFO  org.apache.pig.Main - Apache Pig version 0.
12.0-cdh5.13.0 (rexported) compiled Oct 04 2017, 11:09:03
2026-02-11 02:14:02,714 [main] INFO  org.apache.pig.Main - Logging error message
s to: /home/cloudera/pig_1770804842674.log
2026-02-11 02:14:02,748 [main] INFO  org.apache.pig.impl.util.Utils - Default bo
otup file /home/cloudera/.pigbootup not found
2026-02-11 02:14:03,360 [main] INFO  org.apache.hadoop.conf.Configuration.deprec
ation - mapred.job.tracker is deprecated. Instead, use mapreduce.jobtracker.addr
ess
2026-02-11 02:14:03,360 [main] INFO  org.apache.hadoop.conf.Configuration.deprec
ation - fs.default.name is deprecated. Instead, use fs.defaultFS
2026-02-11 02:14:03,360 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.HExecutionEngine - Connecting to hadoop file system at: hdfs://quickstart.clo
udera:8020
2026-02-11 02:14:05,176 [main] INFO  org.apache.hadoop.conf.Configuration.deprec
ation - mapred.job.tracker is deprecated. Instead, use mapreduce.jobtracker.addr
ess
2026-02-11 02:14:05,176 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.HExecutionEngine - Connecting to map-reduce job tracker at: localhost:8021
2026-02-11 02:14:05,176 [main] INFO  org.apache.hadoop.conf.Configuration.deprec
ation - fs.default.name is deprecated. Instead, use fs.defaultFS
2026-02-11 02:14:05,275 [main] INFO  org.apache.hadoop.conf.Configuration.deprec
ation - fs.default.name is deprecated. Instead, use fs.defaultFS
2026-02-11 02:14:05,276 [main] INFO  org.apache.hadoop.conf.Configuration.deprec
ation - mapred.job.tracker is deprecated. Instead, use mapreduce.jobtracker.addr
ess
```

```
grunt> employee = LOAD '/user/cloudera/input/Sample_data.txt'
>> USING PigStorage(',')
>> AS (
>>     cmpcode:int,
>>     cmpfname:chararray,
>>     emplname:chararray,
>>     job:chararray,
>>     manager:chararray,
>>     hiredate:chararray,
>>     salary:int,
>>     commission:int,
>>     deptcode:int
>> );
```

```
grunt> DUMP employee;
```

Output:

```
mplete
2026-02-11 02:32:25,650 [main] INFO  org.apache.pig.tools.pigstats.SimplePigStats - Script Statistics:

HadoopVersion   PigVersion      UserId  StartedAt       FinishedAt      Features
2.6.0-cdh5.13.0 0.12.0-cdh5.13.0        cloudera        2026-02-11 02:31:54     2026-02-11 02:32:25     UNKNOWN

Success!

Job Stats (time in seconds):
JobId   Maps    Reduces MaxMapTime      MinMapTIme      AvgMapTime      MedianMapTime   MaxReduceTime   MinReduceTime   AvgRe
duceTime        MedianReducetime        Alias   Feature Outputs
job_1770798975700_0007  1       0       7       7       7       7       n/a     n/a     n/a     n/a     employee        MAP_O
NLY     hdfs://quickstart.cloudera:8020/tmp/temp1548826103/tmp-1179320103,

Input(s):
Successfully read 11 records (910 bytes) from: "/user/cloudera/input/Sample_data.txt"

Output(s):
Successfully stored 11 records (606 bytes) in: "hdfs://quickstart.cloudera:8020/tmp/temp1548826103/tmp-1179320103"

Counters:
Total records written : 11
Total bytes written : 606
Spillable Memory Manager spill count : 0
Total bags proactively spilled: 0
Total records proactively spilled: 0

Job DAG:
job_1770798975700_0007
```

```
2026-02-11 02:32:25,761 [main] WARN  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Encount
ered Warning ACCESSING_NON_EXISTENT_FIELD 8 time(s).
2026-02-11 02:32:25,761 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Success
!
2026-02-11 02:32:25,766 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instea
d, use fs.defaultFS
2026-02-11 02:32:25,766 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - mapred.job.tracker is deprecated. Ins
tead, use mapreduce.jobtracker.address
2026-02-11 02:32:25,767 [main] INFO  org.apache.pig.data.SchemaTupleBackend - Key [pig.schematuple] was not set... will not g
enerate code.
2026-02-11 02:32:25,788 [main] INFO  org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1
2026-02-11 02:32:25,789 [main] INFO  org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to pro
cess : 1
(1,Shruti,Panchal,DA,Samyak,12oct2004,700000,5000,1)
(2,Samyak,Chandanshive,DS,Shruti,1oct2005,900000,7000,1)
(3,Preksha,Magdum,CS,Samyak,7nov2004,800000,3000,2)
(4,Manali,Kotgir,DA,Pinkymam,26jan2004,400000,1000,1)
(5,Nikita,Pawar,Developer,Gaurimam,15apr2004,700000,7000,3)
(6,Amit,Patil,DA,Rajesh,5mar2003,650000,4000,2)
(7,Pooja,Sharma,DS,Nehamam,18jul2004,850000,6000,1)
(8,Rohan,Jadhav,Developer,Gaurimam,22dec2002,900000,8000,3)
(9,Neha,Kulkarni,CS,Samyak,9sep2005,780000,3500,2)
(10,Kunal,Deshmukh,DA,Pinkymam,14feb2004,720000,4500,1)
(,,,,,,,,)
```

Grouping Employees by dept:

```
grunt> grp = GROUP employee BY deptcode;
2026-02-11 02:32:39,408 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instea
d, use fs.defaultFS
2026-02-11 02:32:39,408 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - mapred.job.tracker is deprecated. Ins
```

Max_salary:

```
grunt> maxsal = FOREACH grp GENERATE
>>          group AS deptcode,
>>          MAX(employee.salary) AS max_salary;
```

Dump for final O/p:

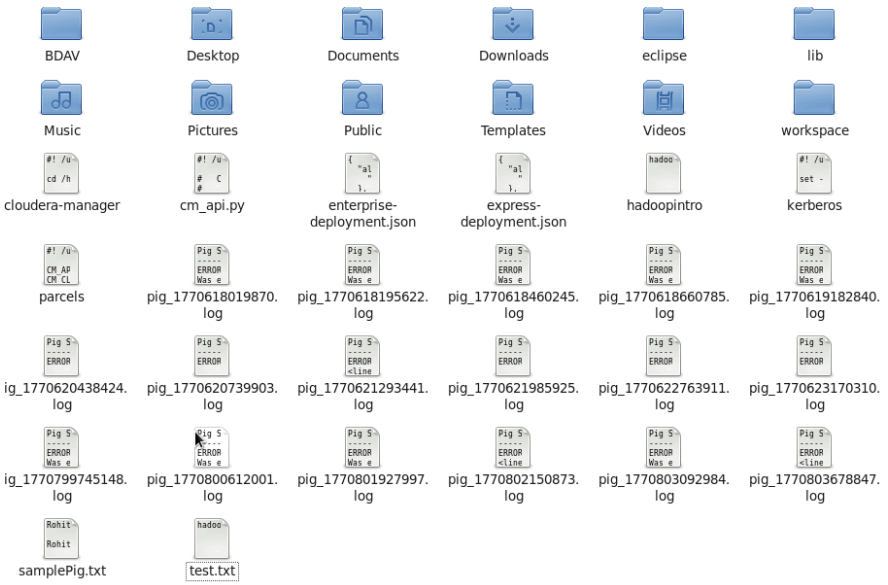
```
grunt> DUMP maxsal;
2026-02-11 02:33:45,229 [main] INFO  org.apache.pig.tools.pigstats.ScriptState - Pig features used in the script: GROUP_BY
2026-02-11 02:33:45,231 [main] INFO  org.apache.pig.newplan.logical.optimizer.LogicalPlanOptimizer - {RULES_ENABLED=[AddForEa
ch, ColumnMapKeyPrune, DuplicateForEachColumnRewrite, GroupByConstParallelSetter, ImplicitSplitInserter, LimitOptimizer, Load
TypeCastInserter, MergeFilter, MergeForEach, NewPartitionFilterOptimizer, PushDownForEachFlatten, PushUpFilter, SplitFilter,
StreamTypeCastInserter], RULES_DISABLED=[FilterLogicExpressionSimplifier, PartitionFilterOptimizer]}
2026-02-11 02:33:45,256 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MRCompiler - File concatena
tion threshold: 100 optimistic? false
2026-02-11 02:33:45,262 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.CombinerOptimizer - Choosin
g to move algebraic foreach to combiner
2026-02-11 02:33:45,281 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MultiQueryOptimizer - MR pl
an size before optimization: 1
2026-02-11 02:33:45,281 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MultiQueryOptimizer - MR pl
an size after optimization: 1
2026-02-11 02:33:45,307 [main] INFO  org.apache.hadoop.yarn.client.RMProxy - Connecting to ResourceManager at /0.0.0.0:8032
2026-02-11 02:33:45,310 [main] INFO  org.apache.pig.tools.pigstats.ScriptState - Pig script settings are added to the job
2026-02-11 02:33:45,346 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.JobControlCompiler - mapred
.job.reduce.markreset.buffer.percent is not set, set to default 0.3
2026-02-11 02:33:45,346 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.JobControlCompiler - Reduce
 phase detected, estimating # of required reducers.
2026-02-11 02:33:45,347 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.JobControlCompiler - Using
reducer estimator: org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.InputSizeReducerEstimator
```

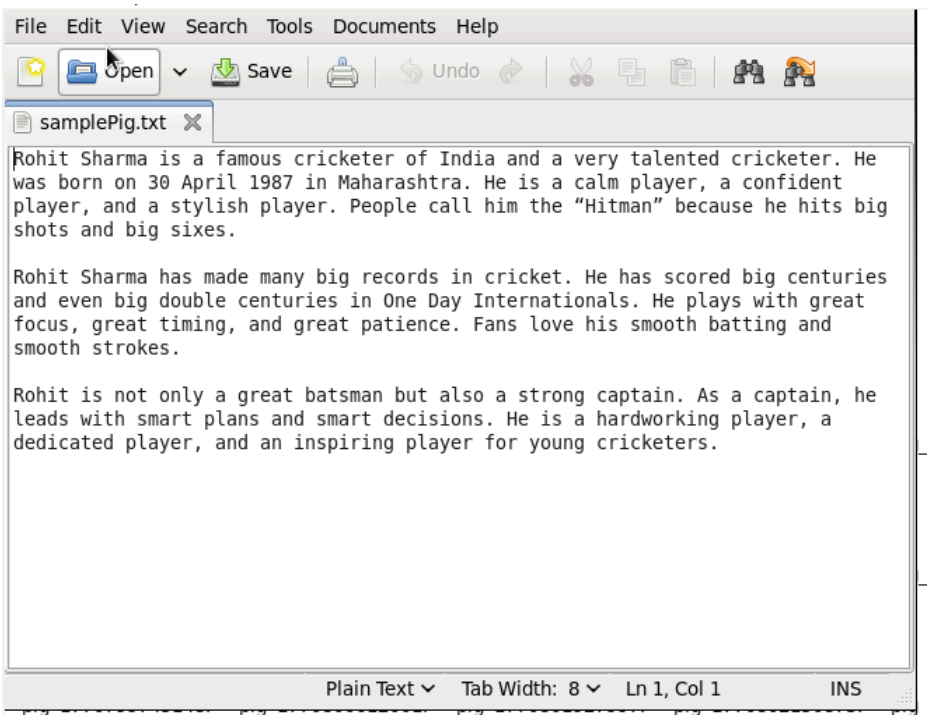
```
2026-02-11 02:33:49,395 [JobControl] INFO  org.apache.hadoop.yarn.client.api.impl.YarnClientImpl - Submitted application appl
ication_1770798975700_0008
2026-02-11 02:33:49,400 [JobControl] INFO  org.apache.hadoop.mapreduce.Job - The url to track the job: http://quickstart.clou
dera:8088/proxy/application_1770798975700_0008/
2026-02-11 02:33:49,617 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - HadoopJ
obId: job_1770798975700_0008
2026-02-11 02:33:49,617 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Process
ing aliases employee,grp,maxsal
2026-02-11 02:33:49,617 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - detaile
d locations: M: employee[1,11],employee[-1,-1],maxsal[27,9],grp[26,6] C: maxsal[27,9],grp[26,6] R: maxsal[27,9]
2026-02-11 02:33:49,617 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - More in
formation at: http://localhost:50030/jobdetails.jsp?jobid=job_1770798975700_0008
2026-02-11 02:33:49,655 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - 0% comp
lete
2026-02-11 02:34:04,972 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - 50% com
plete
2026-02-11 02:34:14,723 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - 100% co
mplete
2026-02-11 02:34:14,724 [main] INFO  org.apache.pig.tools.pigstats.SimplePigStats - Script Statistics:

HadoopVersion   PigVersion      UserId  StartedAt       FinishedAt      Features
2.6.0-cdh5.13.0 0.12.0-cdh5.13.0        cloudera        2026-02-11 02:33:45     2026-02-11 02:34:14     GROUP_BY

Success!

Job Stats (time in seconds):
JobId   Maps    Reduces MaxMapTime      MinMapTIme      AvgMapTime      MedianMapTime   MaxReduceTime   MinReduceTime   AvgRe
duceTime        MedianReducetime        Alias   Feature Outputs
job_1770798975700_0008  1       1       5       5       5       5       4       4       4       4       employee,grp,maxsal G
ROUP_BY,COMBINER        hdfs://quickstart.cloudera:8020/tmp/temp1548826103/tmp-82180312,

Input(s):
Successfully read 11 records (910 bytes) from: "/user/cloudera/input/Sample_data.txt"
```

```
2026-02-11 02:34:14,817 [main] WARN  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Encount
ered Warning ACCESSING_NON_EXISTENT_FIELD 8 time(s).
2026-02-11 02:34:14,817 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Success
!
2026-02-11 02:34:14,817 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instea
d, use fs.defaultFS
2026-02-11 02:34:14,818 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - mapred.job.tracker is deprecated. Ins
tead, use mapreduce.jobtracker.address
2026-02-11 02:34:14,818 [main] INFO  org.apache.pig.data.SchemaTupleBackend - Key [pig.schematuple] was not set... will not g
enerate code.
2026-02-11 02:34:14,824 [main] INFO  org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1
2026-02-11 02:34:14,824 [main] INFO  org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to pro
cess : 1
(1,900000)
(2,800000)
(3,900000)
(,)
```

Practical No: 2
Aim: Implement a MapReduce program that processes a weather dataset. with pig and spark

Pig:
Create a text file in Cloudera:





```
[cloudera@quickstart ~]$ hadoop fs -ls /
Found 8 items
drwxrwxrwx   - hdfs     supergroup          0 2017-10-23 09:15 /benchmarks
drwxr-xr-x   - hbase    supergroup          0 2026-02-11 00:36 /hbase
drwxr-xr-x   - cloudera supergroup          0 2025-12-10 02:38 /samples
drwxr-xr-x   - solr     solr                0 2017-10-23 09:18 /solr
drwxrwxrwt   - hdfs     supergroup          0 2026-02-11 01:40 /tmp
drwxr-xr-x   - cloudera supergroup          0 2025-12-10 02:30 /try
drwxr-xr-x   - hdfs     supergroup          0 2017-10-23 09:17 /user
drwxr-xr-x   - hdfs     supergroup          0 2017-10-23 09:17 /var
[cloudera@quickstart ~]$ sudo su hdfs -l -c 'hdfs dfsadmin -safemode leave'\
>
```

```
[cloudera@quickstart ~]$ sudo su hdfs -l -c 'hdfs dfsadmin -safemode leave''
>


Safe mode is OFF
```

```
[cloudera@quickstart ~]$ hadoop fs -put /home/cloudera/samplePig.txt
```

```
[cloudera@quickstart ~]$ pig
log4j:WARN No appenders could be found for logger (org.apache.hadoop.util.Shell).
log4j:WARN Please initialize the log4j system properly.
log4j:WARN See http://logging.apache.org/log4j/1.2/faq.html#noconfig for more info.
2026-02-11 01:44:53,003 [main] INFO  org.apache.pig.Main - Apache Pig version 0.12.0-cdh5.13.0 (rexported) compiled Oct 04 20
17, 11:09:03
2026-02-11 01:44:53,003 [main] INFO  org.apache.pig.Main - Logging error messages to: /home/cloudera/pig_1770803092984.log
2026-02-11 01:44:53,026 [main] INFO  org.apache.pig.impl.util.Utils - Default bootup file /home/cloudera/.pigbootup not found
2026-02-11 01:44:53,523 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - mapred.job.tracker is deprecated. Ins
tead, use mapreduce.jobtracker.address
2026-02-11 01:44:53,523 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instea
d, use fs.defaultFS
2026-02-11 01:44:53,523 [main] INFO  org.apache.pig.backend.hadoop.executionengine.HExecutionEngine - Connecting to hadoop fi
le system at: hdfs://quickstart.cloudera:8020
2026-02-11 01:44:54,540 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - mapred.job.tracker is deprecated. Ins
tead, use mapreduce.jobtracker.address
2026-02-11 01:44:54,540 [main] INFO  org.apache.pig.backend.hadoop.executionengine.HExecutionEngine - Connecting to map-reduc
e job tracker at: localhost:8021
2026-02-11 01:44:54,541 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instea
d, use fs.defaultFS
2026-02-11 01:44:54,596 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instea
d, use fs.defaultFS
2026-02-11 01:44:54,597 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - mapred.job.tracker is deprecated. Ins
tead, use mapreduce.jobtracker.address
2026-02-11 01:44:54,647 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instea
```

```
grunt> lines = LOAD '/user/cloudera/samplePig.txt' AS (line:chararray);
grunt> words = FOREACH lines GENERATE FLATTEN(TOKENIZE(line));
grunt> grouped = GROUP words BY $0;
grunt> counts = FOREACH grouped GENERATE group,COUNT(words);
grunt> STORE counts INTO '/user/cloudera/wordcount';
2026-02-11 01:55:27,046 [main] INFO  org.apache.pig.tools.pigstats.ScriptState - Pig features used in the script: GROUP_BY
2026-02-11 01:55:27,094 [main] INFO  org.apache.pig.newplan.logical.optimizer.LogicalPlanOptimizer - {RULES_ENABLED=[AddForEa
ch, ColumnMapKeyPrune, DuplicateForEachColumnRewrite, GroupByConstParallelSetter, ImplicitSplitInserter, LimitOptimizer, Load
TypeCastInserter, MergeFilter, MergeForEach, NewPartitionFilterOptimizer, PushDownForEachFlatten, PushUpFilter, SplitFilter,
StreamTypeCastInserter], RULES_DISABLED=[FilterLogicExpressionSimplifier, PartitionFilterOptimizer]}
2026-02-11 01:55:27,116 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - mapred.textoutputformat.separator is
deprecated. Instead, use mapreduce.output.textoutputformat.separator
2026-02-11 01:55:27,209 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MRCompiler - File concatena
tion threshold: 100 optimistic? false
2026-02-11 01:55:27,225 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.CombinerOptimizer - Choosin
g to move algebraic foreach to combiner
2026-02-11 01:55:27,256 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MultiQueryOptimizer - MR pl
an size before optimization: 1
2026-02-11 01:55:27,256 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MultiQueryOptimizer - MR pl
an size after optimization: 1
2026-02-11 01:55:27,368 [main] INFO  org.apache.hadoop.yarn.client.RMProxy - Connecting to ResourceManager at /0.0.0.0:8032
```

Final Output:

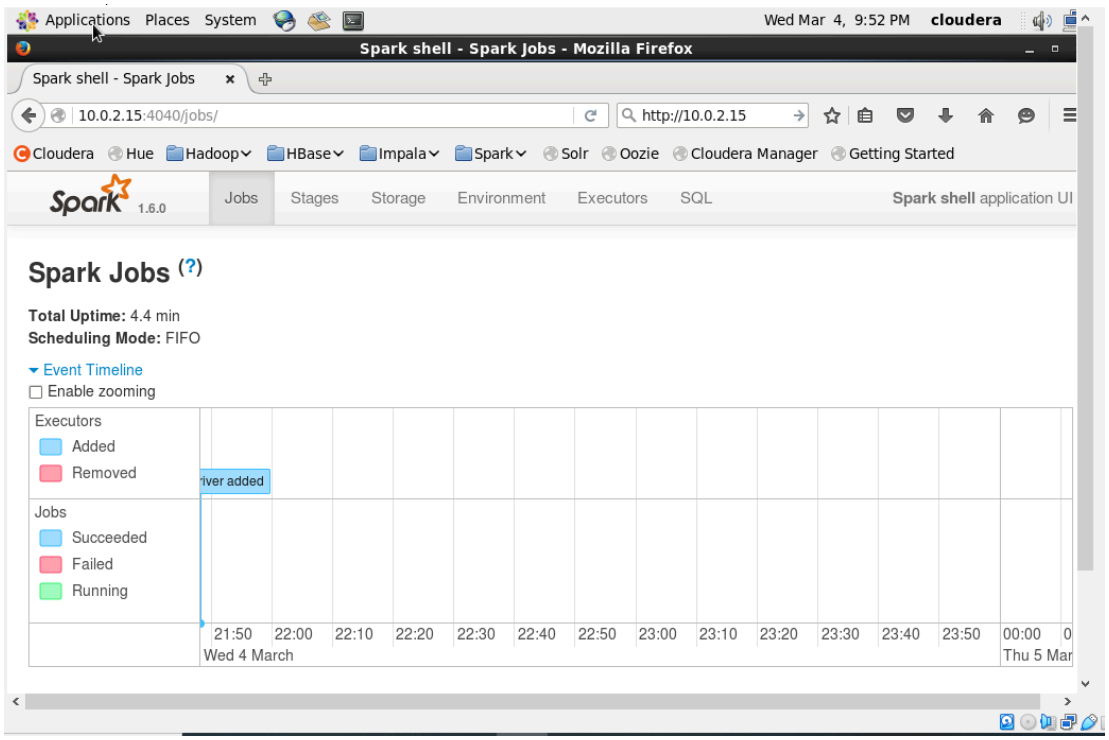
```
2026-02-11 01:55:32,265 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - 0% comp
lete
2026-02-11 01:55:46,065 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - 50% com
plete
2026-02-11 01:55:57,597 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - 100% co
mplete
2026-02-11 01:55:57,600 [main] INFO  org.apache.pig.tools.pigstats.SimplePigStats - Script Statistics:

HadoopVersion   PigVersion      UserId  StartedAt       FinishedAt      Features
2.6.0-cdh5.13.0 0.12.0-cdh5.13.0        cloudera        2026-02-11 01:55:27     2026-02-11 01:55:57     GROUP_BY

Success!

Job Stats (time in seconds):
JobId   Maps    Reduces MaxMapTime      MinMapTIme      AvgMapTime      MedianMapTime   MaxReduceTime   MinReduceTime   AvgRe
duceTime        MedianReducetime        Alias   Feature Outputs
job_1770798975700_0003  1       1       5       5       5       5       3       3       3       3       counts,grouped,lines,
words   GROUP_BY,COMBINER       /user/cloudera/wordcount,

Input(s):
Successfully read 6 records (1100 bytes) from: "/user/cloudera/samplePig.txt"

Output(s):
Successfully stored 82 records (689 bytes) in: "/user/cloudera/wordcount"

Counters:
Total records written : 82
Total bytes written : 689
```

Spark:

Open cloudera-> create text file -> then turn on spark shell by cmd prompt

Command: spark-shell

```
File  Edit  View  Search  Terminal  Help
[cloudera@quickstart ~]$ spark-shell
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel).
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/lib/zookeeper/lib/slf4j-log4j12-1.7.5.jar
!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/lib/flume-ng/lib/slf4j-log4j12-1.7.5.jar!
/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/lib/parquet/lib/slf4j-log4j12-1.7.5.jar!/
org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/lib/avro/avro-tools-1.7.6-cdh5.13.0.jar!/
org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
Welcome to

      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 1.6.0
      /_/

Using Scala version 2.10.5 (Java HotSpot(TM) 64-Bit Server VM, Java 1.7.0_67)
Type in expressions to have them evaluated.
Type :help for more information.
26/03/04 21:42:38 WARN util.NativeCodeLoader: Unable to load native-hadoop libra
ry for your platform... using builtin-java classes where applicable
26/03/04 21:42:39 WARN util.Utils: Your hostname, quickstart.cloudera resolves t
o a loopback address: 127.0.0.1; using 10.0.2.15 instead (on interface eth1)
26/03/04 21:42:39 WARN util.Utils: Set SPARK_LOCAL_IP if you need to bind to ano
ther address
Spark context available as sc (master = local[*], app id = local-1772689361057).
26/03/04 21:42:43 WARN shortcircuit.DomainSocketFactory: The short-circuit local
 reads feature cannot be used because libhadoop cannot be loaded.
SQL context available as sqlContext.

scala>
```



Command:
val inputRDD = sc.textFile("file:///home/cloudera/samplePig.txt")
Command:
inputRDD.collect()

```
scala> val inputRDD = sc.textFile("file:///home/cloudera/samplePig.txt")
inputRDD: org.apache.spark.rdd.RDD[String] = file:///home/cloudera/samplePig.txt MapPartitionsRDD[1] at textFile at <console>
:27

scala> inputRDD.collect()
res0: Array[String] = Array(Rohit Sharma is a famous cricketer of India and a very talented cricketer.)

scala>
```

Command:
val wordsRDD = inputRDD.flatMap(line => line.split(" "))
Command:
wordsRDD.collect()

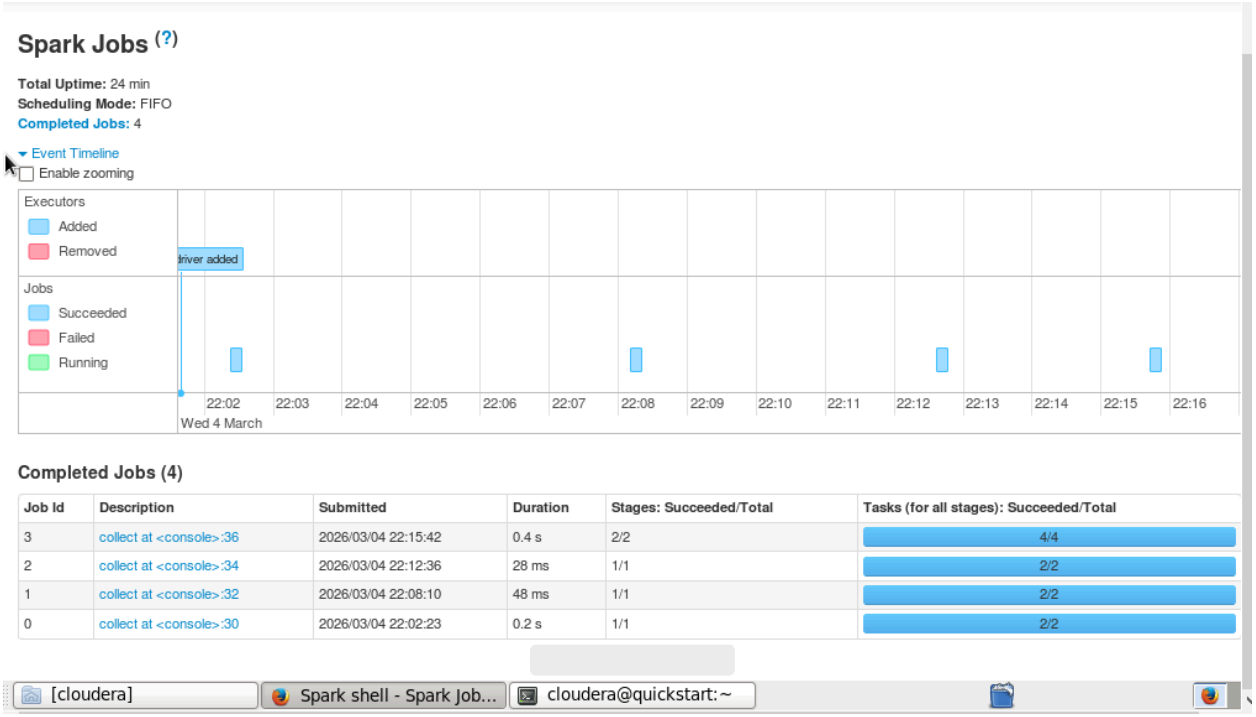
```
scala> val wordsRDD = inputRDD.flatMap(line => line.split(" "))
wordsRDD: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[6] at flatMap at <console>:29

scala> wordsRDD.collect()
res1: Array[String] = Array(Rohit, Sharma, is, a, famous, cricketer, of, India, and, a, very, talented, cricketer.)

scala>
```

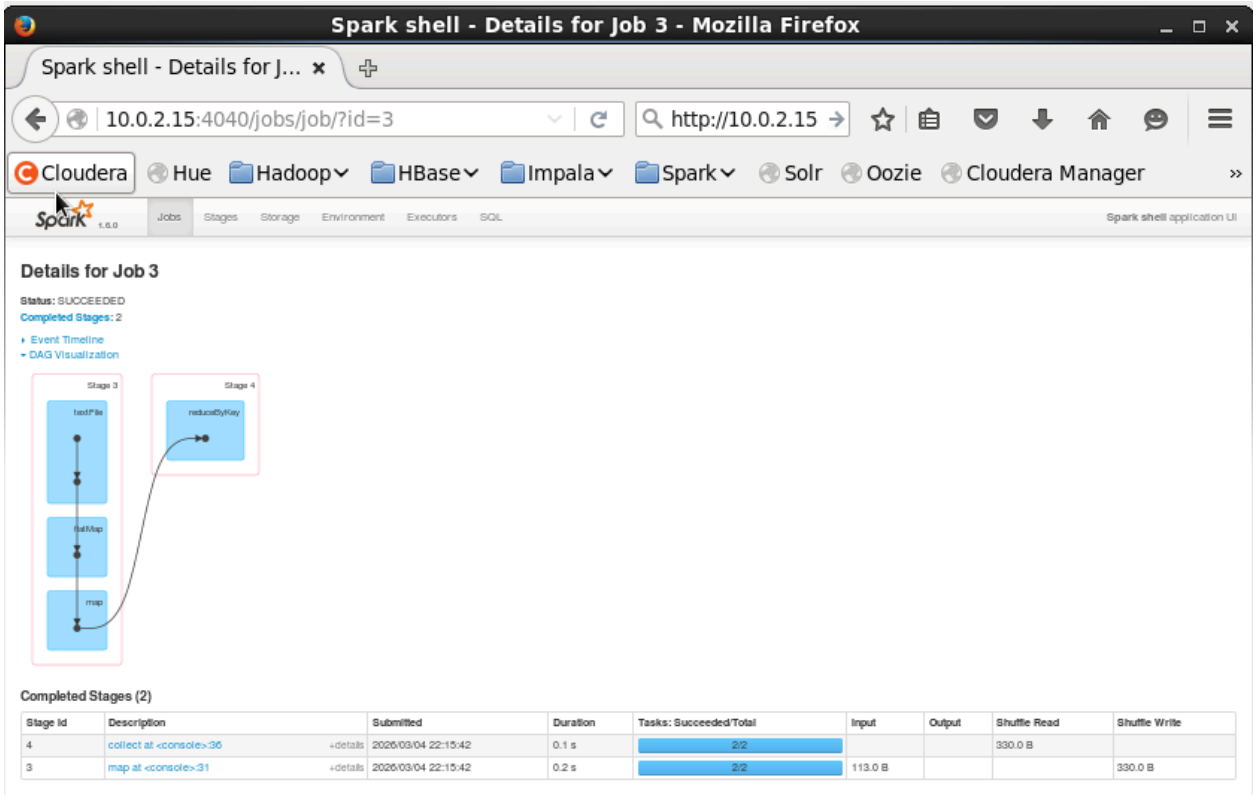
Command: val uniqueCountRDD = wordsRDD.map(word => (word,1))
Command: uniqueCountRDD.collect

```
scala> val uniqueCountRDD = wordsRDD.map(word => (word,1))
uniqueCountRDD: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[7] at map at <console>:31

scala> uniqueCountRDD.collect
res2: Array[(String, Int)] = Array((Rohit,1), (Sharma,1), (is,1), (a,1), (famous,1), (cricketer,1), (of,1), (India,1), (and,1
), (a,1), (very,1), (talented,1), (cricketer.,1))

scala>
```

Command: val finalRDD = uniqueCountRDD.reduceByKey(_ + _)
Command: finalRDD.collect

```
scala> val finalRDD = uniqueCountRDD.reduceByKey(_ + _)
finalRDD: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[8] at reduceByKey at <console>:33

scala> finalRDD.collect
res3: Array[(String, Int)] = Array((Rohit,1), (is,1), (Sharma,1), (cricketer,1), (cricketer.,1), (very,1), (a,2), (of,1), (fa
mous,1), (and,1), (India,1), (talented,1))

scala>
```




Spark shell - Details for Job 3 - Mozilla Firefox
Spark shell - Details for J...
10.0.2.15:4040/jobs/job/?id=3
http://10.0.2.15
Cloudera
Hue
Hadoop
HBase
Impala
Spark
Solr
Oozie
Cloudera Manager
Jobs
Stages
Storage
Environment
Executors
SQL
Spark shell application UI
Details for Job 3
Status: SUCCEEDED
Completed Stages: 2
Event Timeline
DAG Visualization
Stage 3
Stage 4
textFile
reduceByKey
flatMap
map
Completed Stages (2)
Stage Id
Description
Submitted
Duration
Tasks: Succeeded/Total
Input
Output
Shuffle Read
Shuffle Write
4
collect at <console>:36
+details
2026/03/04 22:15:42
0.1 s
2/2
330.0 B
3
map at <console>:31
+details
2026/03/04 22:15:42
0.2 s
2/2
113.0 B
330.0 B

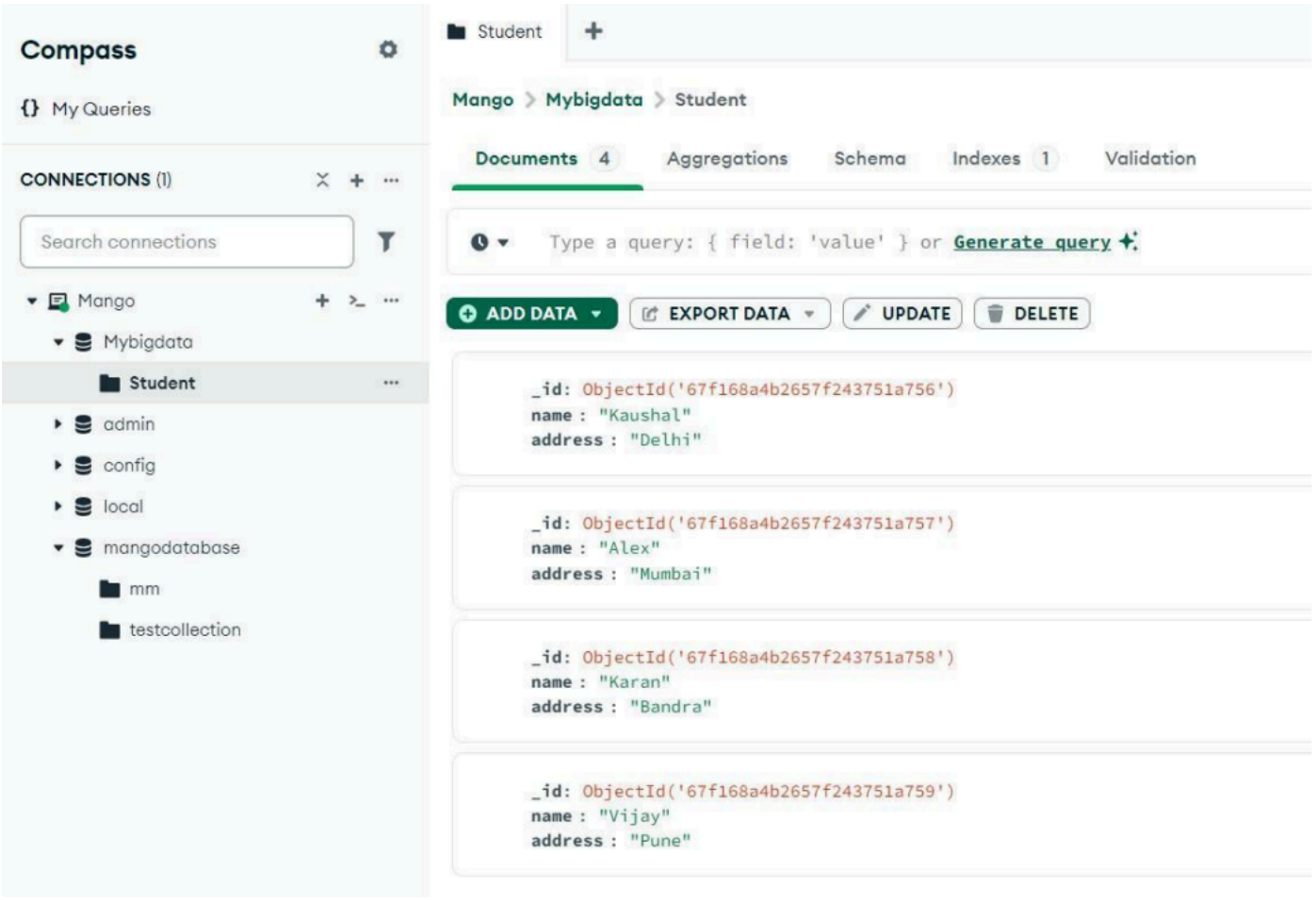
Practical No: 03
Implement an application that stores big data in MongoDB and manipulate it using Python

Code:

```
# First install using command prompt (NOT in Python file):
# pip install pymongo

import pymongo

# Connect to MongoDB server
myclient = pymongo.MongoClient("mongodb://localhost:27017/")

# Create or access database
mydb = myclient["Mybigdata"]

# Print all database names
print("Databases:", myclient.list_database_names())

# Create or access collection
mycol = mydb["Student"]

# Print all collections
print("Collections:", mydb.list_collection_names())

# Insert one document
mydict = {"name": "Kaushal", "address": "Delhi"}
x = mycol.insert_one(mydict)
print("Inserted ID (single):", x.inserted_id)

# Insert multiple documents
mylist = [
    {"name": "Alex", "address": "Mumbai"},
    {"name": "Karan", "address": "Bandra"},
    {"name": "Vijay", "address": "Pune"}
]

x = mycol.insert_many(mylist)
print("Inserted IDs (multiple):", x.inserted_ids)
```

Output:

```
= RESTART: C:/Users/suraj/AppData/Local/Programs/Python/Python312/MongoDB.py
###############
Name: Suraj Mohite
Roll No: A-414
###############
['admin', 'config', 'local', 'mangodatabase']
[]
```



Practical No: 04
Implement Apriori algorithm for market basket analysis.

Code:

```
# Install packages (run once)
install.packages("arules")
install.packages("arulesViz")
install.packages("RColorBrewer")
install.packages("plotly")

# Load libraries
library(arules)
library(arulesViz)
library(RColorBrewer)
library(plotly)

# Load dataset
data(Groceries)
Groceries

# Summary
summary(Groceries)
class(Groceries)

# Generate association rules
rules <- apriori(Groceries, parameter = list(supp = 0.02, conf = 0.2))
summary(rules)

# Inspect rules safely
inspect(head(rules, 10))

# Item frequency plot
itemFrequencyPlot(
  Groceries,
  topN = 20,
  col = brewer.pal(8, 'Pastel2'),
  main = 'Relative Item Frequency Plot',
  type = "relative",
  ylab = "Item Frequency (Relative)"
)

# Frequent itemsets (length = 2)
itemsets_2 <- apriori(
  Groceries,
  parameter = list(minlen = 2, maxlen = 2, support = 0.02),
  target = "frequent itemsets"
)
summary(itemsets_2)

# Safe inspect
if (length(itemsets_2) > 0) {
  inspect(head(itemsets_2, 10))
} else {
```

```
  print("No itemsets of length 2 found. Try lowering support.")
}

# Frequent itemsets (length = 3)
itemsets_3 <- apriori(
  Groceries,
  parameter = list(minlen = 3, maxlen = 3, support = 0.01),  # reduced support
  target = "frequent itemsets"
)
summary(itemsets_3)

# Safe inspect
if (length(itemsets_3) > 0) {
  inspect(head(itemsets_3, 10))
} else {
  print("No itemsets of length 3 found. Try lowering support further.")
}

# Convert rules to dataframe
rules_df <- as(rules, "data.frame")

# 3D interactive plot
plotly_graph <- plot_ly(
  data = rules_df,
  x = ~support,
  y = ~confidence,
  z = ~lift,
  text = ~paste("Rule:", rules),
  type = "scatter3d",
  mode = "markers",
  marker = list(size = 5, color = ~lift, colorscale = "Viridis", showscale = TRUE)
) %>%
  layout(
    title = "Interactive Graph of Association Rules",
    scene = list(
      xaxis = list(title = "Support"),
      yaxis = list(title = "Confidence"),
      zaxis = list(title = "Lift")
    )
  )

# Show plot
plotly_graph

# Top 10 rules by lift
top10subRules <- head(sort(rules, by = "lift"), 10)

# Plot graph
plot(
  top10subRules,
  method = "graph",
  engine = "htmlwidget"
)
```

Output:

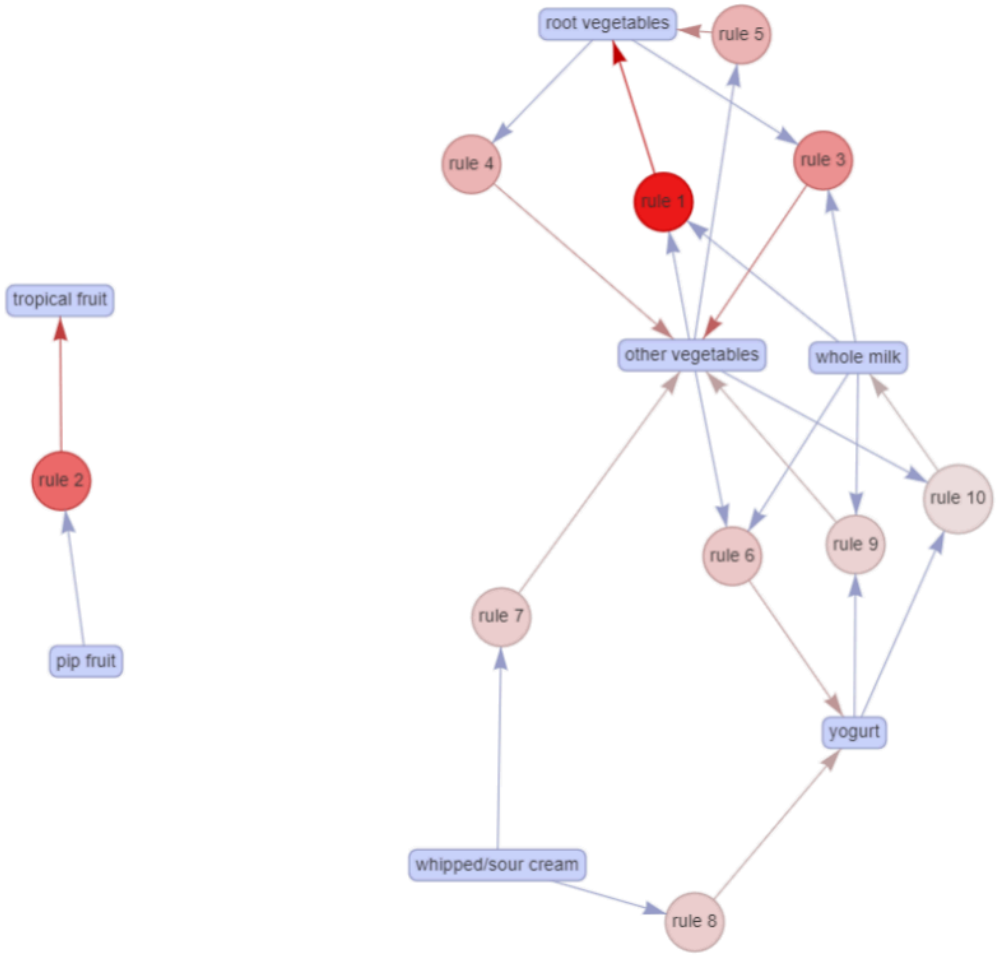
```
> # Summarize and examine the dataset
> summary(Groceries)
transactions as itemMatrix in sparse format with
 9835 rows (elements/itemsets/transactions) and
 169 columns (items) and a density of 0.02609146

most frequent items:
      whole milk other vegetables       rolls/buns             soda           yogurt
            2513             1903             1809             1715             1372
         (Other)
           34055
```

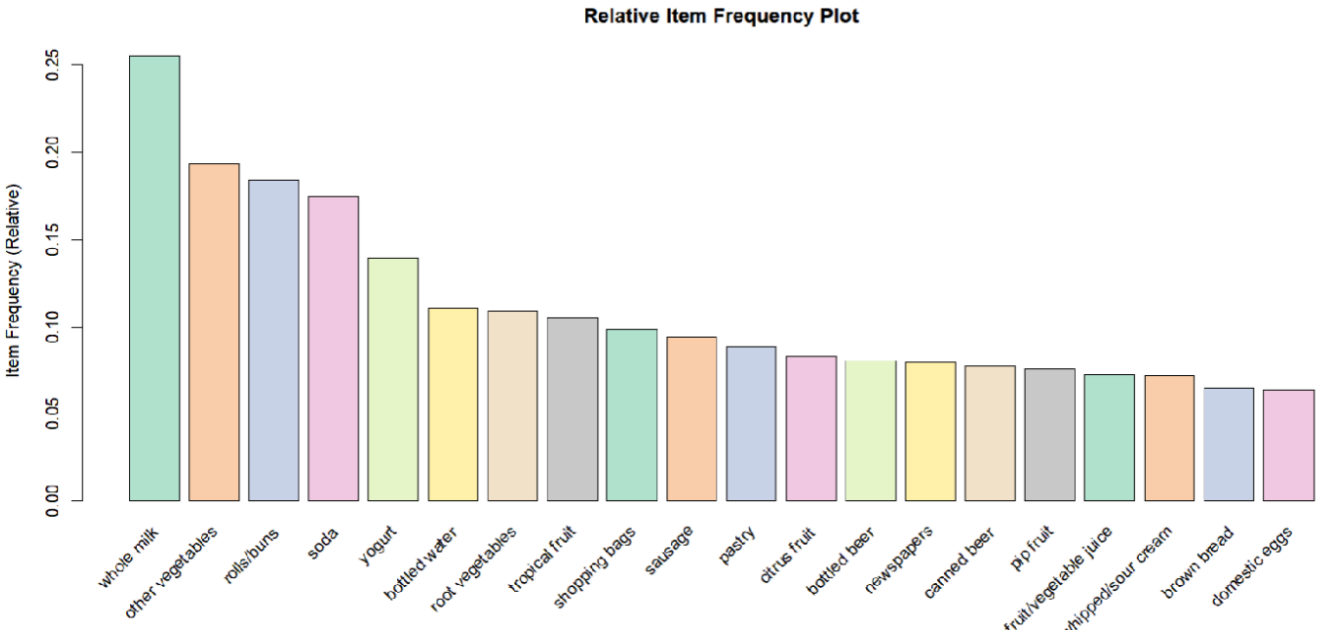
```
> summary(rules)
set of 73 rules
```

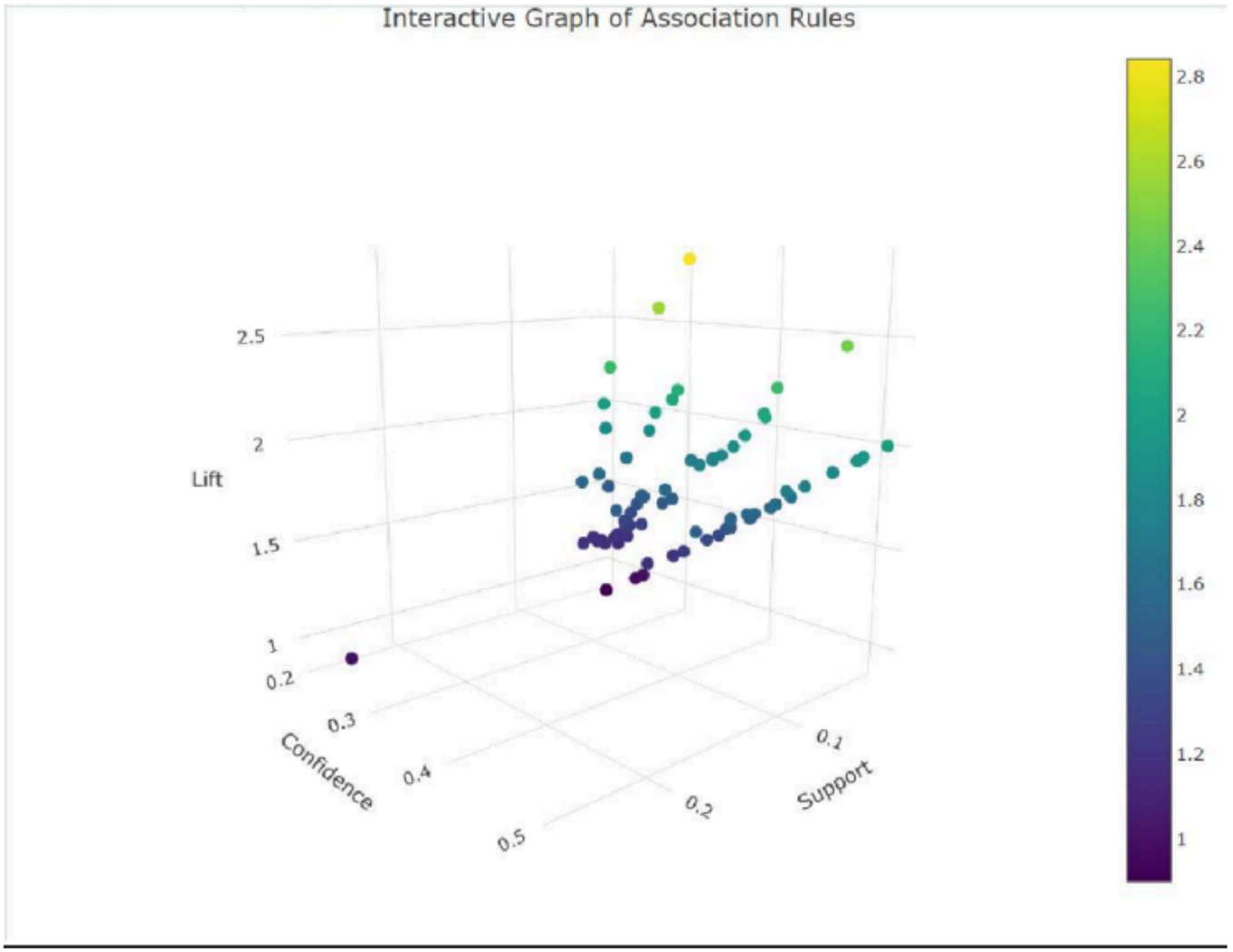
```
  > # Inspect the first 10 rules
  > inspect(rules[1:10])
      lhs                    rhs                support    confidence coverage   lift
  [1] {}                  => {whole milk}       0.25551601 0.2555160  1.00000000 1.0000000
  [2] {frozen vegetables} => {whole milk}       0.02043721 0.4249471  0.04809354 1.6630940
  [3] {beef}              => {whole milk}       0.02125064 0.4050388  0.05246568 1.5851795
  [4] {curd}              => {whole milk}       0.02613116 0.4904580  0.05327911 1.9194805
  [5] {pork}              => {other vegetables} 0.02165735 0.3756614  0.05765125 1.9414764
  [6] {pork}              => {whole milk}       0.02216573 0.3844797  0.05765125 1.5047187
  [7] {frankfurter}       => {whole milk}       0.02053889 0.3482759  0.05897306 1.3630295
  [8] {bottled beer}      => {whole milk}       0.02043721 0.2537879  0.08052872 0.9932367
  [9] {brown bread}       => {whole milk}       0.02521607 0.3887147  0.06487036 1.5212930
  [10] {margarine}        => {whole milk}       0.02419929 0.4131944  0.05856634 1.6170980
      count
  [1] 2513
  [2]  201
  [3]  209
  [4]  257
  [5]  213
  [6]  218
  [7]  202
  [8]  201
  [9]  248
  [10] 238
```

```
> inspect(itemsets_2[1:10])
      items                                 support    count
[1]  {whole milk, frozen vegetables}  0.02043721 201
[2]  {beef, whole milk}               0.02125064 209
[3]  {whole milk, curd}               0.02613116 257
[4]  {pork, other vegetables}         0.02165735 213
[5]  {pork, whole milk}               0.02216573 218
[6]  {frankfurter, whole milk}        0.02053889 202
[7]  {whole milk, bottled beer}       0.02043721 201
[8]  {whole milk, brown bread}        0.02521607 248
[9]  {whole milk, margarine}          0.02419929 238
[10] {other vegetables, butter}       0.02003050 197
```

```
> inspect(itemsets_3)
    items                                                 support    count
[1] {root vegetables, other vegetables, whole milk} 0.02318251 228
[2] {other vegetables, whole milk, yogurt}          0.02226741 219
```


root vegetables
rule 5
rule 4
rule 3
rule 1
tropical fruit
other vegetables
whole milk
rule 2
rule 10
rule 6
rule 9
rule 7
pip fruit
yogurt
whipped/sour cream
rule 8


Relative Item Frequency Plot
Item Frequency (Relative)
0.00
0.05
0.10
0.15
0.20
0.25
whole milk
other vegetables
rolls/buns
soda
yogurt
bottled water
root vegetables
tropical fruit
shopping bags
sausage
pastry
citrus fruit
bottled beer
newspapers
canned beer
pip fruit
brown bread
domestic eggs


Interactive Graph of Association Rules
Lift
2.5
2
1.5
1
Confidence
0.2
0.3
0.4
0.5
Support
0.1
0.2
2.8
2.6
2.4
2.2
2
1.8
1.6
1.4
1.2
1

Practical No: 05
Configure the Hive and implement the application in Hive.

```
# Start the Hive command-line interface
-> hive

# Create a new database named 'demo'
-> CREATE DATABASE demo;

# List all available databases
-> SHOW DATABASES;

# Show detailed information about the 'demo' database
-> DESCRIBE DATABASE EXTENDED demo;

# Create an internal table 'suraj' in the 'demo' database
->CREATE TABLE IF NOT EXISTS demo.suraj (
  id INT,
  name STRING,
  salary FLOAT
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ',';


# Describe the structure of the 'suraj' table
->DESCRIBE demo.suraj;

# Load local data from the specified path into the 'suraj' table
-> LOAD DATA LOCAL INPATH '/home/cloudera/hive-demo/emp-details'
INTO TABLE demo.suraj;

# Query all records from the 'suraj' table
-> SELECT * FROM demo.suraj;

# Create a partitioned internal table 'student'
->CREATE TABLE student (
  id INT,
  name STRING,
  age INT,
  institute STRING
)
PARTITIONED BY (course STRING)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ',';
```

```
# Describe the structure of the 'student' table
-> DESCRIBE student;

# Load local data into the 'student' table under partition "IT"
-> LOAD DATA LOCAL INPATH '/home/suraj/student-details'
INTO TABLE student
PARTITION (course='IT');

# Create an external table 'studentlist' that points to a location in HDFS
-> CREATE EXTERNAL TABLE studentlist (
  id INT,
  name STRING,
  salary FLOAT
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
LOCATION '/HiveDirectory';
```

Practical No: 07

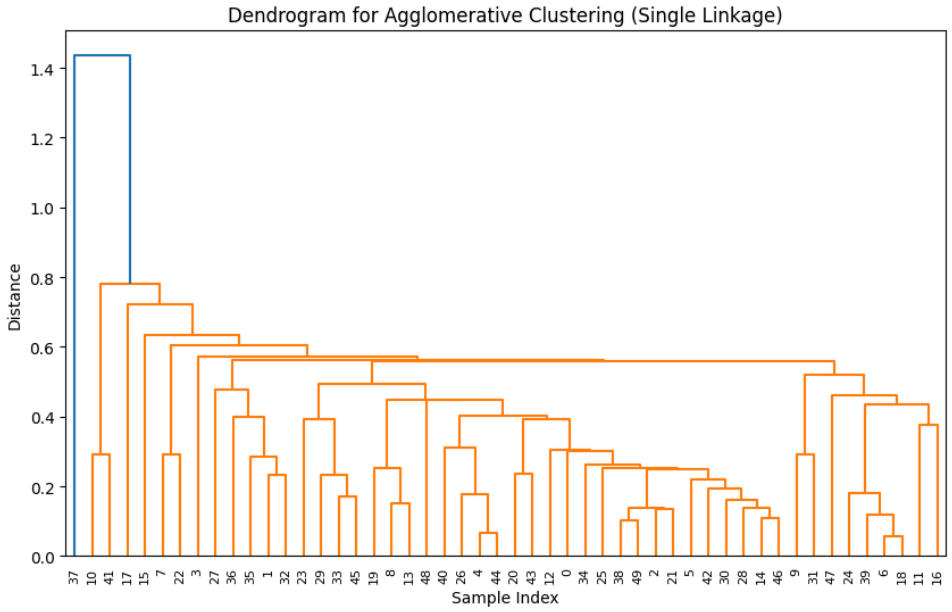
Aim: Implement the agglomerative hierarchy clustering algorithm

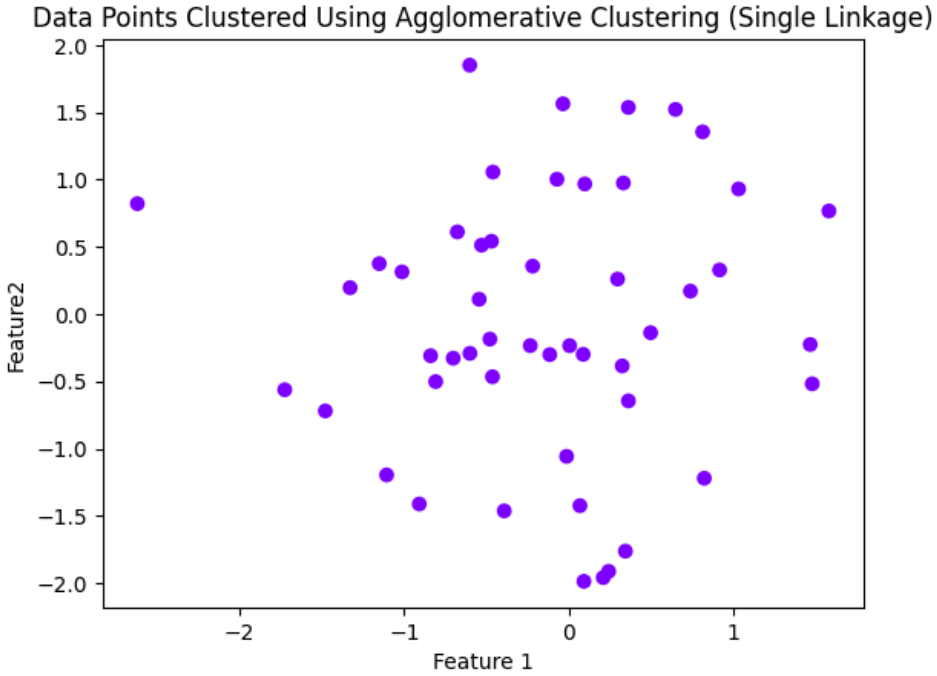
A) Aim: Implement Agglomerative Hierarchy clustering algorithm with Single Linkage Method

Code:

```
# Agglomerative Clustering with Single Linkage Method
# Step 1: Import Required
import numpy as np
import matplotlib.pyplot as plt
from scipy.cluster.hierarchy import dendrogram, linkage
from scipy.spatial.distance import pdist
from scipy.cluster.hierarchy import fcluster
# Step 2: Generate Sample Data
# Generate random data points for clustering
np.random.seed(42)
data = np.random.randn(50, 2)
# Step 3: Compute the Linkage Matrix (using single linkage method)
Z = linkage(data, method='single') # Change to 'single' for single linkage
# Step 4: Visualize the Dendrogram
plt.figure(figsize=(10, 6))
dendrogram(Z)
plt.title('Dendrogram for Agglomerative Clustering (Single Linkage)')
plt.xlabel('Sample Index')
plt.ylabel('Distance')
plt.show()
# Step 5: Form Clusters based on a Distance Threshold
# Form flat clusters by cutting the dendrogram at a specified distance
max_distance = 1.5
clusters = fcluster(Z, max_distance, criterion='distance')
# Step 6: Plot the clustered data
plt.scatter(data[:, 0], data[:, 1], c=clusters, cmap='rainbow')
plt.title('Data Points Clustered Using Agglomerative Clustering (Single Linkage)')
plt.xlabel('Feature 1')
plt.ylabel('Feature2')
plt.show()
```

output:





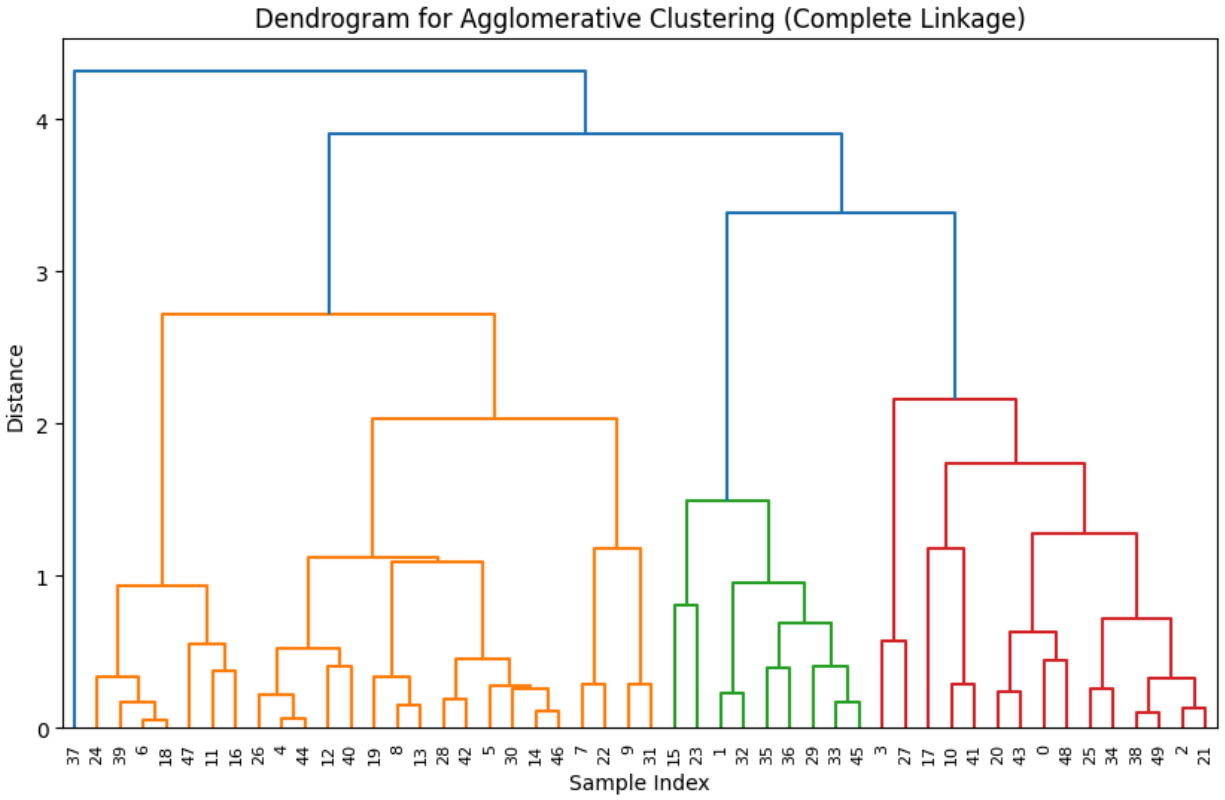
B) Aim: Implement Agglomerative Hierarchy clustering algorithm with Complete Linkage

Code:

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.cluster.hierarchy import dendrogram, linkage, fcluster

# Step 2: Generate Sample Data
np.random.seed(42)
data = np.random.randn(50, 2)

# Step 3: Compute the Linkage Matrix (Complete Linkage)
```

```
Z = linkage(data, method='complete')

# Step 4: Visualize the Dendrogram
plt.figure(figsize=(10, 6))
dendrogram(Z)
plt.title('Dendrogram for Agglomerative Clustering (Complete Linkage)')
plt.xlabel('Sample Index')
plt.ylabel('Distance')
plt.show()

# Step 5: Form Clusters based on a Distance Threshold
max_distance = 1.5  # You can adjust this to control number of clusters
clusters = fcluster(Z, max_distance, criterion='distance')

# Step 6: Plot the clustered data
plt.scatter(data[:, 0], data[:, 1], c=clusters, cmap='rainbow')
plt.title('Data Points Clustered Using Agglomerative Clustering (Complete
Linkage)')
plt.xlabel('Feature 1')
plt.ylabel('Feature 2')
plt.show()
```

Output:

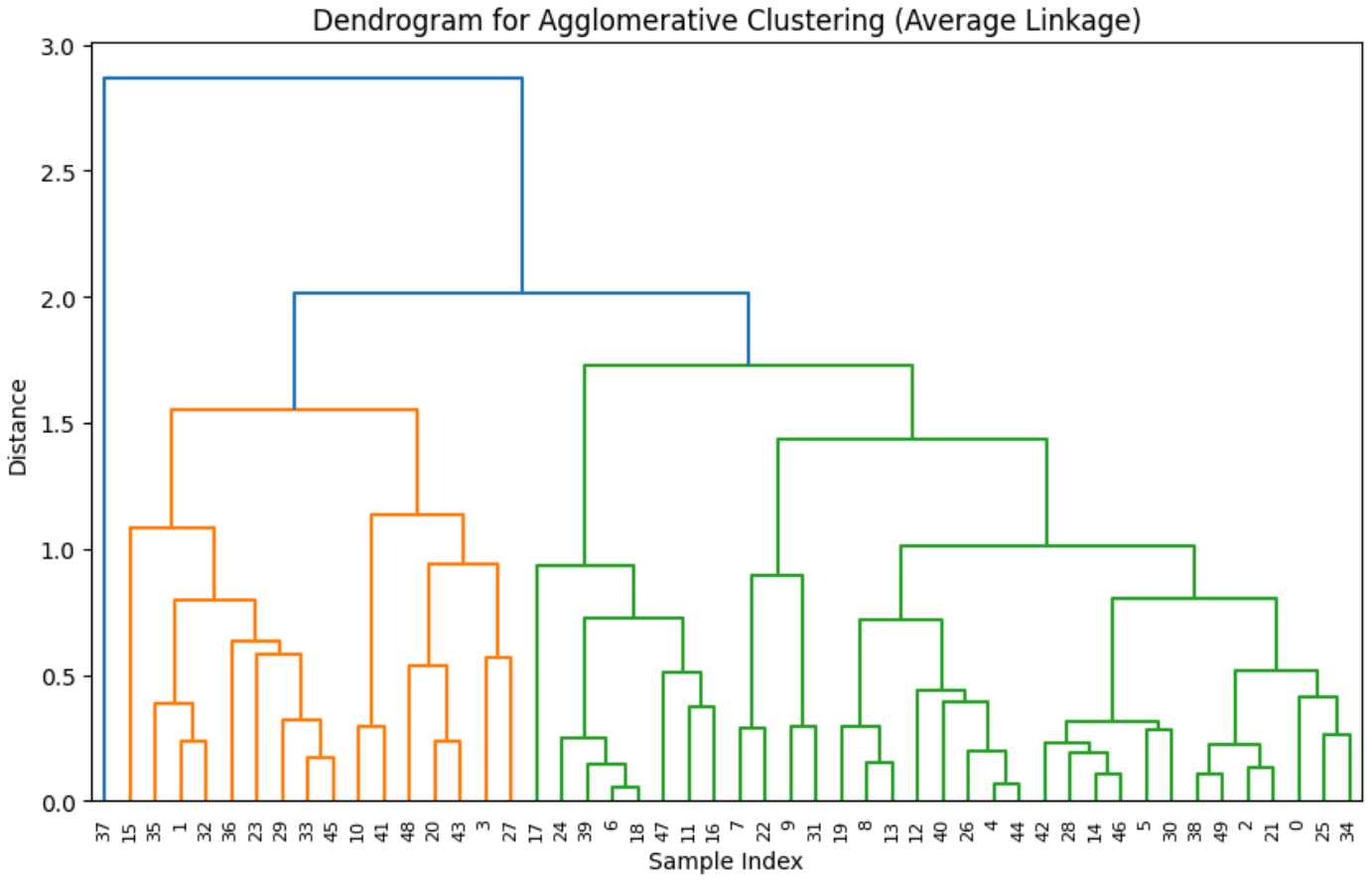


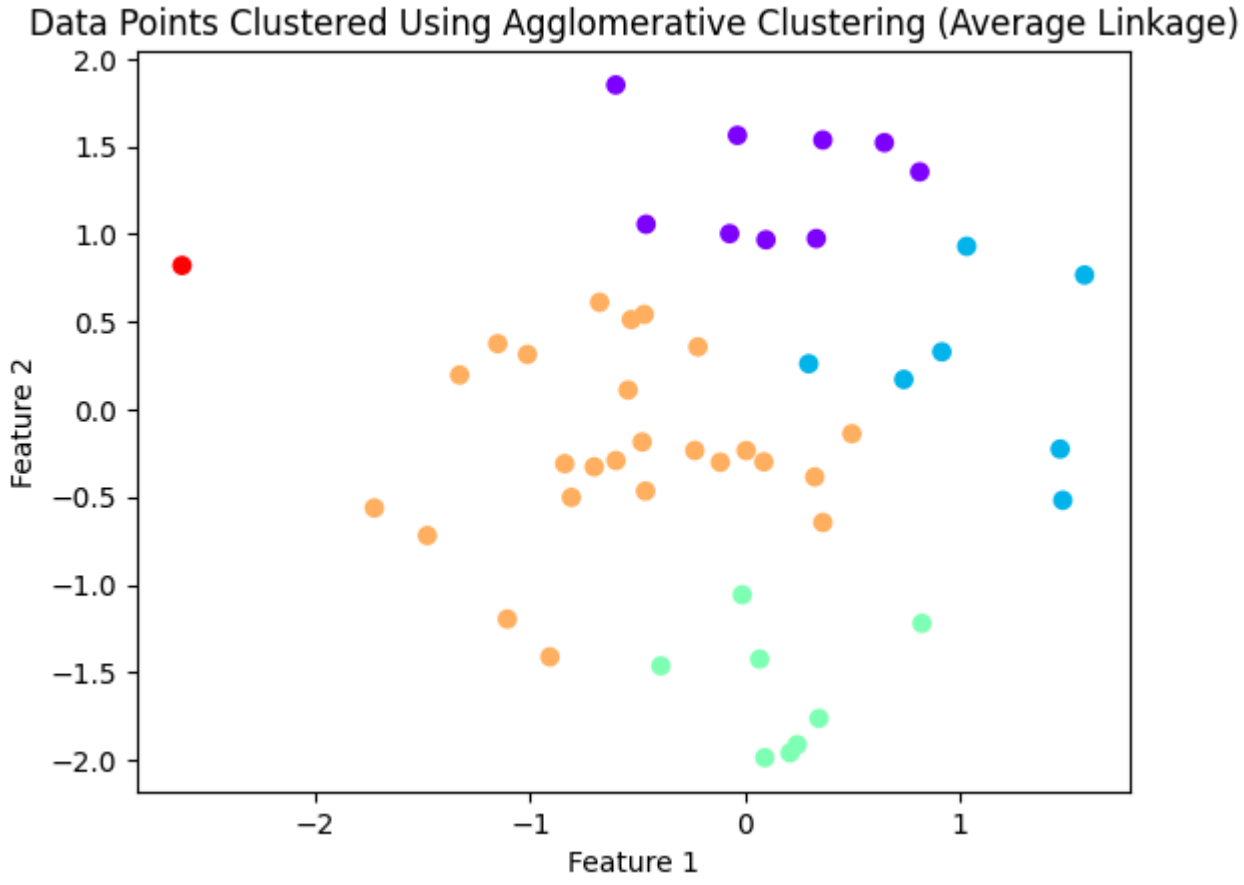
C) Aim: Implement Agglomerative Hierarchy clustering algorithm with Average Linkage Method

Code:

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.cluster.hierarchy import dendrogram, linkage, fcluster

# Step 2: Generate Sample Data
np.random.seed(42)
data = np.random.randn(50, 2)

# Step 3: Compute the Linkage Matrix (Average Linkage)
Z = linkage(data, method='average')

# Step 4: Visualize the Dendrogram
plt.figure(figsize=(10, 6))
dendrogram(Z)
plt.title('Dendrogram for Agglomerative Clustering (Average Linkage)')
plt.xlabel('Sample Index')
plt.ylabel('Distance')
plt.show()

# Step 5: Form Clusters based on a Distance Threshold
max_distance = 1.5  # You can adjust this to control the number of clusters
clusters = fcluster(Z, max_distance, criterion='distance')

# Step 6: Plot the clustered data
plt.scatter(data[:, 0], data[:, 1], c=clusters, cmap='rainbow')
plt.title('Data Points Clustered Using Agglomerative Clustering (Average Linkage)')
plt.xlabel('Feature 1')
plt.ylabel('Feature 2')
```

```
plt.show()
```


Dendrogram for Agglomerative Clustering (Average Linkage)
Distance
Sample Index
3.0
2.5
2.0
1.5
1.0
0.5
0.0
37 15 35 1 32 36 23 29 33 45 10 41 48 20 43 3 27 17 24 39 6 18 47 11 16 7 22 9 31 19 8 13 12 40 26 4 44 42 28 14 46 5 30 38 49 2 21 0 25 34

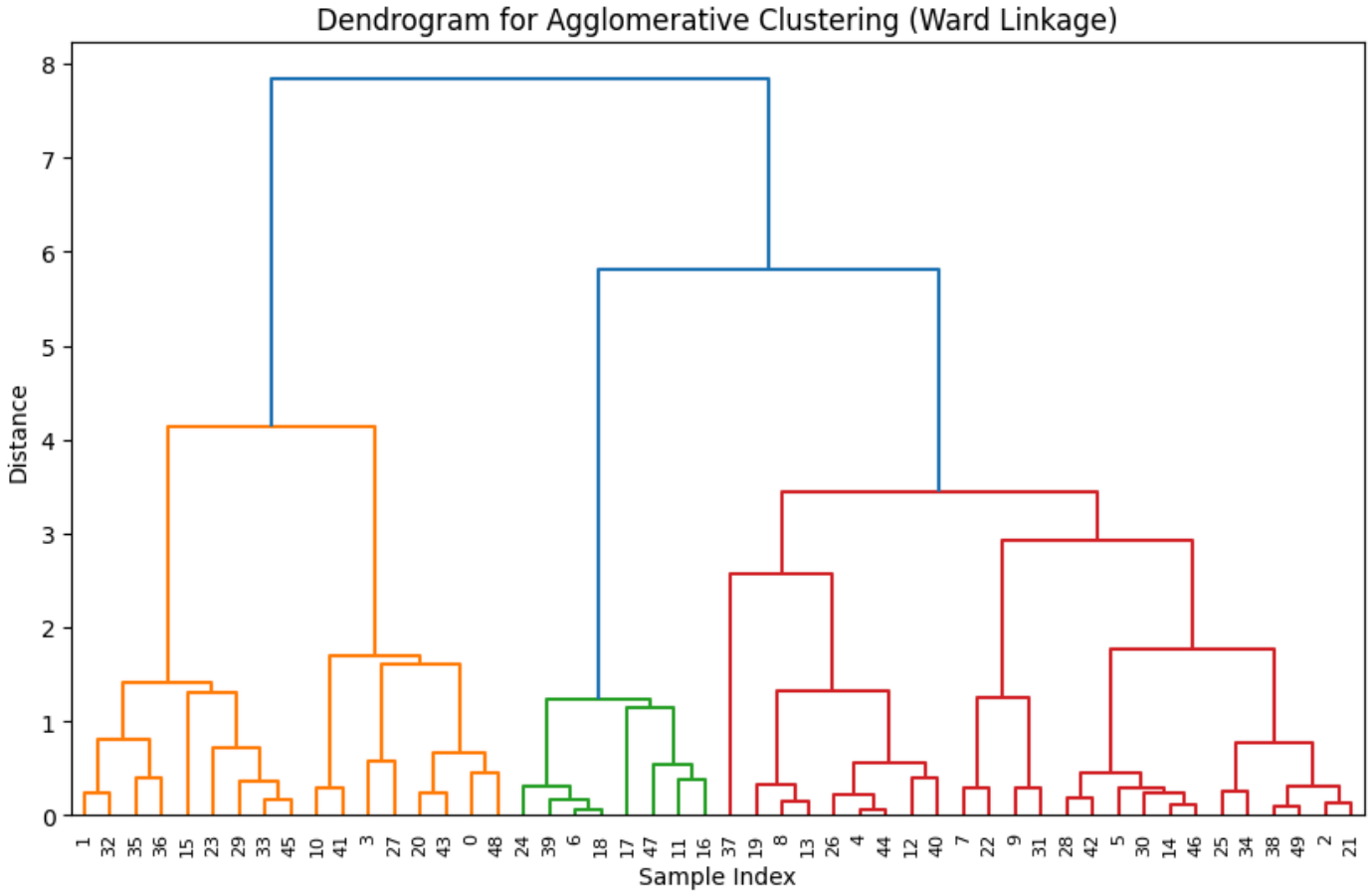

Data Points Clustered Using Agglomerative Clustering (Average Linkage)
Feature 2
Feature 1
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
−2
−1
0
1

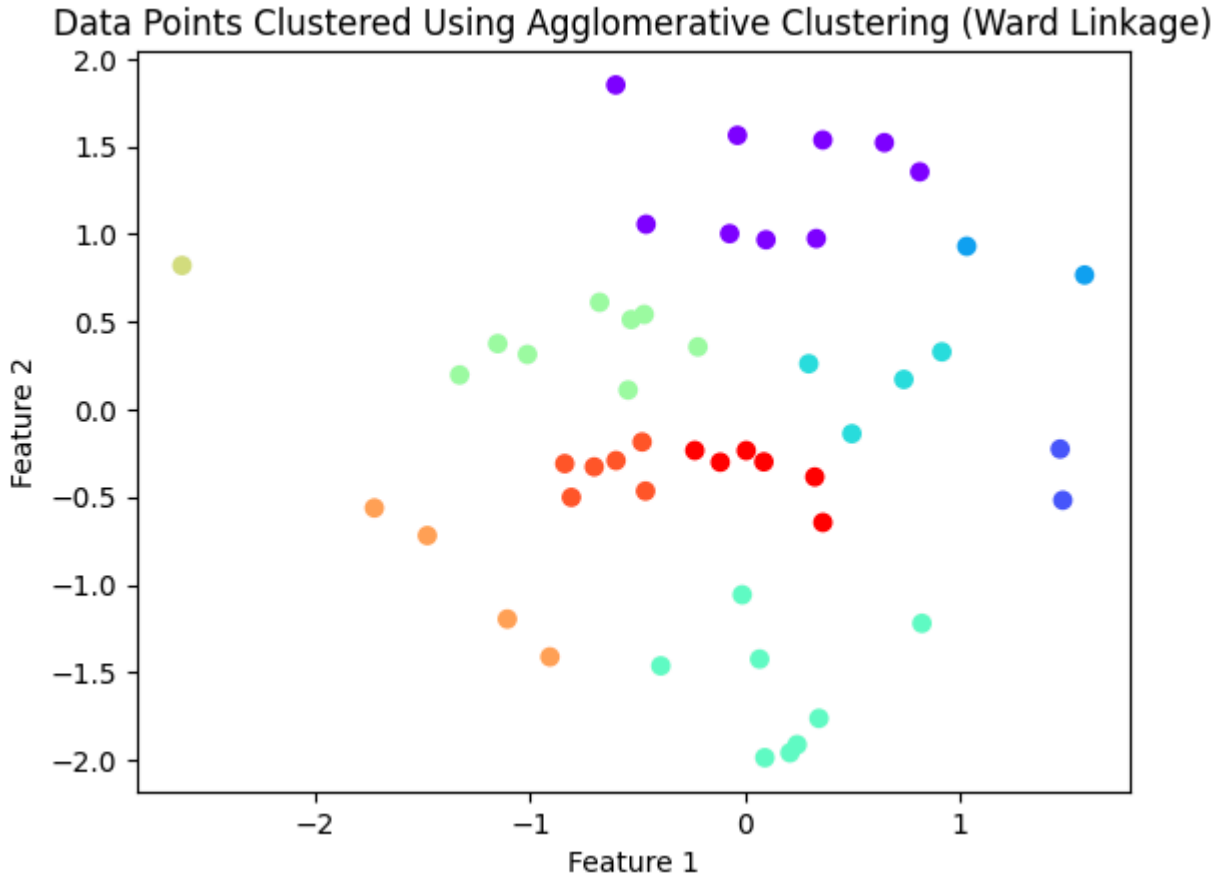
D) Aim: Implement Agglomerative Hierarchy clustering algorithm with Ward Linkage Method

Code:

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.cluster.hierarchy import dendrogram, linkage, fcluster

# Step 2: Generate Sample Data
np.random.seed(42)
data = np.random.randn(50, 2)

# Step 3: Compute the Linkage Matrix (Ward Linkage)
Z = linkage(data, method='ward')

# Step 4: Visualize the Dendrogram
plt.figure(figsize=(10, 6))
dendrogram(Z)
plt.title('Dendrogram for Agglomerative Clustering (Ward Linkage)')
plt.xlabel('Sample Index')
plt.ylabel('Distance')
plt.show()

# Step 5: Form Clusters based on a Distance Threshold
max_distance = 1.5
clusters = fcluster(Z, max_distance, criterion='distance')

# Step 6: Plot the clustered data
plt.scatter(data[:, 0], data[:, 1], c=clusters, cmap='rainbow')
plt.title('Data Points Clustered Using Agglomerative Clustering (Ward Linkage)')
plt.xlabel('Feature 1')
plt.ylabel('Feature 2')
plt.show()
```

Output:


Dendrogram for Agglomerative Clustering (Ward Linkage)
Distance
Sample Index
0
1
2
3
4
5
6
7
8
1 32 35 36 15 23 29 33 45 10 41 3 27 20 43 0 48 24 39 6 18 17 47 11 16 37 19 8 13 26 4 44 12 40 7 22 9 31 28 42 5 30 14 46 25 34 38 49 2 21

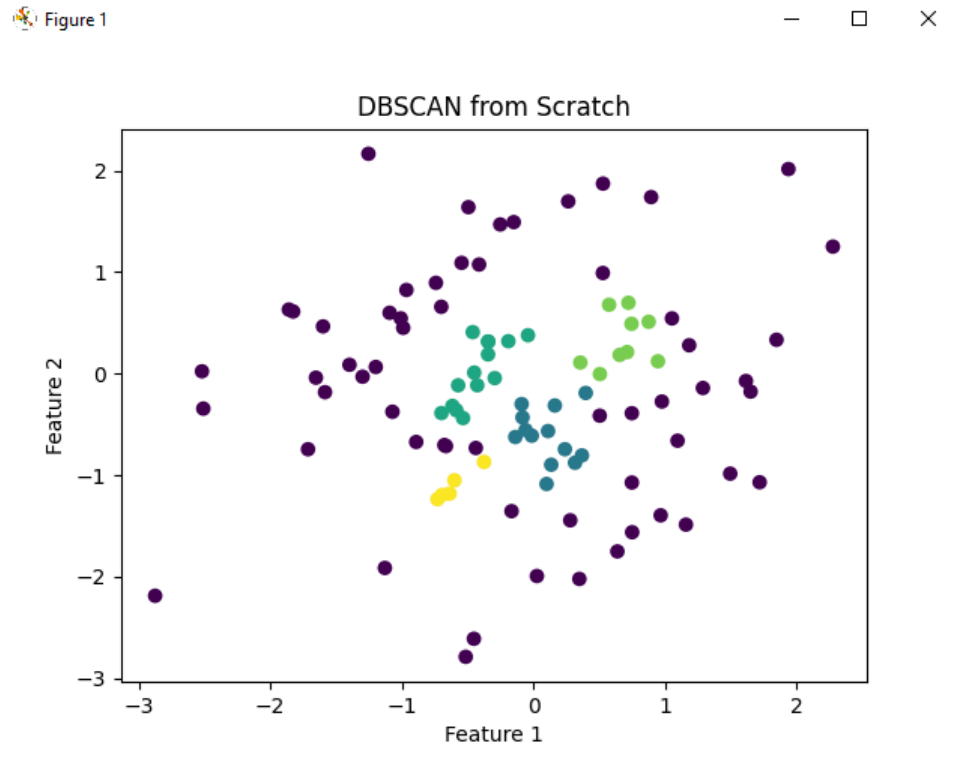

Data Points Clustered Using Agglomerative Clustering (Ward Linkage)
Feature 1
Feature 2
−2
−1
0
1
−2.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0

Practical No: 07
Aim: Implement the DBSCAN density-based clustering algorithm.
code:

```
# Import necessary libraries
import numpy as np
from collections import deque
import matplotlib.pyplot as plt

# Function to calculate Euclidean distance between two points
def euclidean_distance(p1, p2):
    return np.sqrt(np.sum((p1 - p2) ** 2))

# Function to find all neighbors within 'eps' distance of a given point
def region_query(X, point_idx, eps):
    neighbors = []
    for i, point in enumerate(X):
        if euclidean_distance(X[point_idx], point) <= eps:
            neighbors.append(i)
    return neighbors

# Function to expand a cluster starting from a core point
def expand_cluster(X, labels, point_idx, neighbors, cluster_id, eps, min_pts):
    labels[point_idx] = cluster_id
    queue = deque(neighbors)

    while queue:
        current_point = queue.popleft()

        if labels[current_point] == -1:
            labels[current_point] = cluster_id
        elif labels[current_point] == 0:
            labels[current_point] = cluster_id
            current_neighbors = region_query(X, current_point, eps)

            if len(current_neighbors) >= min_pts:
                queue.extend(current_neighbors)

# Main DBSCAN algorithm implementation
def dbscan(X, eps, min_pts):
    labels = np.zeros(len(X))  # 0 means unvisited
    cluster_id = 0

    for point_idx in range(len(X)):
        if labels[point_idx] != 0:
            continue
        neighbors = region_query(X, point_idx, eps)
```

```
        if len(neighbors) < min_pts:
            labels[point_idx] = -1  # Mark as noise
        else:
            cluster_id += 1
            expand_cluster(X, labels, point_idx, neighbors, cluster_id, eps,
min_pts)
    return labels

# Generate 100 random 2D points
X = np.random.randn(100, 2)

# Set DBSCAN parameters
eps = 0.3  # Maximum distance to consider as neighbor
min_pts = 5  # Minimum number of points to form a dense region

# Run DBSCAN algorithm
labels = dbscan(X, eps, min_pts)

# Plotting the clustered data
plt.scatter(X[:, 0], X[:, 1], c=labels, cmap='viridis')
plt.title("DBSCAN from Scratch")
plt.xlabel("Feature 1")
plt.ylabel("Feature 2")
plt.show()
```

Output:



Practical No: 9
Implement SVM classification techniques.
Code:

```
install.packages("caret")
library('caret')
heart <- read.csv("C:\\Users\\ADMIN\\Downloads\\heart.csv",sep=",")
str(heart)
head(heart)
colSums(is.na(heart))
heart$Ca[is.na(heart$Ca)] <- median(heart$Ca, na.rm = TRUE)
heart$Thal[is.na(heart$Thal)] <- names(sort(table(heart$Thal), decreasing = TRUE))[1]
factor_vars <- c("Sex","ChestPain","Fbs","RestECG","ExAng","Slope","Thal")
heart[factor_vars] <- lapply(heart[factor_vars], as.factor)
heart$AHD <- as.factor(heart$AHD)
set.seed(123)
intrain <- createDataPartition(heart$AHD, p = 0.7, list = FALSE)
training <- heart[intrain, ]
testing <- heart[-intrain, ]
trctrl <- trainControl(method = "cv", number = 10)
grid <- expand.grid(
  C = c(0.01, 0.05, 0.1, 0.25, 0.5, 1, 2, 5)
)
svm_Linear_Grid <- train(
  AHD ~ .,
  data = training,
  method = "svmLinear",
  trControl = trctrl,
  preProcess = c("center", "scale"),
  tuneGrid = grid
)
pred <- predict(svm_Linear_Grid, testing)
confusionMatrix(pred, testing$AHD)
plot(svm_Linear_Grid)
```

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

marketbasket.R

```
10
11  # Load dataset
12  heart <- read.csv("C:\\Users\\Shrut\\Downloads\\heart.csv", sep = ",")
```

48:22 (Top Level) R Script

Console Terminal Background Jobs

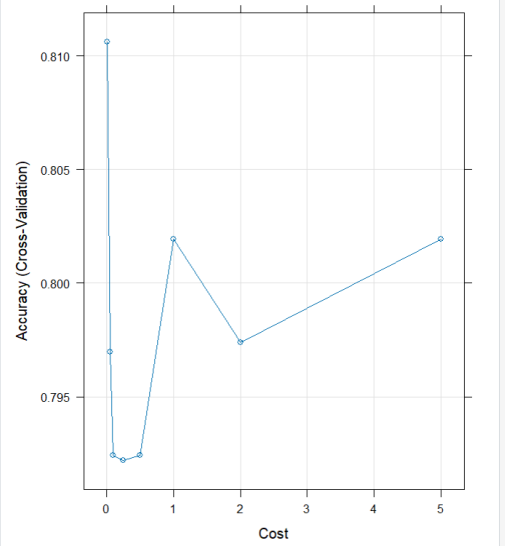
R 4.5.3 · ~/

```
> # Load libraries
> library(caret)
> library(e1071)
> library(kernlab)
>
> # Load dataset
> heart <- read.csv("C:\\Users\\Shrut\\Downloads\\heart.csv", sep = ",")
>
> # Check structure
> str(heart)
'data.frame':   303 obs. of  14 variables:
 $ age     : int  63 37 41 56 57 57 56 44 52 57 ...
 $ sex     : int  1 1 0 1 0 1 0 1 1 1 ...
 $ cp      : int  3 2 1 1 0 0 1 1 2 2 ...
 $ trestbps: int  145 130 130 120 120 140 140 120 172 150 ...
 $ chol    : int  233 250 204 236 354 192 294 263 199 168 ...
 $ fbs     : int  1 0 0 0 0 0 0 0 1 0 ...
 $ restecg : int  0 1 0 1 1 1 0 1 1 1 ...
 $ thalach : int  150 187 172 178 163 148 153 173 162 174 ...
 $ exang   : int  0 0 0 0 1 0 0 0 0 0 ...
 $ oldpeak : num  2.3 3.5 1.4 0.8 0.6 0.4 1.3 0 0.5 1.6 ...
 $ slope   : int  0 0 2 2 2 1 1 2 2 2 ...
 $ ca      : int  0 0 0 0 0 0 0 0 0 0 ...
 $ thal    : int  1 2 2 2 2 1 2 3 3 2 ...
 $ target  : int  1 1 1 1 1 1 1 1 1 1 ...
> head(heart)
  age sex cp trestbps chol fbs restecg thalach exang oldpeak slope ca thal target
1  63   1  3      145  233   1       0     150     0     2.3     0  0    1      1
2  37   1  2      130  250   0       1     187     0     3.5     0  0    2      1
3  41   0  1      130  204   0       0     172     0     1.4     2  0    2      1
4  56   1  1      120  236   0       1     178     0     0.8     2  0    2      1
5  57   0  0      120  354   0       1     163     1     0.6     2  0    2      1
6  57   1  0      140  192   0       1     148     0     0.4     1  0    1      1
>
```

Environment History Connections Tutorial

Files Plots Packages Help Viewer Presentation

Zoom Export Publish



Search ENG IN 71% 11:26 PM 4/8/2026

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

marketbasket.R

```
10
11  # Load dataset
12  heart <- read.csv("C:\\Users\\Shrut\\Downloads\\heart.csv", sep = ",")
```

48:22 (Top Level) R Script

Console Terminal Background Jobs

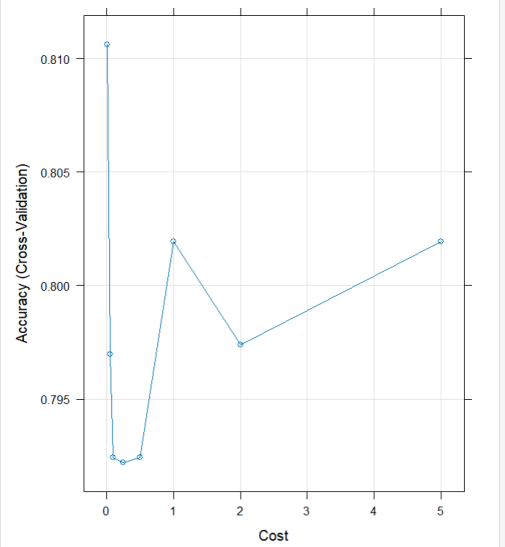
R 4.5.3 · ~/

```
>
> # Check missing values
> colSums(is.na(heart))
     age      sex       cp trestbps     chol      fbs  restecg  thalach    exang  oldpeak    slope       ca
       0        0        0        0        0        0        0        0        0        0        0        0
    thal   target
       0        0
>
> factor_vars <- c("sex","cp","fbs","restecg","exang","slope","thal")
> heart[factor_vars] <- lapply(heart[factor_vars], as.factor)
>
> # Target variable
> heart$target <- as.factor(heart$target)
>
> set.seed(123)
> intrain <- createDataPartition(heart$target, p = 0.7, list = FALSE)
> training <- heart[intrain, ]
> testing  <- heart[-intrain, ]
>
> trctrl <- trainControl(method = "cv", number = 10)
>
> grid <- expand.grid(C = c(0.01, 0.05, 0.1, 0.25, 0.5, 1, 2, 5))
>
> svm_Linear_Grid <- train(
+   target ~ .,
+   data = training,
+   method = "svmLinear",
+   trControl = trctrl,
+   preProcess = c("center", "scale"),
+   tuneGrid = grid
+ )
>
> pred <- predict(svm_Linear_Grid, testing)
>
```

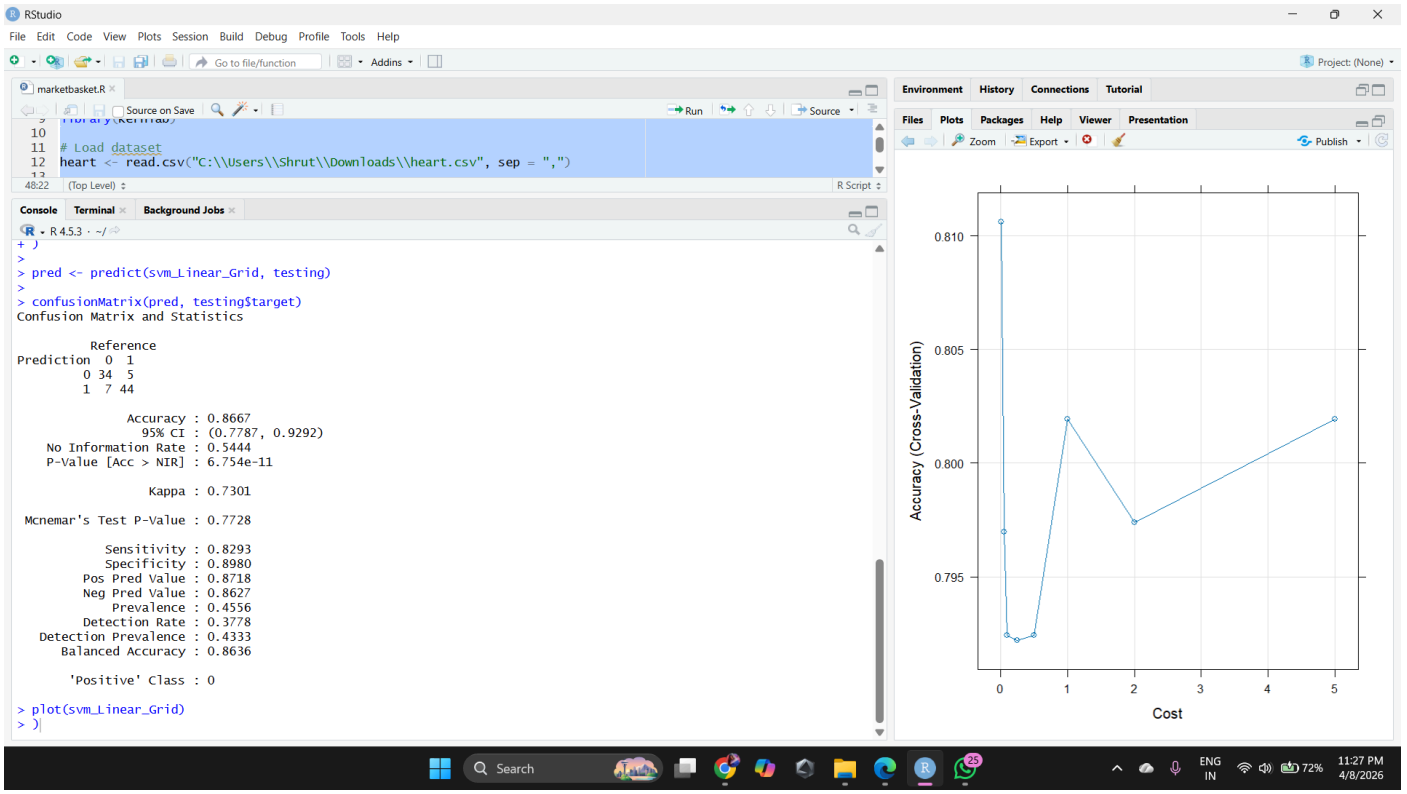
Environment History Connections Tutorial

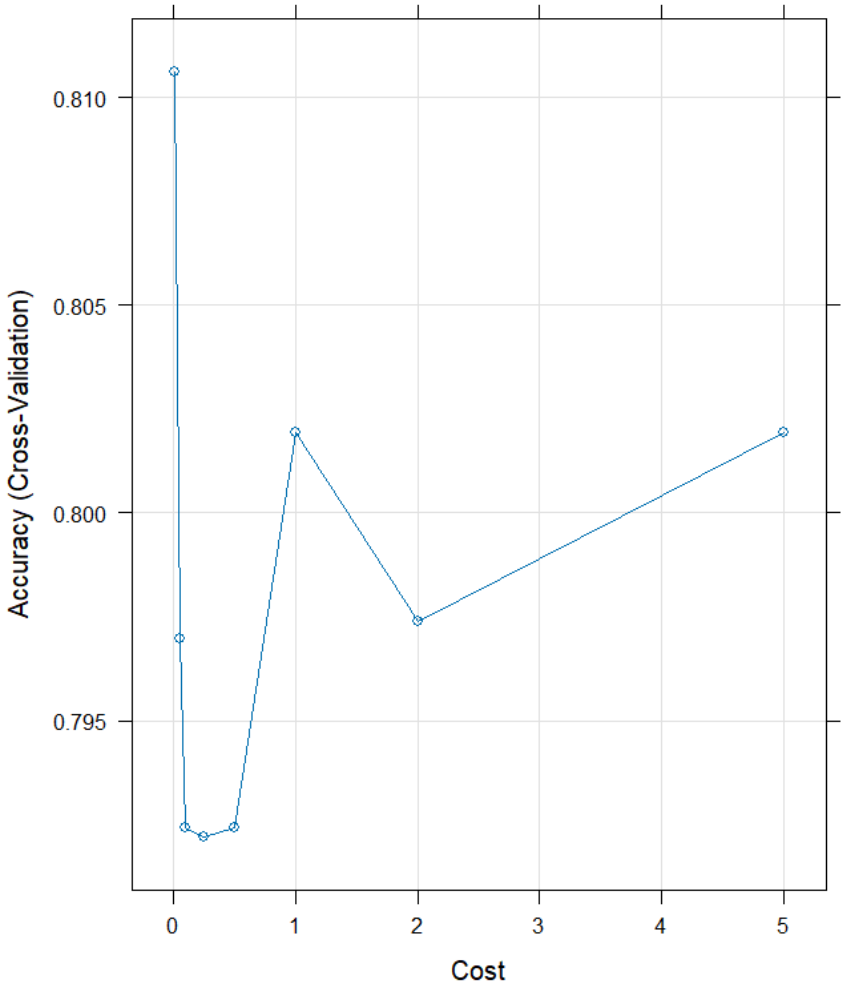
Files Plots Packages Help Viewer Presentation

Zoom Export Publish



Search ENG IN 71% 11:26 PM 4/8/2026





Load libraries

```
library(caret)
library(ggplot2)

# Load dataset
heart <- read.csv("C:\\Users\\Shrut\\Downloads\\heart.csv", sep=",")

# Check structure
str(heart)
colnames(heart)

# ---- Fix column names (standard dataset assumption) ----
# target = output variable

# Handle missing values (only if present)
if("ca" %in% colnames(heart)){
  heart$ca[is.na(heart$ca)] <- median(heart$ca, na.rm = TRUE)
}

if("thal" %in% colnames(heart)){
  heart$thal[is.na(heart$thal)] <- names(sort(table(heart$thal), decreasing = TRUE))[1]
}

# Convert categorical columns (adjusted names)
factor_vars <- c("sex","cp","fbs","restecg","exang","slope","thal")

# Keep only existing columns
factor_vars <- factor_vars[factor_vars %in% colnames(heart)]

# Convert to factor
heart[factor_vars] <- lapply(heart[factor_vars], as.factor)

# Target column
heart$target <- as.factor(heart$target)

# ---- Train-Test Split ----
set.seed(123)
intrain <- createDataPartition(heart$target, p = 0.7, list = FALSE)
training <- heart[intrain, ]
testing  <- heart[-intrain, ]

# ---- Train Control ----
trctrl <- trainControl(method = "cv", number = 10)

# ---- Hyperparameter Grid ----
grid <- expand.grid(C = c(0.01, 0.1, 1, 5))
```

```
# ---- Train Model ----
svm_Linear_Grid <- train(
  target ~ oldpeak + thalach,   # thalach = MaxHR
  data = training,
  method = "svmLinear",
  trControl = trctrl,
  preProcess = c("center", "scale"),
  tuneGrid = grid
)

# ---- Predictions ----
pred <- predict(svm_Linear_Grid, testing)
print(confusionMatrix(pred, testing$target))

# ---- Model Plot ----
plot(svm_Linear_Grid)

# ---- Decision Boundary Visualization ----

# Create grid
x_range <- range(training$oldpeak, na.rm = TRUE)
y_range <- range(training$thalach, na.rm = TRUE)

grid_df <- expand.grid(
  oldpeak = seq(x_range[1], x_range[2], length.out = 100),
  thalach = seq(y_range[1], y_range[2], length.out = 100)
)

# Predict
grid_df$pred <- predict(svm_Linear_Grid, newdata = grid_df)
grid_df$pred_num <- as.numeric(grid_df$pred)

# Plot
ggplot(training, aes(oldpeak, thalach, color = target)) +
  geom_point(alpha = 0.7) +
  geom_contour(
    data = grid_df,
    aes(z = pred_num),
    breaks = 1.5,
    color = "black"
  ) +
  labs(title = "Linear SVM Hyperplane (Oldpeak vs Thalach)")
```

Practical No: 10

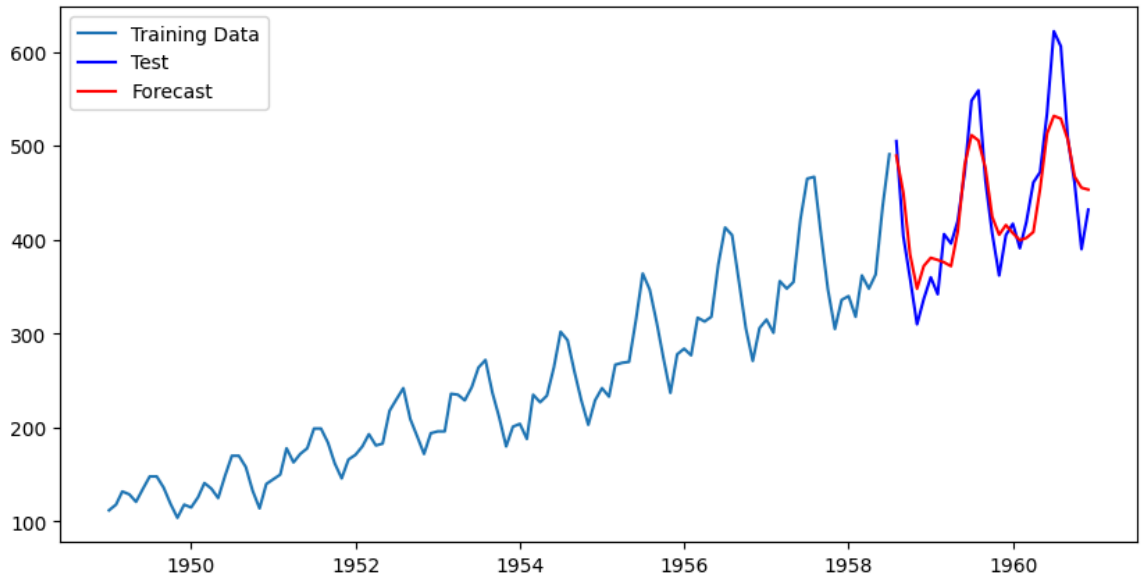
Aim: Create an ARIMA Model for Time Series Forecasting in Python

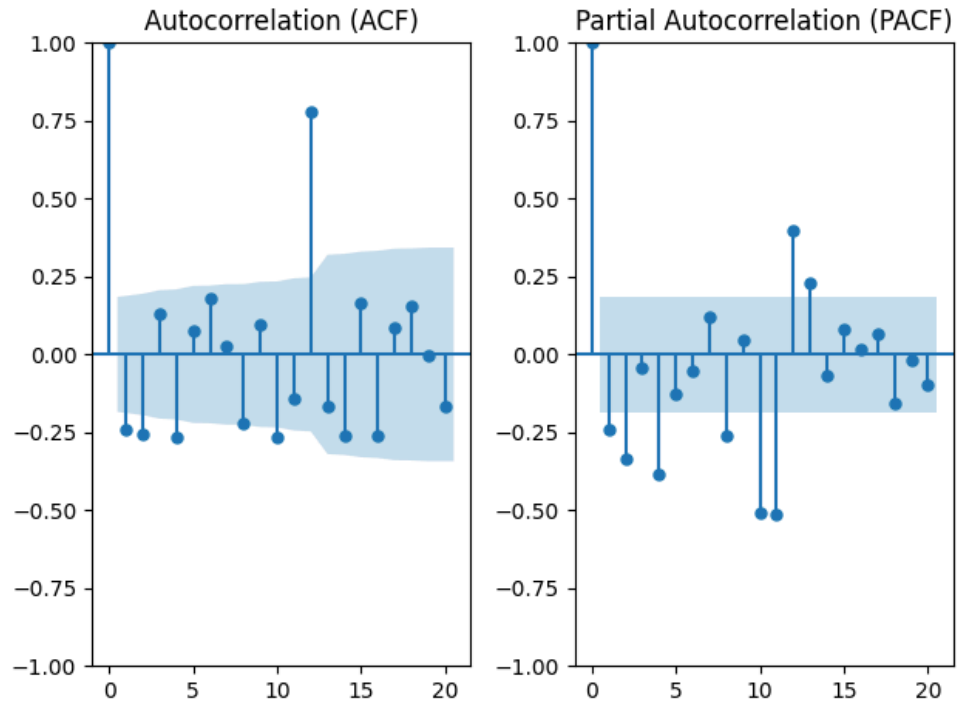
Code:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from statsmodels.tsa.arima.model import ARIMA
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
import matplotlib.pyplot as plt

# Load dataset
df = pd.read_csv(r"/content/airline-passengers.csv", parse_dates=['month'], index_col='month') # Corrected 'Month' to
'month'
ts = df['total_passengers']
# Split data into training and testing sets
train_size = int(len(ts) * 0.8)
train, test = ts[:train_size], ts[train_size:]
# Fit ARIMA model
model = ARIMA(train, order=(8,2,7)) # (p,d,q) parameters
model_fit = model.fit()
# Forecast on test data
forecast = model_fit.forecast(steps=len(test))
# Plot results
plt.figure(figsize=(12, 6))
plt.figure(figsize=(10,5))
plt.plot(train, label='Training Data')
plt.plot(test, label='Test', color='blue')
plt.plot(test.index, forecast, label='Forecast', color='red')
plt.legend()
plt.show()

# Calculate differenced data for ACF/PACF plots
data_diff = train.diff().diff().dropna()

# ACF Plot
plt.subplot(1, 2, 1)
plot_acf(data_diff, lags=20, ax=plt.gca())
plt.title("Autocorrelation (ACF)")

# PACF Plot
plt.subplot(1, 2, 2)
plot_pacf(data_diff, lags=20, ax=plt.gca())
plt.title("Partial Autocorrelation (PACF)")
```

```
plt.tight_layout()
plt.show()
# Calculate MSE
mse = mean_squared_error(test, forecast)
# Calculate RMSE
rmse = np.sqrt(mse)
print(f"Mean Squared Error (MSE): {mse}")
print(f"Root Mean Squared Error (RMSE): {rmse}")
```

Output:






```
Mean Squared Error (MSE): 1332.3657882549583
Root Mean Squared Error (RMSE): 36.501586106016795
```

Practical No: 08
Aim:Implement a program in python to demonstrate how to compute tf-idf values of words from a corpus

Code:

```
from sklearn.feature_extraction.text import TfidfVectorizer
d0 = 'Geeks for geeks'
d1 = 'Geeks'
d2 = 'r2j'
string = [d0, d1, d2]
tfidf = TfidfVectorizer()
result = tfidf.fit_transform(string)
print('\nidf values:')
for ele1, ele2 in zip(tfidf.get_feature_names_out(), tfidf.idf_):
    print(ele1, ':', ele2)
print('\nWord indexes:')
print(tfidf.vocabulary_)
print('\ntf-idf value:')
print(result)
print('\ntf-idf values in matrix form:')
print(result.toarray())
```

Output:

```
...
    idf values:
    for : 1.6931471805599454

    Word indexes:
    geeks : 1.2876820724517808

    Word indexes:
    r2j : 1.6931471805599454

    Word indexes:
    {'geeks': 1, 'for': 0, 'r2j': 2}

    tf-idf value:
    <Compressed Sparse Row sparse matrix of dtype 'float64'
            with 4 stored elements and shape (3, 3)>
      Coords        Values
      (0, 1)        0.8355915419449176
      (0, 0)        0.5493512310263033
      (1, 1)        1.0
      (2, 2)        1.0

    tf-idf values in matrix form:
    [[0.54935123 0.83559154 0.        ]
     [0.         1.         0.        ]
     [0.         0.         1.        ]]
```