Data Warehousing and Mining Lab (DWML)-

Case study on OLAP & ETL process tools

Data Warehousing - OLAP

Online Analytical Processing Server (OLAP) is based on the multidimensional data model. It allows managers, and analysts to get an insight of the information through fast, consistent, and interactive access to information. This chapter covers the types of OLAP, operations on OLAP, differences between OLAP, and statistical databases and OLTP.

>>Types of OLAP Servers

We have four types of OLAP servers –

- Relational OLAP (ROLAP)
- Multidimensional OLAP (MOLAP)
- Hybrid OLAP (HOLAP)
- Specialized SQL Servers

>Relational OLAP

ROLAP servers are placed between relational back-end server and client front-end tools. To store and manage warehouse data, ROLAP uses relational or extended-relational DBMS.

- □ ROLAP includes the following –
- □ Implementation of aggregation navigation logic.
- □ Optimization for each DBMS back end.
- □ Additional tools and services.

>Multidimensional OLAP

MOLAP uses array-based multidimensional storage engines for multidimensional views of data. With multidimensional data stores, the storage utilization may be low if the data set is sparse. Therefore, many MOLAP server use two levels of data storage representation to handle dense and sparse data sets.

>Hybrid OLAP

Hybrid OLAP is a combination of both ROLAP and MOLAP. It offers higher scalability of ROLAP and faster computation of MOLAP. HOLAP servers allow large data volumes of detailed information. The aggregations are stored separately in the MOLAP store.

>Specialized SQL Servers

Specialized SQL servers provide advanced query language and query processing support for SQL queries over star and snowflake schemas in a read-only environment.

OLAP Operations Since OLAP servers are based on a multidimensional view of data, we will discuss OLAP operations in multidimensional data.

Here is the list of OLAP operations −

- □ Roll-up
- □ Drill-down
- □ Slice and dice
- □ Pivot (rotate)

>>Top OLAP Marketing Tools

You can use OLAP tools to analyze large volumes of multidimensional data from different perspectives. They make it easy to filter, analyze, and visualize key data insights. These tools are often part of a Business Intelligence Suite.

OLAP marketing tools should have the following features:

1. The ability to analyze large volumes of (big) data
2. The ability to perform analytical operations
3. A high degree of interactivity
4. Fast response times
5. Different types of data visualizations
6. The ability to analyze why things happen

Some OLAP tools used in marketing include:

>IBM Cognos

IBM Cognos is a web-based reporting and analytical tool to help you understand your organizational data. It's used to view or create detailed business reports, analyze data, and help you make effective business decisions.

>MicroStrategy

MicroStrategy is a business analytics platform that helps enterprises build and deploy analytics and mobile apps to transform their business. The MicroStrategy platform provides interactive dashboards, highly formatted reports, ad hoc queries, and automated report distribution. The software's ROLAP architecture is a key differentiator from other vendors who offer full-featured solutions.

>Palo OLAP Server

Palo is a MOLAP (Multidimensional Online Analytical Processing) server typically used as a BI tool for controlling and budgeting. It is a Jedox AG product. Palo enables multiple users to share one centralized data storage. It works with real-time data. Data can then be consolidated or written back with the help of multidimensional queries. Palo stores run-time data in its memory to give faster data access to users.

>Sisense

Sisense is an agile business intelligence (BI) solution that provides advanced tools to manage big data in marketing analytics. It helps you simplify complex data and transform it into powerful analytic apps to give you a more comprehensive understanding of your data.

>icCube

icCube owns a business intelligence software that offers an end-to-end BI solution. This is great for software companies looking to embed data analytics, visualization, and reporting into their product. icCube sells an online analytical processing server that is implemented in Java as per J2EE standards. It's an in-memory OLAP server and is compatible with any data source that holds its data in tabular form.

>SAP NetWeaver Business Warehouse

SAP NetWeaver Business Warehouse provides a high-performance infrastructure that helps you evaluate and interpret data. It provides reporting, analysis, and interpretation of business data quickly and in line with market needs.

>Oracle Business Intelligence Enterprise Edition (OBIEE)

Oracle Business Intelligence Enterprise Edition helps customers discover new data insights and make faster business decisions by offering interactive dashboards, powerful operational reporting, and real-time alerts. It reduces the total cost of ownership and increases return on investment for the entire organization.

>Apache Kylin

Apache Kylin is an open-source, distributed Analytical Data Warehouse for Big Data. It provides an SQL interface and MOLAP combined with Hadoop and Spark to support large data. In addition, Kylin reduces query processing time and quickly filters billions of data rows.

>Final Thoughts

Businesses continuously need to plan, analyze, and report on sales and marketing activities to maximize efficiency. OLAP applications can help increase the productivity of business managers, developers, marketing analysts, and whole organizations. In addition, they can also help you transform data into actionable insights.

What is ETL

ETL stands for **Extract Transform and Load**. ETL combines all the three database functions into one tool to fetch data from one database and place it into another database.

>Extract: Extract is the process of fetching (reading) the information from the database. At this stage, data is collected from multiple or different types of sources.

>Transform: Transform is the process of converting the extracted data from its previous form into the required form. Data can be placed into another database. Transformation can occur by using rules or lookup tables or by combining the data with other data.

>Load: Load is the process of writing the data into the target database.

>>ETL is used to integrate the data with the help of three steps **Extract, Transform, and Load**, and it is used to blend the data from multiple sources. It is often used to build a **data warehouse**.In the ETL process, data is extracted from the source system and converted into a format that can be examined and stored into a **data warehouse** or any other system. ETL is an alternative but a related approach which is designed to push processing down to the database to improve the performance.

>>Types of ETL Tools

ETL tools can be categorized into the following main types:

>Batch ETL Tools

In this type of ETL tool, batch processing is used to acquire data from the source systems. The data is extracted, transformed, and loaded into the repository in batches of ETL jobs. It's a cost-effective method because it uses limited resources in a time-bound way.

>Real-Time ETL Tools

Data is extracted, cleansed, enriched, and loaded to the target system in real-time ETL tools. These tools offer you faster access to information and improve time to insights.

As the need to gather and analyze the data in the shortest possible time has augmented, these ETL tools are becoming more popular among businesses.

>On-Premise ETL Tools

Many companies operate legacy systems that have both the data and the repository configured on-premise. The main reason behind such an implementation is data security. That's why companies prefer having an ETL tool deployed on-site.

>Cloud ETL Tools

As the name suggests, these tools are deployed on the cloud as various cloud-based applications form an essential part of enterprise architecture. Companies opt for cloud ETL tools to manage data transfer from these applications. Cloud-based ETL tools let businesses leverage flexibility and agility in the ETL process.

>Oracle

Oracle is the industry-leading database. It offers a wide range of choice of Data Warehouse solutions for both on-premises and in the cloud. It helps to optimize customer experiences by increasing operational efficiency.

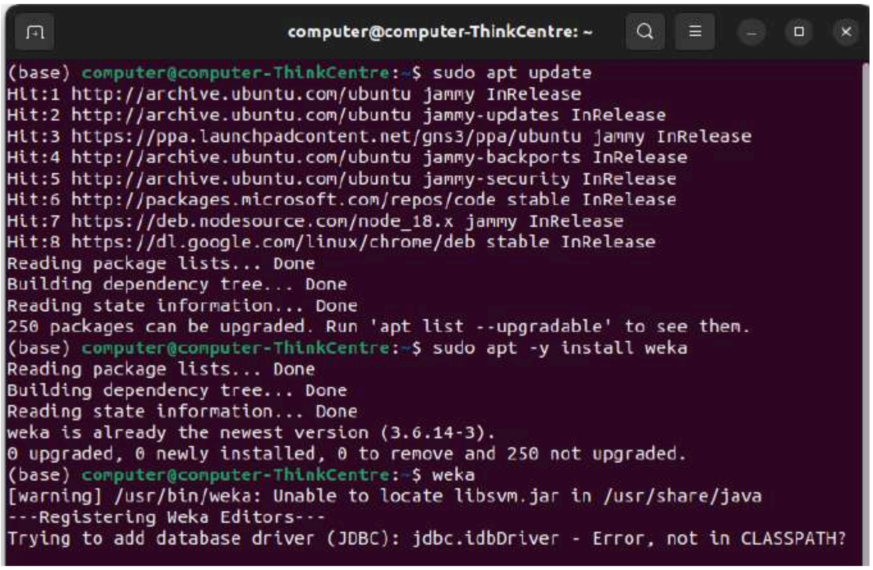
STEPS:

1. Installing WEKA

   Open Terminal

   > sudo apt update

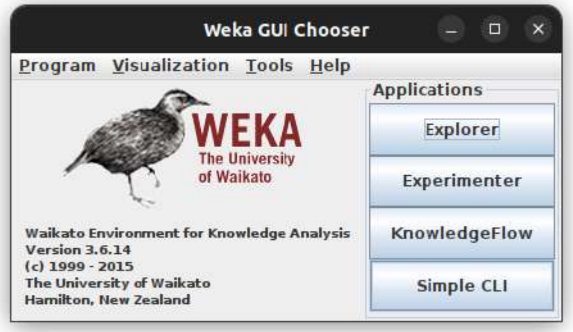
   > sudo apt -y install weka



```
(base) computer@computer-ThinkCentre:~$ sudo apt update
Hit:1 http://archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:3 https://ppa.launchpadcontent.net/gns3/ppa/ubuntu jammy InRelease
Hit:4 http://archive.ubuntu.com/ubuntu jammy-backports InRelease
Hit:5 http://archive.ubuntu.com/ubuntu jammy-security InRelease
Hit:6 http://packages.microsoft.com/repos/code stable InRelease
Hit:7 https://deb.nodesource.com/node_18.x jammy InRelease
Hit:8 https://dl.google.com/linux/chrome/deb stable InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
250 packages can be upgraded. Run 'apt list --upgradable' to see them.
(base) computer@computer-ThinkCentre:~$ sudo apt -y install weka
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
weka is already the newest version (3.6.14-3).
0 upgraded, 0 newly installed, 0 to remove and 250 not upgraded.
(base) computer@computer-ThinkCentre:~$ weka
[warning] /usr/bin/weka: Unable to locate libsvm.jar in /usr/share/java
---Registering Weka Editors---
Trying to add database driver (JDBC): jdbc.idbDriver - Error, not in CLASSPATH?
```

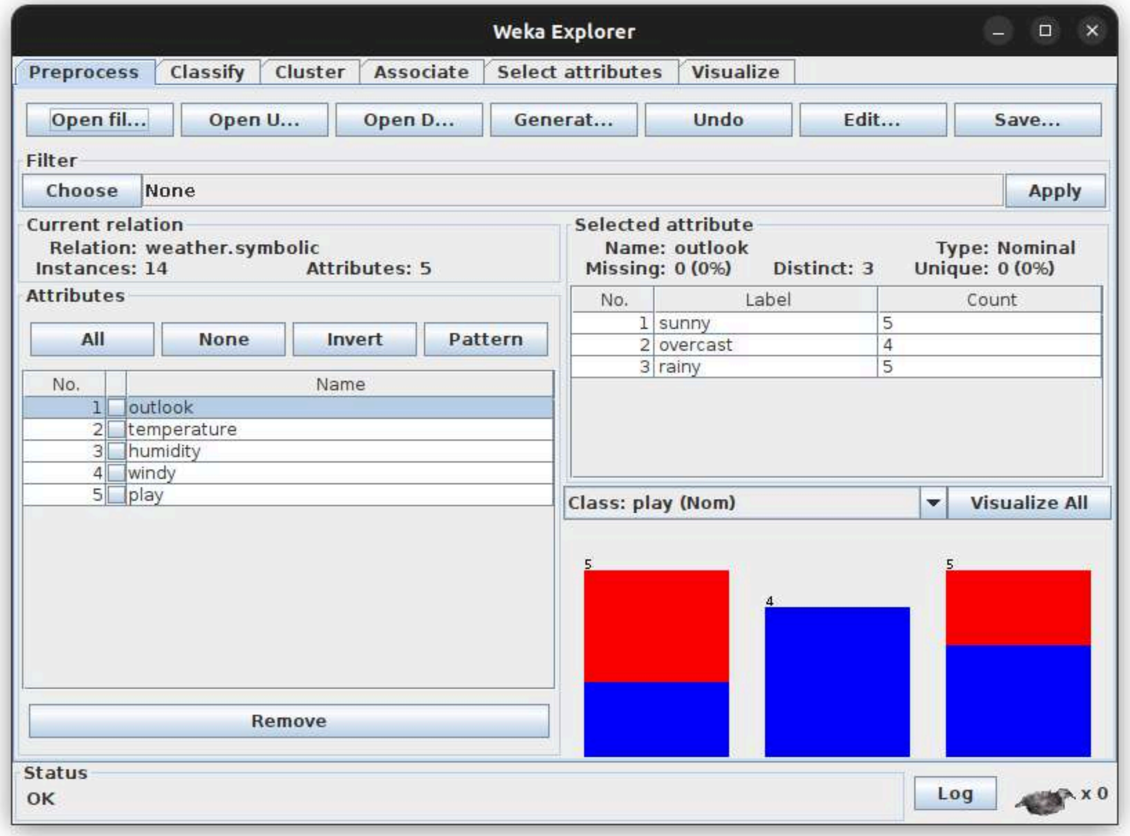
2. Open WEKA.

   > weka



Waikato Environment for Knowledge Analysis
Version 3.6.14
(c) 1999 - 2015
The University of Waikato
Hamilton, New Zealand

3. Open file: /->usr->share>doc-> weka->examples->weather.nominal.arff



Filter: None

Current relation
Relation: weather.symbolic
Instances: 14 Attributes: 5

Attributes

No.		Name
1	□	outlook
2	□	temperature
3	□	humidity
4	□	windy
5	□	play

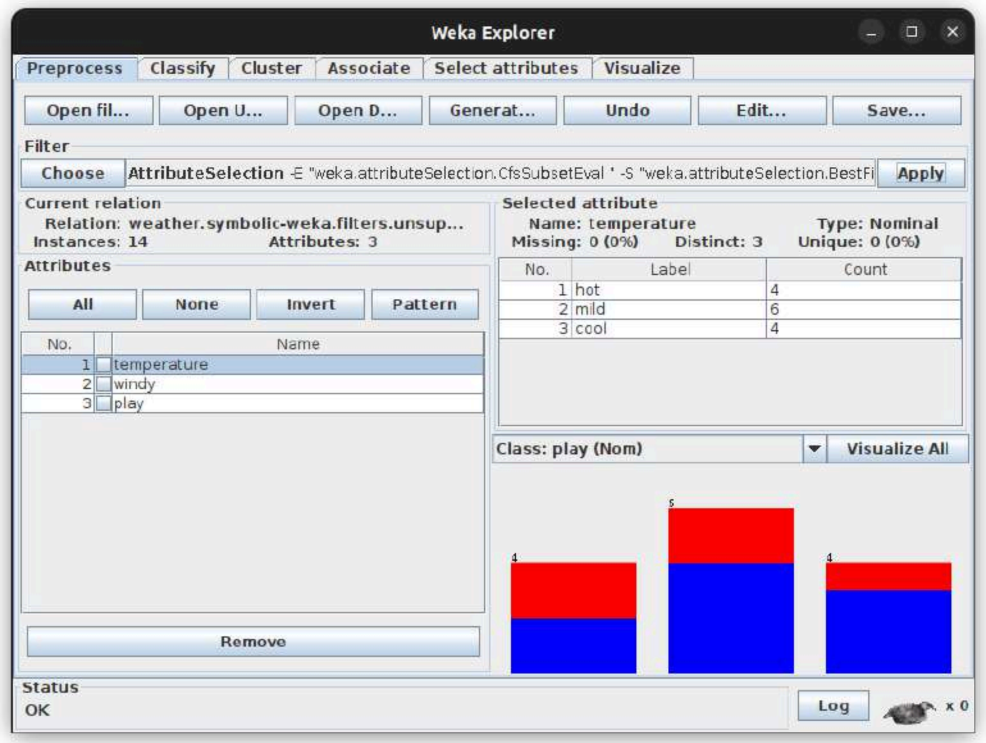
Selected attribute
Name: outlook Type: Nominal
Missing: 0 (0%) Distinct: 3 Unique: 0 (0%)

No.	Label	Count
1	sunny	5
2	overcast	4
3	rainy	5

Class: play (Nom)

Status
OK

4. Removing attributes: To remove Attribute/s select them and click on the Remove button at the bottom.



Filter

AttributeSelection -E "weka.attributeSelection.CfsSubsetEval " -S "weka.attributeSelection.BestFi

Current relation

Relation: weather.symbolic-weka.filters.unsup...
Instances: 14 Attributes: 3

Selected attribute

Name: temperature Type: Nominal
Missing: 0 (0%) Distinct: 3 Unique: 0 (0%)

No.	Label	Count
1	hot	4
2	mild	6
3	cool	4

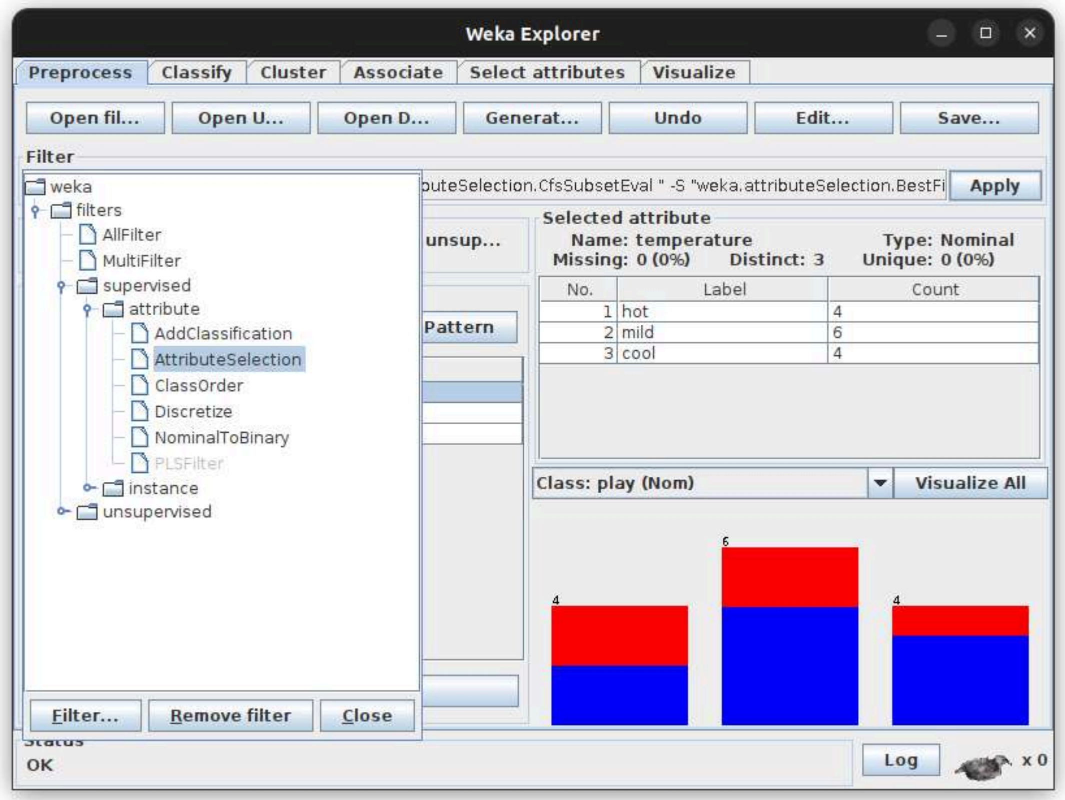
Attributes

No.		Name
1	□	temperature
2	□	windy
3	□	play

Class: play (Nom)

Status
OK

5. Applying FIlters: Click on the Choose button in the Filter subwindow and select the following filter >weka>filters>supervised>attribute>AttributeSelection



Filter

buteSelection.CfsSubsetEval " -S "weka.attributeSelection.BestFi

unsup...

Selected attribute

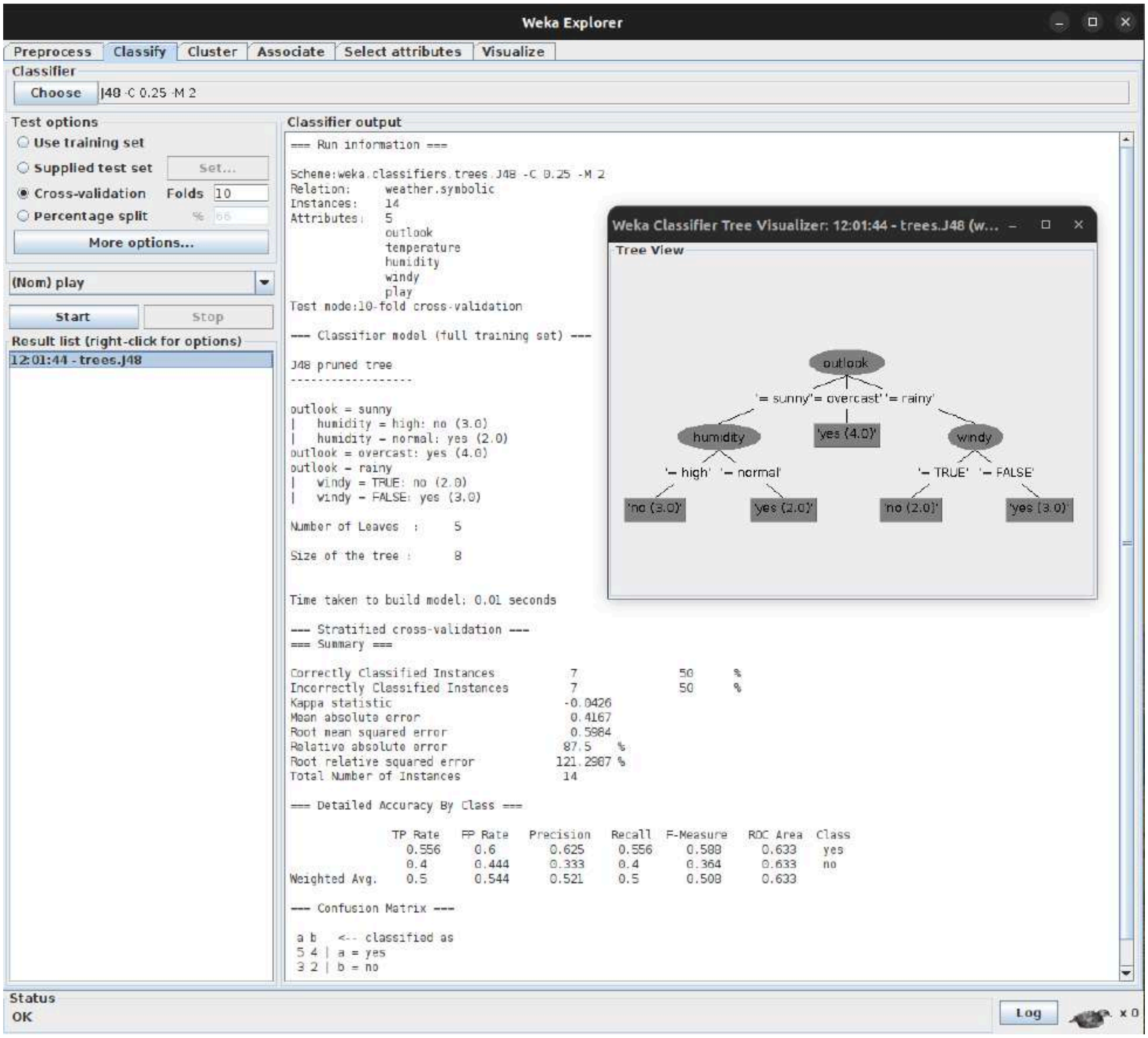
Name: temperature Type: Nominal
Missing: 0 (0%) Distinct: 3 Unique: 0 (0%)

No.	Label	Count
1	hot	4
2	mild	6
3	cool	4

Class: play (Nom)

Status
OK

6. Selecting Classifier: Click on the Choose button and select the following classifier −weka→classifiers>trees>J48



Classifier: J48 -C 0.25 -M 2

Test options:

○ Use training set
○ Supplied test set
● Cross-validation Folds 10
○ Percentage split % 66

(Nom) play

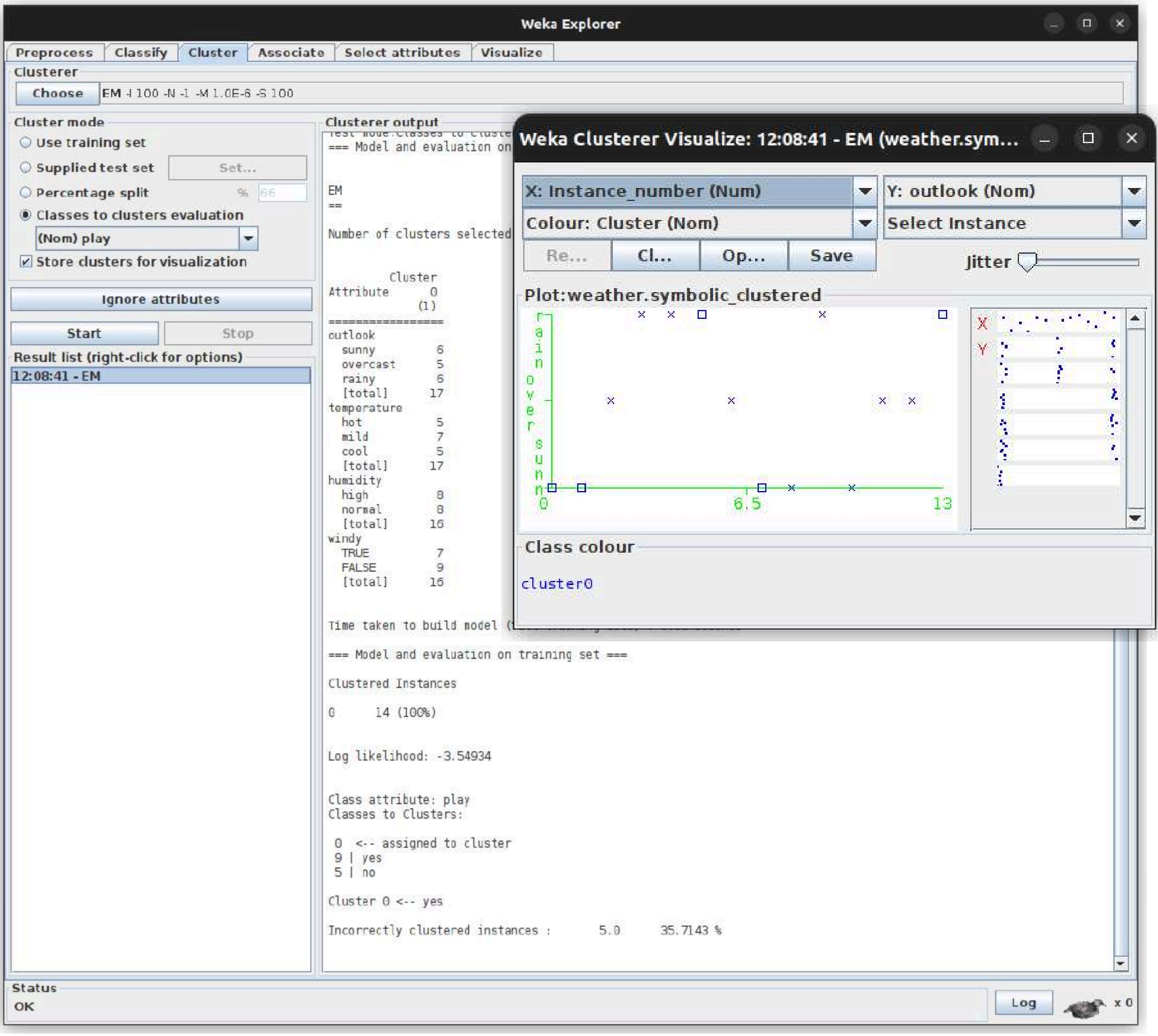
Result list: 12:01:44 - trees.J48

```
=== Run information ===

Scheme:weka.classifiers.trees.J48 -C 0.25 -M 2
Relation:     weather.symbolic
Instances:    14
Attributes:   5
              outlook
              temperature
              humidity
              windy
              play
Test mode:10-fold cross-validation

=== Classifier model (full training set) ===

J48 pruned tree
------------------

outlook = sunny
|   humidity = high: no (3.0)
|   humidity = normal: yes (2.0)
outlook = overcast: yes (4.0)
outlook = rainy
|   windy = TRUE: no (2.0)
|   windy = FALSE: yes (3.0)

Number of Leaves  :     5

Size of the tree :      8


Time taken to build model: 0.01 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances           7               50      %
Incorrectly Classified Instances         7               50      %
Kappa statistic                         -0.0426
Mean absolute error                      0.4167
Root mean squared error                  0.5984
Relative absolute error                 87.5    %
Root relative squared error            121.2987 %
Total Number of Instances               14

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.556     0.6       0.625     0.556     0.588      0.633    yes
                 0.4       0.444     0.333     0.4       0.364      0.633    no
Weighted Avg.    0.5       0.544     0.521     0.5       0.508      0.633

=== Confusion Matrix ===

 a b   <-- classified as
 5 4 | a = yes
 3 2 | b = no
```

Status
OK

STEPS:

1. Open file - weather.arff
2. CLUSTER tab
3. Choose
4. Select EM
5. Start

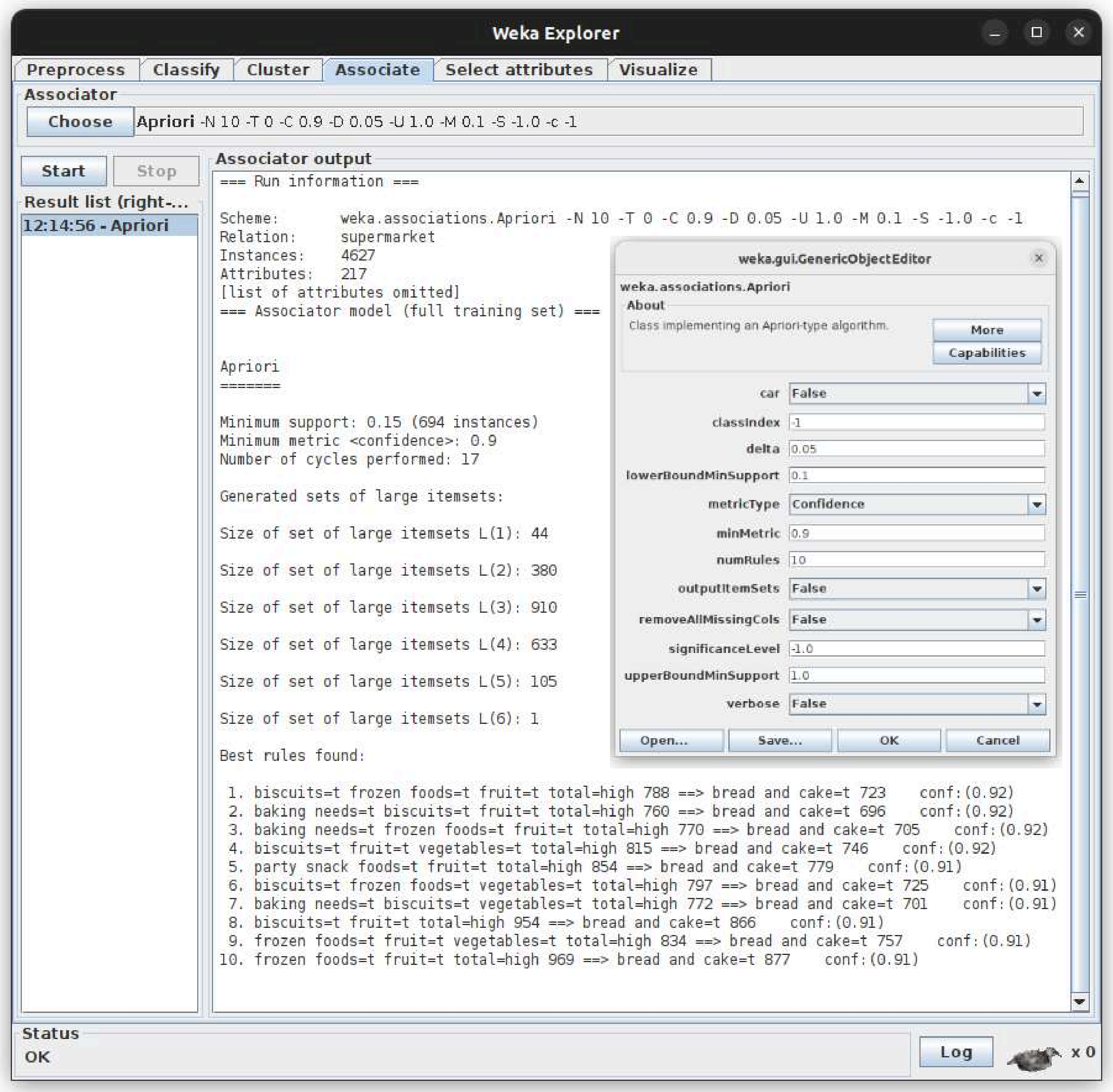


Clusterer: EM -I 100 -N -1 -M 1.0E-6 -S 100

```
=== Model and evaluation on

EM
==

Number of clusters selected


              Cluster
Attribute          0
                 (1)
=================
outlook
  sunny            6
  overcast         5
  rainy            6
  [total]         17
temperature
  hot              5
  mild             7
  cool             5
  [total]         17
humidity
  high             8
  normal           8
  [total]         16
windy
  TRUE             7
  FALSE            9
  [total]         16


Time taken to build model (

=== Model and evaluation on training set ===

Clustered Instances

0      14 (100%)


Log likelihood: -3.54934


Class attribute: play
Classes to Clusters:

 0  <-- assigned to cluster
 9 | yes
 5 | no

Cluster 0 <-- yes

Incorrectly clustered instances :       5.0      35.7143 %
```

STEPS:

1. Open supermarket.arff
2. Open Associate Tab
3. Choose
4. Select Apriori association
5. Start



Apriori -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1

12:14:56 - Apriori

```
=== Run information ===

Scheme:       weka.associations.Apriori -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1
Relation:     supermarket
Instances:    4627
Attributes:   217
[list of attributes omitted]
=== Associator model (full training set) ===


Apriori
=======

Minimum support: 0.15 (694 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 17

Generated sets of large itemsets:

Size of set of large itemsets L(1): 44

Size of set of large itemsets L(2): 380

Size of set of large itemsets L(3): 910

Size of set of large itemsets L(4): 633

Size of set of large itemsets L(5): 105

Size of set of large itemsets L(6): 1

Best rules found:

 1. biscuits=t frozen foods=t fruit=t total=high 788 ==> bread and cake=t 723    conf:(0.92)
 2. baking needs=t biscuits=t fruit=t total=high 760 ==> bread and cake=t 696    conf:(0.92)
 3. baking needs=t frozen foods=t fruit=t total=high 770 ==> bread and cake=t 705    conf:(0.92)
 4. biscuits=t fruit=t vegetables=t total=high 815 ==> bread and cake=t 746    conf:(0.92)
 5. party snack foods=t fruit=t total=high 854 ==> bread and cake=t 779    conf:(0.91)
 6. biscuits=t frozen foods=t vegetables=t total=high 797 ==> bread and cake=t 725    conf:(0.91)
 7. baking needs=t biscuits=t vegetables=t total=high 772 ==> bread and cake=t 701    conf:(0.91)
 8. biscuits=t fruit=t total=high 954 ==> bread and cake=t 866    conf:(0.91)
 9. frozen foods=t fruit=t vegetables=t total=high 834 ==> bread and cake=t 757    conf:(0.91)
10. frozen foods=t fruit=t total=high 969 ==> bread and cake=t 877    conf:(0.91)
```

weka.gui.GenericObjectEditor

weka.associations.Apriori

About

Class implementing an Apriori-type algorithm.

car	False
classIndex	-1
delta	0.05
lowerBoundMinSupport	0.1
metricType	Confidence
minMetric	0.9
numRules	10
outputItemSets	False
removeAllMissingCols	False
significanceLevel	-1.0
upperBoundMinSupport	1.0
verbose	False

Status

OK

CODE:

```
data = [
        ['T100',['I1','I2','I5']],
        ['T200',['I2','I4']],
        ['T300',['I2','I3']],
        ['T400',['I1','I2','I4']],
        ['T500',['I1','I3']],
        ['T600',['I2','I3']],
        ['T700',['I1','I3']],
        ['T800',['I1','I2','I3','I5']],
        ['T900',['I1','I2','I3']]
        ]
init = []
for i in data:
    for q in i[1]:
        if(q not in init):
            init.append(q)
init = sorted(init)
print(init)
sp = 0.4
s = int(sp*len(init))

from collections import Counter
c = Counter()
for i in init:
    for d in data:
        if(i in d[1]):
            c[i]+=1
print("C1:")
for i in c:
    print(str([i])+": "+str(c[i]))
print()
l = Counter()
for i in c:
    if(c[i] >= s):
        l[frozenset([i])]+=c[i]
print("L1:")
for i in l:
    print(str(list(i))+": "+str(l[i]))
print()
pl = l
pos = 1
for count in range (2,1000):
    nc = set()
    temp = list(l)
    for i in range(0,len(temp)):
        for j in range(i+1,len(temp)):
            t = temp[i].union(temp[j])
            if(len(t) == count):
                nc.add(temp[i].union(temp[j]))
    nc = list(nc)
    c = Counter()
    for i in nc:
        c[i] = 0
        for q in data:
            temp = set(q[1])
            if(i.issubset(temp)):
                c[i]+=1
    print("C"+str(count)+":")
    for i in c:
        print(str(list(i))+": "+str(c[i]))
    print()
    l = Counter()
    for i in c:
        if(c[i] >= s):
            l[i]+=c[i]
    print("L"+str(count)+":")
    for i in l:
        print(str(list(i))+": "+str(l[i]))
    print()
    if(len(l) == 0):
        break
    pl = l
    pos = count
print("Result: ")
print("L"+str(pos)+":")
for i in pl:
    print(str(list(i))+": "+str(pl[i]))
print()
from itertools import combinations
for l in pl:
    c = [frozenset(q) for q in combinations(l,len(l)-1)]
    mmax = 0
    for a in c:
        b = l-a
        ab = l
        sab = 0
        sa = 0
        sb = 0
        for q in data:
            temp = set(q[1])
            if(a.issubset(temp)):
                sa+=1
            if(b.issubset(temp)):
                sb+=1
            if(ab.issubset(temp)):
                sab+=1
        temp = sab/sa*100
        if(temp > mmax):
            mmax = temp
        temp = sab/sb*100
        if(temp > mmax):
            mmax = temp
        print(str(list(a))+" -> "+str(list(b))+" = "+str(sab/sa*100)+"%")
        print(str(list(b))+" -> "+str(list(a))+" = "+str(sab/sb*100)+"%")
    curr = 1
    print("choosing:", end=' ')
    for a in c:
        b = l-a
        ab = l
        sab = 0
        sa = 0
        sb = 0
        for q in data:
            temp = set(q[1])
            if(a.issubset(temp)):
                sa+=1
            if(b.issubset(temp)):
                sb+=1
            if(ab.issubset(temp)):
                sab+=1
        temp = sab/sa*100
        if(temp == mmax):
            print(curr, end = ' ')
        curr += 1
        temp = sab/sb*100
        if(temp == mmax):
            print(curr, end = ' ')
        curr += 1
    print()
    print()
```

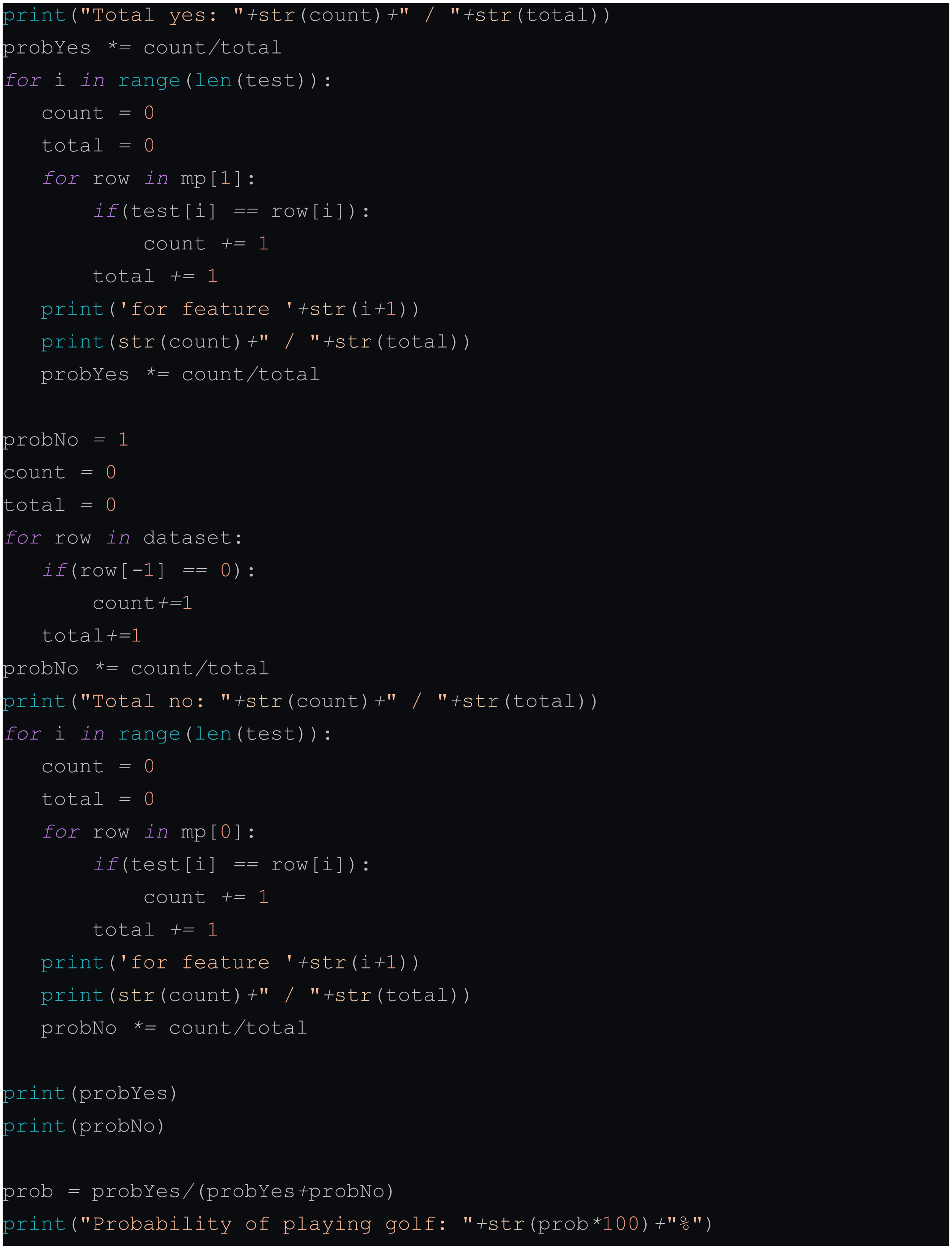
OUTPUT:

```
(base) computer@computer:~/Desktop$ python apriori.py
['I1', 'I2', 'I3', 'I4', 'I5']
C1:
['I1']: 6
['I2']: 7
['I3']: 6
['I4']: 2
['I5']: 2

L1:
['I1']: 6
['I2']: 7
['I3']: 6
['I4']: 2
['I5']: 2

C2:
['I2', 'I4']: 2
['I2', 'I3']: 4
['I3', 'I1']: 4
['I5', 'I4']: 0
['I4', 'I1']: 1
['I5', 'I1']: 2
['I3', 'I4']: 0
['I2', 'I5']: 2
['I5', 'I3']: 1
['I2', 'I1']: 4

L2:
['I2', 'I4']: 2
['I2', 'I3']: 4
['I3', 'I1']: 4
['I5', 'I1']: 2
['I2', 'I5']: 2
['I2', 'I1']: 4

C3:
['I3', 'I2', 'I4']: 0
['I2', 'I5', 'I1']: 2
['I2', 'I4', 'I1']: 1
['I5', 'I3', 'I1']: 1
['I2', 'I5', 'I4']: 0
['I2', 'I3', 'I1']: 2
```

```
['I2', 'I5', 'I3']: 1

L3:
['I2', 'I5', 'I1']: 2
['I2', 'I3', 'I1']: 2

C4:
['I5', 'I1', 'I2', 'I3']: 1

L4:

Result:
L3:
['I2', 'I5', 'I1']: 2
['I2', 'I3', 'I1']: 2

['I2', 'I5'] -> ['I1'] = 100.0%
['I1'] -> ['I2', 'I5'] = 33.33333333333333%
['I2', 'I1'] -> ['I5'] = 50.0%
['I5'] -> ['I2', 'I1'] = 100.0%
['I5', 'I1'] -> ['I2'] = 100.0%
['I2'] -> ['I5', 'I1'] = 28.57142857142857%
choosing: 1 4 5

['I2', 'I3'] -> ['I1'] = 50.0%
['I1'] -> ['I2', 'I3'] = 33.33333333333333%
['I2', 'I1'] -> ['I3'] = 50.0%
['I3'] -> ['I2', 'I1'] = 33.33333333333333%
['I3', 'I1'] -> ['I2'] = 50.0%
['I2'] -> ['I3', 'I1'] = 28.57142857142857%
choosing: 1 3 5
```

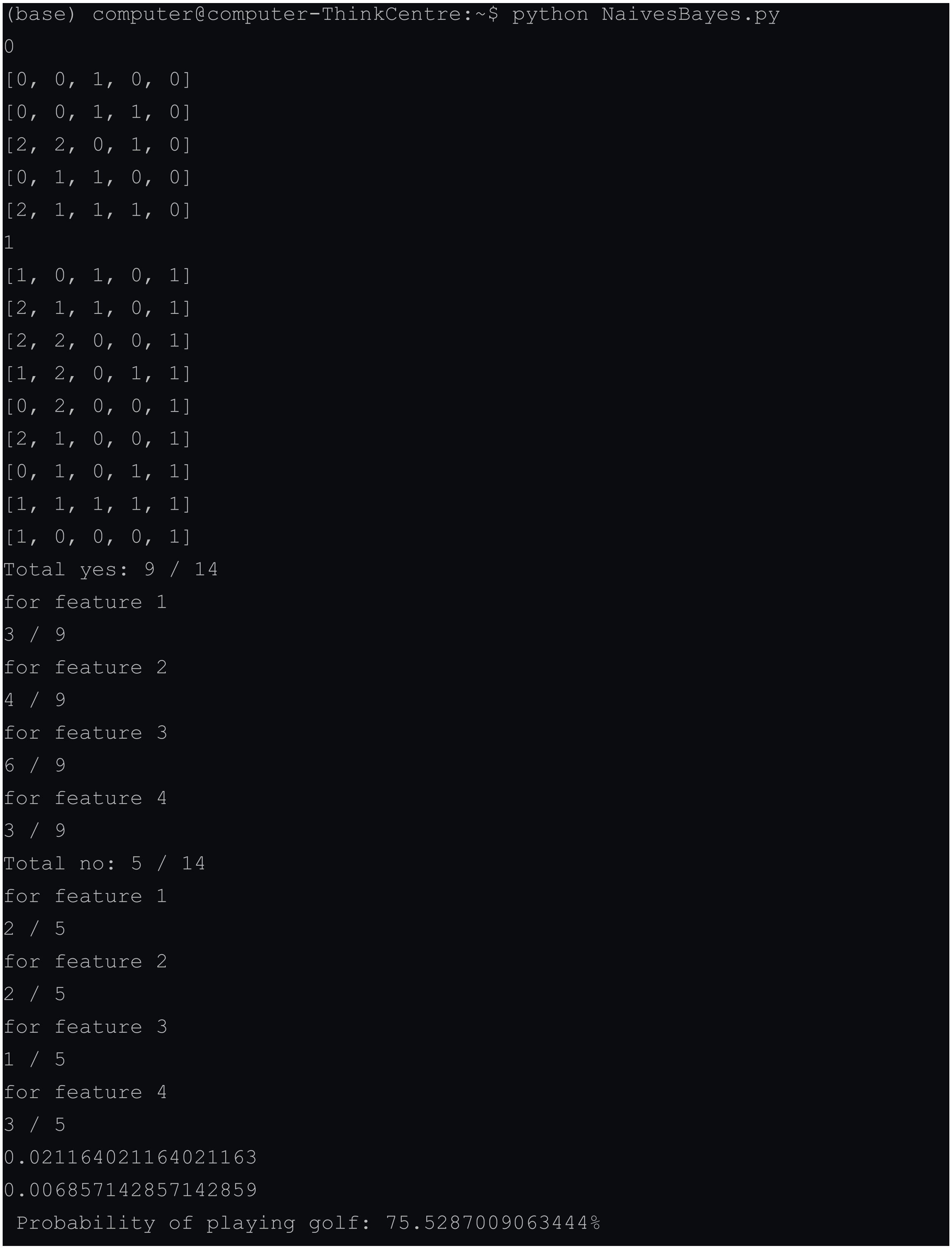
CODE:

```
dataset = [
          [0,0,1,0,0],
          [0,0,1,1,0],
          [1,0,1,0,1],
          [2,1,1,0,1],
          [2,2,0,0,1],
          [2,2,0,1,0],
          [1,2,0,1,1],
          [0,1,1,0,0],
          [0,2,0,0,1],
          [2,1,0,0,1],
          [0,1,0,1,1],
          [1,1,1,1,1],
          [1,0,0,0,1],
          [2,1,1,1,0]
          ]
mp = dict()
for i in range(len(dataset)):
   row = dataset[i]
   y = row[-1]
   if (y not in mp):
       mp[y] = list()
   mp[y].append(row)

for label in mp:
   print(label)
   for row in mp[label]:
       print(row)

test = [2,1,0,1]

probYes = 1

count = 0
total = 0
for row in dataset:
   if(row[-1] == 1):
       count+=1
   total+=1
```



```
print("Total yes: "+str(count)+" / "+str(total))
probYes *= count/total
for i in range(len(test)):
    count = 0
    total = 0
    for row in mp[1]:
        if(test[i] == row[i]):
            count += 1
        total += 1
    print('for feature '+str(i+1))
    print(str(count)+" / "+str(total))
    probYes *= count/total

probNo = 1
count = 0
total = 0
for row in dataset:
    if(row[-1] == 0):
        count+=1
    total+=1
probNo *= count/total
print("Total no: "+str(count)+" / "+str(total))
for i in range(len(test)):
    count = 0
    total = 0
    for row in mp[0]:
        if(test[i] == row[i]):
            count += 1
        total += 1
    print('for feature '+str(i+1))
    print(str(count)+" / "+str(total))
    probNo *= count/total

print(probYes)
print(probNo)

prob = probYes/(probYes+probNo)
print("Probability of playing golf: "+str(prob*100)+"%")
```

OUTPUT:



```
(base) computer@computer-ThinkCentre:~$ python NaivesBayes.py
0
[0, 0, 1, 0, 0]
[0, 0, 1, 1, 0]
[2, 2, 0, 1, 0]
[0, 1, 1, 0, 0]
[2, 1, 1, 1, 0]
1
[1, 0, 1, 0, 1]
[2, 1, 1, 0, 1]
[2, 2, 0, 0, 1]
[1, 2, 0, 1, 1]
[0, 2, 0, 0, 1]
[2, 1, 0, 0, 1]
[0, 1, 0, 1, 1]
[1, 1, 1, 1, 1]
[1, 0, 0, 0, 1]
Total yes: 9 / 14
for feature 1
3 / 9
for feature 2
4 / 9
for feature 3
6 / 9
for feature 4
3 / 9
Total no: 5 / 14
for feature 1
2 / 5
for feature 2
2 / 5
for feature 3
1 / 5
for feature 4
3 / 5
0.021164021164021163
0.006857142857142859
 Probability of playing golf: 75.5287009063444%
```

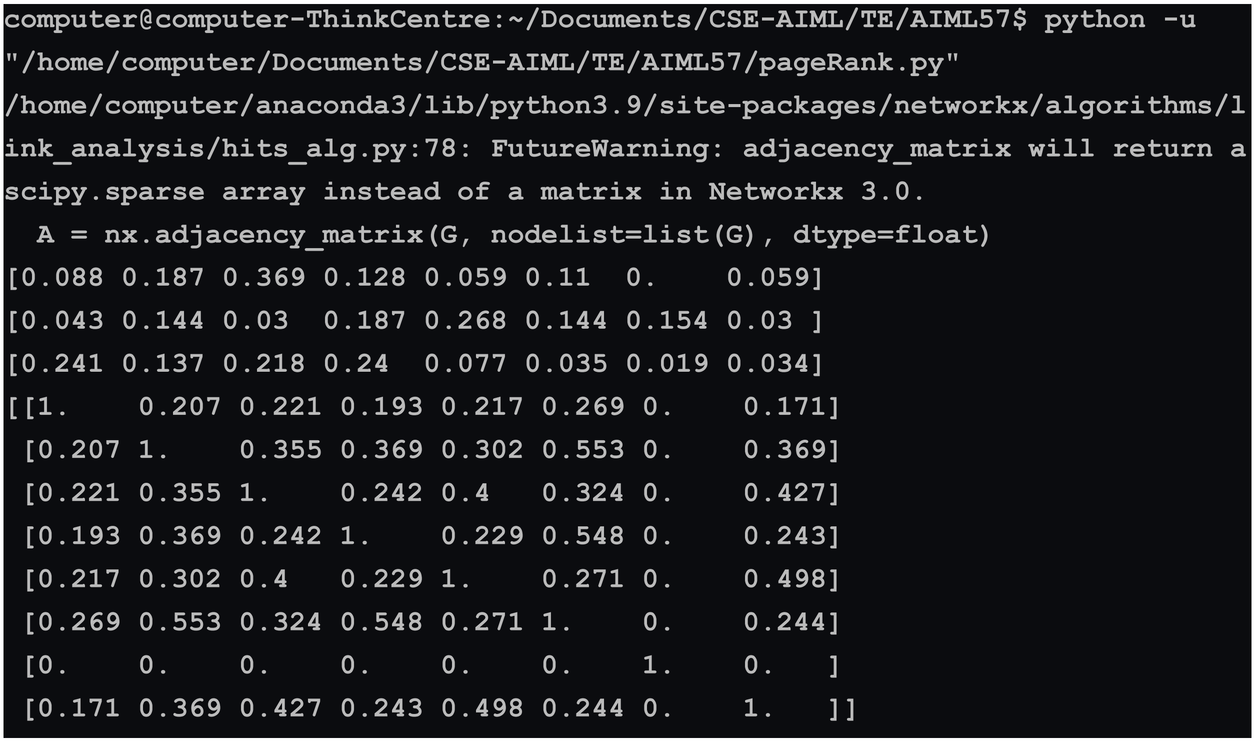
CODE:

```
import networkx as nx
import numpy as np
from numpy import array
import matplotlib.pyplot as plt
with open('hits.txt') as f:
    lines = f.readlines()

G = nx.DiGraph()

for line in lines:
    t = tuple(line.strip().split(','))
    G.add_edge(*t)

h, a = nx.hits(G, max_iter=100)
h = dict(sorted(h.items(), key=lambda x: x[0]))
a = dict(sorted(a.items(), key=lambda x: x[0]))

print(np.round(list(a.values()), 3))
print(np.round(list(h.values()), 3))

pr = nx.pagerank(G)
pr = dict(sorted(pr.items(), key=lambda x: x[0]))
print(np.round(list(pr.values()), 3))


sim = nx.simrank_similarity(G)
lol = [[sim[u][v] for v in sorted(sim[u])] for u in sorted(sim)]
sim_array = np.round(array(lol), 3)
print(sim_array)

nx.draw(G, with_labels=True, node_size=2000, edge_color='#eb4034', width=3,
font_size=16, font_weight=500, arrowsize=20, alpha=0.8)
plt.savefig("graph.png")
```

hits.txt

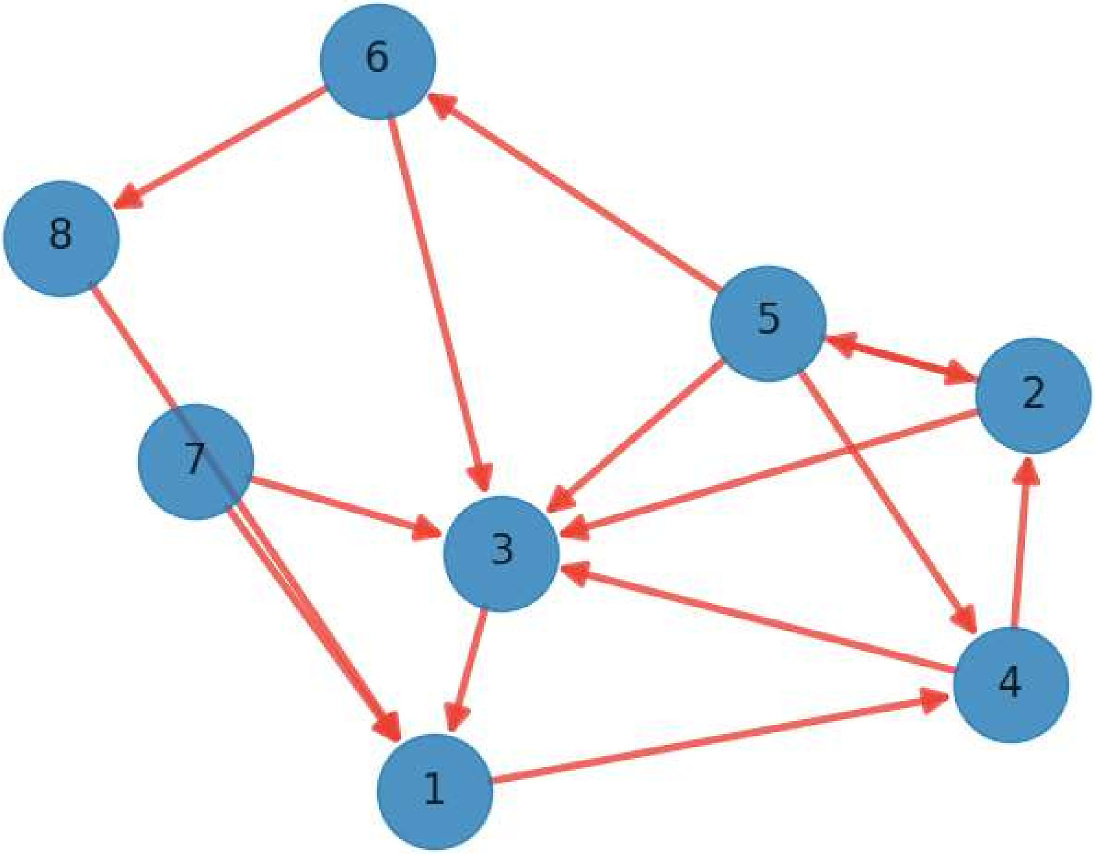
```
1,4
2,3
2,5
3,1
4,2
4,3
5,3
5,2
5,4
5,6
6,3
6,8
7,1
7,3
8,1
```

OUTPUT:



```
computer@computer-ThinkCentre:~/Documents/CSE-AIML/TE/AIML57$ python -u
"/home/computer/Documents/CSE-AIML/TE/AIML57/pageRank.py"
/home/computer/anaconda3/lib/python3.9/site-packages/networkx/algorithms/l
ink_analysis/hits_alg.py:78: FutureWarning: adjacency_matrix will return a
scipy.sparse array instead of a matrix in Networkx 3.0.
  A = nx.adjacency_matrix(G, nodelist=list(G), dtype=float)
[0.088 0.187 0.369 0.128 0.059 0.11  0.    0.059]
[0.043 0.144 0.03  0.187 0.268 0.144 0.154 0.03 ]
[0.241 0.137 0.218 0.24  0.077 0.035 0.019 0.034]
[[1.    0.207 0.221 0.193 0.217 0.269 0.    0.171]
 [0.207 1.    0.355 0.369 0.302 0.553 0.    0.369]
 [0.221 0.355 1.    0.242 0.4   0.324 0.    0.427]
 [0.193 0.369 0.242 1.    0.229 0.548 0.    0.243]
 [0.217 0.302 0.4   0.229 1.    0.271 0.    0.498]
 [0.269 0.553 0.324 0.548 0.271 1.    0.    0.244]
 [0.    0.    0.    0.    0.    0.    1.    0.   ]
 [0.171 0.369 0.427 0.243 0.498 0.244 0.    1.   ]]
```

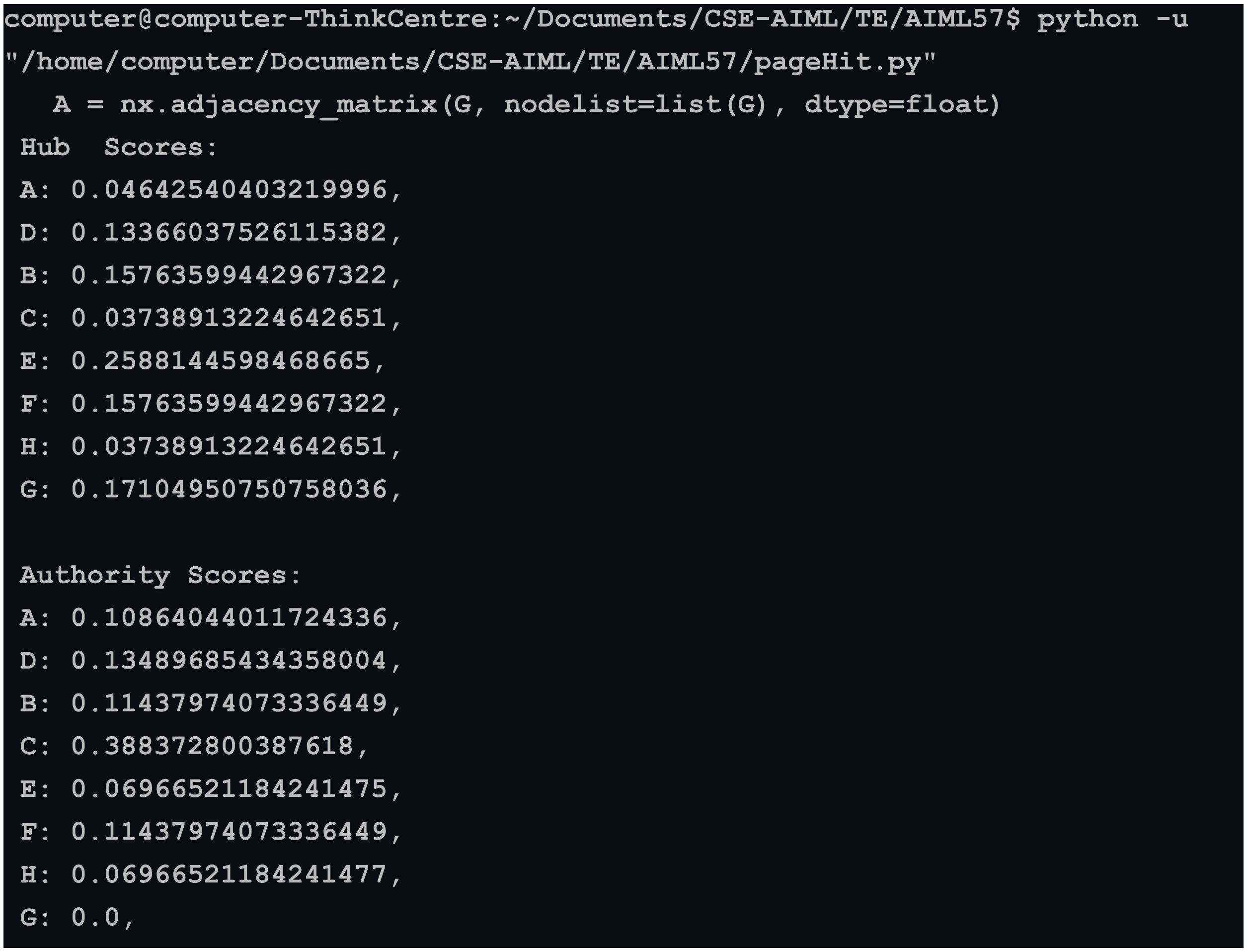
graph.png



CODE(pageHit.py):

```
import networkx as nx
import matplotlib.pyplot as plt
G = nx.DiGraph()
G.add_edges_from([('A', 'D'), ('B', 'C'), ('B', 'E'), ('C', 'A'), ('D', 'C'), ('E', 'D'), ('E', 'B'), ('E',
'F'), ('E', 'C'), ('F', 'C'), ('F', 'H'), ('G', 'A'), ('G', 'C'), ('H', 'A')])
plt.figure(figsize =(10, 10))
nx.draw_networkx(G, with_labels = True)
hubs, authorities = nx.hits(G, max_iter = 50, normalized = True)
print('Hub  Scores: ')
for i in hubs:
   print("{}: {},".format(i, hubs[i]))
print('\nAuthority Scores: ')
for i in authorities:
   print("{}: {},".format(i, authorities[i]))
```

OUTPUT:



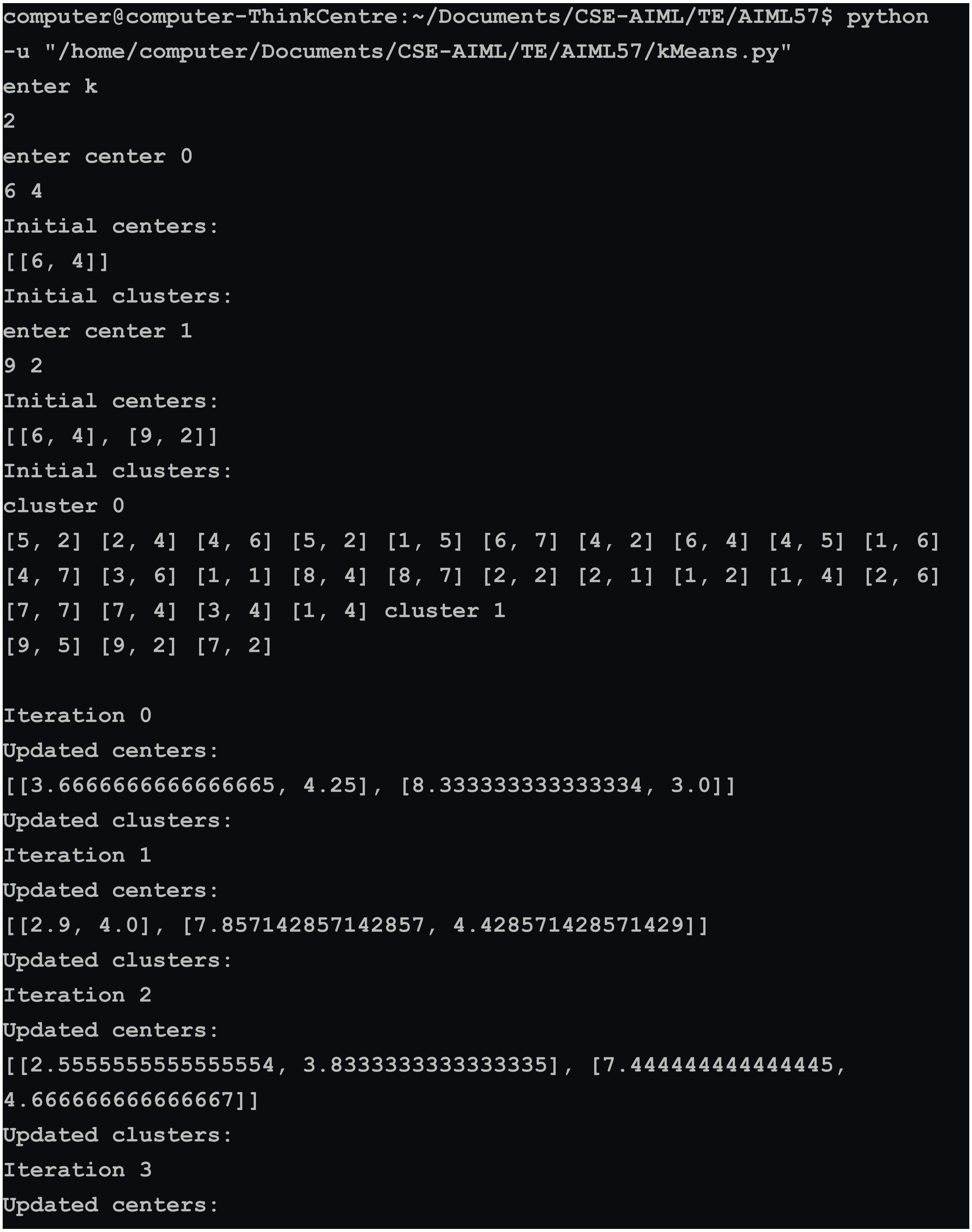
```
computer@computer-ThinkCentre:~/Documents/CSE-AIML/TE/AIML57$ python -u
"/home/computer/Documents/CSE-AIML/TE/AIML57/pageHit.py"
   A = nx.adjacency_matrix(G, nodelist=list(G), dtype=float)
 Hub  Scores:
 A: 0.04642540403219996,
 D: 0.13366037526115382,
 B: 0.15763599442967322,
 C: 0.03738913224642651,
 E: 0.2588144598468665,
 F: 0.15763599442967322,
 H: 0.03738913224642651,
 G: 0.17104950750758036,

 Authority Scores:
 A: 0.10864044011724336,
 D: 0.13489685434358004,
 B: 0.11437974073336449,
 C: 0.388372800387618,
 E: 0.06966521184241475,
 F: 0.11437974073336449,
 H: 0.06966521184241477,
 G: 0.0,
```

CODE:

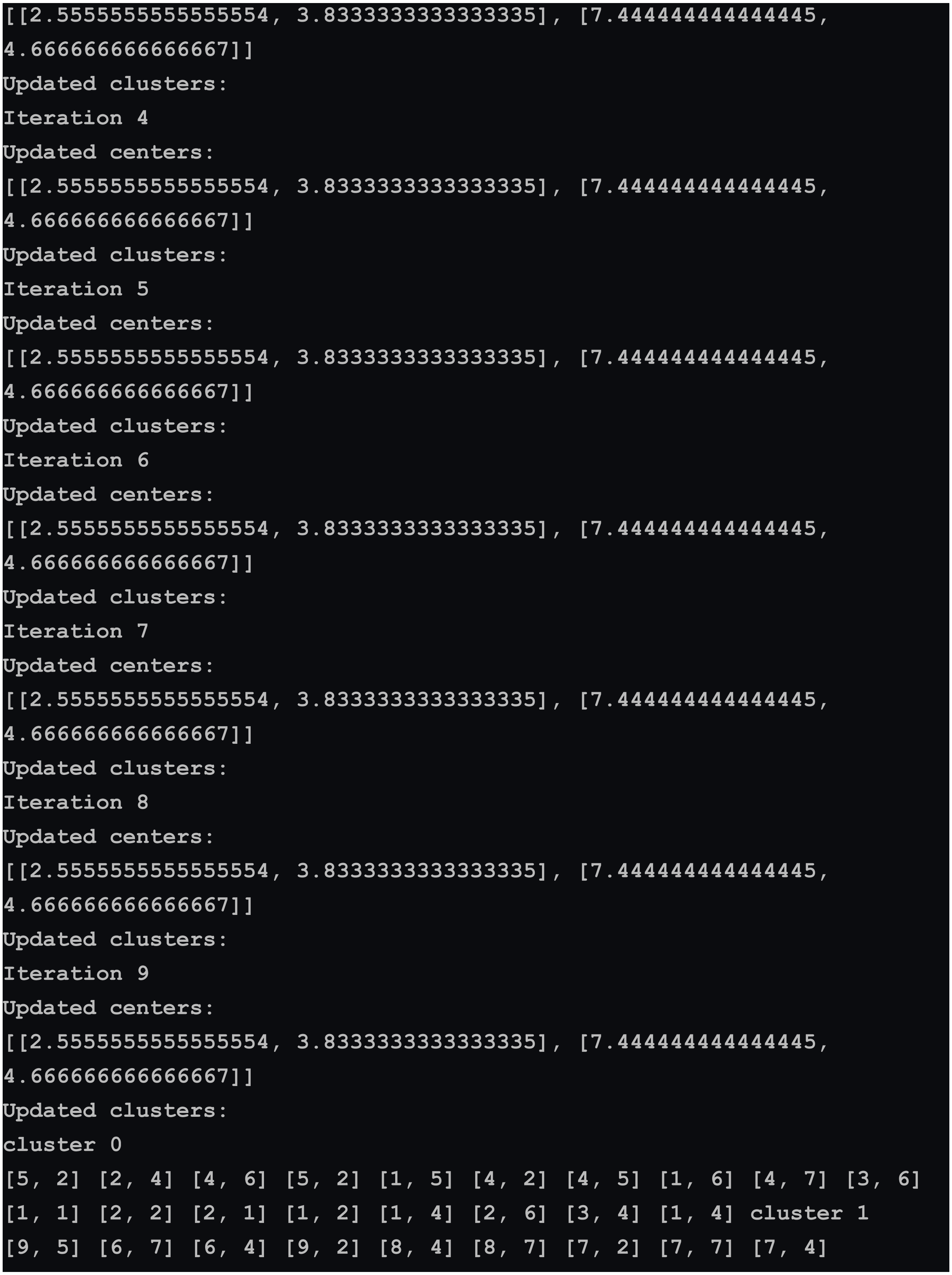
```
data = [
[5,2],[2,4],[9,5],[4,6],[5,2],[1,5],[6,7],[4,2],[6,4],[9,2],[4,5],[1,6],[4,7],[3,6],[1,1],[8,4],[8,7],
          [7,2],[2,2],[2,1],[1,2],[1,4],[2,6],[7,7],[7,4],[3,4],[1,4]
         ]
x = [i[0] for i in data]
y = [i[1] for i in data]
import math
def dist(center, point):
  d = 0.0
  for i in range(0,len(point)):
     d += (center[i]-point[i])**2
  return math.sqrt(d)

def assignCenters(centers, dataset):
  clusters = []
  for i in range(len(dataset)):
     distances = []
     for center in centers:
        distances.append(dist(center, dataset[i]))
        temp = [z for z, val in enumerate(distances) if val==min(distances)]
     clusters.append(temp[0])
  return clusters

def mean_center(k, dataset, clusters):
  nCenters = []
  for i in range(k):
     x = 0.0
     y = 0.0
     count = 0
     for j in range(len(clusters)):
        if(i == clusters[j]):
           x += dataset[j][0]
           y += dataset[j][1]
           count += 1
     x = x/count
     y = y/count
     nCenters.append([x,y])
  return nCenters
print("enter k")
```

```
k = int(input())
centers = []
for i in range(k):
  print("enter center "+str(i))
  temp = [int(x) for x in input().split()]
  centers.append(temp)
  print("Initial centers: ")
  print(centers)
  print("Initial clusters: ")
  clusters = assignCenters(centers, data)
for i in range(k):
  print("cluster "+str(i))
  for j in range(len(clusters)):
     if(i == clusters[j]):
        print(data[j],end=' ')
print()
print()
for itr in range(10):
  print("Iteration "+str(itr))
  centers = mean_center(k,data,clusters)
  print("Updated centers: ")
  print(centers)
  clusters = assignCenters(centers, data)
  print("Updated clusters: ")
for i in range(k):
  print("cluster "+str(i))
  for j in range(len(clusters)):
     if(i == clusters[j]):
        print(data[j],end=' ')
print()
print()
```

OUTPUT:



```
computer@computer-ThinkCentre:~/Documents/CSE-AIML/TE/AIML57$ python
-u "/home/computer/Documents/CSE-AIML/TE/AIML57/kMeans.py"
enter k
2
enter center 0
6 4
Initial centers:
[[6, 4]]
Initial clusters:
enter center 1
9 2
Initial centers:
[[6, 4], [9, 2]]
Initial clusters:
cluster 0
[5, 2] [2, 4] [4, 6] [5, 2] [1, 5] [6, 7] [4, 2] [6, 4] [4, 5] [1, 6]
[4, 7] [3, 6] [1, 1] [8, 4] [8, 7] [2, 2] [2, 1] [1, 2] [1, 4] [2, 6]
[7, 7] [7, 4] [3, 4] [1, 4] cluster 1
[9, 5] [9, 2] [7, 2]

Iteration 0
Updated centers:
[[3.6666666666666665, 4.25], [8.333333333333334, 3.0]]
Updated clusters:
Iteration 1
Updated centers:
[[2.9, 4.0], [7.857142857142857, 4.428571428571429]]
Updated clusters:
Iteration 2
Updated centers:
[[2.5555555555555554, 3.8333333333333335], [7.444444444444445,
4.666666666666667]]
Updated clusters:
Iteration 3
Updated centers:
```



```
[[2.5555555555555554, 3.8333333333333335], [7.444444444444445,
4.666666666666667]]
Updated clusters:
Iteration 4
Updated centers:
[[2.5555555555555554, 3.8333333333333335], [7.444444444444445,
4.666666666666667]]
Updated clusters:
Iteration 5
Updated centers:
[[2.5555555555555554, 3.8333333333333335], [7.444444444444445,
4.666666666666667]]
Updated clusters:
Iteration 6
Updated centers:
[[2.5555555555555554, 3.8333333333333335], [7.444444444444445,
4.666666666666667]]
Updated clusters:
Iteration 7
Updated centers:
[[2.5555555555555554, 3.8333333333333335], [7.444444444444445,
4.666666666666667]]
Updated clusters:
Iteration 8
Updated centers:
[[2.5555555555555554, 3.8333333333333335], [7.444444444444445,
4.666666666666667]]
Updated clusters:
Iteration 9
Updated centers:
[[2.5555555555555554, 3.8333333333333335], [7.444444444444445,
4.666666666666667]]
Updated clusters:
cluster 0
[5, 2] [2, 4] [4, 6] [5, 2] [1, 5] [4, 2] [4, 5] [1, 6] [4, 7] [3, 6]
[1, 1] [2, 2] [2, 1] [1, 2] [1, 4] [2, 6] [3, 4] [1, 4] cluster 1
[9, 5] [6, 7] [6, 4] [9, 2] [8, 4] [8, 7] [7, 2] [7, 7] [7, 4]
```