Lab 2: Association Analysis

Tore Andersson, Zahra Jalil Pour

Some algorithms in data mining need to convert continuous attributes to discrete attributes. In this regard the process of discretization should be used to convert continuous attributes, models or functions to a discrete form by setting various intervals(bins) in which discrete data will be counted.

In this assignment we use Iris data set which consists of 4 numeric continuous attributes and one categorical attribute named class that should be ignored in kmeans clustering. Before starting mining process, discretization should be applied on this dataset.
Iris dataset:

Number of features; four continuous numeric features and one categorical feature(class) Feature information:

- -Sepal Length
- -Sepal width
- -Petal Length
- -Petal width

Number of instances: 150 instances (50 in each class) Class distribution (33.3% in each class)
Before discretization:

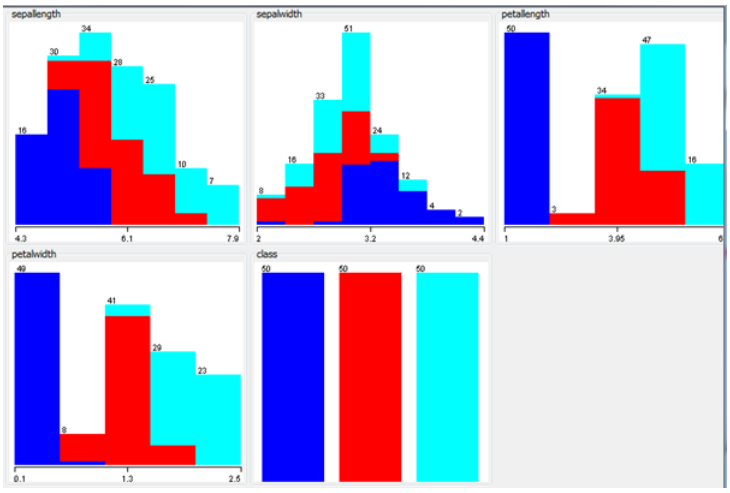

Figure 1: Visualization of the data

Data preprocessing:
Before using Apriori Algorithm, the data set should be discretized. Discretization has several advantages:

- The number of continuous features value will be reduced by discretization.
- By discretization, datamining technique become faster.
- Some data mining algorithm can only deal with discrete features.
- Data will be simplified to understand

Discrete features reduce memory usage and thus increase representation of the knowledge as data is simplified to understand and with this application of mining technique or knowledge retrieval methods become faster and perfect

By selecting Bin=3, data will be discretized:

Relation: iris-weka.filters.unsupervised.attribute.Discretize-B3-M-1.0-R1-4

No.	sepallength Nominal	sepalwidth Nominal	petallength Nominal	petalwidth Nominal	**class** Nominal
1	'(-inf-5.5]'	'(2.8-3.6]'	'(-inf-2.96...'	'(-inf-0.9]'	Iris-se...
2	'(-inf-5.5]'	'(2.8-3.6]'	'(-inf-2.96...'	'(-inf-0.9]'	Iris-se...
3	'(-inf-5.5]'	'(2.8-3.6]'	'(-inf-2.96...'	'(-inf-0.9]'	Iris-se...
4	'(-inf-5.5]'	'(2.8-3.6]'	'(-inf-2.96...'	'(-inf-0.9]'	Iris-se...
5	'(-inf-5.5]'	'(2.8-3.6]'	'(-inf-2.96...'	'(-inf-0.9]'	Iris-se...
6	'(-inf-5.5]'	'(3.6-inf)'	'(-inf-2.96...'	'(-inf-0.9]'	Iris-se...
7	'(-inf-5.5]'	'(2.8-3.6]'	'(-inf-2.96...'	'(-inf-0.9]'	Iris-se...
8	'(-inf-5.5]'	'(2.8-3.6]'	'(-inf-2.96...'	'(-inf-0.9]'	Iris-se...
9	'(-inf-5.5]'	'(2.8-3.6]'	'(-inf-2.96...'	'(-inf-0.9]'	Iris-se...
10	'(-inf-5.5]'	'(2.8-3.6]'	'(-inf-2.96...'	'(-inf-0.9]'	Iris-se...
11	'(-inf-5.5]'	'(3.6-inf)'	'(-inf-2.96...'	'(-inf-0.9]'	Iris-se...
12	'(-inf-5.5]'	'(2.8-3.6]'	'(-inf-2.96...'	'(-inf-0.9]'	Iris-se...
13	'(-inf-5.5]'	'(2.8-3.6]'	'(-inf-2.96...'	'(-inf-0.9]'	Iris-se...
14	'(-inf-5.5]'	'(2.8-3.6]'	'(-inf-2.96...'	'(-inf-0.9]'	Iris-se...
15	'(5.5-6.7]'	'(3.6-inf)'	'(-inf-2.96...'	'(-inf-0.9]'	Iris-se...
16	'(5.5-6.7]'	'(3.6-inf)'	'(-inf-2.96...'	'(-inf-0.9]'	Iris-se...
17	'(-inf-5.5]'	'(3.6-inf)'	'(-inf-2.96...'	'(-inf-0.9]'	Iris-se...
18	'(-inf-5.5]'	'(2.8-3.6]'	'(-inf-2.96...'	'(-inf-0.9]'	Iris-se...
19	'(5.5-6.7]'	'(3.6-inf)'	'(-inf-2.96...'	'(-inf-0.9]'	Iris-se...
20	'(-inf-5.5]'	'(3.6-inf)'	'(-inf-2.96...'	'(-inf-0.9]'	Iris-se...

Figure 2: The data in discrete CSV format

Feature	Subset	Range
Sepal length	S1	{-Inf-5.5}
	S2	{5.5-6.7}
	S3	{6.7-Inf}
Sepal width	S1	{-Inf-2.8}
	S2	{2.8-3.6}
	S3	{3.6-Inf}
Petal length	S1	{-Inf-2.96667}
	S2	{2.96667-4.9333
	S3	{4.9333-Inf}
Petal width	S1	{-Inf-0.9}
	S2	{0.9-1.7}
	S3	{1.7-Inf}

Table 1

Kmeans Clustering:

With 3 bins and 3 clusters, confusion matrix shows that Iris setosa is classified correctly. But Iris versicolor and virginica have error in clustering.

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

Clustered Instances

0       55 ( 37%)
1       45 ( 30%)
2       50 ( 33%)


Class attribute: class
Classes to Clusters:

  0  1  2  <-- assigned to cluster
  0  0 50 | Iris-setosa
 48  2  0 | Iris-versicolor
  7 43  0 | Iris-virginica

Cluster 0 <-- Iris-versicolor
Cluster 1 <-- Iris-virginica
Cluster 2 <-- Iris-setosa

Incorrectly clustered instances :       9.0       6      %
```

Figure 3: Model output with K = 3, bin = 3

Association analysis:
By performing Associate analysis, based on Apriori algorithm and default values, numRules=10, min Metric=0.9 , metricType=Confidence, we obtained the result as below:

```
Apriori
=======

Minimum support: 0.3 (45 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 14

Generated sets of large itemsets:

Size of set of large itemsets L(1): 13

Size of set of large itemsets L(2): 10

Size of set of large itemsets L(3): 5

Size of set of large itemsets L(4): 1

Best rules found:

 1. petalwidth='(-inf-0.9]' 50 ==> petallength='(-inf-2.966667]' 50    conf:(1)
 2. petallength='(-inf-2.966667]' 50 ==> petalwidth='(-inf-0.9]' 50    conf:(1)
 3. class=Iris-setosa 50 ==> petallength='(-inf-2.966667]' 50    conf:(1)
 4. petallength='(-inf-2.966667]' 50 ==> class=Iris-setosa 50    conf:(1)
 5. class=Iris-setosa 50 ==> petalwidth='(-inf-0.9]' 50    conf:(1)
 6. petalwidth='(-inf-0.9]' 50 ==> class=Iris-setosa 50    conf:(1)
 7. petalwidth='(-inf-0.9]' class=Iris-setosa 50 ==> petallength='(-inf-2.966667]' 50    conf:(1)
 8. petallength='(-inf-2.966667]' class=Iris-setosa 50 ==> petalwidth='(-inf-0.9]' 50    conf:(1)
 9. petallength='(-inf-2.966667]' petalwidth='(-inf-0.9]' 50 ==> class=Iris-setosa 50    conf:(1)
10. class=Iris-setosa 50 ==> petallength='(-inf-2.966667]' petalwidth='(-inf-0.9]' 50    conf:(1)
```

Figure 4: Aporiori algorithm output with numRules = 10, minMetric = 0.9, Metric = Confidence

This algorithm has minimum support =0.3, The number of cycles performed=14, number of rules by default=10 and in total, 29 itemsets are identified. For all 10rules, the value of confidence equals 10 which is greater than minMetric=0.9 that we selected for this analysis. Number of observations in each rule equals 50.

Visualization:

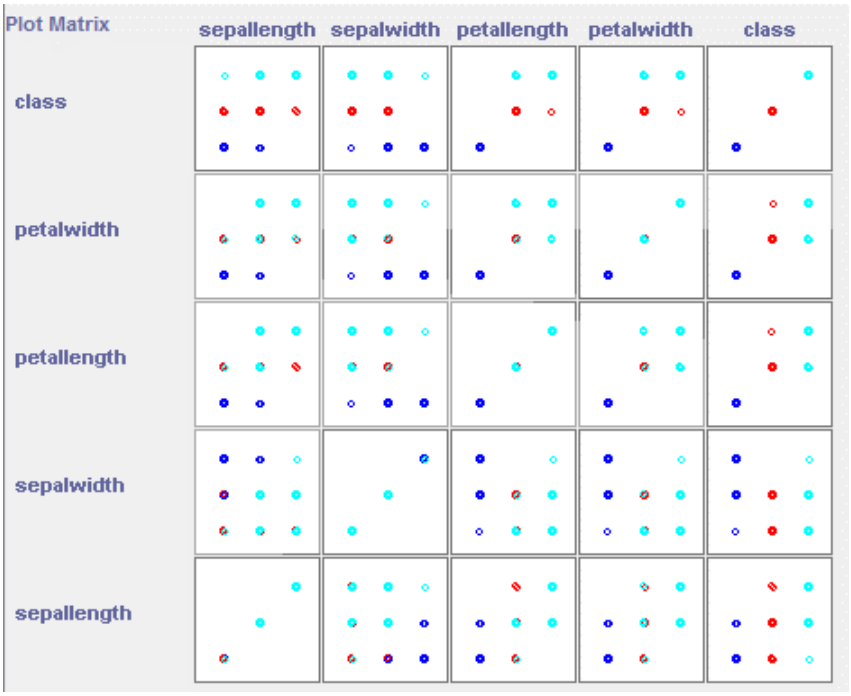

Figure 5: Visualization based on associate analyses.

Describing clustering through association analysis:

After adding new attribute named cluster, we perform associate analysis by selecting K=3, bins=3, Number of rules=100, minMetric=0.9

petallength='(-inf-2.966667]' 50 ==> cluster=cluster3 50 conf:(1)
petalwidth='(-inf-0.9]' 50 ==> cluster=cluster3 50 conf:(1)
petallength='(-inf-2.966667]' petalwidth='(-inf-0.9]' 50 ==> cluster=cluster3 50 conf:(1)
petallength='(2.966667-4.933333]' petalwidth='(0.9-1.7]' 48 ==> cluster=cluster1 48 conf:(1)
sepallength='(-inf-5.5]' petallength='(-inf-2.966667]' 47 ==> cluster=cluster3 47 conf:(1)
sepallength='(-inf-5.5]' petalwidth='(-inf-0.9]' 47 ==> cluster=cluster3 47 conf:(1)
sepallength='(-inf-5.5]' petallength='(-inf-2.966667]' petalwidth='(-inf-0.9]' 47 ==> cluster=cluster3 47 conf:(1)

Figure 6: Table of clusters and the defining values

We see by selecting the number of rules=100, the best rules do not contain the cluster2. Hence, we will increase the number of rules.

Number of rules=1000, minMetric=0.9, k=3, Bins=3

To have sufficient number of rules, we select the rules that have confidence interval $> 90\%$, the rules that do not contain the class attribute and the consequent only contains cluster attribute. So, we will have 43 proper rules by confidence $> 90\%$ We have 32 rules that contain confidence 100%. For each cluster we have at least one rule. We have 16 rules for cluster1, 14 rules for cluster2 and 13 rules for cluster 3 for confidence greater than 90%. Selecting k=3, bin=3 shows that we have a good clustering because the number of rules in each cluster almost equal.

```
1  petallength='(-inf-2.966667]' 50 ==> cluster=cluster3 50    conf:(1)
2  petalwidth='(-inf-0.9]' 50 ==> cluster=cluster3 50    conf:(1)
3  petallength='(-inf-2.966667]' petalwidth='(-inf-0.9]' 50 ==> cluster=cluster3 50    conf:(1)
4  petallength='(2.966667-4.933333]' petalwidth='(0.9-1.7]' 48 ==> cluster=cluster1 48    conf:(1)
5   sepallength='(-inf-5.5]' petallength='(-inf-2.966667]' 47 ==> cluster=cluster3 47    conf:(1)
6  sepallength='(-inf-5.5]' petalwidth='(-inf-0.9]' 47 ==> cluster=cluster3 47    conf:(1)
7  sepallength='(-inf-5.5]' petallength='(-inf-2.966667]' petalwidth='(-inf-0.9]' 47 ==> cluster=cluster3 47    conf:(1)
8  petallength='(4.933333-inf]' petalwidth='(1.7-inf]' 40 ==> cluster=cluster2 40    conf:(1)
9  sepalwidth='(2.8-3.6]' petallength='(-inf-2.966667]' 36 ==> cluster=cluster3 36    conf:(1)
10 sepalwidth='(2.8-3.6]' petalwidth='(-inf-0.9]' 36 ==> cluster=cluster3 36    conf:(1)
11 sepallength='(-inf-5.5]' sepalwidth='(2.8-3.6]' petallength='(-inf-2.966667]' 36 ==> cluster=cluster3 36    conf:(1)
12 sepallength='(-inf-5.5]' sepalwidth='(2.8-3.6]' petalwidth='(-inf-0.9]' 36 ==> cluster=cluster3 36    conf:(1)
13 sepalwidth='(2.8-3.6]' petallength='(-inf-2.966667]' petalwidth='(-inf-0.9]' 36 ==> cluster=cluster3 36    conf:(1)
14 sepallength='(-inf-5.5]' sepalwidth='(2.8-3.6]' petallength='(-inf-2.966667]' petalwidth='(-inf-0.9]' 36 ==> cluster=cluster3 36    conf:(1)
15 sepallength='(5.5-6.7]' petallength='(2.966667-4.933333]' petalwidth='(0.9-1.7]' 33 ==> cluster=cluster1 33    conf:(1)
16 sepalwidth='(-inf-2.8]' petalwidth='(0.9-1.7]' 31 ==> cluster=cluster1 31    conf:(1)
17 sepalwidth='(-inf-2.8]' petallength='(2.966667-4.933333]' 30 ==> cluster=cluster1 30    conf:(1)
18 sepalwidth='(2.8-3.6]' petalwidth='(1.7-inf]' 29 ==> cluster=cluster2 29    conf:(1)
19 sepalwidth='(2.8-3.6]' petallength='(4.933333-inf]' 28 ==> cluster=cluster2 28    conf:(1)
20 sepalwidth='(-inf-2.8]' petallength='(2.966667-4.933333]' petalwidth='(0.9-1.7]' 27 ==> cluster=cluster1 27    conf:(1)
21 sepalwidth='(2.8-3.6]' petallength='(4.933333-inf]' petalwidth='(1.7-inf]' 26 ==> cluster=cluster2 26    conf:(1)
22 sepallength='(5.5-6.7]' petallength='(4.933333-inf]' petalwidth='(1.7-inf]' 24 ==> cluster=cluster2 24    conf:(1)
23 sepalwidth='(2.8-3.6]' petallength='(2.966667-4.933333]' petalwidth='(0.9-1.7]' 21 ==> cluster=cluster1 21    conf:(1)
24 sepallength='(5.5-6.7]' sepalwidth='(-inf-2.8]' petalwidth='(0.9-1.7]' 19 ==> cluster=cluster1 19    conf:(1)
25  sepallength='(5.5-6.7]' sepalwidth='(-inf-2.8]' petallength='(2.966667-4.933333]' 18 ==> cluster=cluster1 18    conf:(1)
26 sepallength='(5.5-6.7]' sepalwidth='(2.8-3.6]' petalwidth='(1.7-inf]' 18 ==> cluster=cluster2 18    conf:(1)
27 sepallength='(5.5-6.7]' sepalwidth='(2.8-3.6]' petallength='(2.966667-4.933333]' petalwidth='(0.9-1.7]' 18 ==> cluster=cluster1 18    conf:(1)
28 sepallength='(6.7-inf]' petallength='(4.933333-inf]' 17 ==> cluster=cluster2 17    conf:(1)
29 sepallength='(6.7-inf]' petalwidth='(1.7-inf]' 16 ==> cluster=cluster2 16    conf:(1)
30 sepallength='(5.5-6.7]' sepalwidth='(2.8-3.6]' petallength='(4.933333-inf]' 16 ==> cluster=cluster2 16    conf:(1)
31 sepallength='(6.7-inf]' petallength='(4.933333-inf]' petalwidth='(1.7-inf]' 16 ==> cluster=cluster2 16    conf:(1)
32 sepallength='(5.5-6.7]' sepalwidth='(-inf-2.8]' petallength='(2.966667-4.933333]' petalwidth='(0.9-1.7]' 15 ==> cluster=cluster1 15    conf:(1)
33 sepallength='(5.5-6.7]' sepalwidth='(2.8-3.6]' petallength='(4.933333-inf]' petalwidth='(1.7-inf]' 15 ==> cluster=cluster2 15    conf:(1)
34 sepallength='(5.5-6.7]' petalwidth='(0.9-1.7]' 38 ==> cluster=cluster1 37    conf:(0.97)
35 sepallength='(-inf-5.5]' sepalwidth='(2.8-3.6]' 37 ==> cluster=cluster3 36    conf:(0.97)
36 petalwidth='(0.9-1.7]' 54 ==> cluster=cluster1 52    conf:(0.96)
37 sepallength='(5.5-6.7]' sepalwidth='(2.8-3.6]' petalwidth='(0.9-1.7]' 19 ==> cluster=cluster1 18    conf:(0.95)
38 petallength='(2.966667-4.933333]' 54 ==> cluster=cluster1 51    conf:(0.94)
39 petalwidth='(1.7-inf]' 46 ==> cluster=cluster2 43    conf:(0.93)
40 sepallength='(5.5-6.7]' petallength='(2.966667-4.933333]' 39 ==> cluster=cluster1 36    conf:(0.92)
41 petallength='(4.933333-inf]' 46 ==> cluster=cluster2 42    conf:(0.91)
42 sepalwidth='(2.8-3.6]' petalwidth='(0.9-1.7]' 23 ==> cluster=cluster1 21    conf:(0.91)
43 sepallength='(5.5-6.7]' petalwidth='(1.7-inf]' 30 ==> cluster=cluster2 27    conf:(0.9)
44
```

Figure 7: Table of rules and maximum occurrence

Table 2 demonstrate the rules with maximum occurrences in each cluster by confidence greater than 90%:

Rules	Cluster	Occurrences	Confidence
petalwidth='(0.9-1.7]'	1	54	96
petallength='(2.966667-4.933333]'	1	54	94
petallength='(2.966667-4.933333]' petalwidth='(0.9-1.7]'	1	48	100
petalwidth='(1.7-inf)'	2	46	93
petallength='(4.933333-inf)'	2	46	91
petallength='(4.933333-inf)' petalwidth='(1.7-inf)'	2	40	100
petallength='(-inf-2.966667]'	3	50	100
petalwidth='(-inf-0.9]'	3	50	100
petallength='(-inf-2.966667]' petalwidth='(-inf-0.9]'	3	50	100

Table 2

We can see in different rules with specific observations in each cluster.

Additional analysis:

In this part we want to see the effect of different number of bins, different number of clustering, different type of clustering on the quality of clustering and best rules.

Different number of bins:

The first we examine different number of bins in discretization. Then apply Kmeans clustering by K=3 to see the confusion matrix and number of incorrectly clustered instances.

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

Clustered Instances

0       54  ( 36%)
1       66  ( 44%)
2       30  ( 20%)


Class attribute: class
Classes to Clusters:

  0  1  2  <-- assigned to cluster
 50  0  0 | Iris-setosa
  4 46  0 | Iris-versicolor
  0 20 30 | Iris-virginica

Cluster 0 <-- Iris-setosa
Cluster 1 <-- Iris-versicolor
Cluster 2 <-- Iris-virginica

Incorrectly clustered instances :       24.0     16      %
```

Figure 8: Model output with K=3, Bin=4

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

Clustered Instances

0       63 ( 42%)
1       35 ( 23%)
2       52 ( 35%)


Class attribute: class
Classes to Clusters:

  0  1  2  <-- assigned to cluster
  0  0 50 | Iris-setosa
 15 33  2 | Iris-versicolor
 48  2  0 | Iris-virginica

Cluster 0 <-- Iris-virginica
Cluster 1 <-- Iris-versicolor
Cluster 2 <-- Iris-setosa

Incorrectly clustered instances :       19.0     12.6667 %
```

Figure 9: Model output with K=3, Bin=5

By applying K=3, number of bins=5, in confusion matrix we see the number of incorrectly clustered instances equal 19(12%). With these values we perform associated analysis by Apriori algorithm. By considering minMetric=0.9, metric type=Confidence, number of rules=100. As we want to predict the class of new instance, we don't select class attribute in the antecedent part, and as we want to use attributes like petal and sepal for prediction, we select cluster attribute in the consequent. By applying these filters we will see 10 number of best rules which only consists of Cluster3. Hence we conclude this type of clustering will not be a proper clustering as it does not contain cluster1 and cluster2.

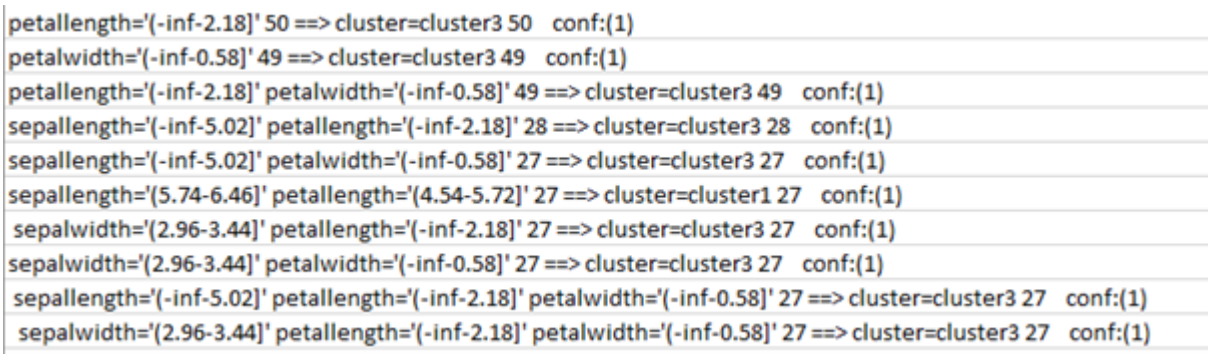

Figure 10: Model output with K=3,Bin=10

By performing different values of bins for discretization, we see that by increasing the number of bins, confusion matrix is getting worse. And for number of bins=5, we try associate analysis for Apriori algorithm, and it demonstrates the best rules for confidence $> 90\%$ do not contain all clusters. It limited to cluster3. Hence, we select Bin=3 for discretization. Now we should examine different number of clustering in Kmeans clustering.

Different number of clusters in Kmeans clustering:

As we have three different species(classes), three clusters are appropriate for clustering. But we can examine different number of clusters to see the confusion matrix and best rules for each cluster.

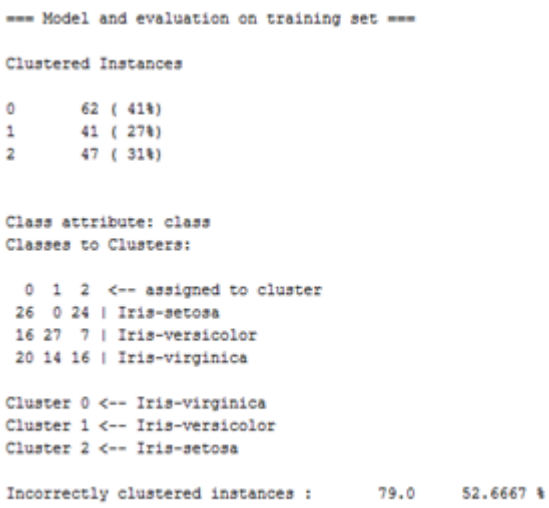

Figure 11: Model output with Bin=3, K=4

When the number of clusters, k= 4, we see in the output, the distribution of instances in each cluster are not equal. In cluster4, we have 4 instances, but the number of instances in three other clusters completely differ. In cluster 3 that related to $Iris_setosa$, all of instances are classified correctly. After associate analysis for Aprioro algorithm, by considering minMetric=0.9, metric type=Confidence, number of rules=100, we see best rules only consist of cluster3.Hence this kind of clustering could not be a good clustering, because in output of rules , two other clusters will not have any role.

```
1  petallength='(-inf-2.966667]' 50 ==> cluster=cluster3 50   conf:(1)
2  petalwidth='(-inf-0.9]' 50 ==> cluster=cluster3 50   conf:(1)
3  petallength='(-inf-2.966667]' petalwidth='(-inf-0.9]' 50 ==> cluster=cluster3 50   conf:(1)
4  sepallength='(-inf-5.5]' petallength='(-inf-2.966667]' 47 ==> cluster=cluster3 47   conf:(1)
5  sepallength='(-inf-5.5]' petalwidth='(-inf-0.9]' 47 ==> cluster=cluster3 47   conf:(1)
6  sepallength='(-inf-5.5]' petallength='(-inf-2.966667]' petalwidth='(-inf-0.9]' 47 ==> cluster=cluster3 47   conf:(1)
```

Figure 12: Model output with N=3, bins=3

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

Clustered Instances

0       52  ( 35%)
1       44  ( 29%)
2       50  ( 33%)
3        4  (  3%)


Class attribute: class
Classes to Clusters:

  0  1  2  3  <-- assigned to cluster
  0  0 50  0 | Iris-setosa
 45  2  0  3 | Iris-versicolor
  7 42  0  1 | Iris-virginica

Cluster 0 <-- Iris-versicolor
Cluster 1 <-- Iris-virginica
Cluster 2 <-- Iris-setosa
Cluster 3 <-- No class

Incorrectly clustered instances :       13.0      8.6667 %
```

Figure 13:

By selecting the number of cluster=6,the number of incorrectly clustered instances will increase in the confusion matrix. If we ignore this issue and try associate analysis, we will see again the output in best rules, do not contain all clusters.
In both, we will not see any rules that contain cluster3 and cluster4. By ex-

amining different number of bins and different number of clustering in Simple Kmeans clustering, we see the k=3, bins=3 are the proper values to have the best rules for prediction.

Clustering:

We examined different values of bin and k for simple kmeans clustering. Now we change kind of clustering.

At first we perform Hierarichal clustering by N=3. As we see, the number of instances are not distributed equally in all three clusters in figure 14. So we change the number of cluster.

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

Clustered Instances

0       50 ( 33%)
1       99 ( 66%)
2        1 (  1%)


Class attribute: class
Classes to Clusters:

  0  1  2  <-- assigned to cluster
 50  0  0 | Iris-setosa
  0 50  0 | Iris-versicolor
  0 49  1 | Iris-virginica

Cluster 0 <-- Iris-setosa
Cluster 1 <-- Iris-versicolor
Cluster 2 <-- Iris-virginica

Incorrectly clustered instances :       49.0     32.6667 %
```

Figure 14: Hierarichal Cluster model output with N=3, bins=3

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

Clustered Instances

0        45 ( 30%)
1        40 ( 27%)
2        14 (  9%)
3         4 (  3%)
4        36 ( 24%)
5        11 (  7%)


Class attribute: class
Classes to Clusters:

  0  1  2  3  4  5  <-- assigned to cluster
  0  0 14  0 36  0 | Iris-setosa
 37  0  0  3  0 10 | Iris-versicolor
  8 40  0  1  0  1 | Iris-virginica

Cluster 0 <-- Iris-versicolor
Cluster 1 <-- Iris-virginica
Cluster 2 <-- No class
Cluster 3 <-- No class
Cluster 4 <-- Iris-setosa
Cluster 5 <-- No class

Incorrectly clustered instances :        37.0      24.6667 %
```

Figure 15: Model output with N=6, bins=3

Like previous test, the instances are not distributed equally in all 6 clusters and incorrectly clustered instances are high. By selecting different values of bins(bins=5) we will not see obvious differences in output. This kind of clustering is not appropriate clustering.

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

Clustered Instances

0       50 ( 33%)
1       99 ( 66%)
2        1 (  1%)


Class attribute: class
Classes to Clusters:

  0  1  2  <-- assigned to cluster
 50  0  0 | Iris-setosa
  0 50  0 | Iris-versicolor
  0 49  1 | Iris-virginica

Cluster 0 <-- Iris-setosa
Cluster 1 <-- Iris-versicolor
Cluster 2 <-- Iris-virginica

Incorrectly clustered instances :       49.0     32.6667 %
```

Figure 16: EM model output with N=3, bins=3

In EM clustering we define N=3, bins=3. The number of instances are divided equally in three different clusters and the value of incorrectly clustered instances are reasonable. So we will do associate analysis by Apriori algorithm:

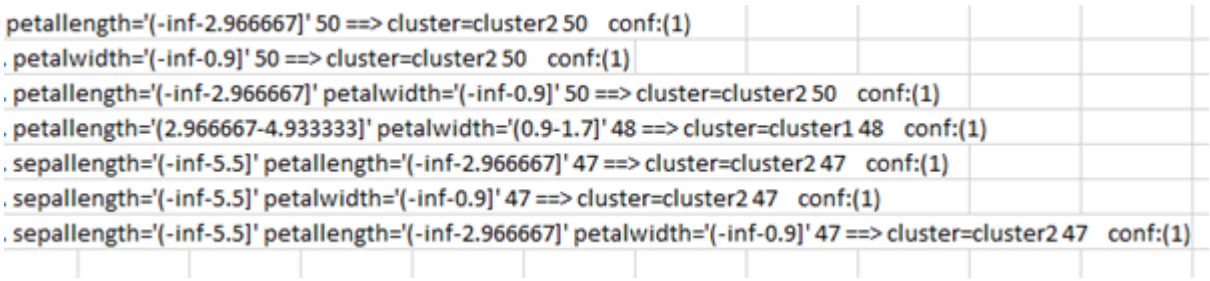

Figure 17:

At first we select the number of rules=100, but the output do not contain all clusters, hence we increase the number of rules to have more best rules to contain all clusters with confidence $> 90\%$
We will see 42 different rules as best rules which have confidence $> 90\%$, and contains all three clusters:
16 rules for cluster 1, 13 rules for cluster2 and 13 rules for cluster3 with confidence $> 90\%$.
Now we select the best rules for each cluster by maximum number of occurrences.

Rules	cluster	occurrences	confidence
petalwidth='(0.9-1.7]'	1	53	98
petallength='(2.966667-4.933333]'	1	51	94
petallength='(2.966667-4.933333]' petalwidth='(0.9-1.7]'	1	48	100
petallength='(-inf-2.966667]'	2	50	100
petalwidth='(-inf-0.9]' 50	2	50	100
petallength='(-inf-2.966667]' petalwidth='(-inf-0.9]'	2	50	100
petalwidth='(1.7-inf)' 46 ==>cluster=cluster3	3	43	93
petallength='(4.933333-inf)' petalwidth='(1.7-inf)'	3	40	100

Table 3

In comparing simple kmeans clustering (k=3,bins=3) and EM clustering(k=3, bins=3), the second method of clustering is better. In both of them by applying 1000 rules, the outputs of both clustering are same, but in EM algorithm , the number of incorrect clustered instances are lower than it in the simple kmeans clustering.