Lab2

Andreas Stasinakis & Jesper Lindberg

March 1, 2019

Contributors: Jesper Lindberg(jesli060) & Andreas Stasinakis(andst745)

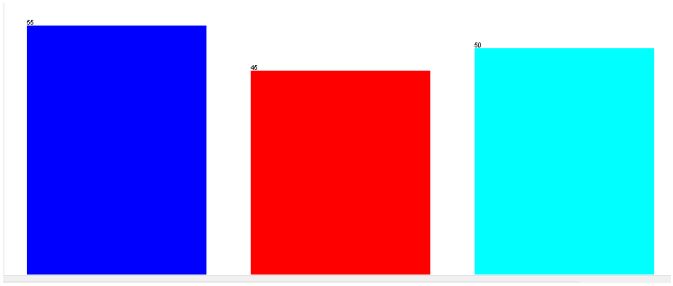
In this lab we perform association analysis using Apriori algorithm in the Iris dataset. The first thing we have to do here is to discreatize our continuous attributes using the discretize filter from weka. We discretize all the continuous attributes and we use 3 bins which is the number of states of the discretized attributes. We also try different values of the parameter bins in the next tasks, but our first choice is 3. In this way now, we have discrete variables we need in order to continue the procedure.

The second step, after discreatize our data, is to create one more attribute which represents the cluster label assigned to each instance. The optimal number of cluster labels is 3(the recommended one). We will also execute this procedure for $k = 3, 4, 5$ and we present the results below.

For now, we split our data in 3 bins, we use k means with 3 clusters and we produce associations rules.

The k means, as discussed in previous labs, did a really decent job while only 9 observations misclassified as can be seen in the figure below. We also plot how the observations are splited, using 3 clusters, and it is really make sense to choose 3 cluster because the observations are divided almost equally.

```
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      %
```



One key parameter for this procedure is the number of rules we want the algorithm to search for. We run it for many different values. Our goal is to find association rules for *ALL* the clusters(in this case 3). For example when we run the algorithm for the default value 10, the obtained association rules occur only for the cluster 1 and 3. In order to find associations rule for all the clusters, we have to choose 23 in the parameter NumRules. In that case we obtain the results below:

```
Best rules found:

 1. sepallength='(-inf-5.5]' petallength='(-inf-2.966667]' 47 ==> cluster=cluster3 47    conf:(1)
 2. sepallength='(-inf-5.5]' petalwidth='(-inf-0.9]' 47 ==> cluster=cluster3 47    conf:(1)
 3. sepallength='(-inf-5.5]' petallength='(-inf-2.966667]' petalwidth='(-inf-0.9]' 47 ==> cluster=cluster3 47    conf:(1)
 4. sepallength='(5.5-6.7]' petallength='(2.966667-4.933333]' 39 ==> cluster=cluster1 39    conf:(1)
 5. sepallength='(-inf-5.5]' sepalwidth='(2.8-3.6]' 37 ==> cluster=cluster3 37    conf:(1)
 6. sepalwidth='(2.8-3.6]' petallength='(-inf-2.966667]' 36 ==> cluster=cluster3 36    conf:(1)
 7. sepalwidth='(2.8-3.6]' petalwidth='(-inf-0.9]' 36 ==> cluster=cluster3 36    conf:(1)
 8. sepallength='(-inf-5.5]' sepalwidth='(2.8-3.6]' petallength='(-inf-2.966667]' 36 ==> cluster=cluster3 36    conf:(1)
 9. sepallength='(-inf-5.5]' sepalwidth='(2.8-3.6]' petalwidth='(-inf-0.9]' 36 ==> cluster=cluster3 36    conf:(1)
10. sepalwidth='(2.8-3.6]' petallength='(-inf-2.966667]' petalwidth='(-inf-0.9]' 36 ==> cluster=cluster3 36    conf:(1)
11. sepallength='(-inf-5.5]' sepalwidth='(2.8-3.6]' petallength='(-inf-2.966667]' petalwidth='(-inf-0.9]' 36 ==> cluster=cluster3 36    conf:(1)
12. sepallength='(5.5-6.7]' petallength='(2.966667-4.933333]' petalwidth='(0.9-1.7]' 33 ==> cluster=cluster1 33    conf:(1)
13. sepallength='(5.5-6.7]' sepalwidth='(-inf-2.8]' 31 ==> cluster=cluster1 31    conf:(1)
14. sepalwidth='(-inf-2.8]' petalwidth='(0.9-1.7]' 31 ==> cluster=cluster1 31    conf:(1)
15. sepalwidth='(-inf-2.8]' petallength='(2.966667-4.933333]' 30 ==> cluster=cluster1 30    conf:(1)
16. sepalwidth='(2.8-3.6]' petallength='(4.933333-inf)' 28 ==> cluster=cluster2 28    conf:(1)
17. sepalwidth='(-inf-2.8]' petallength='(2.966667-4.933333]' petalwidth='(0.9-1.7]' 27 ==> cluster=cluster1 27    conf:(1)
18. sepalwidth='(2.8-3.6]' petallength='(4.933333-inf)' petalwidth='(1.7-inf)' 26 ==> cluster=cluster2 26    conf:(1)
19. sepallength='(5.5-6.7]' petalwidth='(0.9-1.7]' 38 ==> cluster=cluster1 37    conf:(0.97)
20. petallength='(2.966667-4.933333]' 54 ==> cluster=cluster1 51    conf:(0.94)
21. petallength='(-inf-2.966667]' 50 ==> cluster=cluster3 47    conf:(0.94)
22. petalwidth='(-inf-0.9]' 50 ==> cluster=cluster3 47    conf:(0.94)
23. petallength='(-inf-2.966667]' petalwidth='(-inf-0.9]' 50 ==> cluster=cluster3 47    conf:(0.94)
```

For itemsets of length 1: Both cluster 1 and 3 characterized mostly by the Petal length and there is no candidate for cluster 2.

For itemsets with length more than 1, it seems that for cluster 1 and 3 sepal length and petal length occur in the rules with the highest support value and confidence. For cluster 3 this rule has 47 support value and 100% confidence and for cluster 1, 39 support value and also 100% confidence. For the second cluster, only two association rule occurs which are sepal width and petal length(support = 28 and 100% confidence) and sepal width, petal length and petal width(support = 26 and 100% confidence).

Now we will repeat the same procedure but we will change some parameters in order to find the optimal values for them.

As mentioned before we have to discretize our continuous variables before starting the procedure. It is very important to choose a efficient number of bins, which is the number of intervals that we want to split our data. In this initial procedure we use only 3 bins. We now use different number of bins in order to compare the results and comment the optimal one. So we run the algorithm again, using number of bins equal to 3,5,10. We present all the results obtained, below:

For 5 bins and 20 rules:

```
Best rules found:

 1. petallength='(4.54-5.72]' 47 ==> cluster=cluster1 47    conf:(1)
 2. sepallength='(5.74-6.46]' 42 ==> cluster=cluster1 42    conf:(1)
 3. sepallength='(-inf-5.02]' petallength='(-inf-2.18]' 28 ==> cluster=cluster3 28    conf:(1)
 4. sepallength='(-inf-5.02]' petalwidth='(-inf-0.58]' 27 ==> cluster=cluster3 27    conf:(1)
 5. sepallength='(5.74-6.46]' petallength='(4.54-5.72]' 27 ==> cluster=cluster1 27    conf:(1)
 6. sepalwidth='(2.96-3.44]' petallength='(-inf-2.18]' 27 ==> cluster=cluster3 27    conf:(1)
 7. sepalwidth='(2.96-3.44]' petalwidth='(-inf-0.58]' 27 ==> cluster=cluster3 27    conf:(1)
 8. sepallength='(-inf-5.02]' petallength='(-inf-2.18]' petalwidth='(-inf-0.58]' 27 ==> cluster=cluster3 27    conf:(1)
 9. sepalwidth='(2.96-3.44]' petallength='(-inf-2.18]' petalwidth='(-inf-0.58]' 27 ==> cluster=cluster3 27    conf:(1)
10. sepalwidth='(2.96-3.44]' petallength='(4.54-5.72]' 25 ==> cluster=cluster1 25    conf:(1)
11. sepallength='(5.74-6.46]' sepalwidth='(2.48-2.96]' 23 ==> cluster=cluster1 23    conf:(1)
12. sepallength='(-inf-5.02]' sepalwidth='(2.96-3.44]' 22 ==> cluster=cluster3 22    conf:(1)
13. sepallength='(-inf-5.02]' sepalwidth='(2.96-3.44]' petallength='(-inf-2.18]' 22 ==> cluster=cluster3 22    conf:(1)
14. sepallength='(-inf-5.02]' sepalwidth='(2.96-3.44]' petalwidth='(-inf-0.58]' 22 ==> cluster=cluster3 22    conf:(1)
15. sepallength='(-inf-5.02]' sepalwidth='(2.96-3.44]' petallength='(-inf-2.18]' petalwidth='(-inf-0.58]' 22 ==> cluster=cluster3 22    conf:(1)
16. sepalwidth='(2.48-2.96]' petallength='(4.54-5.72]' 21 ==> cluster=cluster1 21    conf:(1)
17. petallength='(4.54-5.72]' petalwidth='(1.54-2.02]' 20 ==> cluster=cluster1 20    conf:(1)
18. sepallength='(6.46-7.18]' petallength='(4.54-5.72]' 18 ==> cluster=cluster1 18    conf:(1)
19. sepallength='(5.02-5.74]' petallength='(3.36-4.54]' 17 ==> cluster=cluster2 17    conf:(1)
20. sepallength='(5.74-6.46]' petalwidth='(1.06-1.54]' 17 ==> cluster=cluster1 17    conf:(1)
```

For 10 bins and 20 rules:

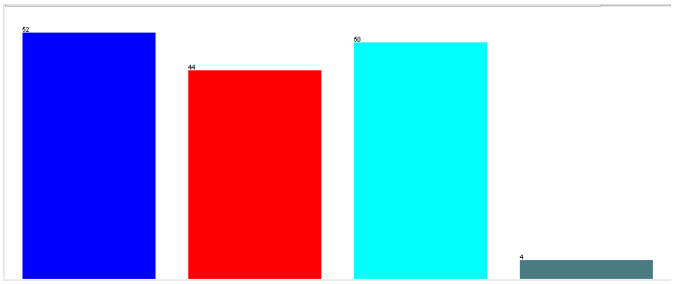
```
Generated sets of large itemsets:

Size of set of large itemsets L(1): 14

Size of set of large itemsets L(2): 4

Best rules found:

 1. sepalwidth='(2.48-2.72]' 22 ==> cluster=cluster1 22    conf:(1)
 2. sepallength='(4.66-5.02]' petalwidth='(-inf-0.34]' 17 ==> cluster=cluster3 17    conf:(1)
 3. sepalwidth='(2.96-3.2]' petalwidth='(-inf-0.34]' 16 ==> cluster=cluster3 16    conf:(1)
 4. sepallength='(4.66-5.02]' 23 ==> cluster=cluster3 22    conf:(0.96)
 5. sepallength='(5.74-6.1]' 22 ==> cluster=cluster1 21    conf:(0.95)
```

First of all it is obvious that cluster 2 is the cluster which is captured really difficult. In all different parameters, we can find many rules for cluster 1 and 3 but not for 2. More specific, as easily can be seen from the figure above, for 10 bins we do not obtain any association rule for cluster 2. Despite the fact, we asked for 20 rules, the algorithm could only find 5. So we need smaller intervals in order to capture all the clusters. Also if we split the data only into 5 intervals we have the picture below, from which we can conclude that the misclassification rate increases while the number of bins increases. For that reason we will continue with 3 bins.

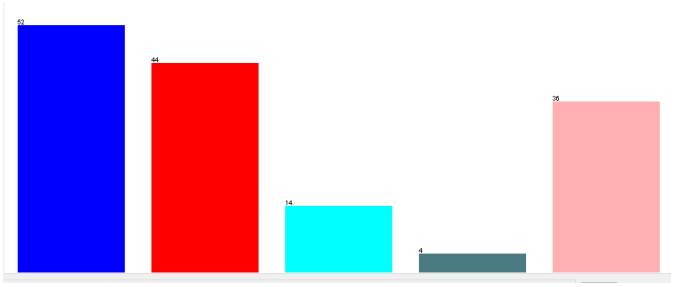
```
Class attribute: class
Classes to Clusters:

  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 %
```

As mentioned before, we will run the procedure for different number of clusters. We plot the histograms of the observations, after using the k means algorithm in order to classify the observations.

For 4 clusters:



For 5 clusters:



It seems that when we use $k = 4$ or $k = 5$, there is one cluster with only 4 observations. This is reasonable because we only have 3 classes of flowers, but when we use a different k, the K- means algorithm tries to "split" the observations in k clusters. Of course we have to also run the appriori algorithm in order to be sure that 3 is the optimal number for clusters. So we will run the procedure for 5 clusters and 3 bins and we will compare it the initial one(3 clusters and 3 bins).

First of all, we should expect that the misclassification rate will rise because we have 3 real classes but we are trying to classify our data using 5 means algorithm. The figure below confirms this statement.

```
Clustered Instances

0       52 ( 35%)
1       44 ( 29%)
2       14 (  9%)
3        4 (  3%)
4       36 ( 24%)


Class attribute: class
Classes to Clusters:

  0  1  2  3  4  <-- assigned to cluster
  0  0 14  0 36 | Iris-setosa
 45  2  0  3  0 | Iris-versicolor
  7 42  0  1  0 | Iris-virginica

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

Incorrectly clustered instances :       27.0     18      %
```

For the appriori algorithm, we could only obtain 31 associations rules despite the fact that we add as number of rules 1000. From the plot it is clear that there is no rule for the cluster 3 and 4. If we want rules for all

clusters, we should probably change the minimum support parameter we use.

```
Best rules found:

 1. sepallength='(5.5-6.7]' petallength='(2.966667-4.933333]' 39 ==> cluster=cluster1 39    conf:(1)
 2. sepalwidth='(2.8-3.6]' petallength='(-inf-2.966667]' 36 ==> cluster=cluster5 36    conf:(1)
 3. sepalwidth='(2.8-3.6]' petalwidth='(-inf-0.9]' 36 ==> cluster=cluster5 36    conf:(1)
 4. sepallength='(-inf-5.5]' sepalwidth='(2.8-3.6]' petallength='(-inf-2.966667]' 36 ==> cluster=cluster5 36    conf:(1)
 5. sepallength='(-inf-5.5]' sepalwidth='(2.8-3.6]' petalwidth='(-inf-0.9]' 36 ==> cluster=cluster5 36    conf:(1)
 6. sepalwidth='(2.8-3.6]' petallength='(-inf-2.966667]' petalwidth='(-inf-0.9]' 36 ==> cluster=cluster5 36    conf:(1)
 7. sepallength='(-inf-5.5]' sepalwidth='(2.8-3.6]' petallength='(-inf-2.966667]' petalwidth='(-inf-0.9]' 36 ==> cluster=cluster5 36    conf:(1)
 8. sepallength='(5.5-6.7]' petallength='(2.966667-4.933333]' petalwidth='(0.9-1.7]' 33 ==> cluster=cluster1 33    conf:(1)
 9. sepallength='(5.5-6.7]' sepalwidth='(-inf-2.8]' 31 ==> cluster=cluster1 31    conf:(1)
10. sepalwidth='(-inf-2.8]' petalwidth='(0.9-1.7]' 31 ==> cluster=cluster1 31    conf:(1)
11. sepalwidth='(-inf-2.8]' petallength='(2.966667-4.933333]' 30 ==> cluster=cluster1 30    conf:(1)
12. sepalwidth='(2.8-3.6]' petallength='(4.933333-inf)' 28 ==> cluster=cluster2 28    conf:(1)
13. sepalwidth='(-inf-2.8]' petallength='(2.966667-4.933333]' petalwidth='(0.9-1.7]' 27 ==> cluster=cluster1 27    conf:(1)
14. sepalwidth='(2.8-3.6]' petallength='(4.933333-inf)' petalwidth='(1.7-inf)' 26 ==> cluster=cluster2 26    conf:(1)
15. sepallength='(5.5-6.7]' sepalwidth='(2.8-3.6]' petallength='(2.966667-4.933333]' 21 ==> cluster=cluster1 21    conf:(1)
16. sepallength='(5.5-6.7]' sepalwidth='(-inf-2.8]' petalwidth='(0.9-1.7]' 19 ==> cluster=cluster1 19    conf:(1)
17. sepallength='(5.5-6.7]' sepalwidth='(-inf-2.8]' petallength='(2.966667-4.933333]' 18 ==> cluster=cluster1 18    conf:(1)
18. 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)
19. sepallength='(6.7-inf)' petallength='(4.933333-inf)' 17 ==> cluster=cluster2 17    conf:(1)
20. sepallength='(6.7-inf)' petalwidth='(1.7-inf)' 16 ==> cluster=cluster2 16    conf:(1)
21. sepallength='(5.5-6.7]' sepalwidth='(2.8-3.6]' petallength='(4.933333-inf)' 16 ==> cluster=cluster2 16    conf:(1)
22. sepallength='(6.7-inf)' petallength='(4.933333-inf)' petalwidth='(1.7-inf)' 16 ==> cluster=cluster2 16    conf:(1)
23. 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)
24. sepallength='(5.5-6.7]' sepalwidth='(2.8-3.6]' petallength='(4.933333-inf)' petalwidth='(1.7-inf)' 15 ==> cluster=cluster2 15    conf:(1)
25. sepallength='(5.5-6.7]' petalwidth='(0.9-1.7]' 38 ==> cluster=cluster1 37    conf:(0.97)
26. sepallength='(-inf-5.5]' sepalwidth='(2.8-3.6]' 37 ==> cluster=cluster5 36    conf:(0.97)
27. sepallength='(5.5-6.7]' sepalwidth='(2.8-3.6]' petalwidth='(0.9-1.7]' 19 ==> cluster=cluster1 18    conf:(0.95)
28. petallength='(2.966667-4.933333]' 54 ==> cluster=cluster1 51    conf:(0.94)
29. petallength='(2.966667-4.933333]' petalwidth='(0.9-1.7]' 48 ==> cluster=cluster1 45    conf:(0.94)
30. sepalwidth='(-inf-2.8]' 47 ==> cluster=cluster1 43    conf:(0.91)
31. petalwidth='(0.9-1.7]' 54 ==> cluster=cluster1 49    conf:(0.91)
```

For the cluster 1 we have the same rule as with 3 clusters, sepal length and petal length with support 39 and confidence 100%. For cluster 2, 5 the highest support is for the rule sepal width, petal length with support 28 and 36 and confidence 100%.

From the results above it is obvious that we should choose 3 clusters. Therefore, in the end we should choose 3 clusters and 3 bins in order to find association rules for all clusters. We now choose the "best" rule for each cluster. Our choice depends on the maximum confidence and maximum support. So for the first cluster we have the rule:

For the first cluster:

```
4. sepallength='(5.5-6.7]' petallength='(2.966667-4.933333]' 39 ==> cluster=cluster1 39    conf:(1)
```

For the second cluster:

```
16. sepalwidth='(2.8-3.6]' petallength='(4.933333-inf)' 28 ==> cluster=cluster2 28    conf:(1)
```

For the third cluster:

```
1. sepallength='(-inf-5.5]' petallength='(-inf-2.966667]' 47 ==> cluster=cluster3 47    conf:(1)
```