UMBC CSEE

Foundations of Data Mining

CS-691

Homework Assignment - 3

Sandipan Dey

2009

1. First the iris text data provided needs to be converted to **weka-compatible attribute relation file format (.arff)** that looks like the following:

% Iris

@RELATION iris

@ATTRIBUTE Sepal_length NUMERIC
@ATTRIBUTE Sepal_width NUMERIC
@ATTRIBUTE Petal_length NUMERIC
@ATTRIBUTE Petal_width NUMERIC

@DATA

2	14	33	50
24	56	31	67
23	51	31	69
2	10	36	46
20	52	30	65
19	51	27	58
13	45	28	57
16	47	33	63
…	…	…	…
…	…	…	…
…	…	…	…

As can be seen from above, only the last four attributes from the text data are selected.

If we run KMeans algorithm in Weka with the .arff file as the input file,

weka.clusterers.SimpleKMeans -V -N <numClusters> -A <distanceFunction> -I maxIteration -S seed

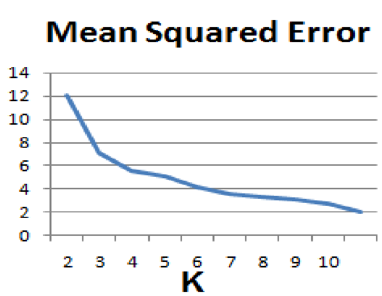
- Initial number of clusters is chosen as **3** (since we suspect that **specifies_name** attribute, or equivalently specifies_no attribute denotes the **class labels** for the data – those already obtained by applying some supervised method and there are exactly **3 different class labels**, namely, **I.Setosa, I.Verginica, I.Versicolor**)
- distanceFunction as **Euclidian**
- maximum iteration (to converge) as **500**
- random seed (to select initial cluster centers) as **10**
- display standard deviation option is selected

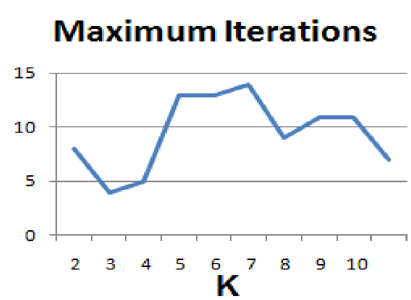
The following output is obtained after the algorithm terminates in 4 iterations, with **MSE** 7.12:

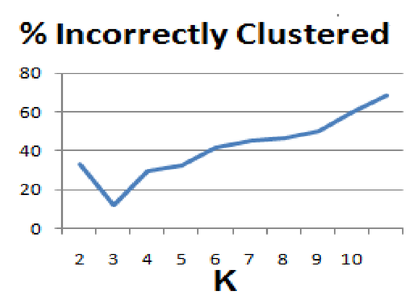
```
=== Run information ===

Scheme:       weka.clusterers.SimpleKMeans -V -N 3 -A "weka
Relation:     iris
Instances:    150
Attributes:   4
              Sepal_length
              Sepal_width
              Petal_length
              Petal_width
Test mode:    evaluate on training data

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


kMeans
======

Number of iterations: 4
Within cluster sum of squared errors: 7.115548372424189
Missing values globally replaced with mean/mode

Cluster centroids:
                           Cluster#
Attribute       Full Data          0          1          2
                    (150)       (38)       (62)       (50)
==========================================================
Sepal_length      11.9267    20.7368    14.1613       2.46
                 +/-7.569  +/-2.8254  +/-2.7051  +/-1.0539

Sepal_width       37.7867    57.1579    44.5968      14.62
               +/-17.7762  +/-5.1963  +/-5.6784  +/-1.7366

Petal_length      30.5533    30.8421     27.371      34.28
                +/-4.3728  +/-2.8335  +/-2.9266  +/-3.7906

Petal_width       58.4467       68.5    59.0161       50.1
                +/-8.2686  +/-5.0871  +/-4.5682  +/-3.5355



Clustered Instances

0       38 ( 25%)
1       62 ( 41%)
2       50 ( 33%)
```

If the **classes to cluster evaluation option** from weka (with class label as **specifies_name**, that is to be **ignored** when **running KMeans** but the resulting **cluster-membership** for each instance is to be **validated against** the corresponding **class labels**) is used to verify the clusters generated, the following output is obtained additionally:

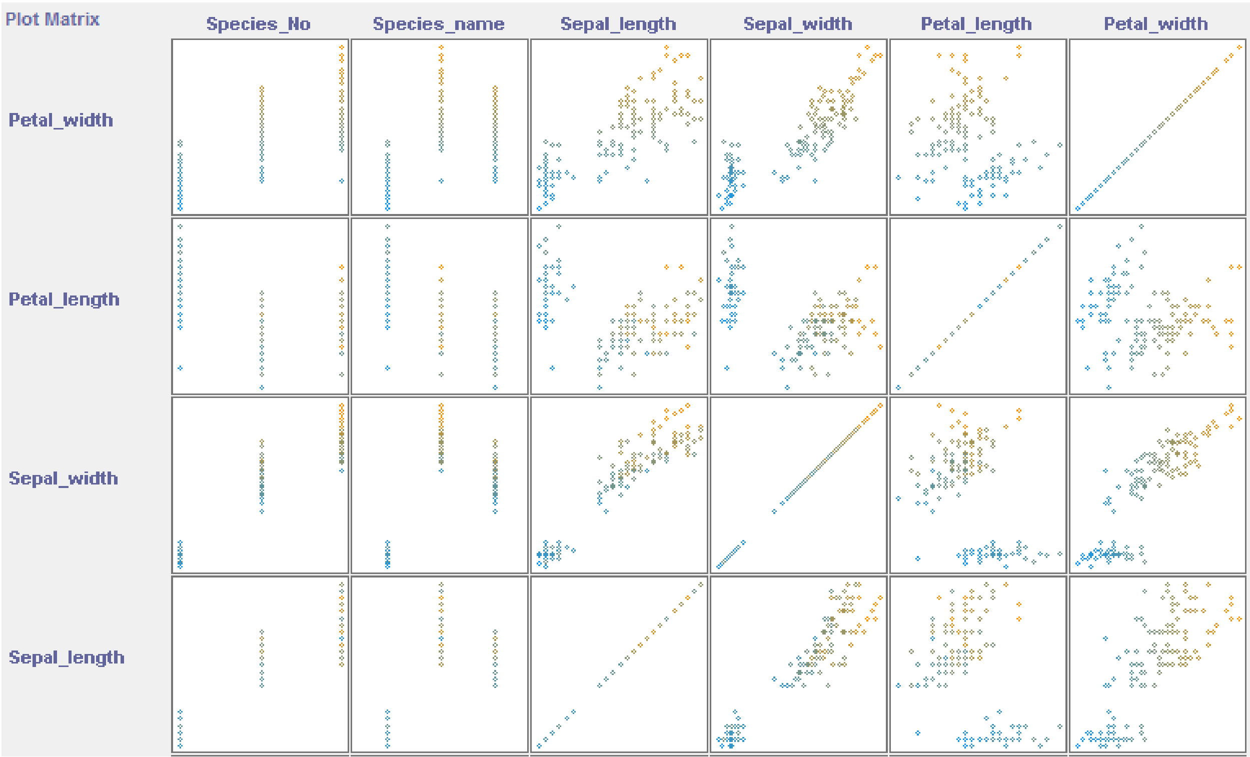
```
                                     Class attribute: Species_name          Total: 150 Instances
K       Clustered Instances          Classes to Clusters:                   Incorrectly clustered instances

2       0      100 ( 67%)             0  1  <-- assigned to cluster            50.0     33.3333 %
        1       50 ( 33%)             0 50 | I.Setosa
                                     50  0 | I.Verginica
                                     50  0 | I.Versicolor

                                     Cluster 0 <-- I.Verginica
                                     Cluster 1 <-- I.Setosa

3       0       38 ( 25%)             0  1  2  <-- assigned to cluster         18.0     12       %
        1       62 ( 41%)             0  0 50 | I.Setosa
        2       50 ( 33%)            35 15  0 | I.Verginica
                                      3 47  0 | I.Versicolor

                                     Cluster 0 <-- I.Verginica
                                     Cluster 1 <-- I.Versicolor
                                     Cluster 2 <-- I.Setosa

4       0       28 ( 19%)             0  1  2  3  <-- assigned to cluster      45.0     30       %
        1       43 ( 29%)             0  0 50  0 | I.Setosa
        2       50 ( 33%)            28 20  0  2 | I.Verginica
        3       29 ( 19%)             0 23  0 27 | I.Versicolor

                                     Cluster 0 <-- I.Verginica
                                     Cluster 1 <-- No class
                                     Cluster 2 <-- I.Setosa
                                     Cluster 3 <-- I.Versicolor

5       0       26 ( 17%)             0  1  2  3  4  <-- assigned to cluster   49.0     32.6667 %
        1       24 ( 16%)             0  0 50  0  0 | I.Setosa
        2       50 ( 33%)            26  2  0  1 21 | I.Verginica
        3       26 ( 17%)             0 22  0 25  3 | I.Versicolor
        4       24 ( 16%)

                                     Cluster 0 <-- I.Verginica
                                     Cluster 1 <-- No class
                                     Cluster 2 <-- I.Setosa
                                     Cluster 3 <-- I.Versicolor
                                     Cluster 4 <-- No class

6       0       26 ( 17%)             0  1  2  3  4  5  <-- assigned to cluster   63.0   42     %
        1       24 ( 16%)             0  0 36  0  0 14 | I.Setosa
        2       36 ( 24%)            26  2  0  1 21  0 | I.Verginica
        3       26 ( 17%)             0 22  0 25  3  0 | I.Versicolor
        4       24 ( 16%)
        5       14 (  9%)            Cluster 0 <-- I.Verginica
                                     Cluster 1 <-- No class
                                     Cluster 2 <-- I.Setosa
                                     Cluster 3 <-- I.Versicolor
                                     Cluster 4 <-- No class
                                     Cluster 5 <-- No class
```







As can be seen from above, the number of incorrectly clustered instances is minimum when **K = 3**



As seen from the Weka **Visualize**, there are clearly 3 classes (pre-determined labels) in the data, so in unsupervised grouping one can expect 3 clusters and start with **K = 3** for KMeans.

<u>Compression (dimensionality reduction / feature extraction technique)</u>

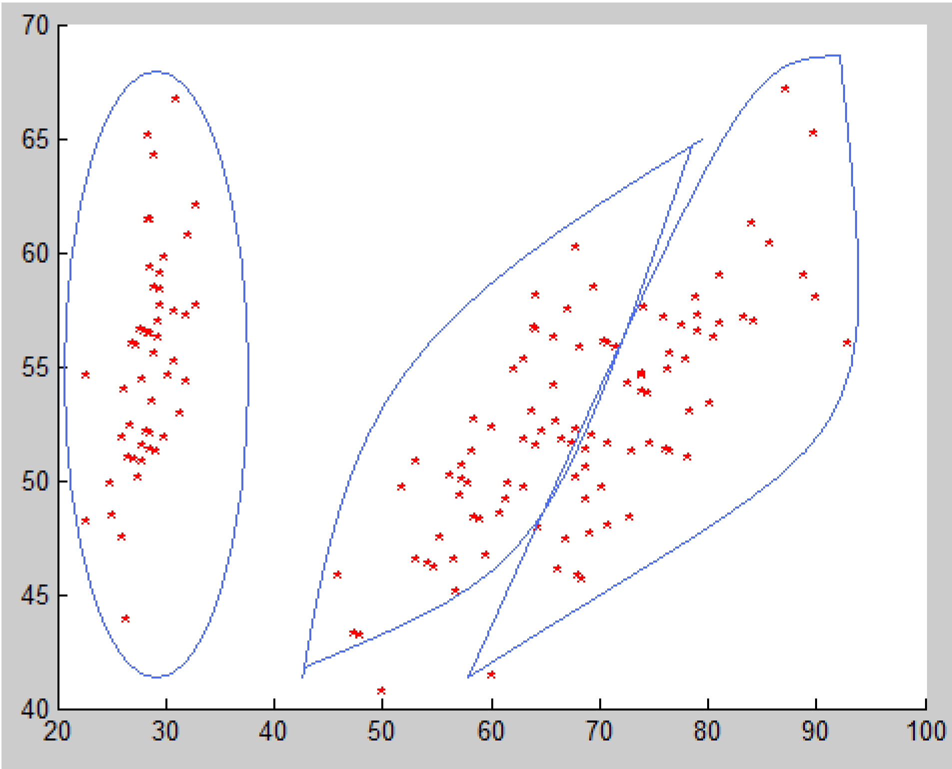
PCA can be used for compression. For Iris data it's enough to use 2 most dominant Eigen vectors for projection in the feature space, since the Eigen values obtained are 2.9127, 0.9150, 0.1483, 0.0241, the 1st two dominant Eigen Vectors preserve

$$\frac{2.9127+0.9150}{2.9127+0.9150+0.1483+0.0241}=95.69\%$$ of variance.

First we shall be interested to see the direction of maximum variations in the data along the feature space. To find it a **scatter plot** is done in between the data projections along the 2nd most dominant eigen vectors in the transformed feature space, with the following output:

```
% Matlab
% iris data already loaded in X
[COEFF,SCORE,latent] = princomp(X); % PCA
A = COEFF(:, 1:2);
Y = X * A;                                  % Projection on 2 most dominat PCs
scatter(Y(:,1), Y(:, 2), 15, [1,0,0], 'p')  % scatter plot in the projected space
```

<u>Scatter plot in the PCA feature (projection) space</u>



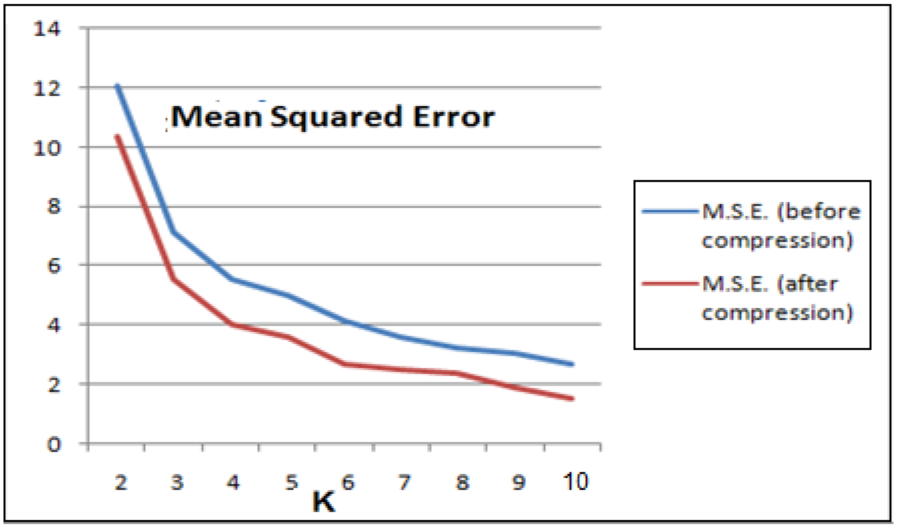
There are clearly **3 different groups** in the data (outlined approximately), hence **k = 3 is the best choice** for KMeans.

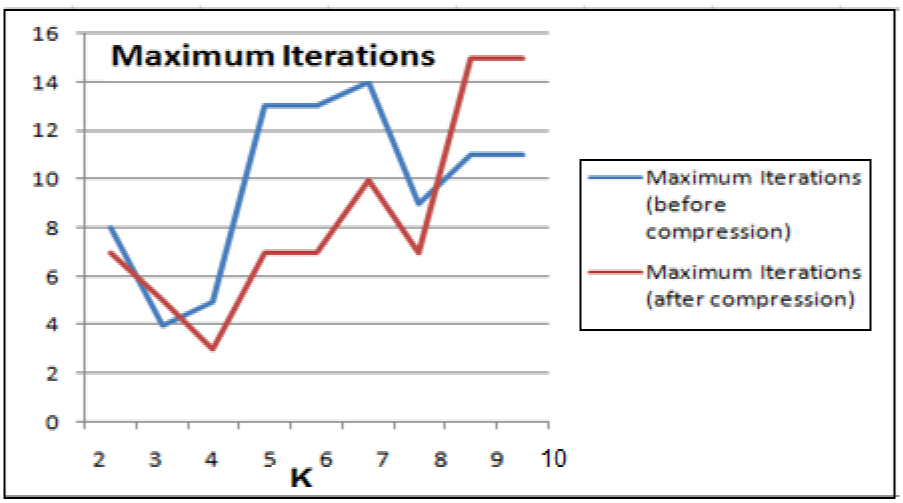
Altrnatively, PCA can be done using weka as well:
java weka.filters.AttributeSelectionFilter -S "weka.attributeSelection.Ranker" -E "weka.attributeSelection.PrincipalComponents -R 0.5" -i iris.arff -o fs_iris.arff

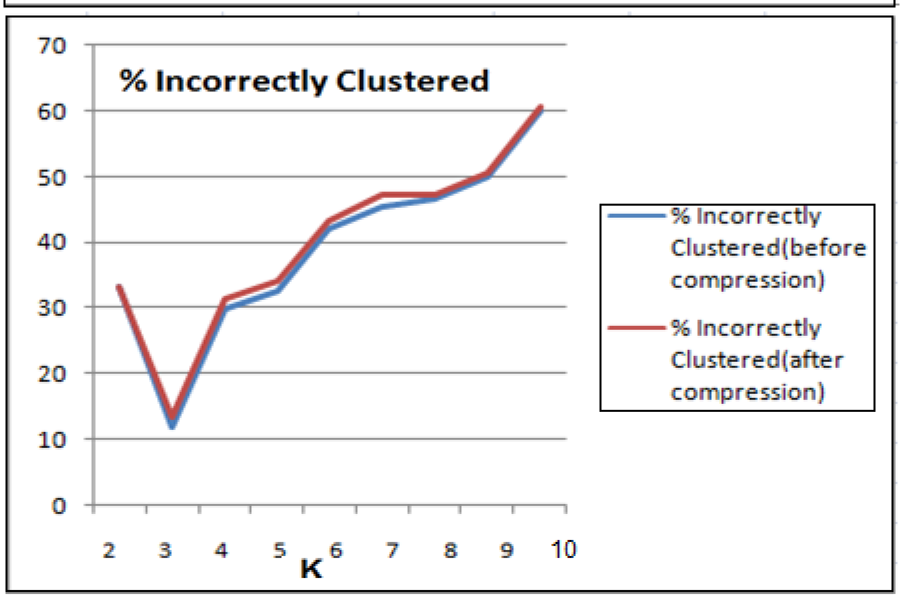
Project in the feature space and compress by dimesionality reduction

```
% Matlab
X = zscore(X); % Normalize
[COEFF,SCORE,latent] = princomp(X); % PCA
A = COEFF(:, 1:2);
Y = X * A;                          % Projection on 2 most dominat PCs
Xc = Y * A';                        % Inverse projection
mean(Xc - X);                       % Nearly Zero
```

Now, if we compare the KMeans algorithm's output in Weka, in between before and after compression, we get the following results:



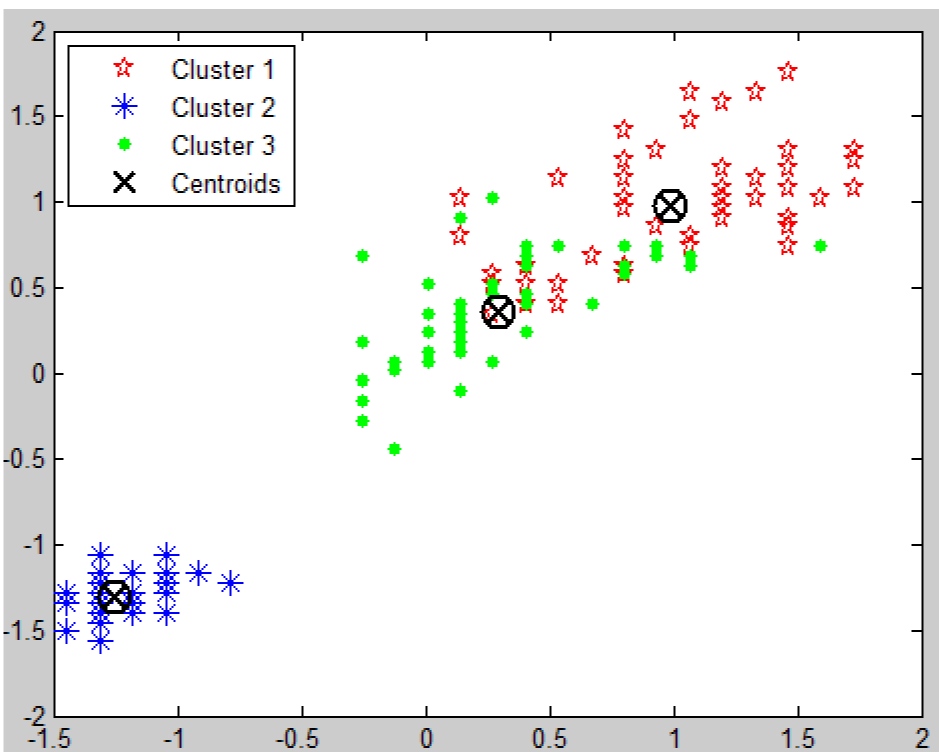




Again, it can be seen that even after compression, the # incorrectly-clustered instances are minimized when **K = 3** (assuming **Specis_name** to be class labels). Also, M.S.E. reduces after PCA compression.

Running KMeans on the compressed data and plotting the output:

```
% matlab
% iris data already loaded in X
opts = statset('Display','final');
k = 3;                                          % # clusters
[idx,ctrs] = kmeans(X,k,...                     % kmeans
                    'Distance','sqEuclidean',...
                    'Replicates',5,...
                    'Options',opts);

plot(X(idx==1,1),X(idx==1,2),'rp','MarkerSize',8)
hold on
plot(X(idx==2,1),X(idx==2,2),'b*','MarkerSize',10)
hold on
plot(X(idx==3,1),X(idx==3,2),'g.','MarkerSize',15)
plot(ctrs(:,1),ctrs(:,2),'kx',...
    'MarkerSize',12,'LineWidth',2)
plot(ctrs(:,1),ctrs(:,2),'ko',...
    'MarkerSize',12,'LineWidth',2)
legend('Cluster 1','Cluster 2','Cluster 3','Centroids',...
      'Location','NW')
```



Evidently there are 3 clusters (2 overlapping).

2.

Table 6.23. Market basket transactions.

Transaction ID	Items Bought
1	{Milk, Beer, Diapers}
2	{Bread, Butter, Milk}
3	{Milk, Diapers, Cookies}
4	{Bread, Butter, Cookies}
5	{Beer, Cookies, Diapers}
6	{Milk, Diapers, Bread, Butter}
7	{Bread, Butter, Diapers}
8	{Beer, Diapers}
9	{Milk, Diapers, Bread, Butter}
10	{Beer, Cookies}

6. Consider the market basket transactions shown in Table 6.23.

(a) What is the maximum number of association rules that can be extracted from this data (including rules that have zero support)?

(b) What is the maximum size of frequent itemsets that can be extracted (assuming $minsup > 0$)?

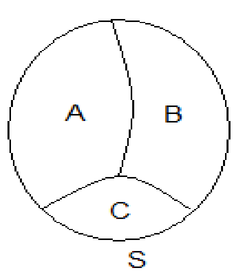
(c) Write an expression for the maximum number of size-3 itemsets that can be derived from this data set.

(d) Find an itemset (of size 2 or larger) that has the largest support.

(e) Find a pair of items, a and b, such that the rules $\{a\} \longrightarrow \{b\}$ and $\{b\} \longrightarrow \{a\}$ have the same confidence.

Answer

If $S = \{I_1, I_2, \ldots, I_d\}$, $|S| = d$, i.e., with *d* items, association rules will be of the form $A \to B$, where $A, B \subseteq S$, $A, B \neq \Phi$, $A \cap B = \Phi$, $A \cup B \subseteq S$. This basically implies the division of *S* into *3* partitions A, B, C s.t. $A, B, C \subseteq S$, $A \cap B \cap C = \Phi$, $A \cup B \cup C = S$, $A, B \neq \Phi$.



Association rules $A \to B$

A, B may not contain all the elements of S, hence we need to consider another set (possibly non-empty) C, not taking part into association rule formation.

Number of different ways it can be done
= (Total #ways *S* can be partitioned into A, B, C) – (#ways A is empty) – (#ways B is empty)
+ (#ways both A, B are empty)

$$= 3^d - 2^d - 2^d + 1 = 3^d - 2^{d+1} + 1.$$

Thought in slightly different manner, let's consider all possible association rules of the form $A \to B$. The L.H.S. can consist of any number of items (k) staring from 1 to d – 1 (since both L.H.S and R.H.S. must be non-empty but their intersection is the null set) and has to be chosen from the d-item set and for each of these choices, the R.H.S. can have 1 to d – k items (l) in it.

Hence, total number of different ways

$$= \sum_{k=1}^{d-1}\sum_{l=1}^{d-k}\binom{d}{k}\binom{d-k}{l}$$

$$= \sum_{k=1}^{d-1}\binom{d}{k}\sum_{l=1}^{d-k}\binom{d-k}{l} = \sum_{k=1}^{d-1}\binom{d}{k}\left(2^{d-k}-1\right) = \sum_{k=1}^{d-1}\binom{d}{k}\left(2^{d-k}\right) - \sum_{k=1}^{d-1}\binom{d}{k}$$

$$= 3^d - 2^d - 1 - \left(2^d - 2\right) = 3^d - 2^{d+1} + 1$$

Since by Binomial theorem,

$$(x+1)^d - x^d - 1 = \sum_{k=1}^{d-1}\binom{d}{k}\left(x^{d-k}\right) \Rightarrow 3^d = \sum_{k=0}^{d}\binom{d}{k}\left(2^{d-k}\right) = 2^d + 1 + \sum_{k=1}^{d-1}\binom{d}{k}\left(2^{d-k}\right)$$

(a) Here, *d = 6*. Hence, maximum number of association rules

$$= 3^6 - 2^{6+1} + 1 = 729 - 128 + 1 = 602$$

(b)If *minsup > 0*, maximum size of frequent item-sets that can be extracted = maximum width of a record in the transaction = *4* here.

(c)Maximum number of size-*k* item-sets that can be derived from the dataset containing *d* items $= \binom{d}{k}$. Here, we have, d = 6 and k = 3. Hence, maximum # of item-sets $= \binom{6}{3}$ =20.

(d)

In	Items	#
1	Beer	4
2	Bread	5
3	Butter	5
4	Cookies	4
5	Diapers	7
6	Milk	5

1-itemset

In	Items	#
1	{Beer, Bread}	0
2	{Beer, Butter}	0
3	{Beer, Cookies}	2
4	{Beer, Diapers}	3
5	{Beer, Milk}	1
6	{Bread, Butter}	5
7	{Bread, Cookies}	1
8	{Bread, Diapers}	3
9	{Bread, Milk}	3
10	{Butter, Cookies}	1
11	{Butter, Diapers}	3
12	{Butter, Milk}	3
13	{Cookies, Diapers}	2
14	{Cookies, Milk}	1
15	{Diapers, Milk}	4

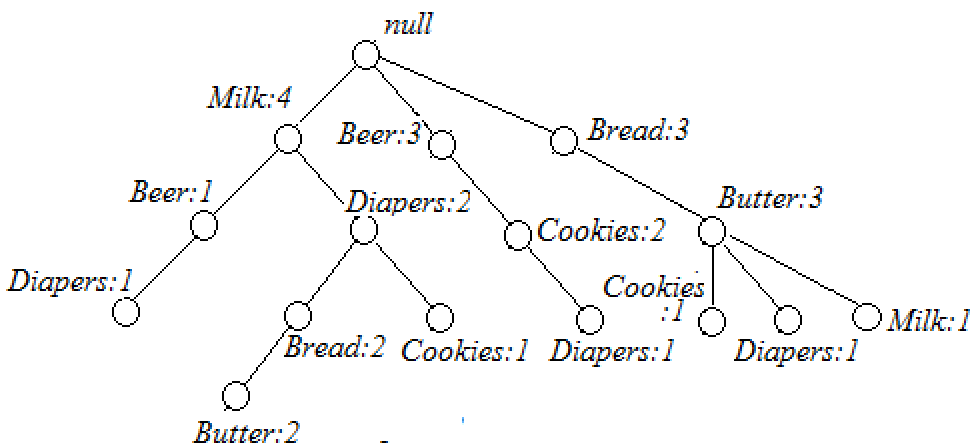
2-itemset

The highest support-count in set of all 2-itemsets = 5, i.e., for {Bread, Butter}. Considering only the item-sets with support count 5 or higher for 3-itemsets (because by anti-monotone property, the support-count for any superset of an item-set is less than or equal to the support count of that item-set). Also, {Beer, Bread} support count is 0, we need not consider its superset. As we can see, support counts for all other 2-item-sets are strictly less than 5, **all 3-item sets will have support count strictly less than 5**.

In	Items	#
1	{ Bread, Butter, Cookies }	1
2	{Bread, Butter, Diapers}	3
3	{Bread, Butter, Milk}	3

Hence, the item-set (of size-2 or larger) with the largest support count (5) is the 2-item-set {Bread, Butter}.

Let's construct the FP-tree,



From the FP tree it's clear that only {Bread, Butter} is having the support count as high as 5.

(e) $\because\ confidence(\{a\} \rightarrow \{b\}) = \dfrac{\sigma(\{a\}\ \ and\ \ \{b\})}{\sigma(\{a\})}$,

$$confidence(\{a\} \rightarrow \{b\}) = confidence(\{b\} \rightarrow \{a\}) \Rightarrow \sigma(\{a\}) = \sigma(\{b\})$$

Consider the following two rules:

$\{\text{Bread,Diapers}\} \rightarrow \{Butter, Milk\}$ and $\{Butter, Milk\} \rightarrow \{\text{Bread,Diapers}\}$

$$\because\ \sigma(\{\text{Bread,Diapers}, Butter, Milk\}) = 2 \ \wedge\ \sigma(\{\text{Bread,Diapers}\}) = \sigma(\{Butter, Milk\}) = 3,$$

$$confidence(\{\text{Bread,Diapers}\} \rightarrow \{Butter, Milk\}) = confidence(\{Butter, Milk\} \rightarrow \{\text{Bread,Diapers}\})$$

$$= \frac{2}{3}$$

3. Consider the training examples shown in Table 4.8 for a binary classification problem.

   (a) What is the entropy of this collection of training examples with respect to the positive class?

Table 4.8. Data set for Exercise 3.

Instance	a_1	a_2	a_3	Target Class
1	T	T	1.0	+
2	T	T	6.0	+
3	T	F	5.0	−
4	F	F	4.0	+
5	F	T	7.0	−
6	F	T	3.0	−
7	F	F	8.0	−
8	T	F	7.0	+
9	F	T	5.0	−

   (b) What are the information gains of a_1 and a_2 relative to these training examples?

   (c) For a_3, which is a continuous attribute, compute the information gain for every possible split.

   (d) What is the best split (among a_1, a_2, and a_3) according to the information gain?

   (e) What is the best split (between a_1 and a_2) according to the classification error rate?

   (f) What is the best split (between a_1 and a_2) according to the Gini index?

Answer

(a) Here the target class is a binary random variable *X* with the following probability mass function

$$p(x) = P(X = x) = \begin{cases} \frac{4}{9}, & x \text{ is } +ve \\ \frac{5}{9}, & x \text{ is } -ve \end{cases}$$

$$H(X) = \sum_{x \in X} p(x) \log\left(\frac{1}{p(x)}\right)$$

$$= \sum_{x>0} p(x) \log\left(\frac{1}{p(x)}\right) + \sum_{x<0} p(x) \log\left(\frac{1}{p(x)}\right)$$

$$= I(4,5) = \frac{4}{9}\log_2\left(\frac{9}{4}\right) + \frac{5}{9}\log_2\left(\frac{9}{5}\right) = 0.52 + 0.47 = 0.99$$

where *I(p, n)* is defined as $-\frac{p}{p+n}\log\left(\frac{p}{p+n}\right) - \frac{n}{p+n}\log\left(\frac{n}{p+n}\right)$.

(b)

$$E(a_1) = \frac{|a_1 = T|}{|a_1|} H(X \mid a_1 = T) + \frac{|a_1 = F|}{|a_1|} H(X \mid a_1 = F)$$

$$= \frac{4}{9} I(3,1) + \frac{5}{9} I(1,4) = \frac{4}{9}\left(\frac{3}{4}\log\left(\frac{4}{3}\right) + \frac{1}{4}\log\left(\frac{4}{1}\right)\right) + \frac{5}{9}\left(\frac{1}{5}\log\left(\frac{5}{1}\right) + \frac{4}{5}\log\left(\frac{5}{4}\right)\right)$$

$$= \frac{4}{9} \times 0.8113 + \frac{5}{9} \times 0.7219 = 0.36057 + 0.40106 = 0.76163$$

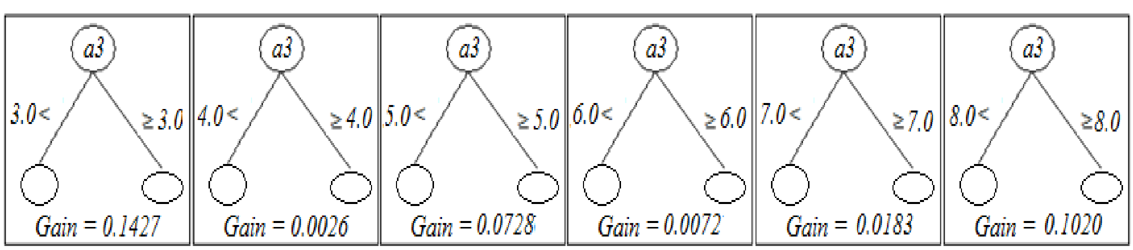
$$\therefore Gain(a_1) = H(X) - E(a_1) = I(4,5) - E(a_1) = 0.9911 - 0.7616 = 0.2294$$

$$E(a_2) = \frac{5}{9} I(2,3) + \frac{4}{9} I(2,2) = 0.9839$$

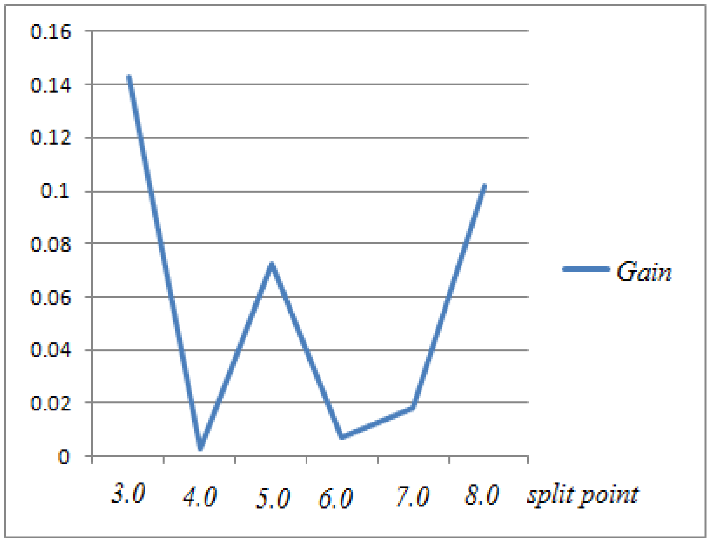
$$\therefore Gain(a_2) = H(X) - E(a_2) = I(4,5) - E(a_2) = 0.9911 - 0.9839 = 0.0072$$

(c) Since *a3* is continuous valued, we try all possible split points, in order to split the set into 2 subsets. For instance, if <6.0 then left subset, ow create right subset. For all different choises of the split-points, the information gain is as follows:

	a3					
Split point	3.0	4.0	5.0	6.0	7.0	8.0
Gain	0.1427	0.0026	0.0728	0.0072	0.0183	0.1020

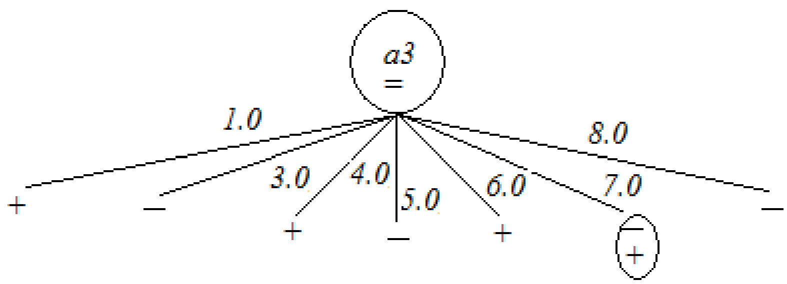


Splitting the data set into 2 subset using different values of the attribute a3 as split point



Splitting the data set into 2 subset using different values of the attribute a3 as split point

If the number of split points is increased, the gain will also increase in general, resulting in **highest gain** in the worst case if there are 5 splits (at the cost of 7 branches in the tree), as shown:

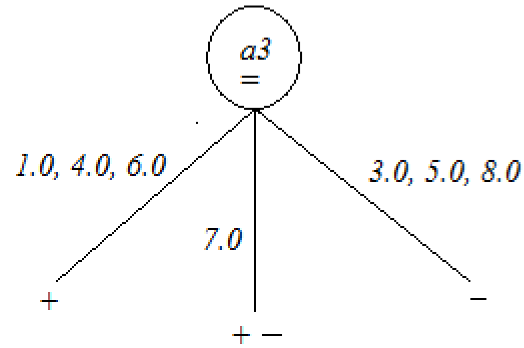


$$E(a_3) = \frac{1}{9}I(1,0) + \frac{1}{9}I(0,1) + \frac{1}{9}I(1,0) + \frac{2}{9}I(0,2) + \frac{1}{9}I(1,0) + \frac{2}{9}I(1,1) + \frac{1}{9}I(0,1)$$

$$= \frac{1}{9}.0 + \frac{1}{9}.0 + \frac{1}{9}.0 + \frac{2}{9}.0 + \frac{1}{9}.0 + \frac{2}{9}.1 + \frac{1}{9}.0 = 0.2222$$

$$\therefore Gain(a_3) = H(X) - E(a_3) = I(4,5) - E(a_3) = 0.9911 - 0.2222 = 0.7689$$

So far the **values** of the attribute *a3* were considered in **sorted order**. From the above it can be easily seen that the split can be minimized to 3 and yet the same gain can be achieved (if we get rid of the order in the attribute values)

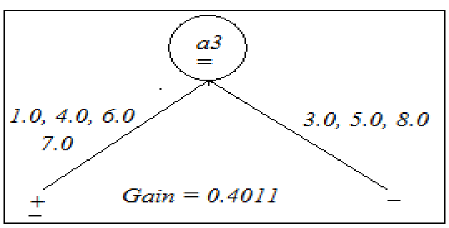


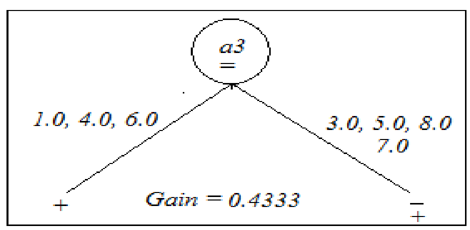
$$E(a_3) = \frac{3}{9}I(1,0) + \frac{2}{9}I(0,2) + \frac{4}{9}I(0,1)$$

$$= \frac{3}{9}.0 + \frac{2}{9}.1 + \frac{4}{9}.0 = 0.2222$$

$$\therefore Gain(a_3) = H(X) - E(a_3) = I(4,5) - E(a_3) = 0.9911 - 0.2222 = 0.7689$$

If the value 7.0 is included in any of the subsets, the gain decreases.



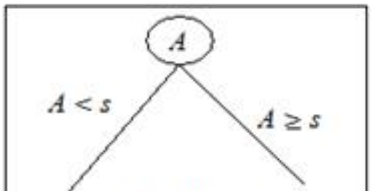


[matlab]

```
a1 = [
    true     true     true     false     false     false     false     true     false
];

a2 = [
    true     true     false     false     true     true     false     false     true
];

C = [
    +1     +1     -1     +1     -1     -1     -1     +1     -1
];

Gain(a1, C, true)   % split point true / false
Gain(a2, C, true)   % split point true / false

a3 = [
    1.0     6.0     5.0     4.0     7.0     3.0     8.0     7.0     5.0
];

N = size(a3, 2);

for i = 1 : N
    Gain(a3, C, a3(i)); % a3 continuous, test every possible split point
end
```

[matlab]

```matlab
% attribute A
% target class C
% split value s
function [G] = Gain(A, C, s)
    nc = size(A, 2); % size of target class labels
    posC = 0;
    for i = 1 : nc
        if (C(i) > 0)
            posC = posC + 1;
        end
    end
    G = I(posC, nc - posC) - E(A, C, s);
end
```



attribute A with split point s

```matlab
% attribute A
% target class C
% split value s
function [WH] = E(A, C, s)
    WH = 0;             % initialize weighted average entropy E(A)
    n = size(A, 2);  % size of attribute values
    nc = size(A, 2); % size of corresponding target class labels

    if (n == nc)      % must be equal
        % attribute set A to be split into 2 subsets A1, A2: s being the
        % split point: all tuples with values of A < s will fall in A1,
        % otherwise fall into A2.
        nA1 = 0;
        nA2 = 0;
        posA1 = 0;  % positive class labels in A1
        posA2 = 0;  % positive class labels in A2
        for i = 1 : n
            if (A(i) >= s)
                nA1 = nA1 + 1;  % number of tuples in A1
                if (C(i) > 0)    % if corresponding class label is positive
                    posA1 = posA1 + 1;
                end
            else
                nA2 = nA2 + 1;  % number of tuples in A2: nA2 = N - nA1
                if (C(i) > 0)       % if corresponding class label is positive
                    posA2 = posA2 + 1;
                end
            end
        end
        WH = (nA1 / n) * I(posA1, nA1 - posA1) + (nA2 / n) * I(posA2, nA2 - posA2);
    end
end

% find entropy
function [H] = I(a, b)
    p = a / (a + b);
    n = b / (a + b);
    logp = 0;
    if (p ~= 0)
        logp = log2(p);
    end
    logn = 0;
    if (n ~= 0)
        logn = log2(n);
    end
    H = -p * logp - n * logn
end
```

(d)The attribute corresponding to the largest information gain should be chosen. If we allow **binary splitting** only (maintaining attribute order), the information gain is highest in case of the attribute a1, we should choose attribute *a1* for splitting.
If we allow n-ary splitting without maintaining attribute order, *a3* has the highest gain in the 3-split shown above, hence *a3* should be chosen.

(e)

$$\because ClassErr(t) = 1 - \max_i [p(i \mid t)]$$

$$ClassErr(X \mid a_1 = T) = 1 - \max\left(\frac{3}{4}, \frac{1}{4}\right) = \frac{1}{4}$$

$$ClassErr(X \mid a_1 = F) = 1 - \max\left(\frac{1}{5}, \frac{4}{5}\right) = \frac{1}{5}$$

$$\therefore ClassErr(a_1) = \frac{|a_1 = T|}{|a_1|} ClassErr(X \mid a_1 = T) + \frac{|a_1 = F|}{|a_1|} ClassErr(X \mid a_1 = F)$$

$$= \frac{4}{9} . \frac{1}{4} + \frac{5}{9} . \frac{1}{5} = \frac{2}{9} = 0.2222$$

$$ClassErr(a_2)$$

$$= \frac{5}{9}\left(1 - \frac{3}{5}\right) + \frac{4}{9}\left(1 - \frac{2}{4}\right) = \frac{5}{9}\left(\frac{2}{5}\right) + \frac{4}{9}\left(\frac{2}{4}\right) = \frac{4}{9} = 0.4444$$

The best split is the attribute with minimum classification error rate, hence *a1*.

(f)

$$\because gini(t) = 1 - \sum_i [p(i \mid t)]^2$$

$$gini(X \mid a_1 = T) = 1 - \left(\frac{3}{4}\right)^2 - \left(\frac{1}{4}\right)^2 = \frac{6}{16}$$

$$gini(X \mid a_1 = F) = 1 - \left(\frac{1}{5}\right)^2 - \left(\frac{4}{5}\right)^2 = \frac{8}{25}$$

$$\therefore gini(a_1) = \frac{|a_1 = T|}{|a_1|} gini(X \mid a_1 = T) + \frac{|a_1 = F|}{|a_1|} gini(X \mid a_1 = F)$$

$$= \frac{4}{9} . \frac{6}{16} + \frac{5}{9} . \frac{8}{25} = \frac{1}{9}\left(\frac{3}{2} + \frac{8}{5}\right) = \frac{3.1}{9} = 0.3444$$

$$gini(a_2) = \frac{5}{9}\left(\frac{12}{25}\right) + \frac{4}{9}\left(\frac{1}{2}\right) = \frac{1}{9}\left(\frac{12}{5} + 2\right) = \frac{4.4}{9} = 0.4889$$

The best split is the attribute with minimum gini index, hence *a1*.