1. 10.2 A_1(2,10), A_2(2,5), A_3(8,4), B_1(5,8), B_2(7,5), B_3(6,4), C_1(1,2), C_2(4,9). Euclidean distance, initially assign A_1, B_1, C_1 as the center of each cluster. Use k-means,

   (a) The three cluster centers after the first round of execution.

Dis.	A_1(2,10)	B_1(5,8)	C_1(1,2)	Belong
A_2(2,5)	5.0000	4.2426	3.1623	C_1
A_3(8,4)	8.4853	5.0000	7.2801	B_1
B_2(7,5),	7.0711	3.6056	6.7082	B_1
B_3(6,4)	7.2111	4.1231	5.3852	B_1
C_2(4,9)	2.2361	1.4142	7.6158	B_1
1st round	A_1^1(2,10)	B_1^1(6,6)	C_1^1(1.5, 3.5)	$E = 67.0004$

   The three cluster centers after the first round of execution are A_1^1, B_1^1, C_1^1.

   (b) The final three clusters.

      i. Cluster1centroid (3.6667,9.0000); Cluster 2 centroid (7.0000,4.3333); Cluster 3 centroid (1.5000, 3.5000);

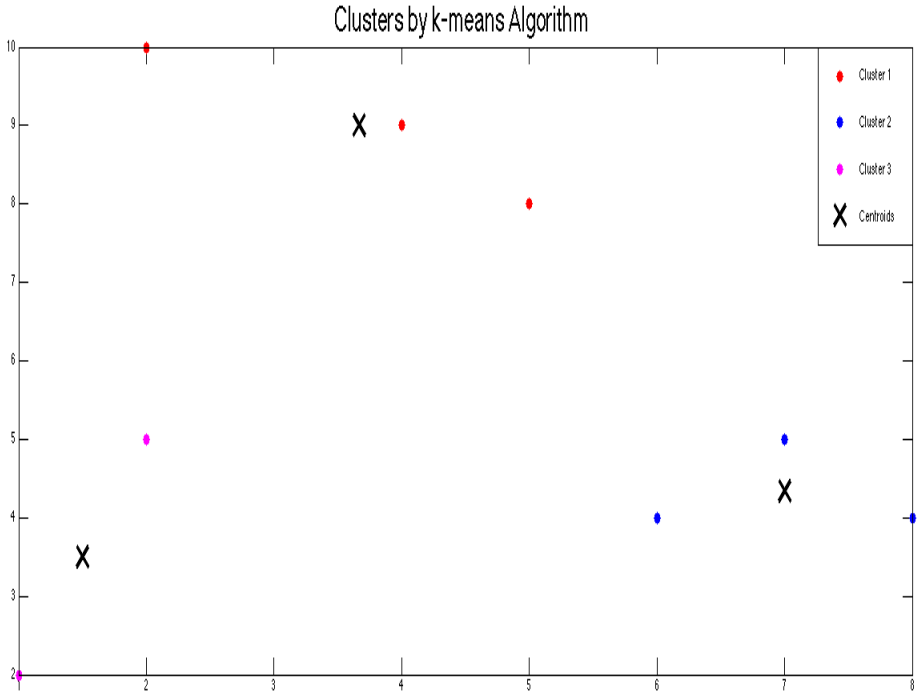
   Use matlab k-means,

   X=[2,10;2,5;8,4;5,8;7,5;6,4;1,2;4,9;], seeds=[2,10;5,8;1,2;], [idx,C]=kmeans(X,3,'Start',seeds),

   gives that C=3.6667 9.0000, 7.0000 4.3333, 1.5000 3.5000.

```
hold on
plot(X(idx==2,1),X(idx==2,2),'b.','MarkerSize',24);
plot(X(idx==3,1),X(idx==3,2),'m.','MarkerSize',24);
plot(C(:,1),C(:,2),'kx','MarkerSize',20,'LineWidth',3);
legend('Cluster 1','Cluster 2','Cluster 3', 'Centroids','Location','NW');
title 'Clusters by k-means Algorithm'
hold off
```

Figure 1: Plot of K-means



2. 10.4 k-means++ has also been integrated into matlab. For m = 1,..., n and p = 1,..., j − 1, select centroid j at random from X with probability: $P(x_m, c_p) = \frac{d^2(x_m, c_p)}{\sum_{h, x_h \in C_p} d^2(x_h, c_p)}$, where C_p is the set of all observations closest to centroid c_p and x_m belongs to C_p. This method will not only speed up the convergence of the k-means algorithm, but also guarantee the quality of the final clustering results. This is because for each loop generating the new centroid, the denominate will be the same. The larger the distance of $x_m$to c_p, the more possible $x_m$will be select. In another word, the probability on choosing centroid j (c_j) ensure that the points near previous points$c_1, c_2, \cdots, c_{j-1}$have small possibilities to be chosen, the ones that are far away from previous chosen centroids will be more likely to be select as a new centroids.

   (a) To calculate speed of convergence, we know that k-means has complexity of O(tkn), while kmeans++ always attains an O(log k) approximation in expectation.

   (b) As for quality of the final clustering results, kmeans++ is better because the possibility of choosing points in k-means++ ($P(x_m, c_p) = \frac{d^2(x_m, c_p)}{\sum_{h, x_h \in C_p} d^2(x_h, c_p)}$) will guarantee that outliers and noisy data will have larger chance to be chosen as new cluster points rather than forcing to be included in one cluster partition as in k-means.

3. 10.6 k-means vs k-medoids

(a) Strength and weakness of k-means comparing with k-medoids:

i. Strength: Efficient: $O(tkn)$, normally k,t << n. n is # of objects, k is # of clusters, t is # of iterations. While typical k-medoids methods PAM takes $O(k(n-k)^2)$, CLARA takes $O(ks^2 + k(n-k))$, CLARA seems to cost smaller but it sacrifices accuracy for efficiency.

ii. Weakness: Applicable only to objects in a continuous n-dimensional space, while k-medoids can be applied to a wider range of data. K-means algorithm is sensitive to noisy data and outliers, and it is not guaranteed to converge to the global optimum and often terminates at a local optimum.

(b) k-means and k-medoids in comparation with hierarchical clustering methods(e.g. AGNES)

i. Strength of k-means and k-medoids methods: they are partitioning-based algorithms, and can undo previous clustering steps. However, hierarchical clustering cannot, and are sensitive to the split and merge decisions, which may lead to low-quality clusters if not well-chosen.

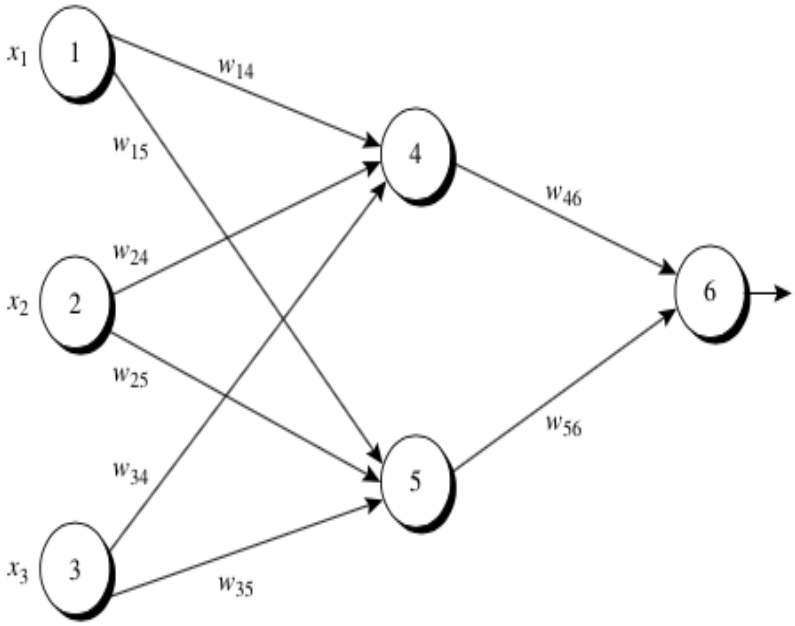
ii. Weakness: k-means and k-medoids require initial input of k partition while hierarchical methods can use any partition number.

4. Using Weka, solve 9.1 with MLNN, SVM, and another classifier of your choice

(a) not use weka,

Design a multilayer feed-forward neural network. Let status be the class-label attribute, let X1(department)={0,1,2,3} : 0,1,2,3 stands for sales, systems, marketing, secretary separately; X2(age)={23,28,33,38,43,48}, which are the median of each range; X3(salary)={28,33,38,43,48,68}, which are the median of each range and its unit is K dollars. And node 6 is the output junior or senior. For simplicity, I just use the example with 1 hidden layer in the textbook figure 9.5, for that the idea under MLNN is not changed by the structure.

Figure 2: MLNN



Example of a multilayer feed-forward neural network.

Initial inputs, weights and biases

x1(sales)	x2(31..35)	x3(46K..50K)	w_{14}	w_{15}	w_{24}	w_{25}	w_{34}	w_{35}	w_{46}	w_{56}	θ_4	θ_5	θ_6	T_6
0	33	48	0.2	-0.3	0.4	0.1	-0.5	0.2	-0.3	-0.2	-0.4	0.2	0.1	1

Net input and output calculation ($I_j = \sum_i w_{ij} O_i + \theta_i$)

Unit	Input	Output
4	$I_4 = 0.2O_1 + 0.4O_2 - 0.5O_3 - 0.4 = -9.88$	$O_4 = \frac{1}{(1+e^{9.88})} = 5.118E-5$
5	$I_5 = -0.3O_1 + 0.1O_2 - 0.2O_3 + 0.2 = -6.1$	$O_5 = \frac{1}{(1+e^{6.1})} = 2.238E-3$
6	$I_6 = (5.118E-5)w_{46} + (2.238E-3)w_{56} + 0.1 = 0.099537$	$O_6 = \frac{1}{1+e^{-0.099537}} = 0.52486$

Error at Each Node $Err_j = O_j(1-O_j)(T_j - O_j)$ or $Err_j = O_j(1-O_j)\sum_k Err_k w_{jk}$

Unit	Err_j
6	0.52486(1-0.52486)(1-0.52486)=0.11849
5	2.238E-3(1-2.238E-3)(0.11849)(-0.2)=-5.291E-5
4	5.118E-5(1-5.118E-5)(0.11849)(-0.3)=-1.819E-6

Learning rate $l = 0.9$, $\Delta w_{ij} = l * Err_j O_i$, $w_{ij} = w_{ij} + \Delta w_{ij}$

Weight/Bias	New value
w_{46}	-0.3+(0.9)(0.52486)(5.118E-5)=-0.2438347
w_{56}	-0.2+(0.9)(0.52486)(2.238E-3)=-0.5595238
w_{14}	0.2+(0.9)(-1.819E-6)(0)=0.2
w_{15}	-0.3+(0.9)(-5.291E-5)(0)=-0.3
w_{24}	0.4+(0.9)(-1.819E-6)(33)=0.3999
w_{25}	0.1+(0.9)(-5.291E-5)(33)=0.0984
w_{34}	-0.5+(0.9)(-1.819E-6)(48)=-0.5001
w_{35}	0.2+(0.9)(-5.291E-5)(48)=0.1977
θ_6	0.1+(0.9)(0.11849)=0.2066
θ_5	0.2+(0.9)(-5.291E-5)=0.2000
θ_4	-0.4+(0.9)(-1.819E-6)=-0.4000

5. Using weka and cross-validation of 10 folds, both MLNN (I set 2 hidden layers) and SVM classified the data extremely good, which supported by that all TP, Precision,Recall,F-Measure,ROC Area =1. However, MLNN seems faster than SVM method because it took 0.12 seconds and SVM took 0.23 seconds. Moreover, by right-clicking on the results, one can choose to visualize classifier errors, margin curve, threshold curve, cost and benefit analysis and curves. (I attached exe91.arff and its outputs from MLNN and SVM method)

   (a) **MLNN output(0.12 seconds)**

```
=== Run information ===

Scheme:weka.classifiers.functions.MultilayerPerceptron -L 0.3 -M 0.2 -N 500
     -V 0 -S 0 -E 20 -H 2
Relation:     employee
Instances:    165
Attributes:   4
              department
              age
              salary
              class
Test mode:10-fold cross-validation

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

Sigmoid Node 0
    Inputs    Weights
    Threshold    -4.8073953221914305
    Node 2    4.8382203463652855
    Node 3    5.70402363875396
Sigmoid Node 1
    Inputs    Weights
    Threshold    4.807590826757485
    Node 2    -4.860362609579104
    Node 3    -5.682467079556528
Sigmoid Node 2
    Inputs    Weights
    Threshold    0.01048287697751337
    Attrib department=sales    -0.8069086081985345
```

```
    Attrib department=systems     1.425432600118906
    Attrib department=marketing     -0.24448825046701989
    Attrib department=secretary     -0.3291027919645579
    Attrib age     -2.540854509568941
    Attrib salary=28K     1.6971848997408574
    Attrib salary=33K     2.0986920072776503
    Attrib salary=38K     -0.5450724150170226
    Attrib salary=43K     1.0625627580401131
    Attrib salary=48K     -1.327291053074795
    Attrib salary=68K     -3.088390152441787
Sigmoid Node 3
    Inputs     Weights
    Threshold     -0.014862143339935638
    Attrib department=sales     -0.9460206257755995
    Attrib department=systems     1.5128202294400876
    Attrib department=marketing     -0.30429527552150126
    Attrib department=secretary     -0.37319477061135836
    Attrib age     -2.8310527601859214
    Attrib salary=28K     1.8176625787155813
    Attrib salary=33K     2.2949183727245135
    Attrib salary=38K     -0.5434852224109147
    Attrib salary=43K     1.1674106929142172
    Attrib salary=48K     -1.4177970737343066
    Attrib salary=68K     -3.4119431847570154
Class junior
    Input
    Node 0
Class senior
    Input
    Node 1


Time taken to build model: 0.12 seconds

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

Correctly Classified Instances         165               100      %
Incorrectly Classified Instances         0                 0      %
Kappa statistic                          1
Mean absolute error                      0.0072
Root mean squared error                  0.0079
Relative absolute error                  1.6568 %
Root relative squared error              1.6939 %
Total Number of Instances              165

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area
                 Class
                 1         0          1         1         1          1
                           junior
                 1         0          1         1         1          1
                           senior
Weighted Avg.    1         0          1         1         1          1

=== Confusion Matrix ===
```

```
  a   b   <-- classified as
113   0 |   a = junior
  0  52 |   b = senior
```

i. **SVM output(0.23 seconds)**

```
=== Run information ===

Scheme:weka.classifiers.functions.SMO -C 1.0 -L 0.001 -P 1.0E-12 -N 0 -V -1
     -W 1 -K "weka.classifiers.functions.supportVector.PolyKernel -C 250007
     -E 1.0"
Relation:     employee
Instances:    165
Attributes:   4
              department
              age
              salary
              class
Test mode:10-fold cross-validation

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

SMO

Kernel used:
  Linear Kernel: K(x,y) = <x,y>

Classifier for classes: junior, senior

BinarySMO

Machine linear: showing attribute weights, not support vectors.

         0.5014 * (normalized) department=sales
 +      -0.9874 * (normalized) department=systems
 +       0.2443 * (normalized) department=marketing
 +       0.2417 * (normalized) department=secretary
 +       1.2815 * (normalized) age
 +      -0.9929 * (normalized) salary=28K
 +      -1.2489 * (normalized) salary=33K
 +       0.2417 * (normalized) salary=38K
 +      -0.9915 * (normalized) salary=43K
 +       0.7521 * (normalized) salary=48K
 +       2.2395 * (normalized) salary=68K
 -       0.7654

Number of kernel evaluations: 5366 (81.643% cached)



Time taken to build model: 0.23 seconds

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

Correctly Classified Instances         165               100      %
Incorrectly Classified Instances         0                 0      %
Kappa statistic                          1
```

```
Mean absolute error                      0
Root mean squared error                  0
Relative absolute error                  0       %
Root relative squared error              0       %
Total Number of Instances              165

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area
                 Class
                 1         0          1         1         1          1
                           junior
                 1         0          1         1         1          1
                           senior
Weighted Avg.    1         0          1         1         1          1

=== Confusion Matrix ===

   a   b   <-- classified as
 113   0 |   a = junior
   0  52 |   b = senior
```

6. 11.2 Ada Bob: products in common, each pick up 7 randomly from 993 products. Cathy 10 randomly from 1000. Euclidean distance, the probability that dist (Ada, Bob) > dist (Ada, Cathy)? What if Jaccard similarity (Chapter 2) is used? What can you learn from this example?

   (a) The first three items of A and B are the same.

      i. The probability of A and B get 1 different product in the rest 997 while A and C get 0 different
      ii. A&B 2 different, A&C 0,1 different
      iii. A&B 3 different, A&C 0,1,2 different
      iv. A&B 4 different, A&C 0,1,2,3 different
      v. A&B 5 different, A&C 0,1,2,3,4 different
      vi. A&B 6 different, A&C 0,1,2,3,4,5 different
      vii. A&B 7 different, A&C 0,1,2,3,4,5,6 different

      One can use basic probability method to calculate the probability and sum them all to get the results. However, the method is extremely complex when the bought products becomes larger. So clustering high-dimensional data should be used to simplify it, dimensionality reduction methods and spectral clustering, Biclustering,correlation based clustering method, subspace search method may be used.