Data Mining Assignment 5

Aradhya Mathur

Textbook

- 10.2, 10.4, 10.6
- Using Weka, solve 9.1 with MLNN, SVM, and another classifier of your choice
- 11.2

10.2 Suppose that the data mining task is to cluster points (with .x, y/ representing location) into three clusters, where the points are

$$A_1(2,10), A_2(2,5), A_3(8,4), B_1(5,8), B_2(7,5), B_3(6,4), C_1(1,2), C_2(4,9).$$

The distance function is Euclidean distance. Suppose initially we assign $A1$, $B1$, and $C1$ as the center of each cluster, respectively. Use the *k-means* algorithm to show *only*

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

In [1]:
```
import numpy as np
from sklearn.cluster import KMeans
import warnings
warnings.filterwarnings('ignore')
```

a)

In [2]:
```
points = np.array([[2,10],[2,5],[8,4],[5,8],[7,5],[6,4],[1,2],[4,9]])
initial_centroid = np.array([[2,10],[5,8],[1,2]])
```

In [3]:
```
#Using max_iter we can check for each iteration.
kmeans = KMeans(n_clusters=3, init=initial_centroid, max_iter=1)
kmeans.fit(points)
```

Out[3]:
```
KMeans
KMeans(init=array([[ 2, 10],
       [ 5,  8],
       [ 1,  2]]), max_iter=1,
       n_clusters=3)
```

In [4]:
```
kmeans.cluster_centers_
```

Out[4]:
```
array([[ 2. , 10. ],
       [ 6. ,  6. ],
       [ 1.5,  3.5]])
```

The three cluster centers after the first round of execution are:

[2,10] , [6,6] and [1.5,3.5]

(b) The final three clusters.

Iteration 2:

b)

In [10]:
```
kmeans = KMeans(n_clusters=3, init=initial_centroid, max_iter=2)
kmeans.fit(points)
```

Out[10]:
```
KMeans
KMeans(init=array([[ 2, 10],
       [ 5,  8],
       [ 1,  2]]), max_iter=2,
       n_clusters=3)
```

In [11]:
```
kmeans.cluster_centers_
```

Out[11]:
```
array([[3.  , 9.5 ],
       [6.5 , 5.25],
       [1.5 , 3.5 ]])
```

Iteration 3:

```
In [12]: kmeans = KMeans(n_clusters=3, init=initial_centroid, max_iter=3)
         kmeans.fit(points)

Out[12]: KMeans
         KMeans(init=array([[ 2, 10],
                [ 5,  8],
                [ 1,  2]]), max_iter=3,
                n_clusters=3)

In [13]: kmeans.cluster_centers_

Out[13]: array([[3.66666667, 9.        ],
                [7.        , 4.33333333],
                [1.5       , 3.5       ]])
```

Iteration 4:

```
In [14]: kmeans = KMeans(n_clusters=3, init=initial_centroid, max_iter=4)
         kmeans.fit(points)

Out[14]: KMeans
         KMeans(init=array([[ 2, 10],
                [ 5,  8],
                [ 1,  2]]), max_iter=4,
                n_clusters=3)

In [15]: kmeans.cluster_centers_

Out[15]: array([[3.66666667, 9.        ],
                [7.        , 4.33333333],
                [1.5       , 3.5       ]])
```

As centers at iteration 3 and iteration 4 were same (remain unchanged), these are the final centers of the cluster.

```
In [16]: kmeans.labels_

Out[16]: array([0, 2, 1, 0, 1, 1, 2, 0])
```

Final three clusters are:

1) [2,10] , [5,8] , [4,9]
2) [8,4] , [7,5] , [6.4]
3) [2,5] , [1,2]

10.4 For the k-means algorithm, it is interesting to note that by choosing the initial cluster centers carefully, we may be able to not only speed up the algorithm's convergence, but also guarantee the quality of the final clustering. The k-means++ algorithm is a variant of k-means, which chooses the initial centers as follows. First, it selects one center uniformly at random from the objects in the data set. Iteratively, for each object p other than the chosen center, it chooses an object as the new center. This object is chosen at random with probability proportional to dist.p/2, where dist.p/ is the distance from p to the closest center that has already been chosen. The iteration continues until k centers are selected.

Explain why this method will not only speed up the convergence of the k-mean algorithm, but also guarantee the quality of the final clustering results.

Answer)

Random initialization of the centroids is the major drawback of k-means algorithm. This results in incorrect and inaccurate formation of clusters. Points that are selected as initial chosen centroids have higher possibility of already existing in different clusters. This ensures decrease in the runtime and provides better quality of the clusters. Initialization of the clusters highly matters in K-means. It will speed up the convergence process as it will avoid initialization of initial clusters which are similar to each other.

10.6 Both *k-means* and *k-medoids* algorithms can perform effective clustering.

(a) Illustrate the strength and weakness of *k-means* in comparison with *k-medoids*.
Answer)

K - Means	K- Medoids
Affected by outliers (Weakness)	More robust to outliers (Strength)
Faster and efficient (Strength)	Comparatively slower (Weakness)

Means are generally more affected by outliers because average can change quickly if a value is too small or too big. While, in case of median, center value is considered and because of which outliers and noise don't have significant impact.

(b) Illustrate the strength and weakness of these schemes in comparison with a hierarchical clustering scheme (e.g., AGNES).

Answer)

Partitioning	**Hierarchical**
Weakness 1. Need to specify the number of clusters to partition which may be unknown information before some testing 2. Less efficient	Strength 1. No need to specify the number of clusters beforehand. 2. More efficient
Strength 1. Steps can be undone 2. Quality is good	Weakness 1. Steps can't be undone 2. Quality can be bad

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

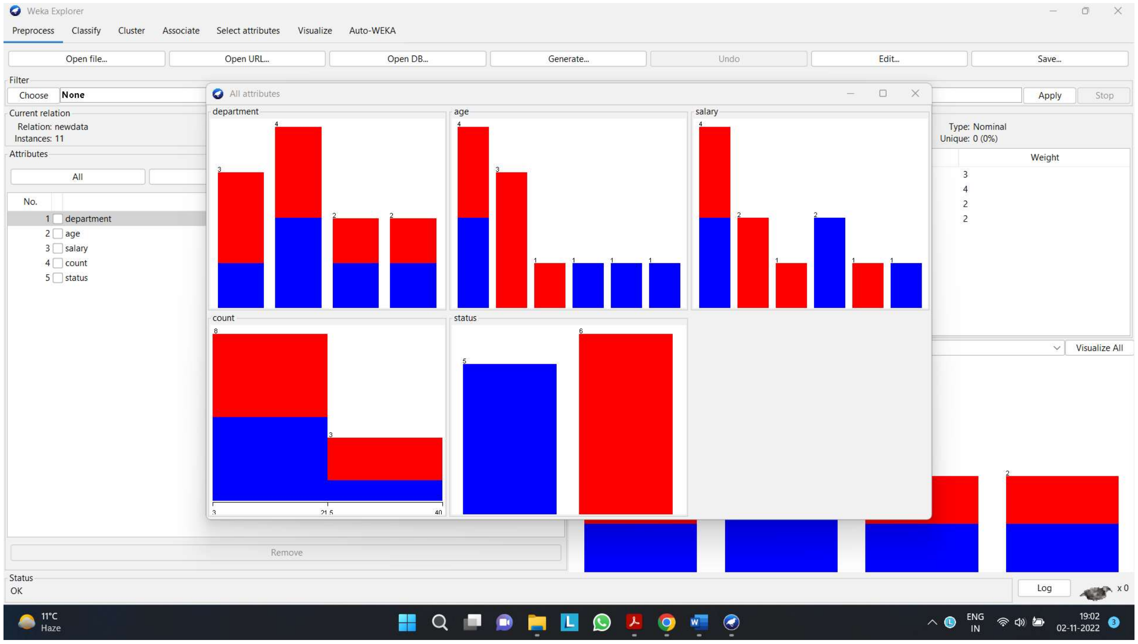
9.1 The following table consists of training data from an employee database. The data have been generalized. For example, "31 ... 35" for *age* represents the age range of 31 to 35. For a given row entry, *count* represents the number of data tuples having the values for *department, status, age,* and *salary* given in that row.

department	*status*	*age*	*salary*	*count*
sales	senior	31 ... 35	46K ... 50K	30
sales	junior	26 ... 30	26K ... 30K	40
sales	junior	31 ... 35	31K ... 35K	40
systems	junior	21 ... 25	46K ... 50K	20
systems	senior	31 ... 35	66K ... 70K	5
systems	junior	26 ... 30	46K ... 50K	3
systems	senior	41 ... 45	66K ... 70K	3
marketing	senior	36 ... 40	46K ... 50K	10
marketing	junior	31 ... 35	41K ... 45K	4
secretary	senior	46 ... 50	36K ... 40K	4
secretary	junior	26 ... 30	26K ... 30K	6

Let *status* be the class-label attribute.

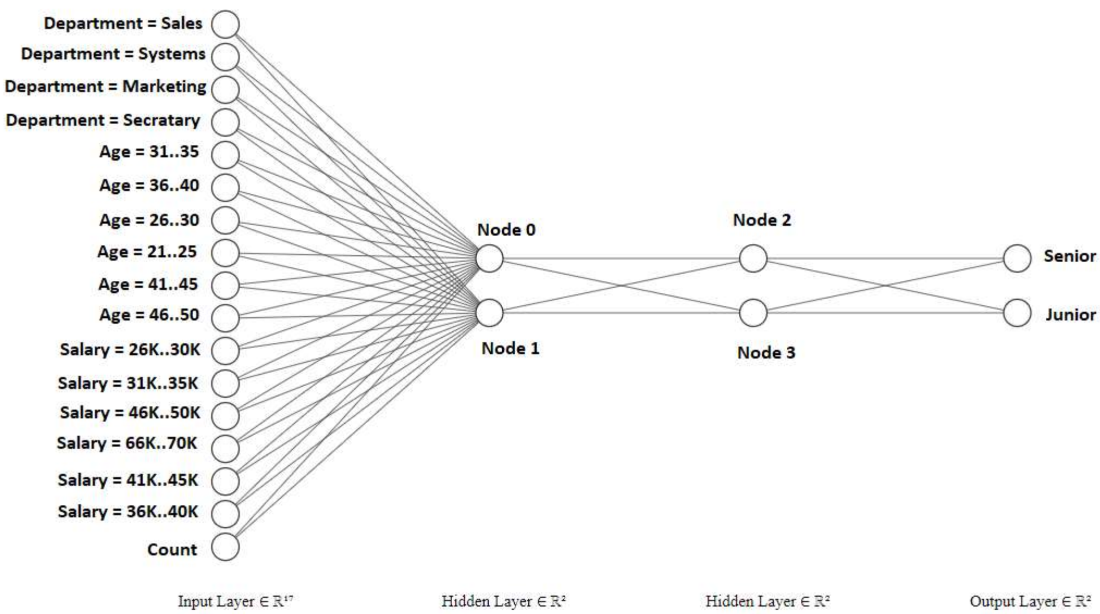
(a) Design a multilayer feed-forward neural network for the given data. Label the nodes in the input and output layers.

(b) Using the multilayer feed-forward neural network obtained in (a), show the weight values after one iteration of the backpropagation algorithm, given the training instance "*(sales, senior, 31 ... 35, 46K ... 50K)*". Indicate your initial weight values and biases and the learning rate used.

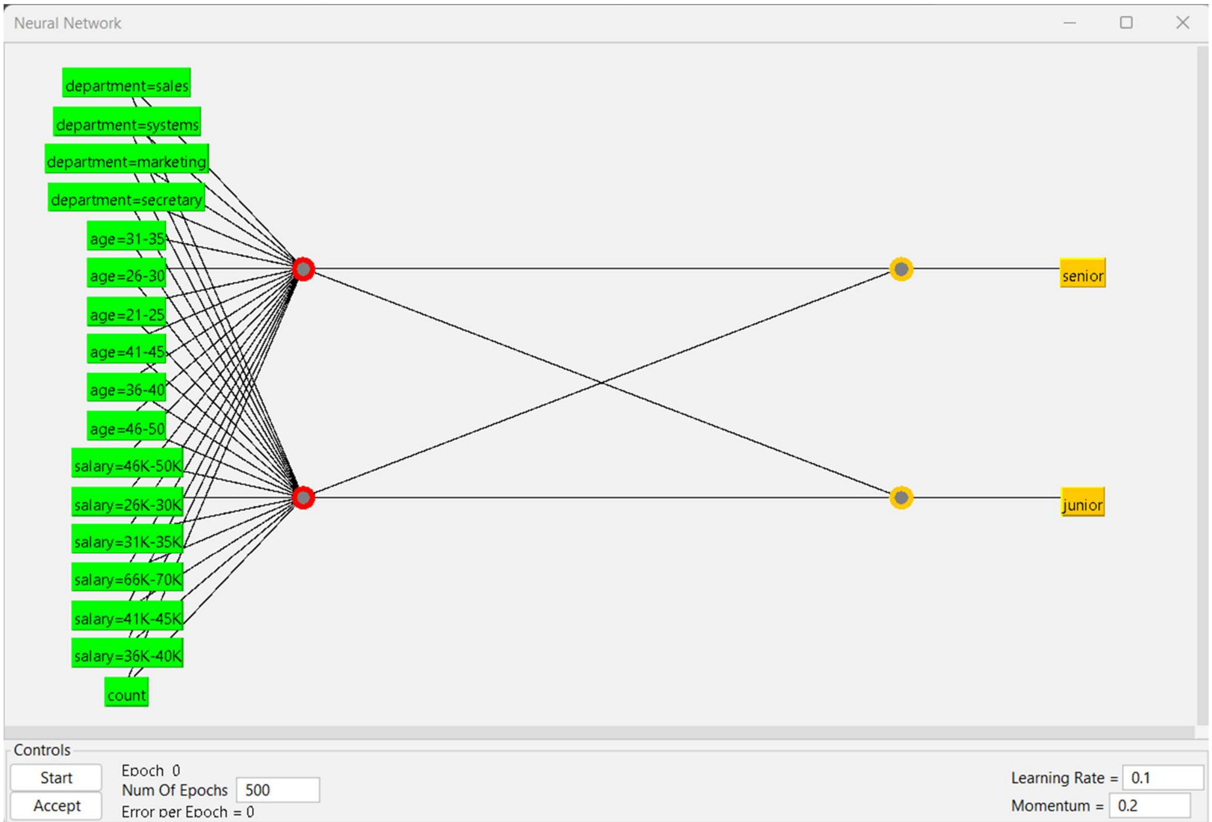


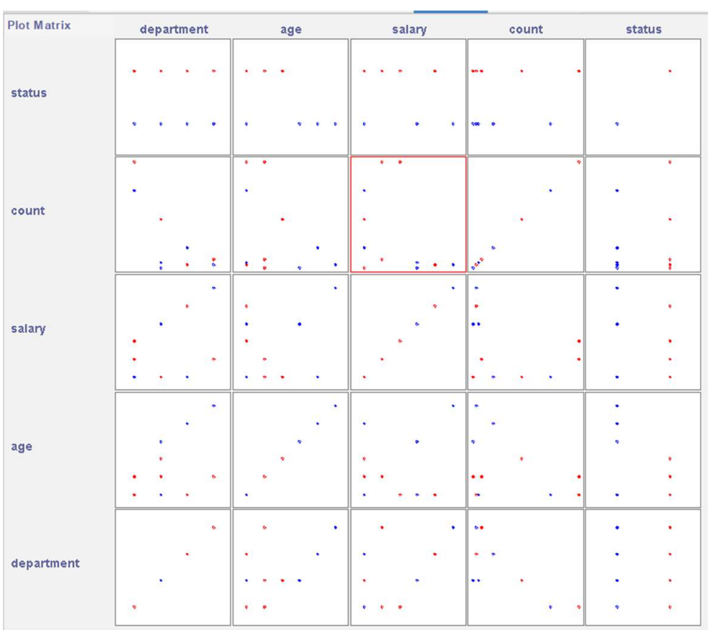
a)

Multilayer feed-forward neural network:



Using Weka

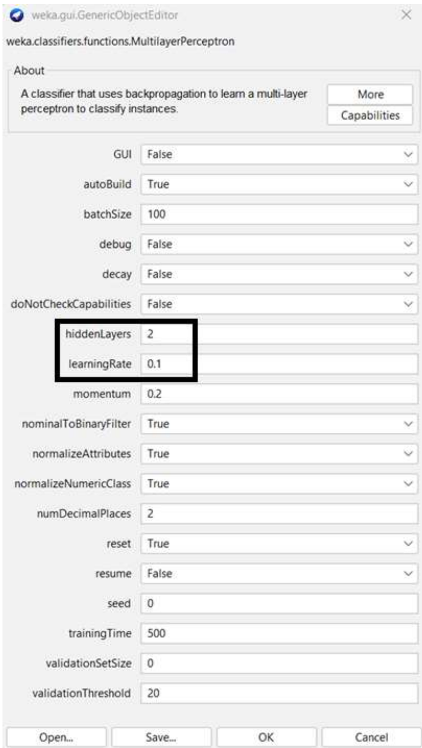

Neural Network
department=sales
department=systems
department=marketing
department=secretary
age=31-35
age=26-30
age=21-25
age=41-45
age=36-40
age=46-50
salary=46K-50K
salary=26K-30K
salary=31K-35K
salary=66K-70K
salary=41K-45K
salary=36K-40K
count
senior
junior
Controls
Start
Accept
Epoch 0
Num Of Epochs 500
Error per Epoch = 0
Learning Rate = 0.1
Momentum = 0.2


Plot Matrix
department
age
salary
count
status

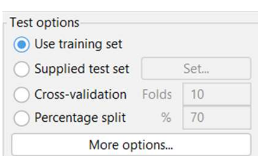
b)

(b) Using the multilayer feed-forward neural network obtained in (a), show the weight values after one iteration of the backpropagation algorithm, given the training instance "*(sales, senior, 31 . . . 35, 46K . . . 50K)*". Indicate your initial weight values and biases and the learning rate used.

For multilayer feed-forward neural network, weights are:



Using training dataset.



Hidden Layers = 2, Learning Rate = 0.1

Weights are:

Classifier output

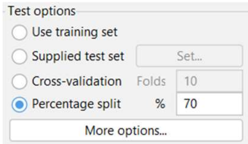
```
=== Run information ===

Scheme:       weka.classifiers.functions.MultilayerPerceptron -L 0.3 -M 0.2 -N 500 -V 0 -S 0 -E 20 -H 2
Relation:     newdata
Instances:    11
Attributes:   5
              department
              age
              salary
              count
              status
Test mode:    evaluate on training data

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

Sigmoid Node 0
    Inputs    Weights
    Threshold    3.4851346848408755
    Node 2    -4.363442578932802
    Node 3    -3.2560705531843728
Sigmoid Node 1
    Inputs    Weights
    Threshold    -3.485352476655363
    Node 2    4.343039891611943
    Node 3    3.277299190573678
Sigmoid Node 2
    Inputs    Weights
    Threshold    0.12841960905221478
    Attrib department=sales    -0.5576542776440633
    Attrib department=systems    0.6331491744498269
    Attrib department=marketing    0.2422522180790835
    Attrib department=secretary    -0.47183089359765656
    Attrib age=31-35    -0.4911086033732522
    Attrib age=26-30    1.7692243958397942
    Attrib age=21-25    1.5078671477123209
    Attrib age=41-45    -0.7822834183690296
    Attrib age=36-40    -1.2821814881474034
```

Classifier output

```
    Attrib age=36-40    -1.2821814881474034
    Attrib age=46-50    -0.9052529604289394
    Attrib salary=46K-50K    -1.1197280374674266
    Attrib salary=26K-30K    0.603707075653472
    Attrib salary=31K-35K    1.6392077812527839
    Attrib salary=66K-70K    -2.128493101438507
    Attrib salary=41K-45K    1.510905496929731
    Attrib salary=36K-40K    -0.9092508064546239
    Attrib count    0.5955476674858969
Sigmoid Node 3
    Inputs    Weights
    Threshold    0.09016921625334368
    Attrib department=sales    -0.4782474015342242
    Attrib department=systems    0.49698630289865714
    Attrib department=marketing    0.21278267210539267
    Attrib department=secretary    -0.4287448956888747
    Attrib age=31-35    -0.39090284241848144
    Attrib age=26-30    1.5649571870867165
    Attrib age=21-25    1.2752871060028073
    Attrib age=41-45    -0.6633160123177847
    Attrib age=36-40    -1.1267201193365655
    Attrib age=46-50    -0.8057395035681446
    Attrib salary=46K-50K    -0.8968887314853673
    Attrib salary=26K-30K    0.5752551319783026
    Attrib salary=31K-35K    1.3948924225197412
    Attrib salary=66K-70K    -1.7755708392366605
    Attrib salary=41K-45K    1.2691247600700892
    Attrib salary=36K-40K    -0.7219787887416631
    Attrib count    0.5041282077292387
Class senior
    Input
    Node 0
Class junior
    Input
    Node 1
```

```
Time taken to build model: 0.02 seconds

=== Evaluation on training set ===

Time taken to test model on training data: 0 seconds

=== Summary ===

Correctly Classified Instances          11              100      %
Incorrectly Classified Instances         0                0      %
Kappa statistic                          1
Mean absolute error                      0.0298
Root mean squared error                  0.0307
Relative absolute error                  5.9974 %
Root relative squared error              6.1709 %
Total Number of Instances               11

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     senior
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     junior
Weighted Avg.    1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000

=== Confusion Matrix ===

 a b   <-- classified as
 5 0 | a = senior
 0 6 | b = junior
```

Using 70-30 split.



Multilayer feed-forward neural network

Weights are:

Classifier output

```
=== Run information ===

Scheme:       weka.classifiers.functions.MultilayerPerceptron -L 0.1 -M 0.2 -N 500 -V 0 -S 0 -E 20 -H 2
Relation:     newdata
Instances:    11
Attributes:   5
              department
              age
              salary
              count
              status
Test mode:    split 70.0% train, remainder test

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

Sigmoid Node 0
    Inputs    Weights
    Threshold    2.7718152052617615
    Node 2    -3.7612727795000502
    Node 3    -2.4756334465425955
Sigmoid Node 1
    Inputs    Weights
    Threshold    -2.7716965117260437
    Node 2    3.7412409700861677
    Node 3    2.4961510271168956
Sigmoid Node 2
    Inputs    Weights
    Threshold    0.11427927065628996
    Attrib department=sales    -0.4905874198192064
    Attrib department=systems    0.5674878193589538
    Attrib department=marketing    0.21676922615655067
    Attrib department=secretary    -0.419472727617253
    Attrib age=31-35    -0.41904617553810164
    Attrib age=26-30    1.6317956458031824
    Attrib age=21-25    1.3684879231458835
    Attrib age=41-45    -0.7259713069235665
    Attrib age=36-40    -1.1554849005186978
```

Classifier output

```
    Attrib age=46-50    -0.8269547591515234
    Attrib salary=46K-50K    -1.005766109748428
    Attrib salary=26K-30K    0.5905861520562155
    Attrib salary=31K-35K    1.4425031362818082
    Attrib salary=66K-70K    -1.9163270671276926
    Attrib salary=41K-45K    1.3728662557744244
    Attrib salary=36K-40K    -0.830952605177208
    Attrib count    0.5456685767301938
Sigmoid Node 3
    Inputs    Weights
    Threshold    0.06703383302989854
    Attrib department=sales    -0.3713443399723155
    Attrib department=systems    0.3969489178990156
    Attrib department=marketing    0.1714314892756246
    Attrib department=secretary    -0.34798862297447886
    Attrib age=31-35    -0.29668085039795056
    Attrib age=26-30    1.352778827643681
    Attrib age=21-25    1.0618954449777438
    Attrib age=41-45    -0.5577207052234878
    Attrib age=36-40    -0.9298016753817471
    Attrib age=46-50    -0.6843636932759224
    Attrib salary=46K-50K    -0.7167135160186119
    Attrib salary=26K-30K    0.5446796518998308
    Attrib salary=31K-35K    1.1135001146807275
    Attrib salary=66K-70K    -1.4574783006232468
    Attrib salary=41K-45K    1.053990516508938
    Attrib salary=36K-40K    -0.6006029784494419
    Attrib count    0.4311599550271979
Class senior
    Input
    Node 0
Class junior
    Input
    Node 1


Time taken to build model: 0.09 seconds
```

```
=== Evaluation on test split ===

Time taken to test model on test split: 0 seconds

=== Summary ===

Correctly Classified Instances           2               66.6667 %
Incorrectly Classified Instances         1               33.3333 %
Kappa statistic                          0.4
Mean absolute error                      0.3894
Root mean squared error                  0.5438
Relative absolute error                 77.8766 %
Root relative squared error            108.7506 %
Total Number of Instances                3

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    0.500    0.500      1.000    0.667      0.500    0.500     0.500     senior
                 0.500    0.000    1.000      0.500    0.667      0.500    0.500     0.833     junior
Weighted Avg.    0.667    0.167    0.833      0.667    0.667      0.500    0.500     0.722

=== Confusion Matrix ===

 a b   <-- classified as
 1 0 | a = senior
 1 1 | b = junior
```

SVM

Using training dataset.

- **Lib SVM**

Classifier output

```
=== Run information ===

Scheme:       weka.classifiers.functions.LibSVM -S 0 -K 2 -D 3 -G 0.0 -R 0.0 -N 0.5 -M 40.0 -C 1.0 -E 0.001 -P 0.1 -model "C:\\Program Files\\Weka-3-8-6" -seed 1
Relation:     newdata
Instances:    11
Attributes:   5
              department
              age
              salary
              count
              status
Test mode:    evaluate on training data

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

LibSVM wrapper, original code by Yasser EL-Manzalawy (= WLSVM)

Time taken to build model: 0 seconds

=== Evaluation on training set ===

Time taken to test model on training data: 0 seconds

=== Summary ===

Correctly Classified Instances          11              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               11
```

```
=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     senior
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     junior
Weighted Avg.    1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000

=== Confusion Matrix ===

 a b   <-- classified as
 5 0 | a = senior
 0 6 | b = junior
```

- **SMO**

Classifier output

```
=== 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 -E 1.0 -C 250007" -calibrator "w
Relation:     newdata
Instances:    11
Attributes:   5
              department
              age
              salary
              count
              status
Test mode:    evaluate on training data

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

SMO

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

Classifier for classes: senior, junior

BinarySMO

Machine linear: showing attribute weights, not support vectors.

        -0.1307 * (normalized) department=sales
 +       0.3128 * (normalized) department=systems
 +       0.0729 * (normalized) department=marketing
 +      -0.255  * (normalized) department=secretary
 +      -0.1664 * (normalized) age=31-35
 +       0.9261 * (normalized) age=26-30
 +       0.7862 * (normalized) age=21-25
 +      -0.2169 * (normalized) age=41-45
 +      -0.8913 * (normalized) age=36-40
 +      -0.4376 * (normalized) age=46-50
 +      -0.3616 * (normalized) salary=46K-50K
 +       0.1826 * (normalized) salary=26K-30K
 +       0.8693 * (normalized) salary=31K-35K
 +      -1.2169 * (normalized) salary=66K-70K
 +       0.9643 * (normalized) salary=41K-45K
 +      -0.4376 * (normalized) salary=36K-40K
 +       0.3071 * (normalized) count
 +       0.1217

Number of kernel evaluations: 61 (92.269% cached)



Time taken to build model: 0.01 seconds

=== Evaluation on training set ===

Time taken to test model on training data: 0 seconds

=== Summary ===

Correctly Classified Instances          11              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               11

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     senior
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     junior
Weighted Avg.    1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000

=== Confusion Matrix ===

 a b   <-- classified as
 5 0 | a = senior
 0 6 | b = junior
```

Using 70-30 split.

- **SMO**

Classifier output

```
=== 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 -E 1.0 -C 250007" -calibrator "w
Relation:     newdata
Instances:    11
Attributes:   5
              department
              age
              salary
              count
              status
Test mode:    split 70.0% train, remainder test

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

SMO

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

Classifier for classes: senior, junior

BinarySMO

Machine linear: showing attribute weights, not support vectors.

        -0.1307 * (normalized) department=sales
 +       0.3128 * (normalized) department=systems
 +       0.0729 * (normalized) department=marketing
 +      -0.255  * (normalized) department=secretary
 +      -0.1664 * (normalized) age=31-35
 +       0.9261 * (normalized) age=26-30
 +       0.7862 * (normalized) age=21-25
 +      -0.2169 * (normalized) age=41-45
 +      -0.8913 * (normalized) age=36-40
 +      -0.4376 * (normalized) age=46-50
 +      -0.3616 * (normalized) salary=46K-50K
```

Classifier output

```
 +      -0.3616 * (normalized) salary=46K-50K
 +       0.1826 * (normalized) salary=26K-30K
 +       0.8693 * (normalized) salary=31K-35K
 +      -1.2169 * (normalized) salary=66K-70K
 +       0.9643 * (normalized) salary=41K-45K
 +      -0.4376 * (normalized) salary=36K-40K
 +       0.3071 * (normalized) count
 +       0.1217

Number of kernel evaluations: 61 (92.269% cached)



Time taken to build model: 0.02 seconds

=== Evaluation on test split ===

Time taken to test model on test split: 0 seconds

=== Summary ===

Correctly Classified Instances           2               66.6667 %
Incorrectly Classified Instances         1               33.3333 %
Kappa statistic                          0.4
Mean absolute error                      0.3333
Root mean squared error                  0.5774
Relative absolute error                 66.6667 %
Root relative squared error            115.4701 %
Total Number of Instances                3

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    0.500    0.500      1.000    0.667      0.500    0.750     0.500     senior
                 0.500    0.000    1.000      0.500    0.667      0.500    0.750     0.833     junior
Weighted Avg.    0.667    0.167    0.833      0.667    0.667      0.500    0.750     0.722

=== Confusion Matrix ===

 a b   <-- classified as
 1 0 | a = senior
 1 1 | b = junior
```

- **Lib SVM**

Classifier output

```
=== Run information ===

Scheme:       weka.classifiers.functions.LibSVM -S 0 -K 2 -D 3 -G 0.0 -R 0.0 -N 0.5 -M 40.0 -C 1.0 -E 0.001 -P 0.1 -model "C:\\Program Files\\Weka-3-8-6" -seed 1
Relation:     newdata
Instances:    11
Attributes:   5
              department
              age
              salary
              count
              status
Test mode:    split 70.0% train, remainder test

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

LibSVM wrapper, original code by Yasser EL-Manzalawy (= WLSVM)

Time taken to build model: 0.01 seconds

=== Evaluation on test split ===

Time taken to test model on test split: 0 seconds

=== Summary ===

Correctly Classified Instances           2               66.6667 %
Incorrectly Classified Instances         1               33.3333 %
Kappa statistic                          0.4
Mean absolute error                      0.3333
Root mean squared error                  0.5774
Relative absolute error                 66.6667 %
Root relative squared error            115.4701 %
Total Number of Instances                3
```

```
=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    0.500    0.500      1.000    0.667      0.500    0.750     0.500     senior
                 0.500    0.000    1.000      0.500    0.667      0.500    0.750     0.833     junior
Weighted Avg.    0.667    0.167    0.833      0.667    0.667      0.500    0.750     0.722

=== Confusion Matrix ===

 a b   <-- classified as
 1 0 | a = senior
 1 1 | b = junior
```

I will be using Logistic Regression classifier.

Logistic Regression

Using training dataset.

Classifier output

```
=== Run information ===

Scheme:       weka.classifiers.functions.Logistic -R 1.0E-8 -M -1 -num-decimal-places 4
Relation:     newdata
Instances:    11
Attributes:   5
              department
              age
              salary
              count
              status
Test mode:    evaluate on training data

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

Logistic Regression with ridge parameter of 1.0E-8
Coefficients...
                                 Class
Variable                        senior
======================================
department=sales               15.4121
department=systems            -11.7066
department=marketing           -7.3458
department=secretary            5.0068
age=31-35                      11.1206
age=26-30                     -19.7511
age=21-25                     -31.6926
age=41-45                      14.3942
age=36-40                      19.1757
age=46-50                      14.3878
salary=46K-50K                 10.7576
salary=26K-30K                 -5.9401
salary=31K-35K                -37.0433
salary=66K-70K                 19.7914
salary=41K-45K                -32.3982
salary=36K-40K                 14.3878
count                           0.0727
```

```
count                            0.0727
Intercept                       -6.1169


Odds Ratios...
                                  Class
Variable                         senior
=====================================
department=sales           4935937.8646
department=systems                    0
department=marketing             0.0006
department=secretary           149.4213
age=31-35                    67546.1661
age=26-30                             0
age=21-25                             0
age=41-45                  1783732.7191
age=36-40                212756728.9276
age=46-50                  1772393.1928
salary=46K-50K               46983.4626
salary=26K-30K                   0.0026
salary=31K-35K                        0
salary=66K-70K           393836095.7563
salary=41K-45K                        0
salary=36K-40K             1772393.1928
count                            1.0754


Time taken to build model: 0.01 seconds

=== Evaluation on training set ===

Time taken to test model on training data: 0 seconds
```

```
=== Summary ===

Correctly Classified Instances          11              100      %
Incorrectly Classified Instances         0                0      %
Kappa statistic                          1
Mean absolute error                      0
Root mean squared error                  0
Relative absolute error                  0.0001 %
Root relative squared error              0.0001 %
Total Number of Instances               11

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     senior
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     junior
Weighted Avg.    1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000

=== Confusion Matrix ===

 a b   <-- classified as
 5 0 | a = senior
 0 6 | b = junior
```

Logistic Regression

Using 70-30 split

```
=== Run information ===

Scheme:       weka.classifiers.functions.Logistic -R 1.0E-8 -M -1 -num-decimal-places 4
Relation:     newdata
Instances:    11
Attributes:   5
              department
              age
              salary
              count
              status
Test mode:    split 70.0% train, remainder test

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

Logistic Regression with ridge parameter of 1.0E-8
Coefficients...
                                  Class
Variable                         senior
=====================================
department=sales                15.4121
department=systems             -11.7066
department=marketing            -7.3458
department=secretary             5.0068
age=31-35                       11.1206
age=26-30                      -19.7511
age=21-25                      -31.6926
age=41-45                       14.3942
age=36-40                       19.1757
age=46-50                       14.3878
salary=46K-50K                  10.7576
salary=26K-30K                  -5.9401
salary=31K-35K                 -37.0433
salary=66K-70K                  19.7914
salary=41K-45K                 -32.3982
salary=36K-40K                  14.3878
count                            0.0727
```

```
Intercept                       -6.1169


Odds Ratios...
                                  Class
Variable                         senior
=======================================
department=sales           4935937.8646
department=systems                    0
department=marketing             0.0006
department=secretary           149.4213
age=31-35                    67546.1661
age=26-30                             0
age=21-25                             0
age=41-45                  1783732.7191
age=36-40                212756728.9276
age=46-50                  1772393.1928
salary=46K-50K               46983.4626
salary=26K-30K                   0.0026
salary=31K-35K                        0
salary=66K-70K           393836095.7563
salary=41K-45K                        0
salary=36K-40K             1772393.1928
count                            1.0754


Time taken to build model: 0.02 seconds

=== Evaluation on test split ===

Time taken to test model on test split: 0 seconds

=== Summary ===
```

```
Correctly Classified Instances           2               66.6667 %
Incorrectly Classified Instances         1               33.3333 %
Kappa statistic                          0.4
Mean absolute error                      0.3333
Root mean squared error                  0.5774
Relative absolute error                 66.6667 %
Root relative squared error            115.4701 %
Total Number of Instances                3

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    0.500    0.500      1.000    0.667      0.500    0.500     0.500     senior
                 0.500    0.000    1.000      0.500    0.667      0.500    0.500     0.833     junior
Weighted Avg.    0.667    0.167    0.833      0.667    0.667      0.500    0.500     0.722

=== Confusion Matrix ===

 a b   <-- classified as
 1 0 | a = senior
 1 1 | b = junior
```

11.2 *AllElectronics* carries 1000 products, $P1, \ldots, P1000$. Consider customers Ada, Bob, and Cathy such that Ada and Bob purchase three products in common, $P1$,$P2$, and $P3$. For the other 997 products, Ada and Bob independently purchase seven of them randomly. Cathy purchases 10 products, randomly selected from the 1000 products. In Euclidean distance, what is the probability that *dist*.Ada,Bob/ > *dist*.Ada,Cathy/?What if Jaccard similarity (Chapter 2) is used?What can you learn from this example?

Answer) Codes are in Jupyter Notebook

For Ada,Bob

```
AB_df #Ada,Bob
```

	I	Probability	Euclidean Distance	Jaccard
0	3.0	9.517334e-01	3.741657	0.176471
1	4.0	4.739323e-02	3.464102	0.250000
2	5.0	8.660691e-04	3.162278	0.333333
3	6.0	7.319719e-06	2.828427	0.428571
4	7.0	2.966451e-08	2.449490	0.538462
5	8.0	5.404466e-11	2.000000	0.666667
6	9.0	3.643051e-14	1.414214	0.818182
7	10.0	5.256928e-18	0.000000	1.000000

For Ada,Cathy

```
#Ada,Cathy
AC_df
```

	J	Probability	Euclidean Distance	Jaccard
0	1.0	9.214765e-02	4.242641	0.052632
1	2.0	3.800387e-03	4.000000	0.111111
2	3.0	8.247703e-05	3.741657	0.176471
3	4.0	1.026772e-06	3.464102	0.250000
4	5.0	7.505338e-09	3.162278	0.333333
5	6.0	3.171627e-11	2.828427	0.428571
6	7.0	7.344917e-14	2.449490	0.538462
7	8.0	8.363392e-17	2.000000	0.666667
8	9.0	3.758406e-20	1.414214	0.818182
9	10.0	3.796369e-24	0.000000	1.000000

Used

$\binom{7}{i-3} * \binom{990}{10-i} / \binom{997}{7}$ for calculation probability for Ada,Bob

$\binom{10}{j} * \binom{990}{10-j} / \binom{1000}{10}$ for calculation probability for Ada,Cathy

Probability that *dist*.Ada,Bob/ > *dist*.Ada,Cathy

Using Euclidean

```
sum_euc = 0
for i in range(2, 9):
    for j in range(i+1,10):
        sum_euc += pr_i[i]*pr_j[j]
sum_euc
```

```
9.847436804201952e-07
```

Using Jaccard

```
sum_jac = 0
for i in range(2, 10):
    for j in range(0,i-1):
        sum_jac += pr_i[i]*pr_j[j]
sum_jac
```

```
0.09233115445920217
```

Used

$\sum_{i=3}^{9}(\Pr(i)\sum_{j=i+1}^{10}\Pr(j))$ for Euclidean Distance. Value is 9.847436804201952e-07

$\sum_{i=3}^{10}(\Pr(i)\sum_{j=1}^{i-1}\Pr(j))$ for Jaccard Similarity. Value is 0.09233115445920217

Higher the Jaccard similarity means similar are the two quantities, while higher the Euclidean distance, more different will be two quantities.

Also, Jaccard similarity has a range between 0 to 1. Euclidean distance can only be non-negative.