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

**

The distance function is Euclidean distance. Suppose initially we assign *A*1, *B*1, and *C*1 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.

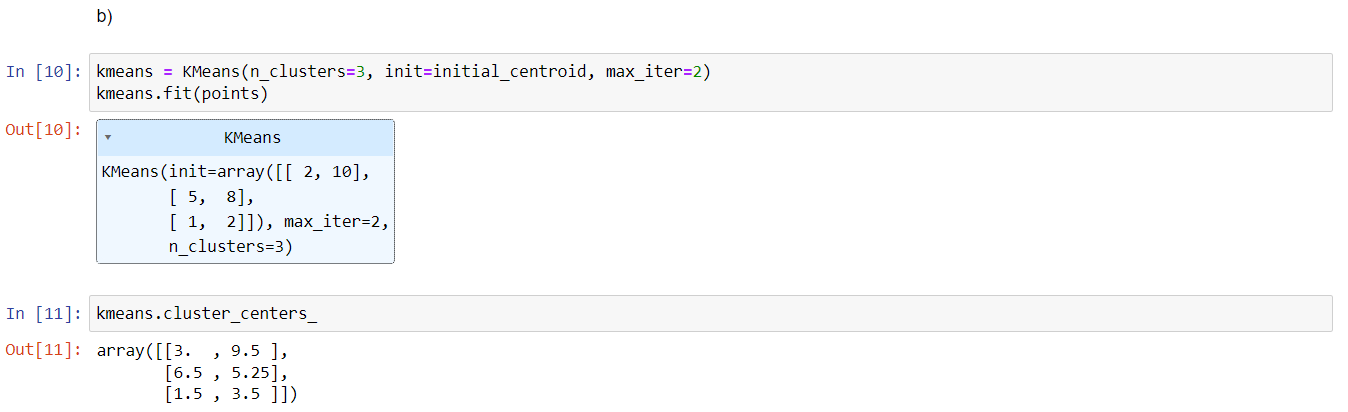


The three cluster centers after the first round of execution are:

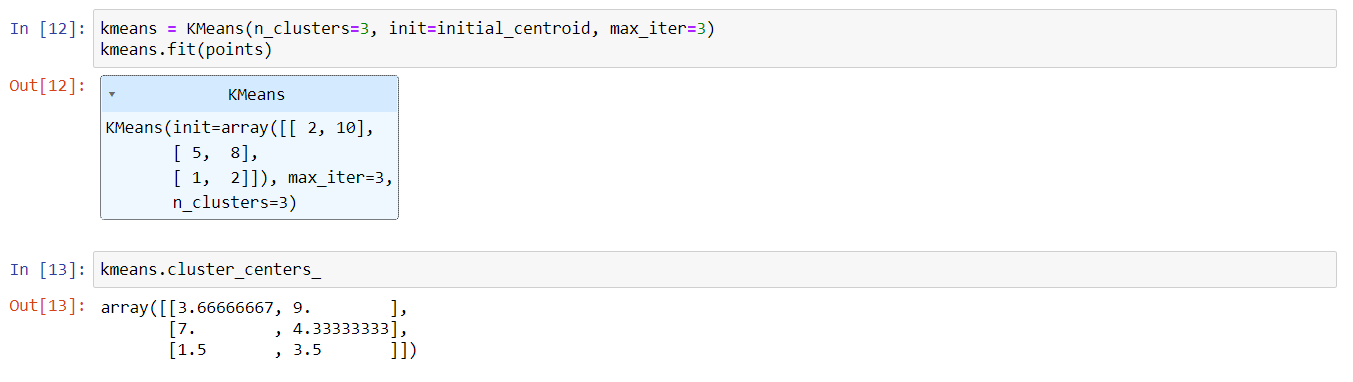
[2,10] , [6,6] and [1.5,3.5]

**(b)** The final three clusters.

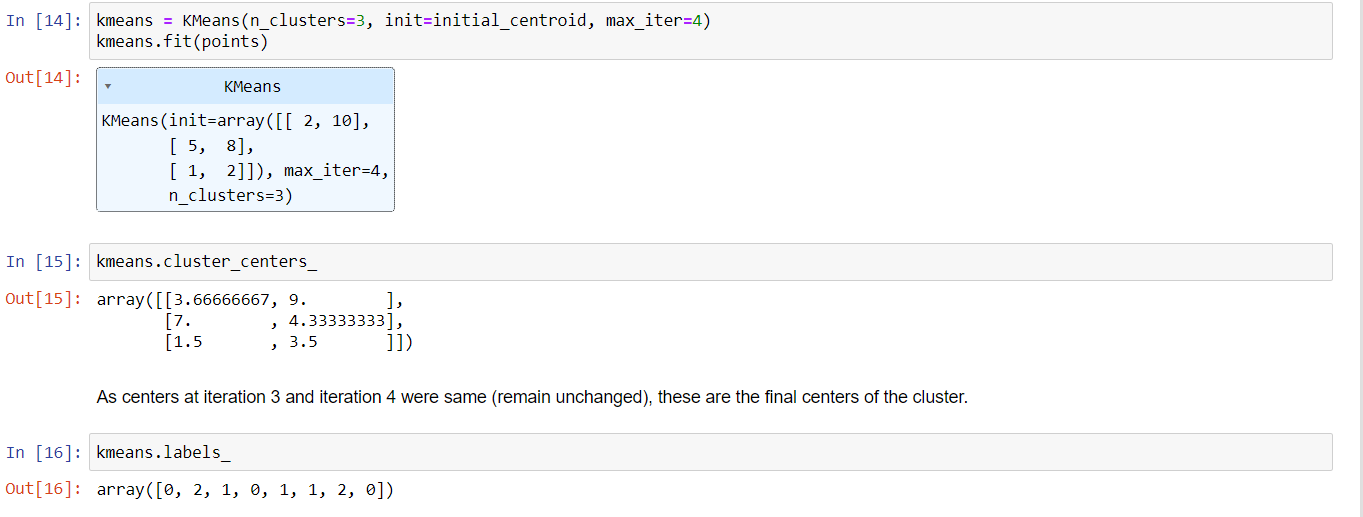
Iteration 2:



Iteration 3:



Iteration 4:



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.

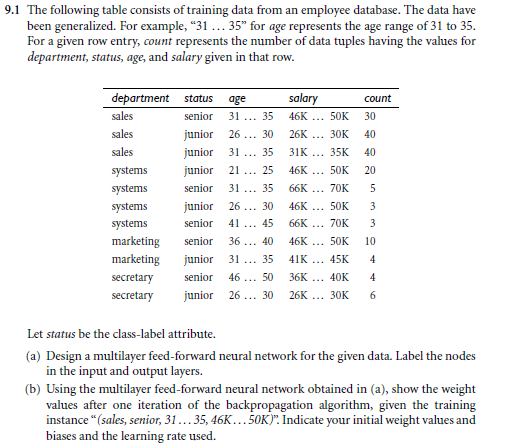
**(b)** Illustrate the strength and weakness of these schemes in comparison with a hierarchical

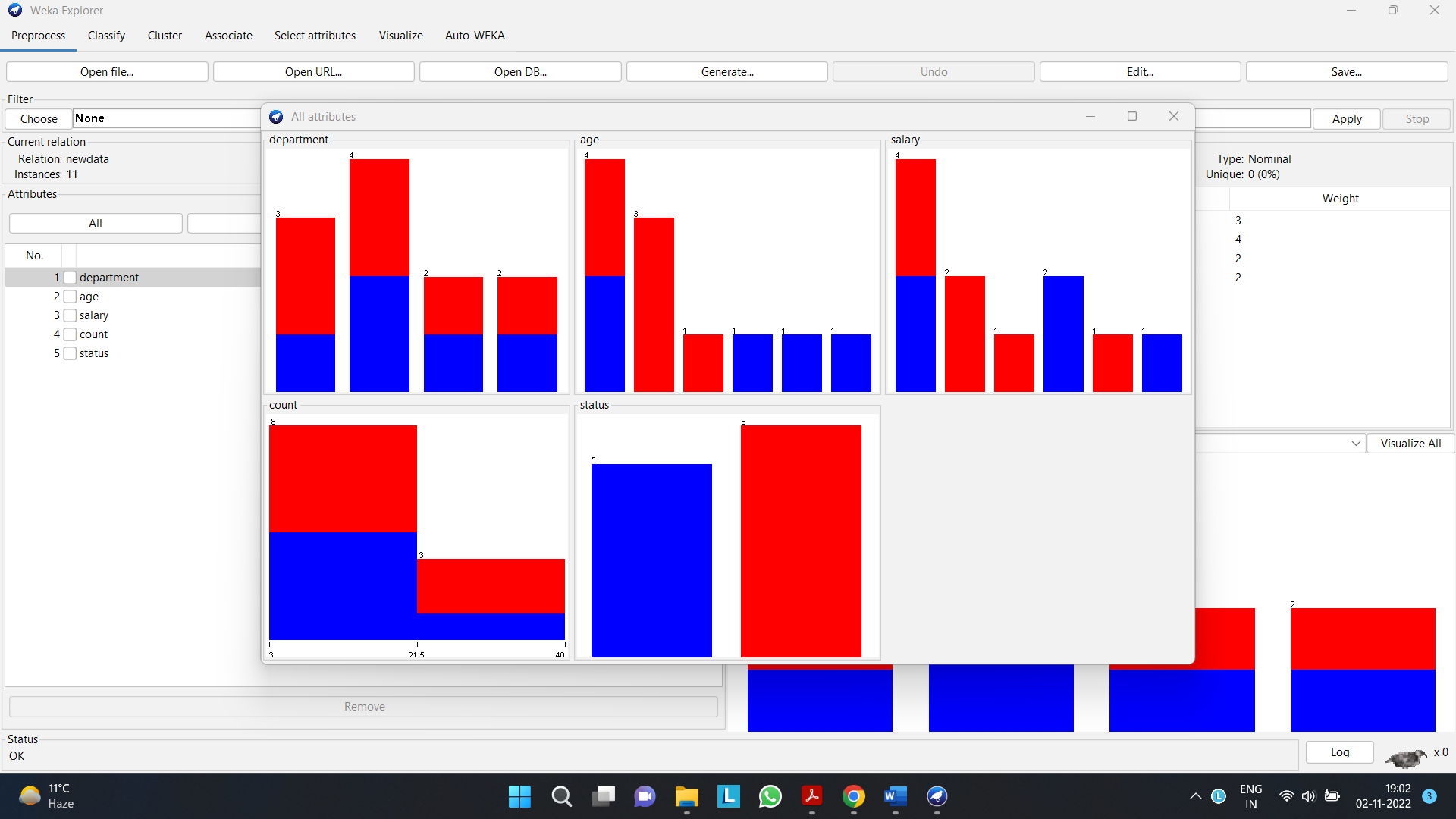
clustering scheme (e.g., AGNES).

**Answer)**

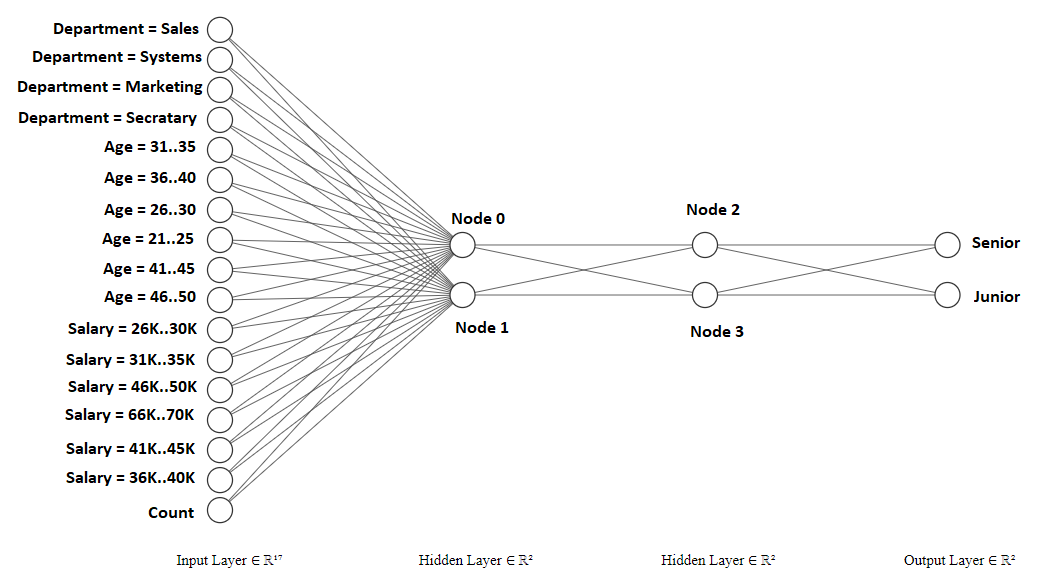
|  |  |
| --- | --- |
| **Partitioning** | **Hierarchical** |
| Weakness   1. Need to specify the number of clusters to partition which may be unknown information before some testing 2. Less efficient | Strength   1. No need to specify the number of clusters beforehand. 2. More efficient |
| Strength   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**

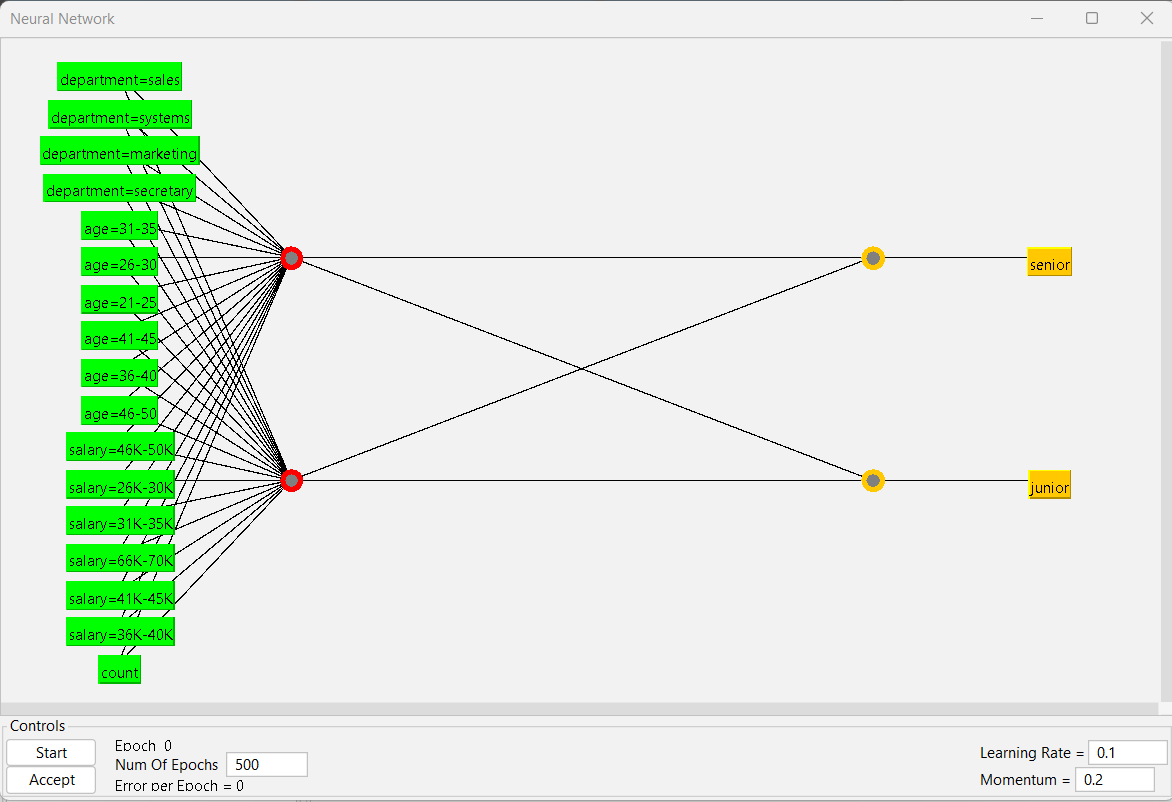


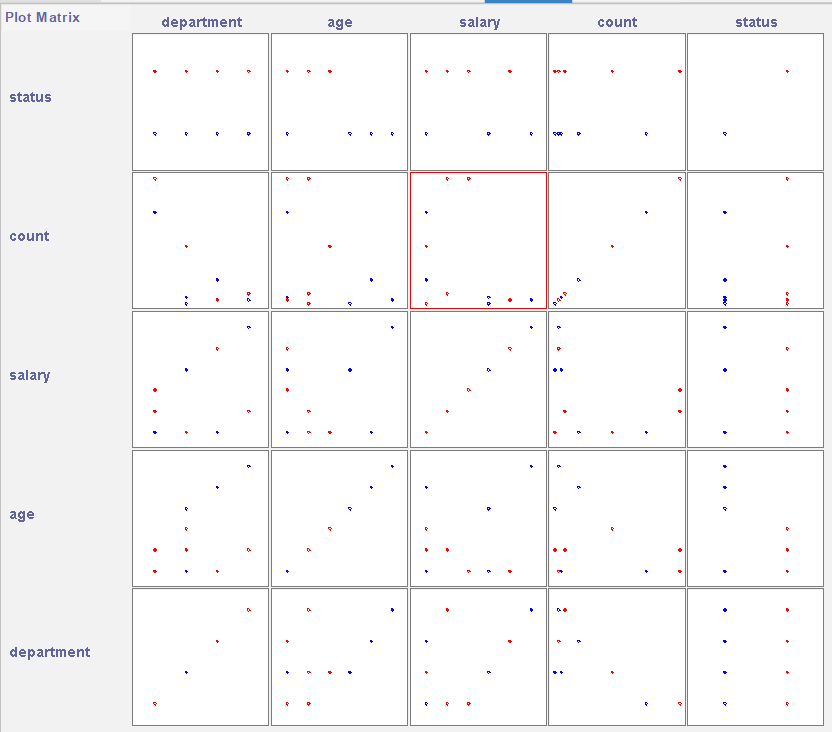


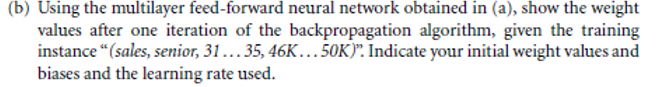
Multilayer feed-forward neural network:



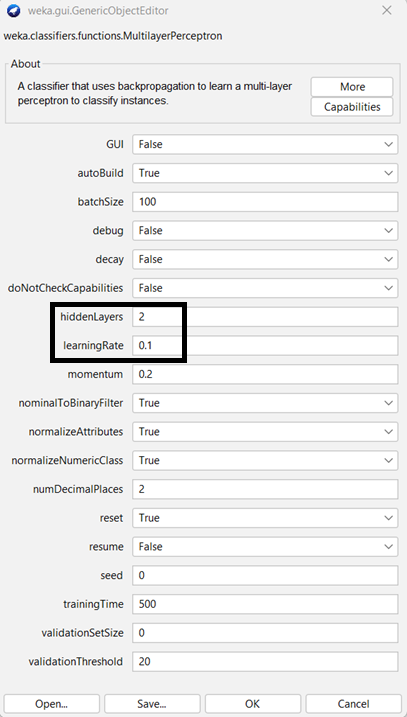
**Using Weka**

****

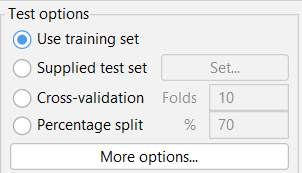




**For multilayer feed-forward neural network, weights are:**

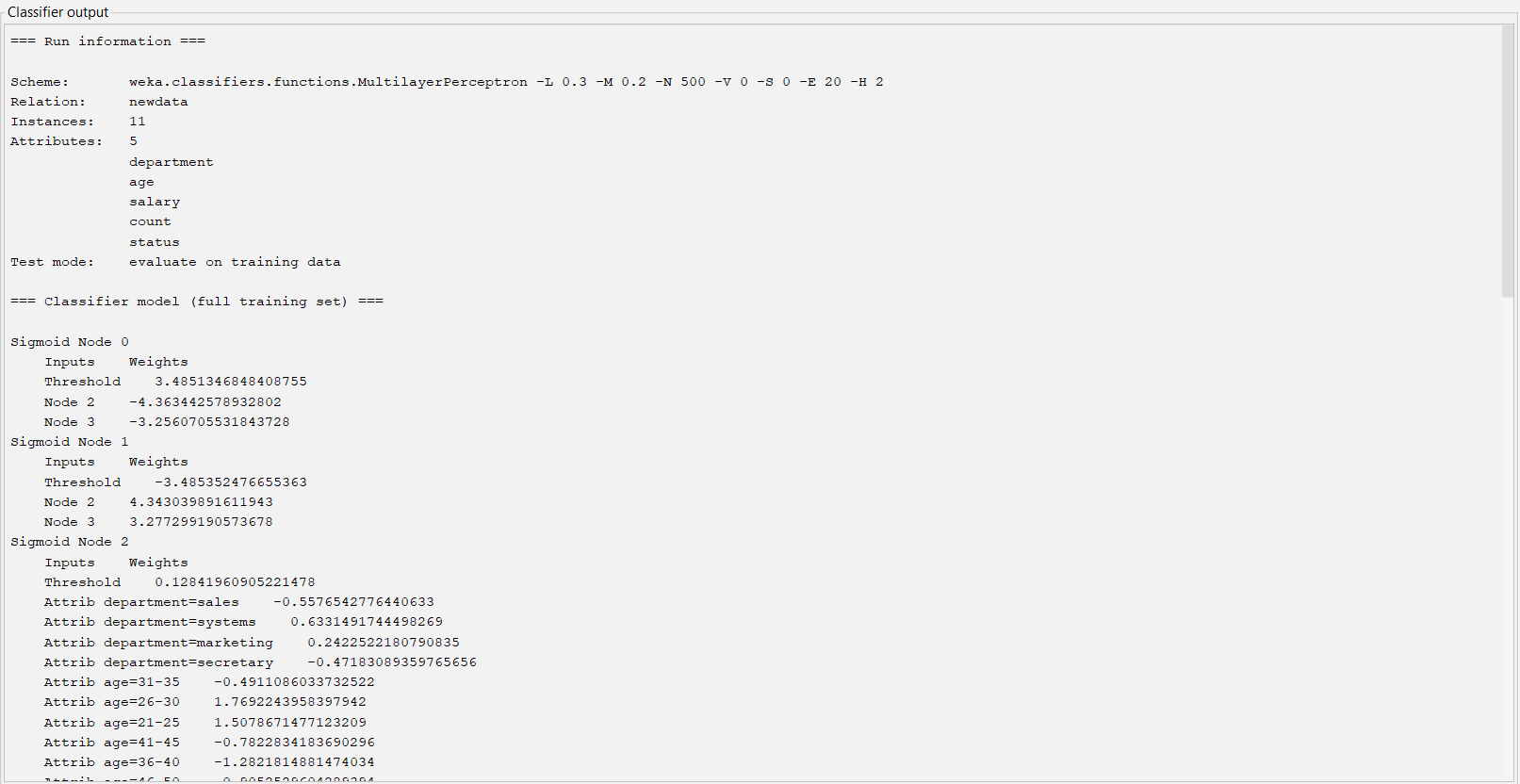
****

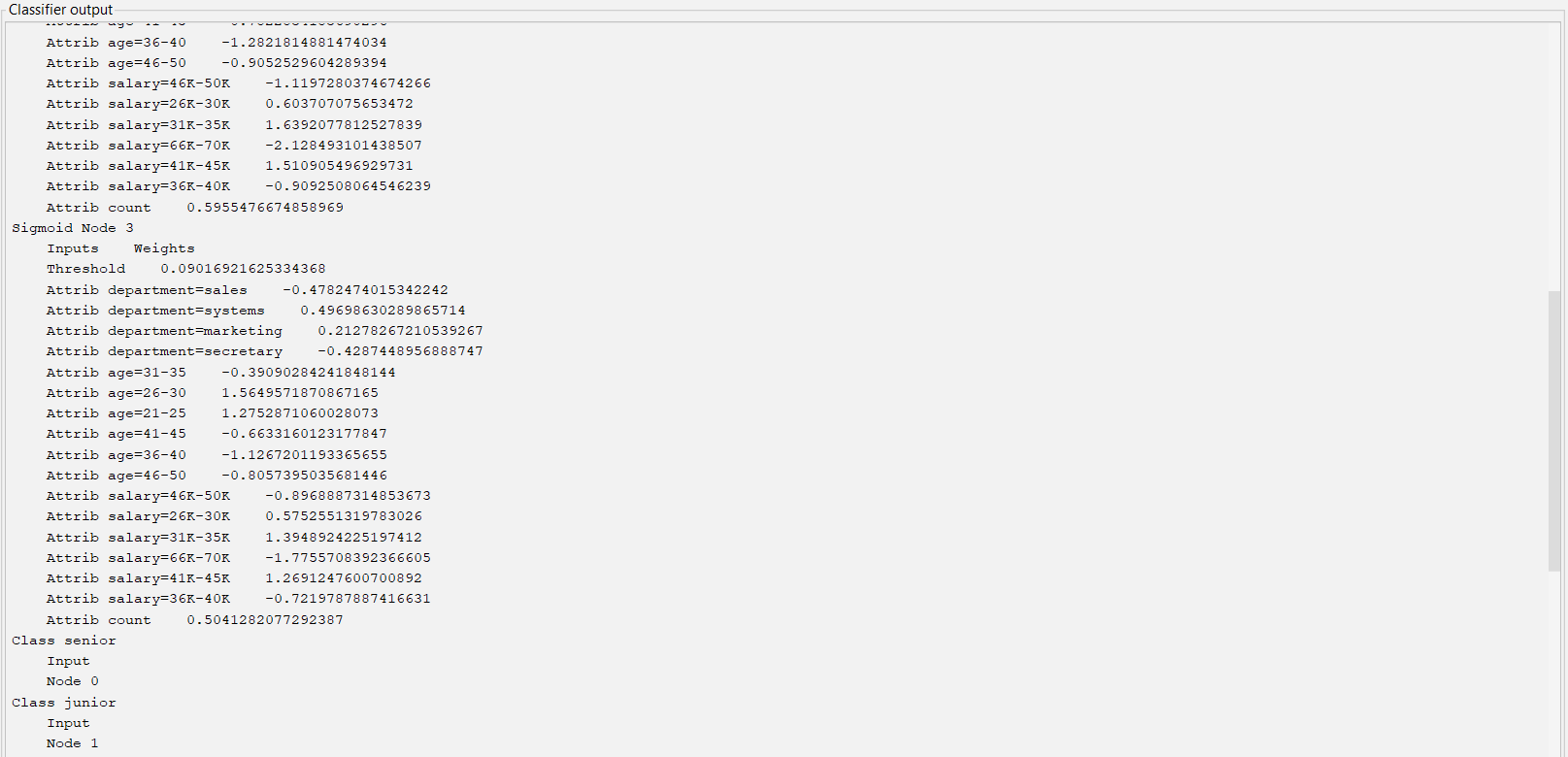
**Using training dataset**.

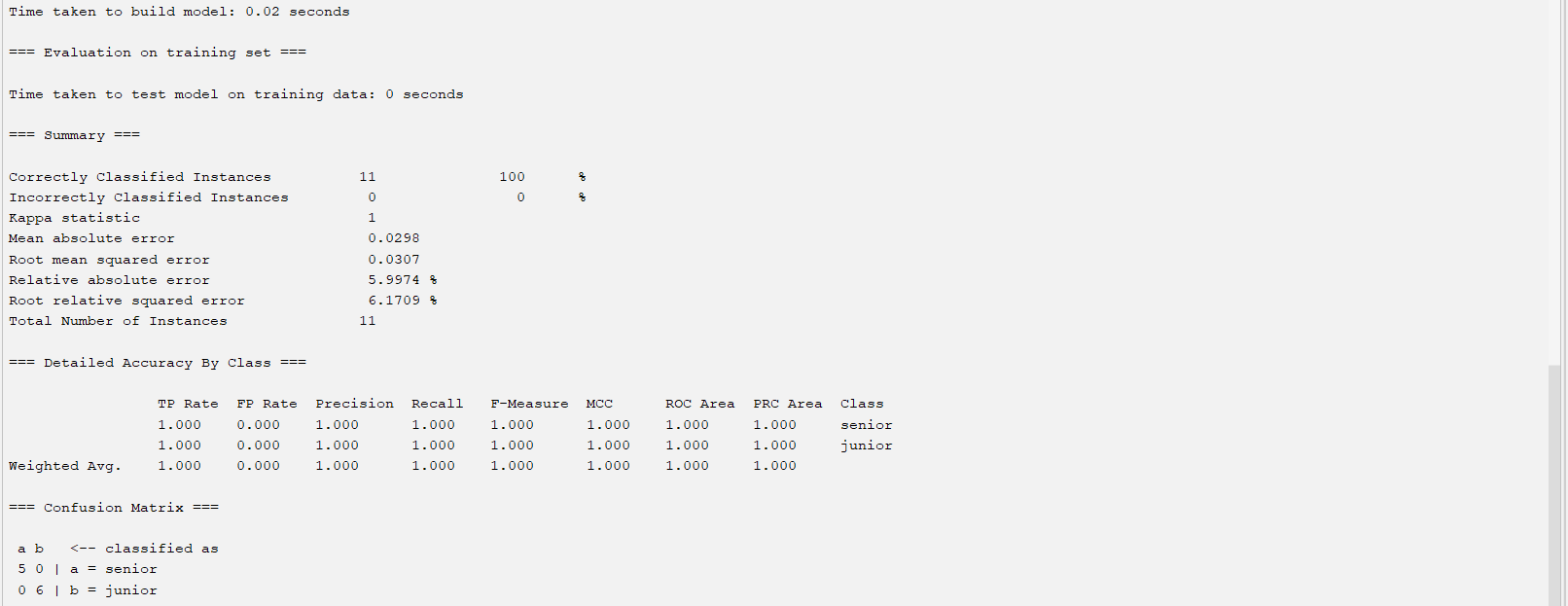


Hidden Layers = 2, Learning Rate = 0.1

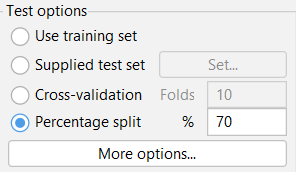
Weights are:





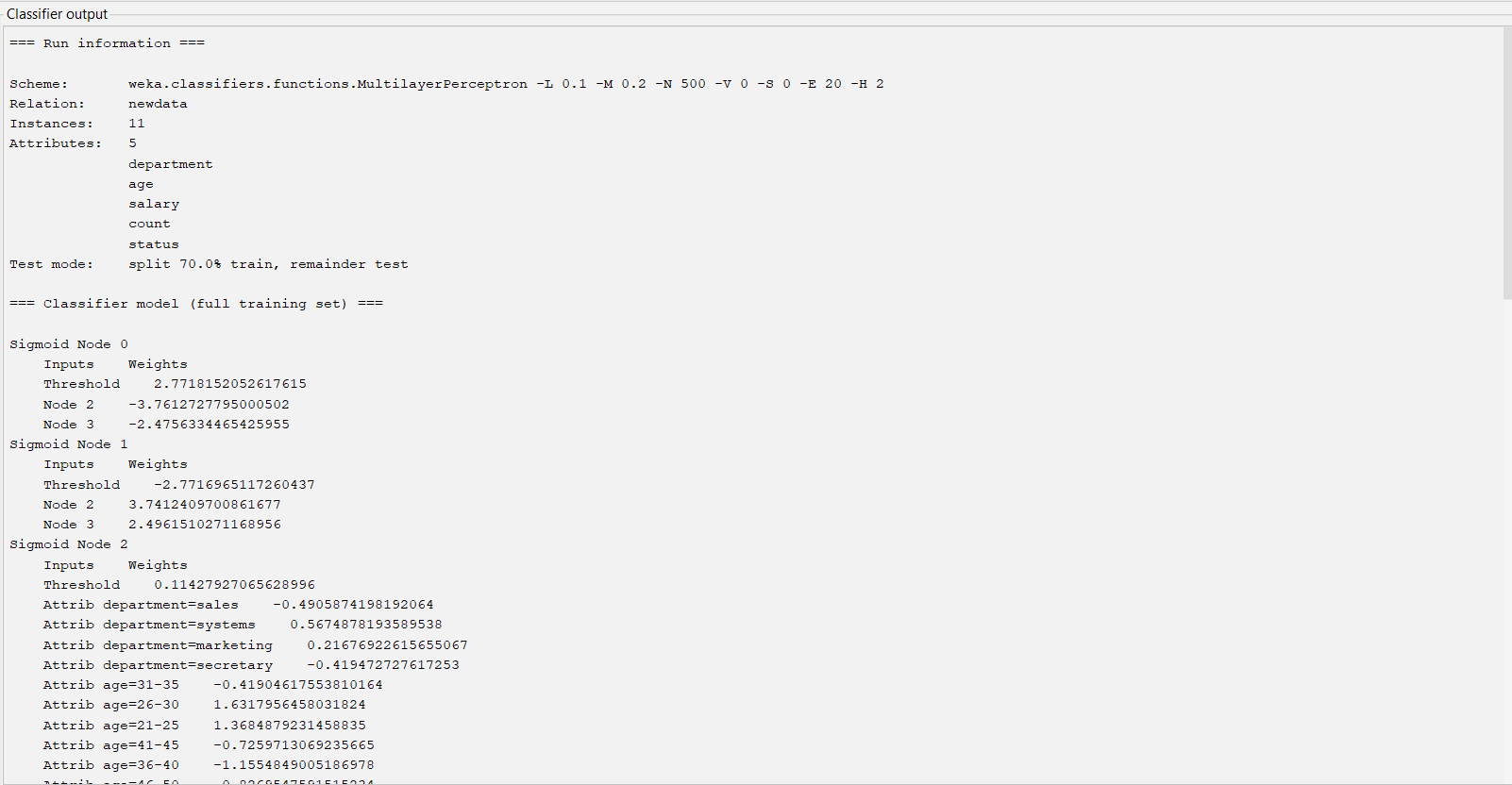


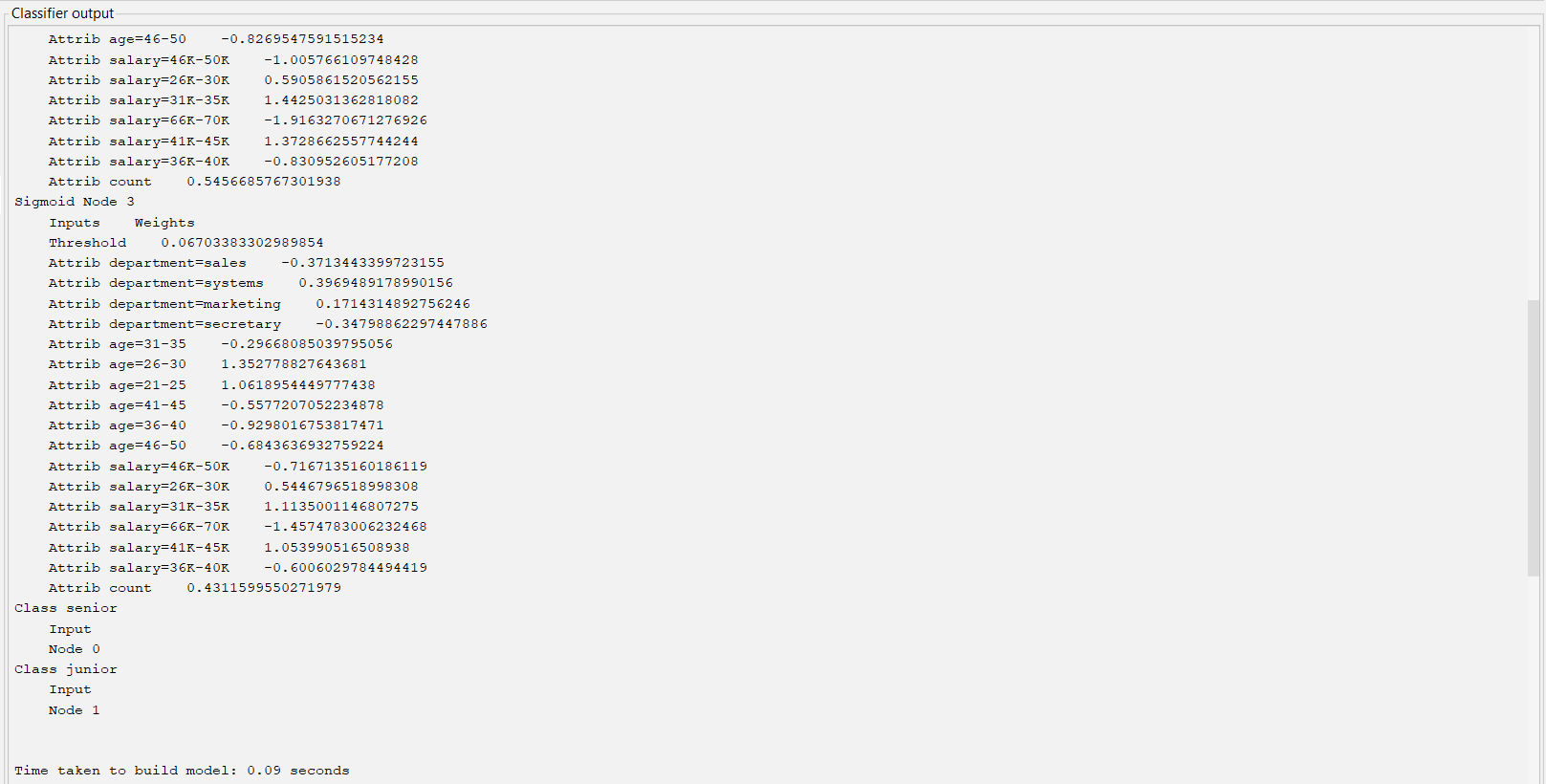
**Using 70-30 split.**

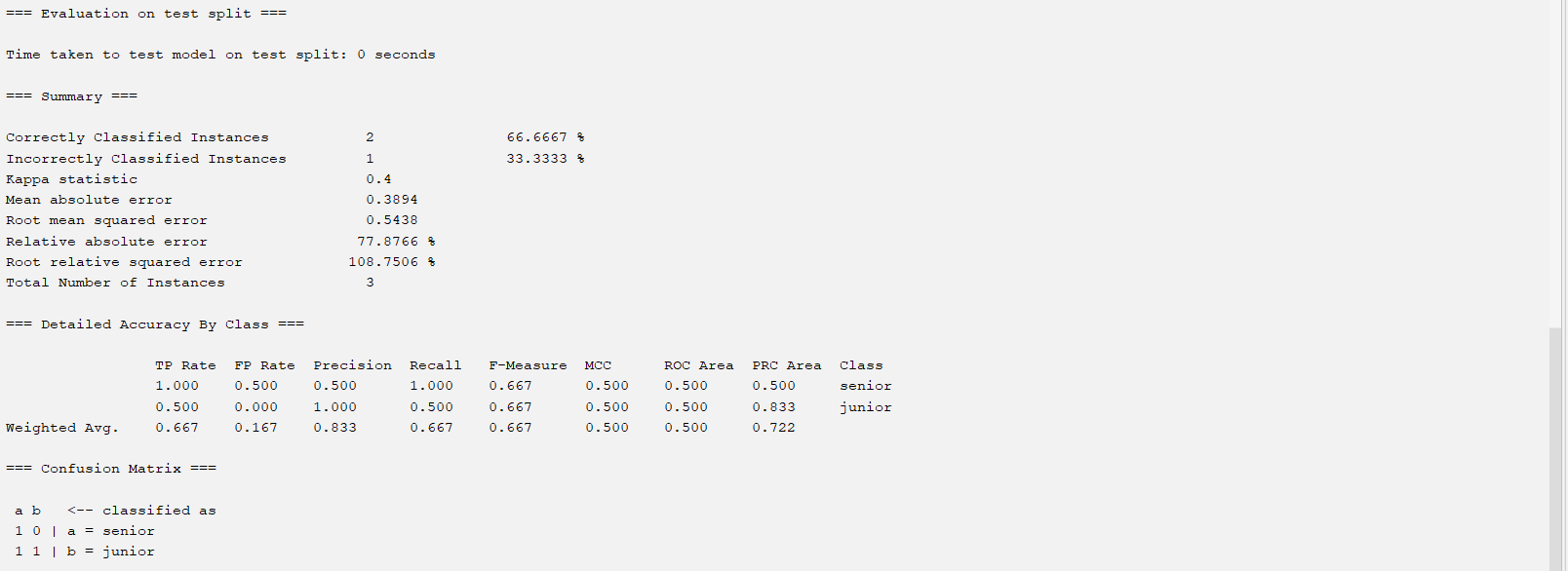


**Multilayer** **feed-forward neural network**

Weights are:



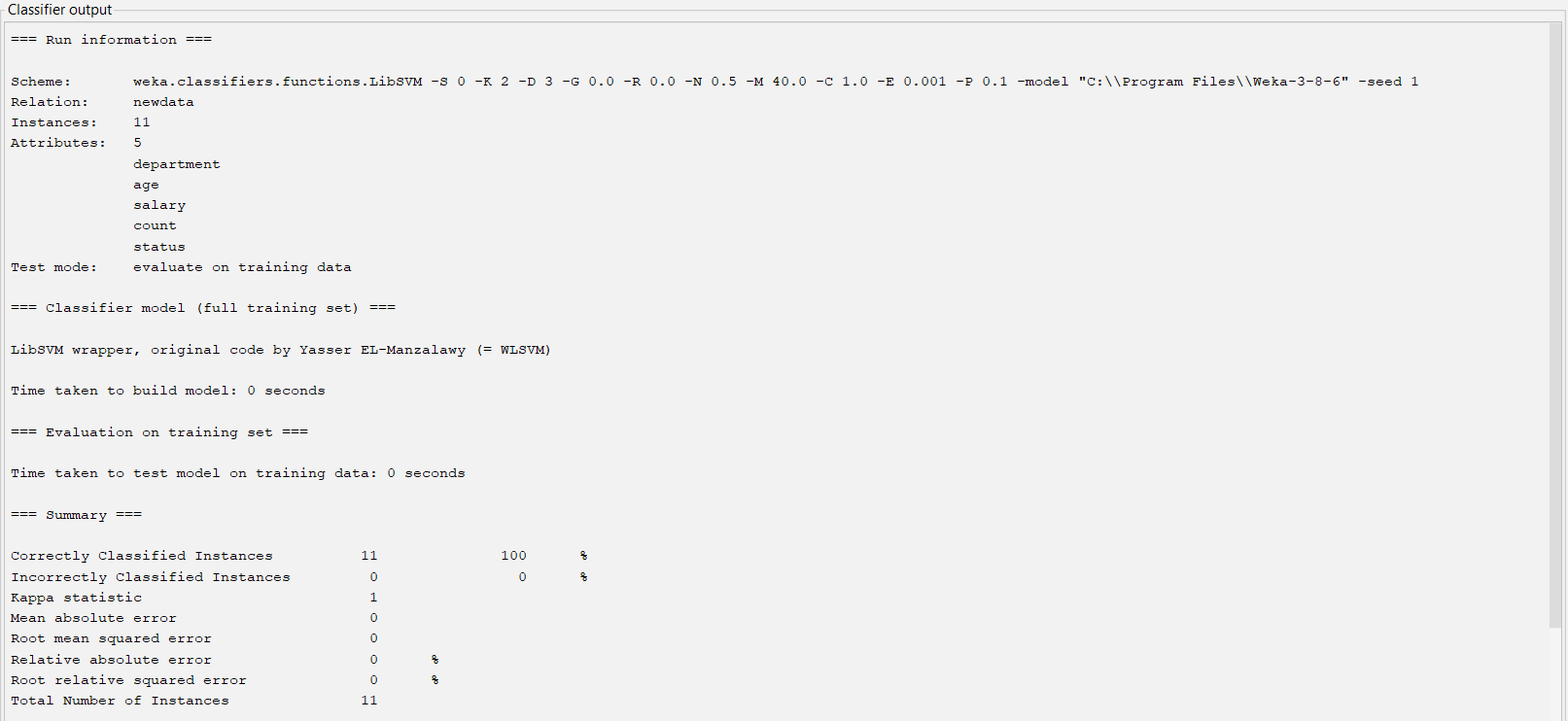


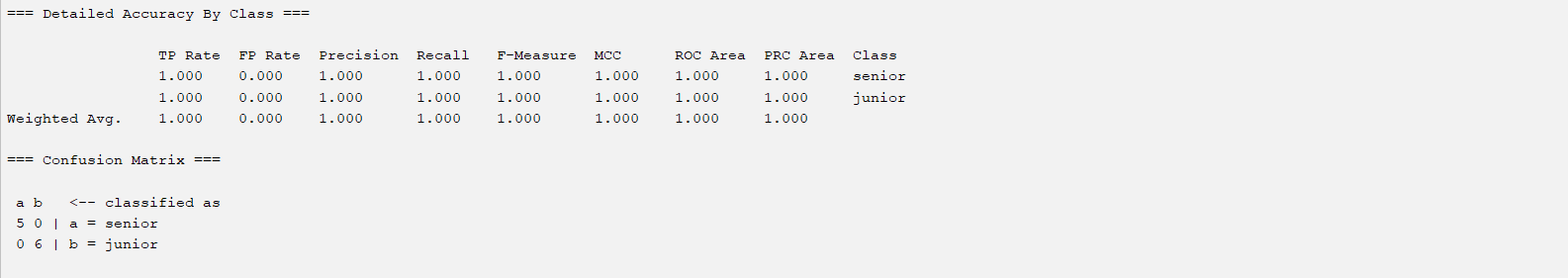


**SVM**

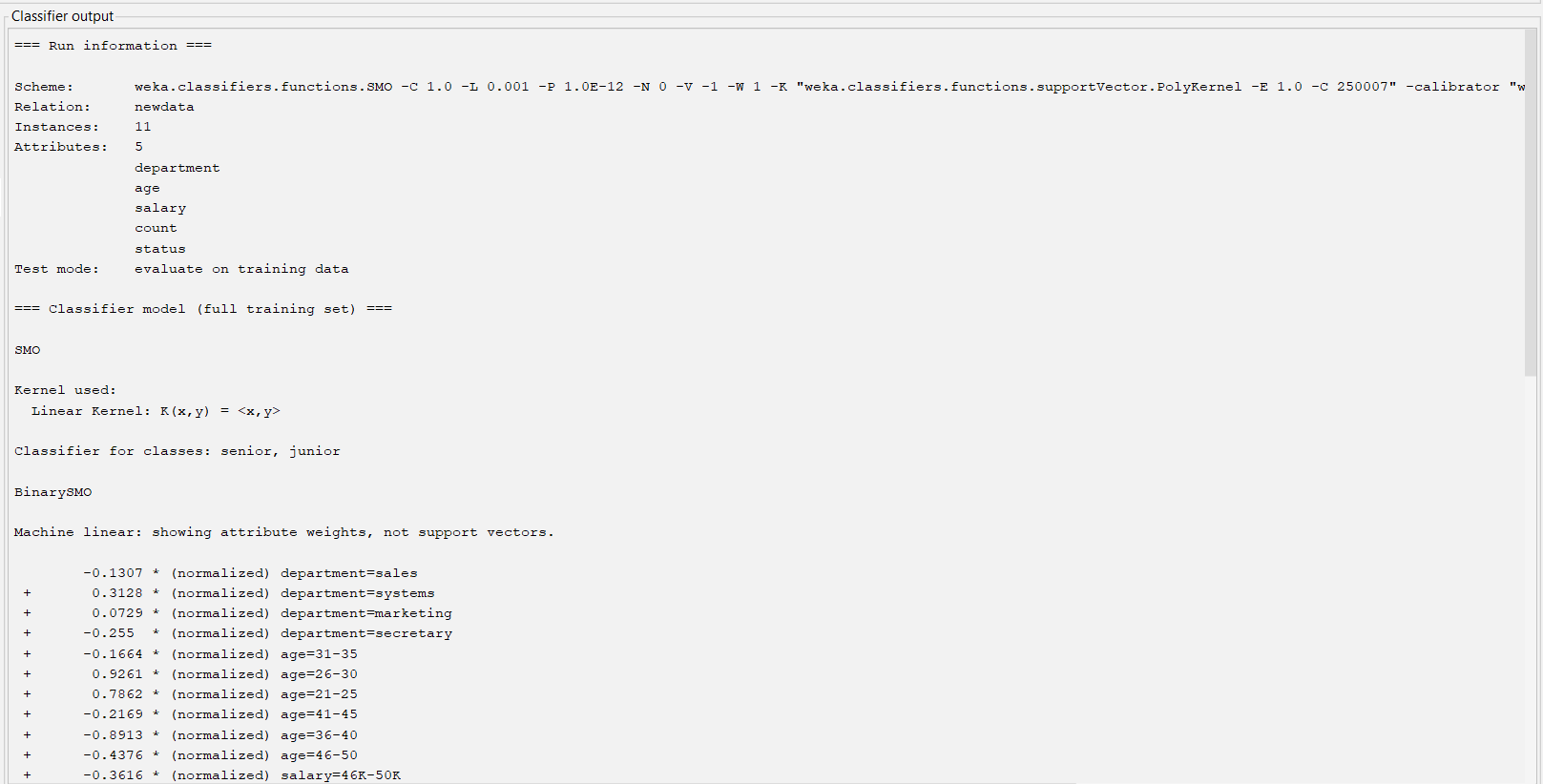
**Using training dataset**.

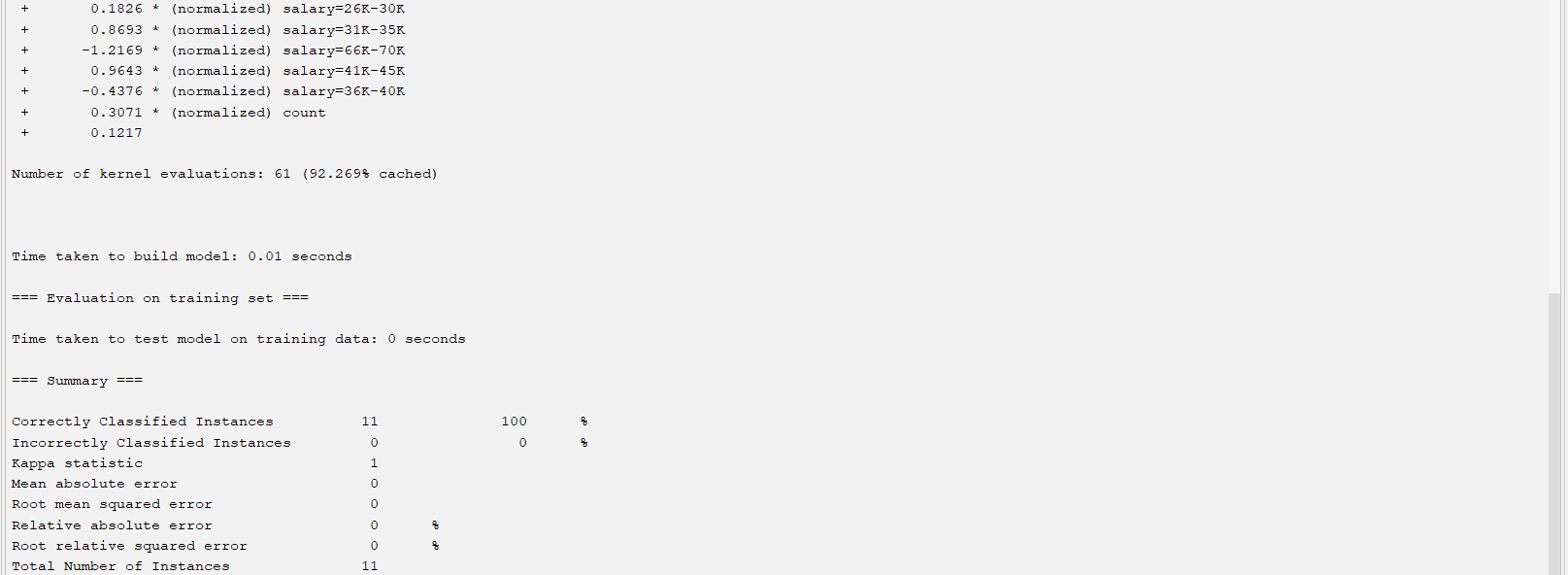
* **Lib SVM**

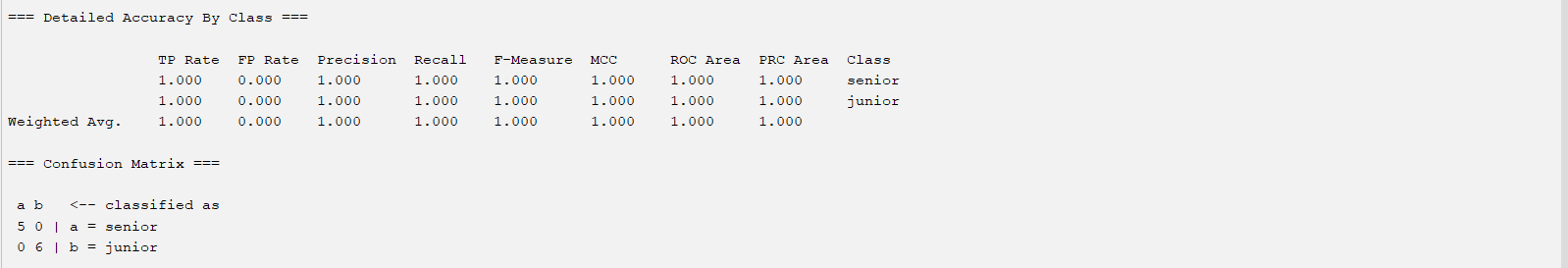




* **SMO**

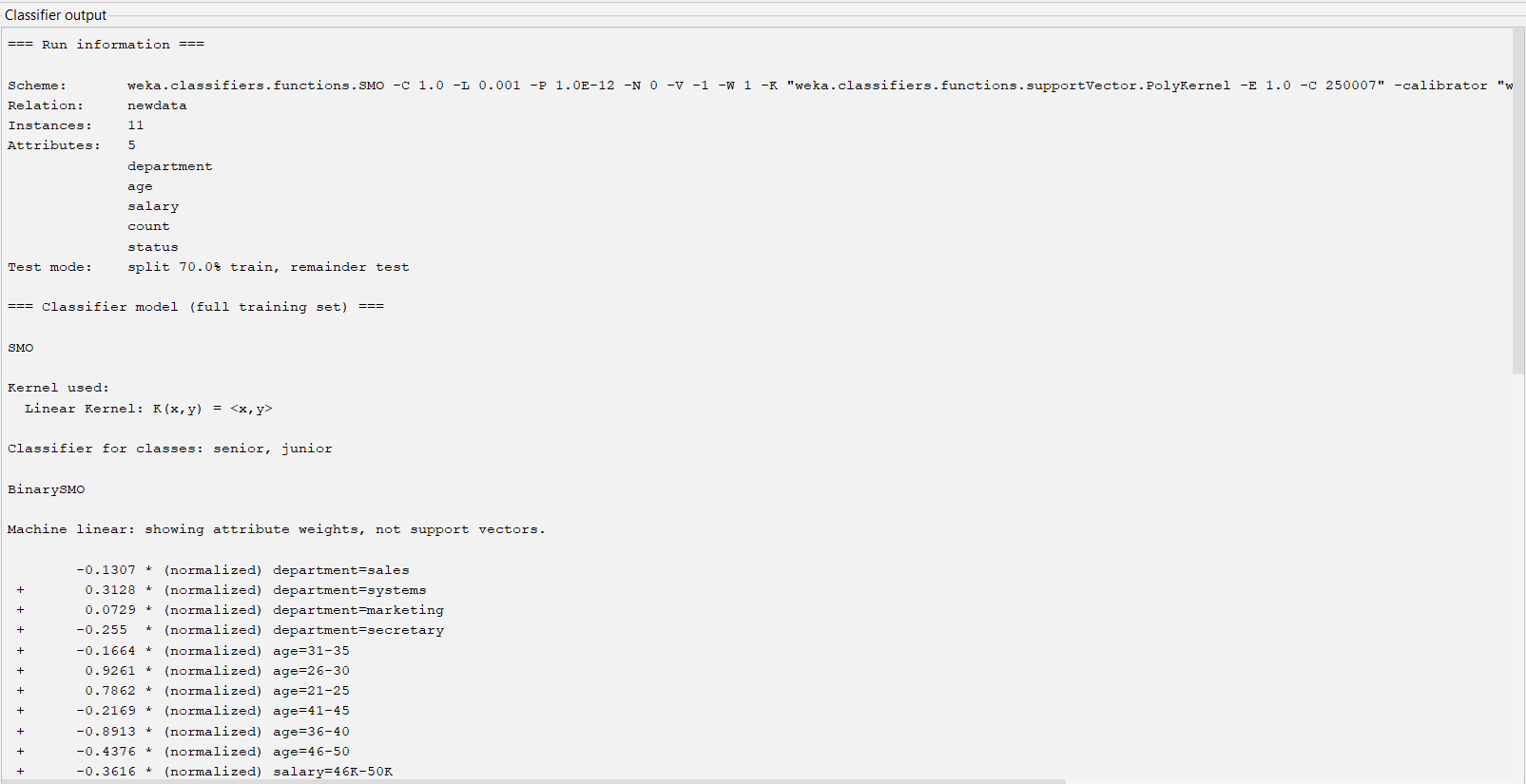


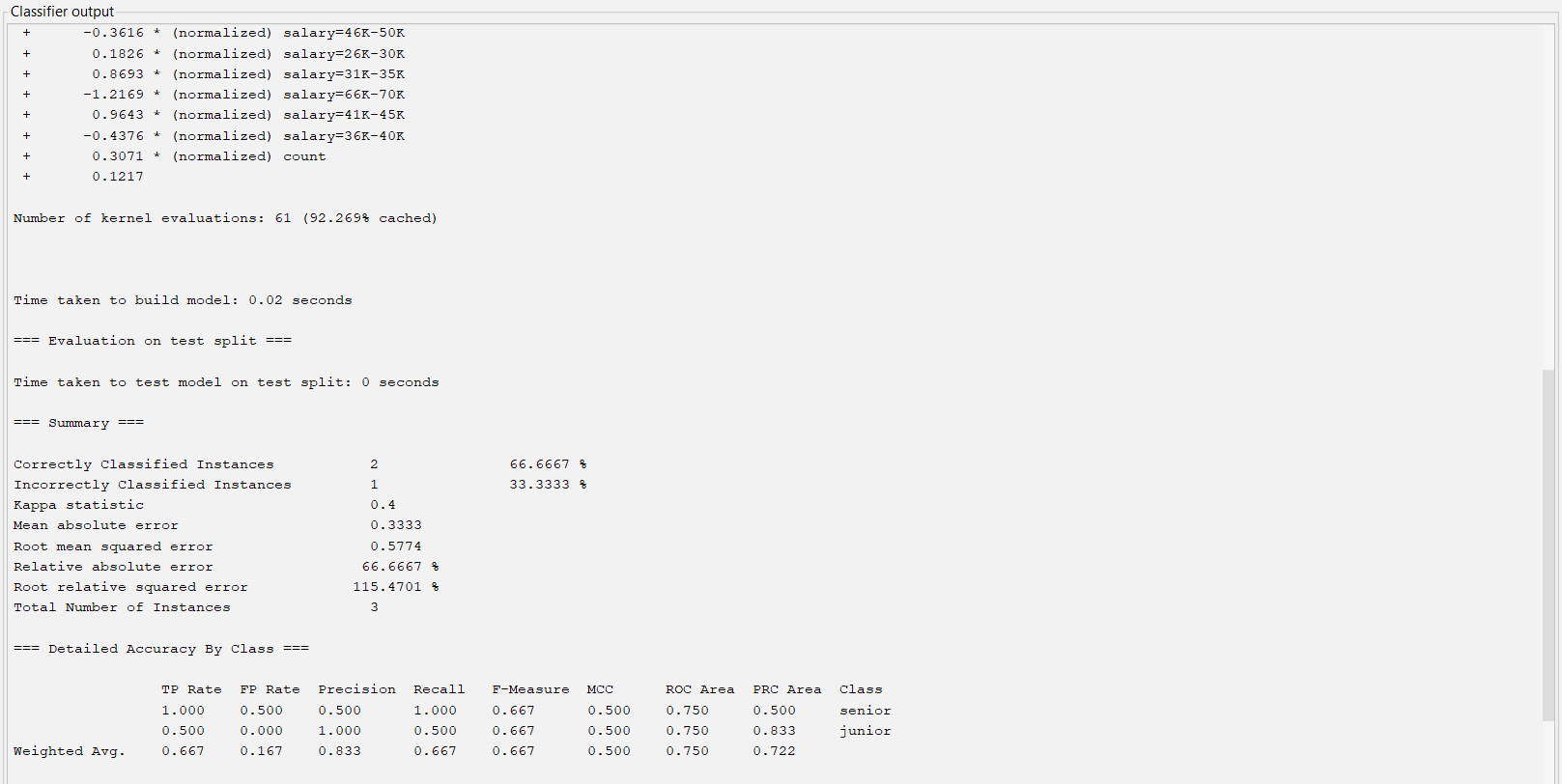


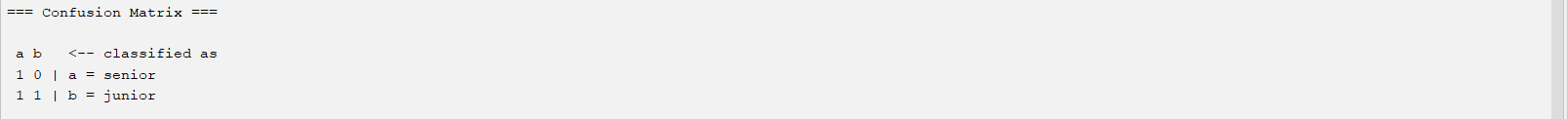


**Using 70-30 split.**

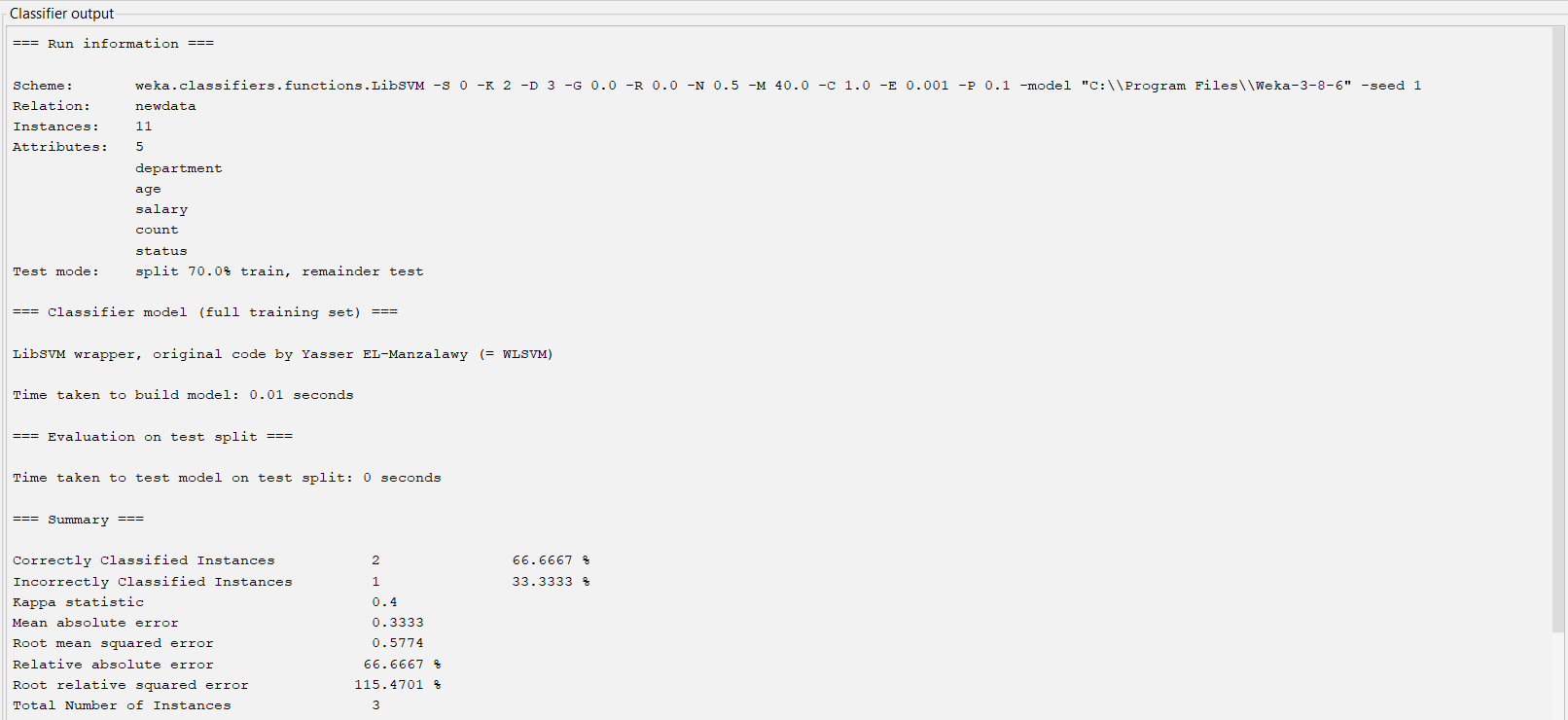
* **SMO**

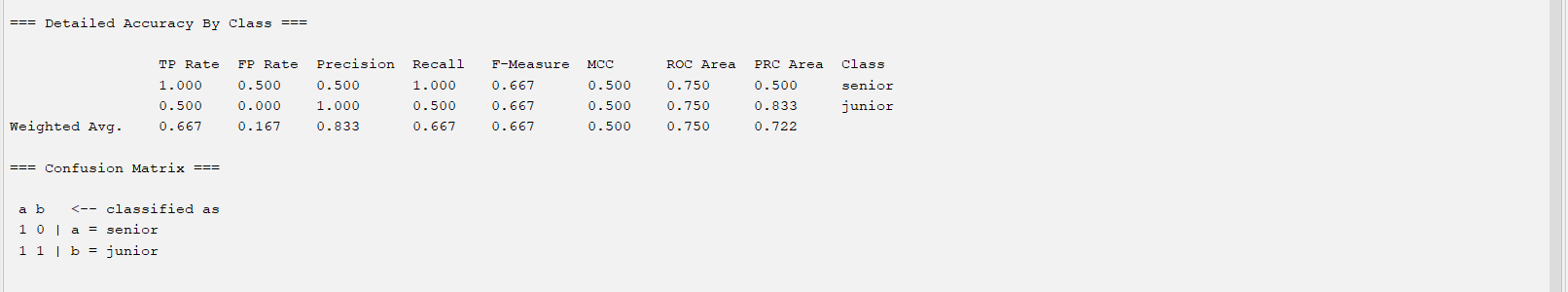






* **Lib SVM**

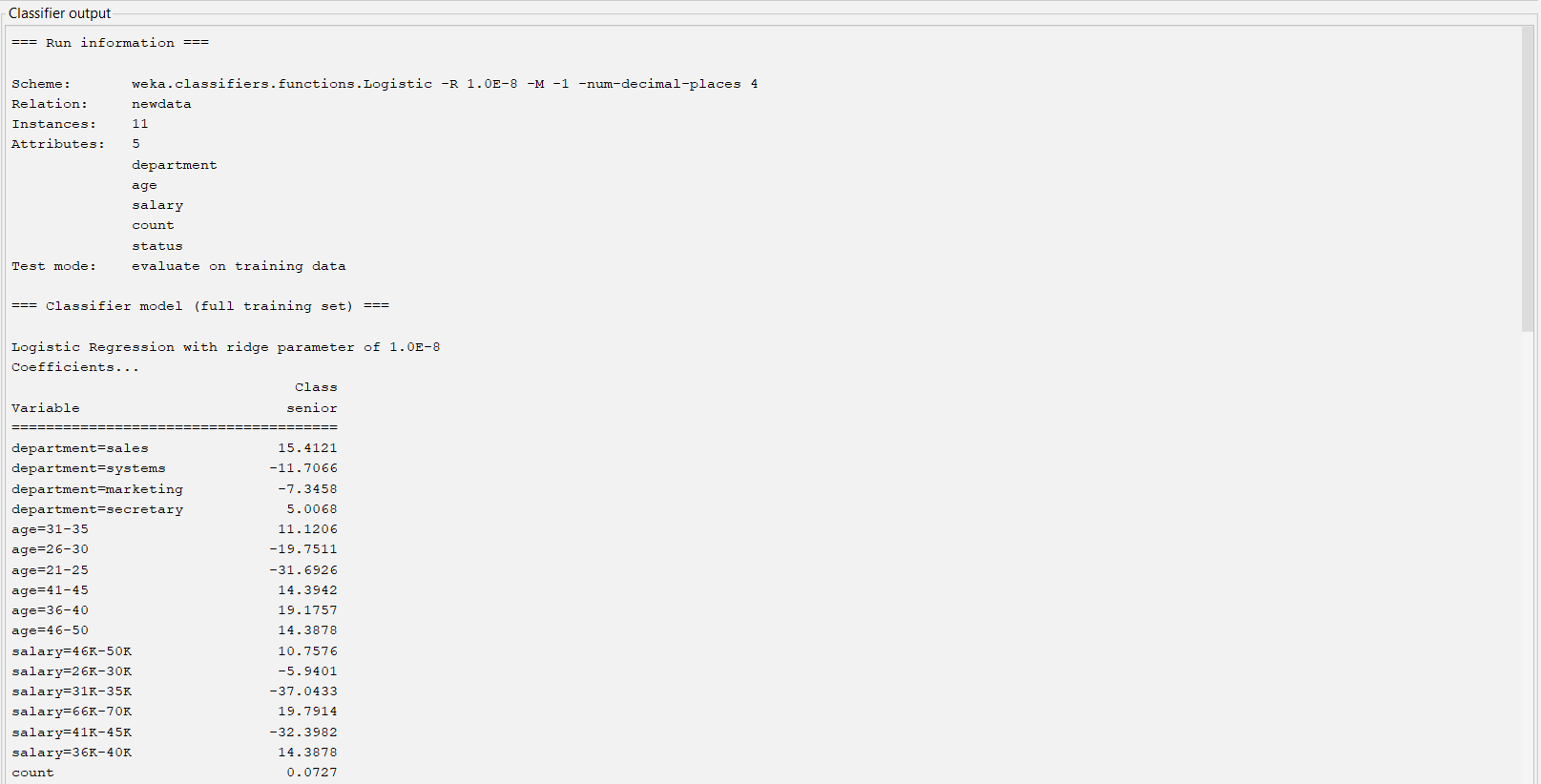




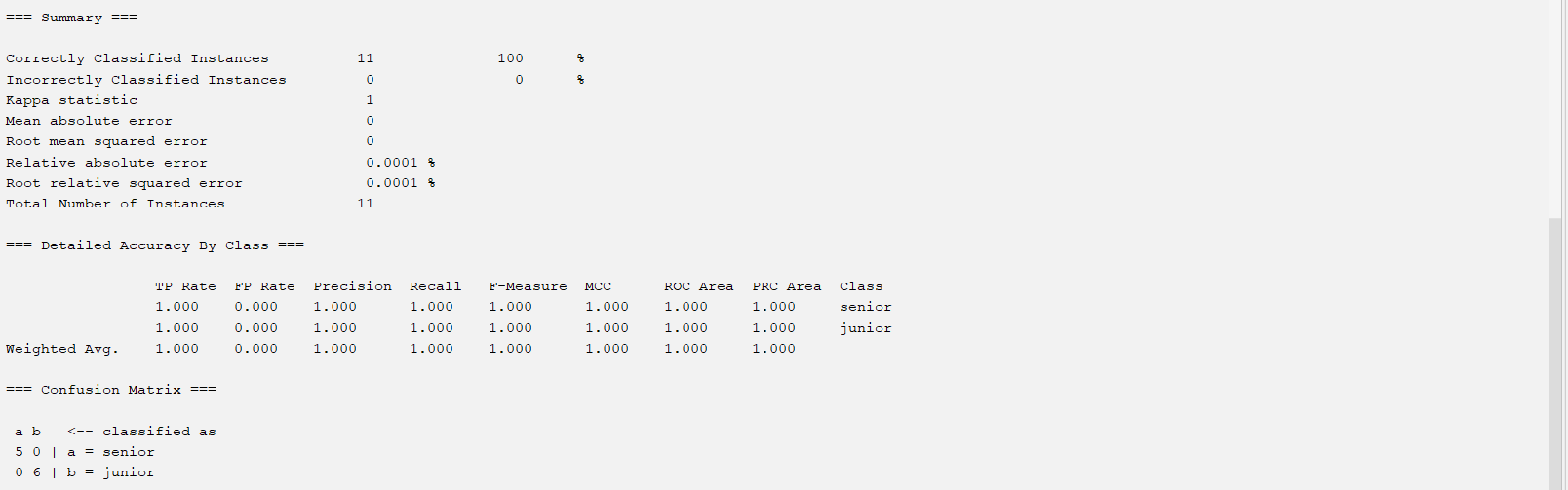
I will be using Logistic Regression classifier.

**Logistic Regression**

**Using training dataset.**

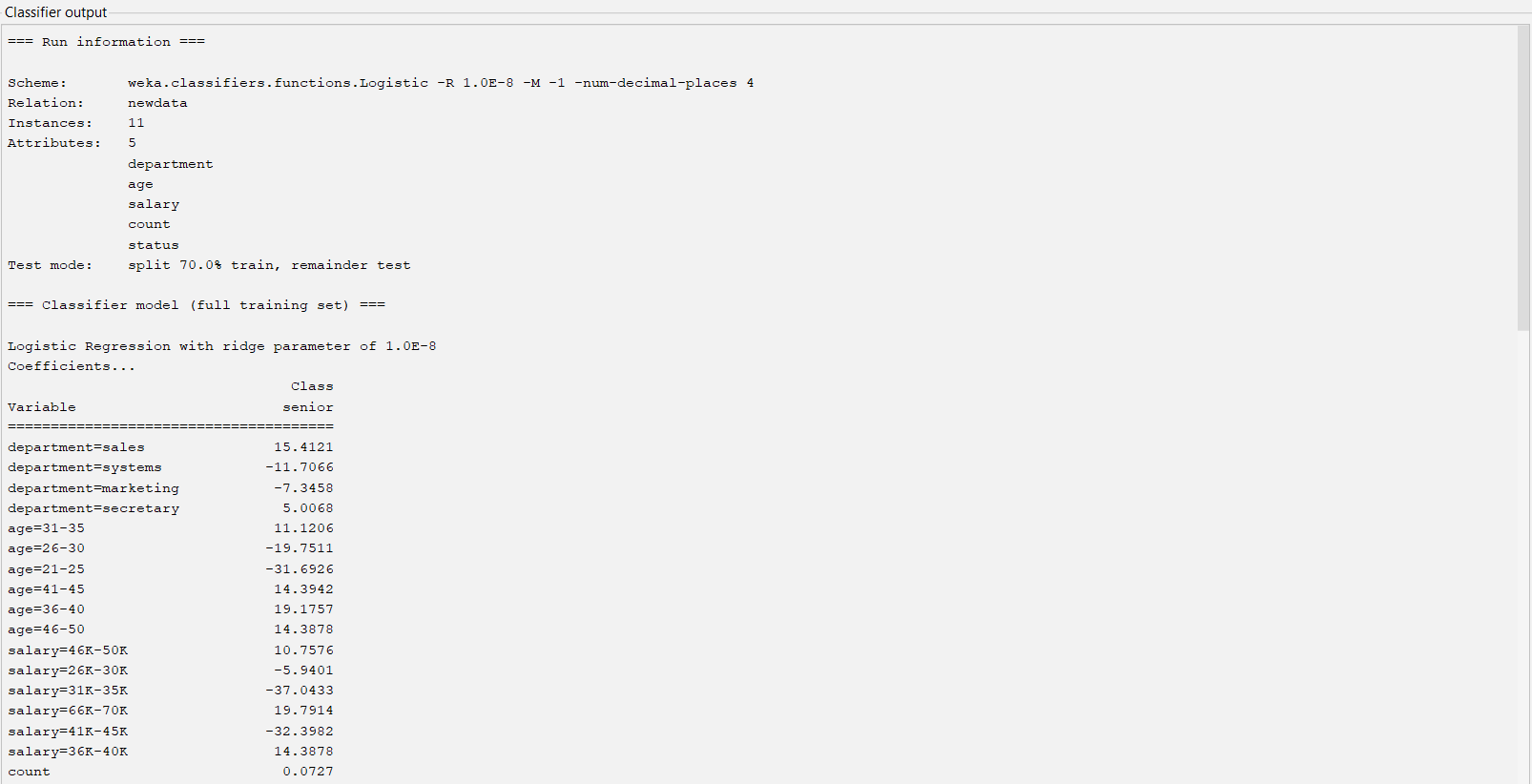


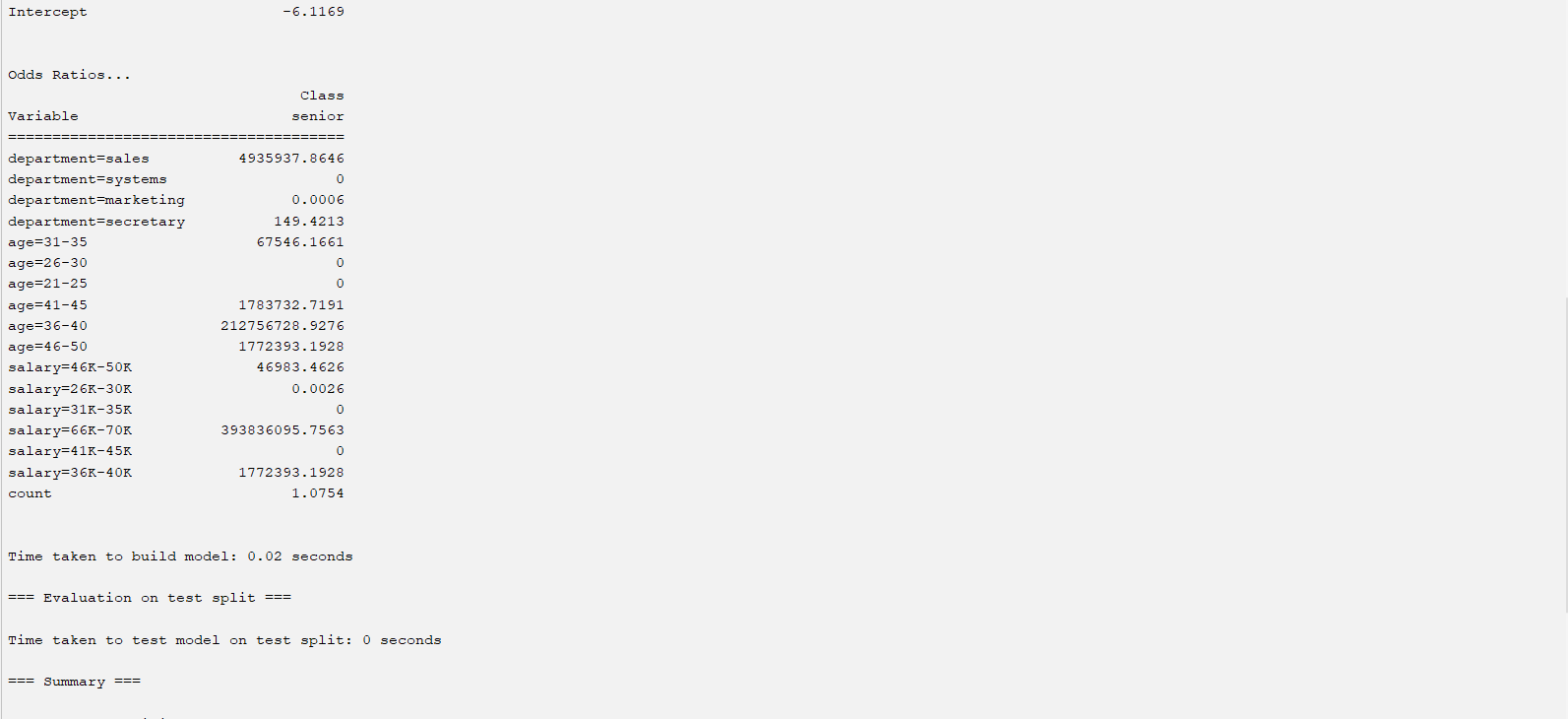


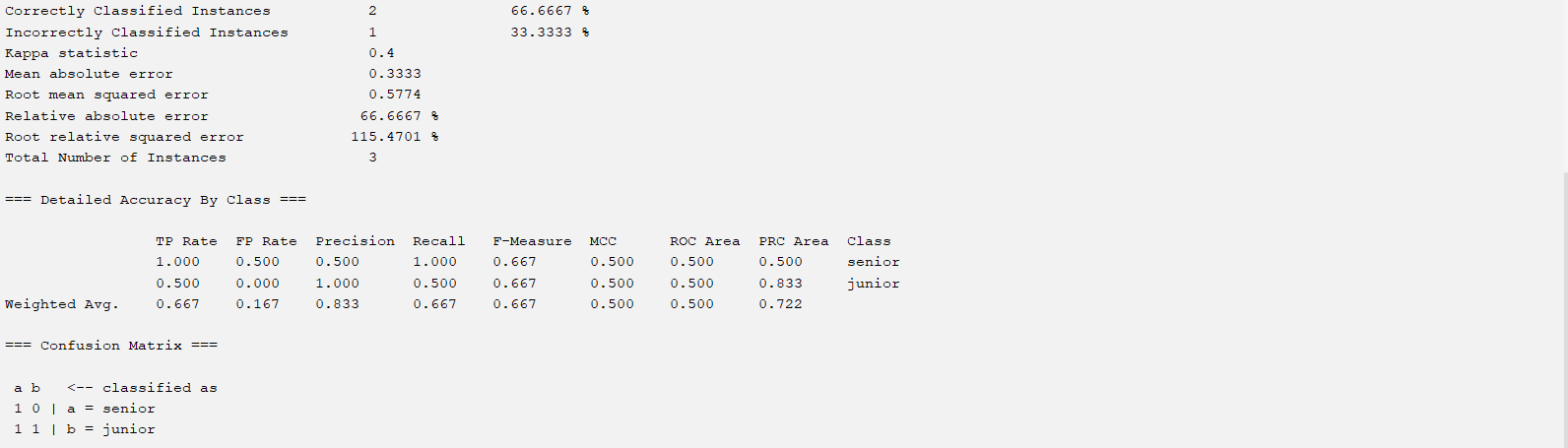


**Logistic Regression**

**Using 70-30 split**



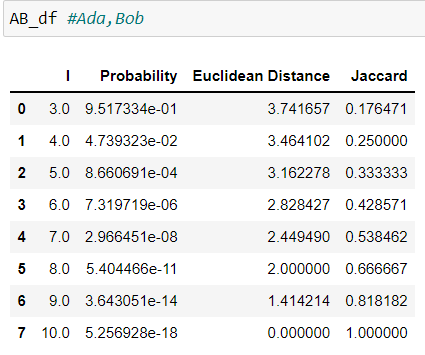
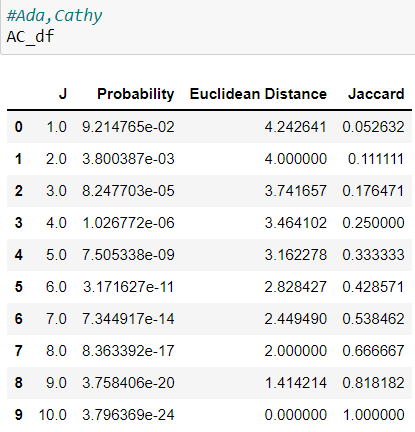




**11.2** *AllElectronics* carries 1000 products, *P*1, . . . , *P*1000. Consider customers Ada, Bob, and Cathy such that Ada and Bob purchase three products in common, *P*1,*P*2, and *P*3. For the other 997 products, Ada and Bob independently purchase seven of them randomly. Cathy purchases 10 products, 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 For Ada,Cathy**

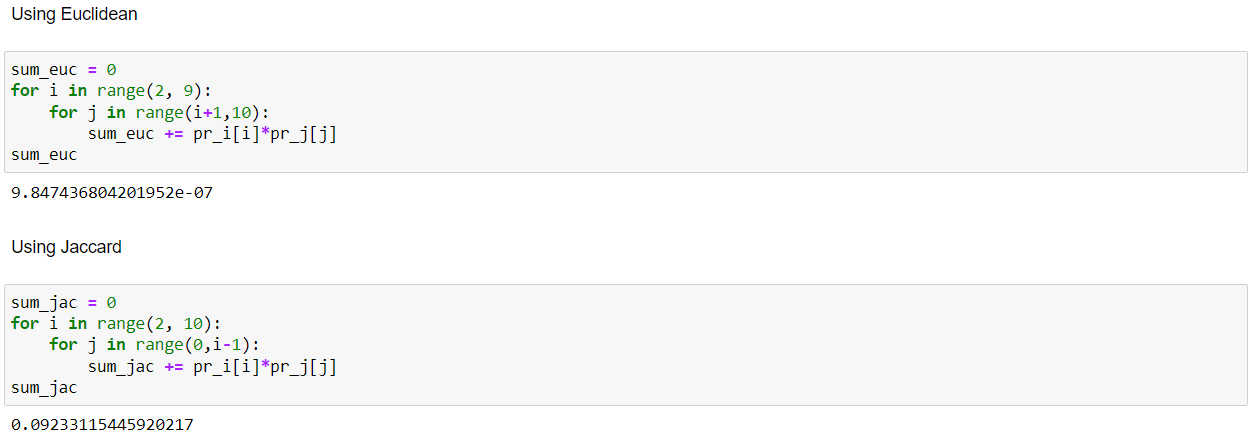
** **

**Used**

**/** for calculation probability for Ada,Bob

**/** for calculation probability for Ada,Cathy

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



Used

for Euclidean Distance. Value is 9.847436804201952e-07

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.