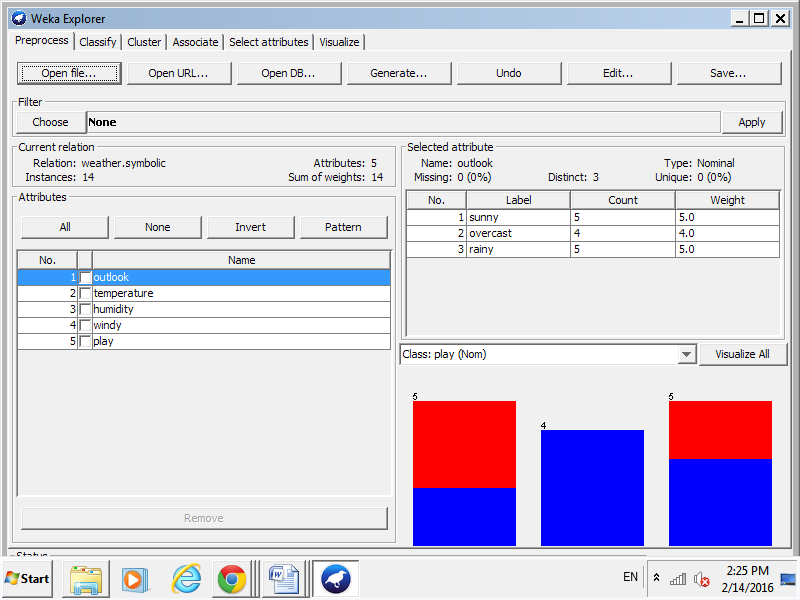
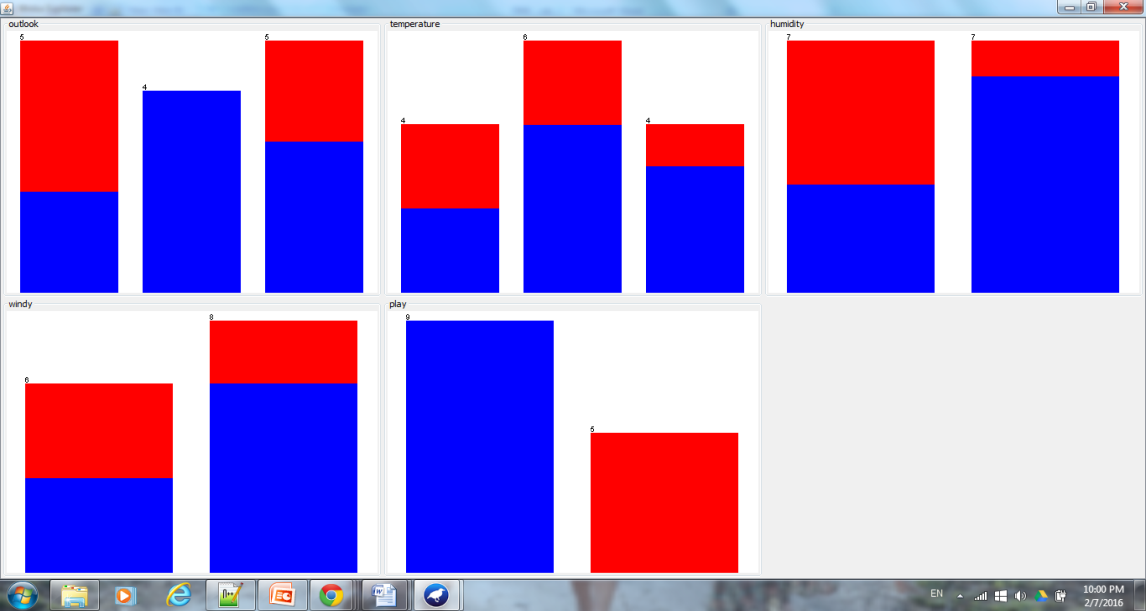
### **Data Mining I: Basic Methods and Techniques**

***Laboratory Assignment #4:***

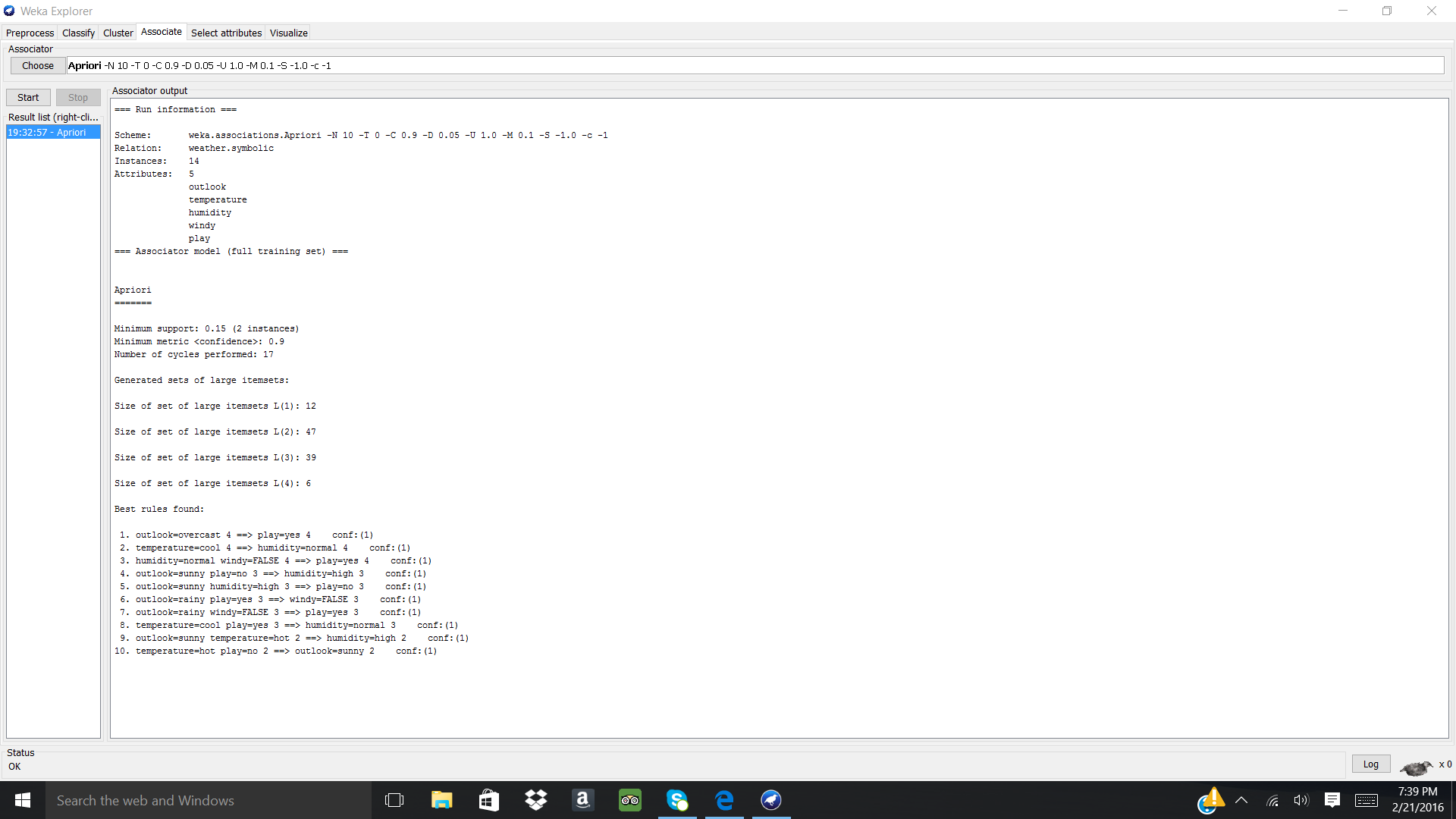
1. Use the Association rule learner APRIORI method to find the association rule in the Weather.nominal data set. How many rules did it produce? How large are the item sets? What was the largest one? What happened when you increased/decreased the confidence level? What about the number of rules? What happens when you increase the confidence parameter to 2? Why?

A. The file Weather.nominal.arff was opened in Weka. The data set contains 5 attributes, with supervised learning to be completed in order to determine the outcome of the class attribute “play”, which provides a nominal answer (yes/no) as to whether or not to play base do weather information. No further filters were necessary to apply, since all of the values are nominal. 

B. Selecting “Visualize All” in order to see the distribution of all the attributes for 14 instances.



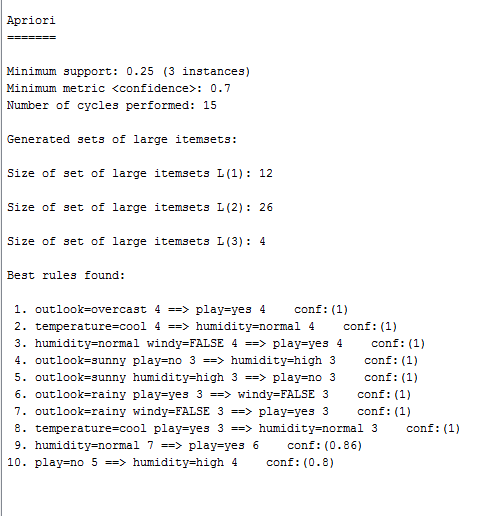
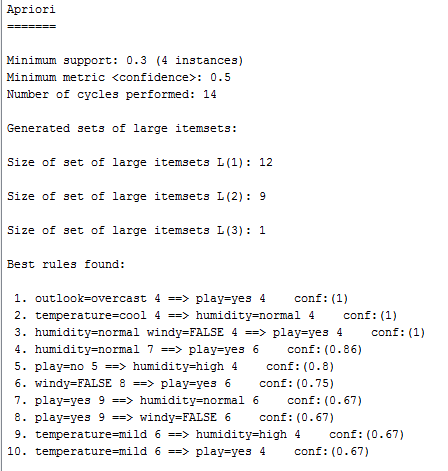
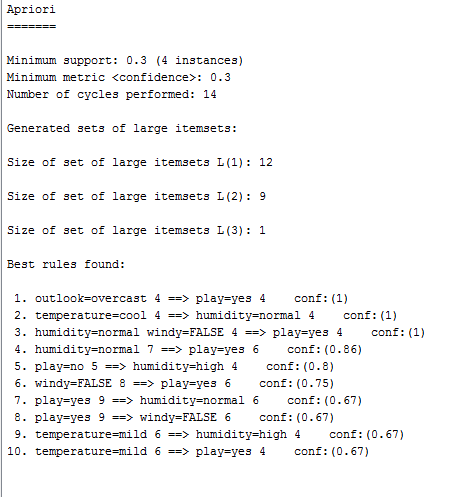
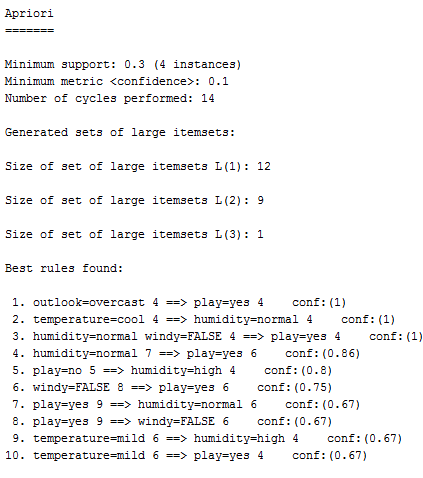
C. Association rules utilize and separate-and-conquer method, in which the algorithm scans each of the instances and creates rules using combination of attribute values with a set support and metric. Selecting Associate/APRIORI to start at default settings, resulting in the following:



Under default settings, “10” was selected as the “numRules”. The 10 rules generated, contain from 4 to 2 supports, each with a confidence of 1. The item sets range from: L(1) with 12, L(2) with 47 (which was the largest), L(3) with 39, and L(4) with 6.

Note the default setting for the “minmetric” or confidence level was 0.9. Using systematic analysis, the confidence level altered, either decreased or increased, for the following values 0.1,0.3, 0.5, 0.7, 1.0, 1.1, 1.3.

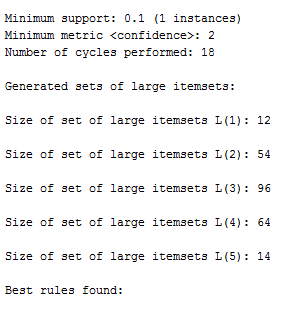
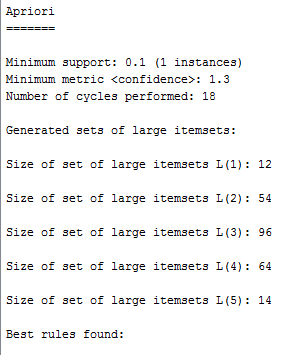
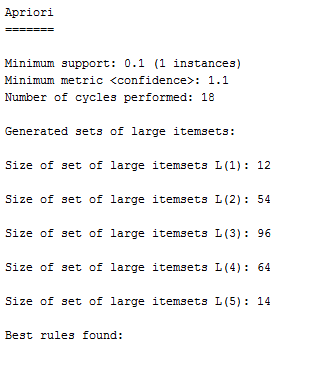
For confidence levels < 0.9:



When the confidence level < 0.9, there are fewer itemsets and the individual itemsets are smaller in size compared to that of confidence level = 0.9. Also, with increasing confidence the best rules generated increase in confidence, as expected. Increasing the confidence increases the number of rules with confidence = 1.

For confidence level = 1.0, there rules and itemsets generated are similar to those of confidence level = 0.9.

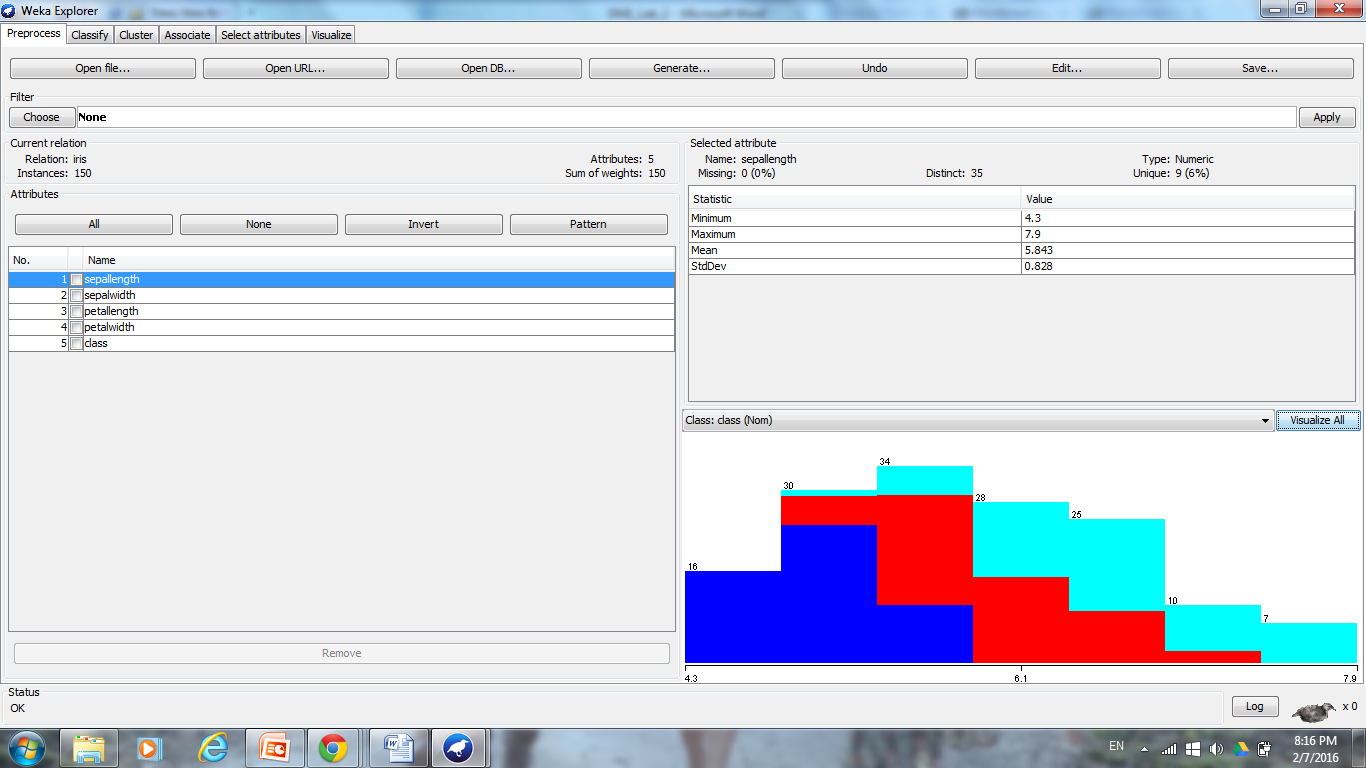
For confidence level >1.0, where confidence level has values of 1.1, 1.3 and 2.0, the following was generated:



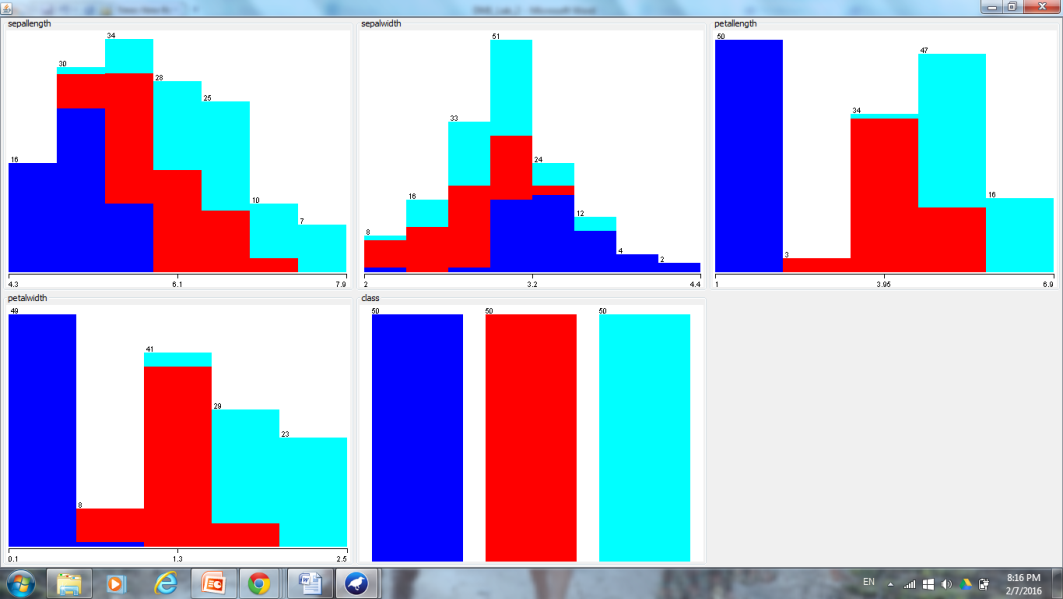
Confidence levels larger than 1.0, result in no “Best rules found”. The itemsets are identical for the above values. A confidence level > 1.0 results in an error in the algorithm thus preventing any rules to become generated for the dataset. In order to have “Best rules found”, the confidence level should be maitained at < or = 1.0 in order for the system to generate rules which are statistically significant.

1. Produce a hierarchical clustering (COBWEB) model for iris data. How many clusters did it produce? Why? Does it make sense? What did you expect? Change the acuity and cutoff parameters in order to produce a model that clusters major Iris types together. Use the classes to cluster evaluation. Examine your findings/understanding of the produced results.

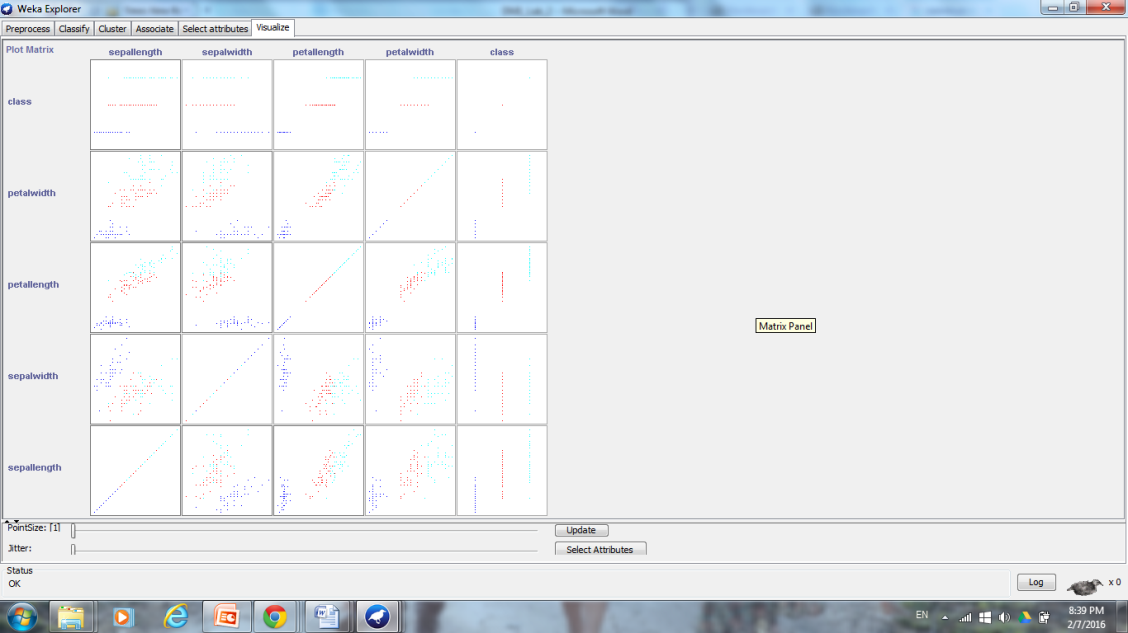
The iris.arff was uploaded into Weka, there was no need to further filter the data during the “Preprocess” phase. Note: 5 attributes exist in this data set.



B. “Visualize all” was used to visual the distributions for each class. Where, attributes “sepalength” and “sepalwidth” follow positive skewed Gaussian distributions, the attributes “petallength” and “petalwidth” follow no concrete distribution, and the “class” attribute distributions are equal.

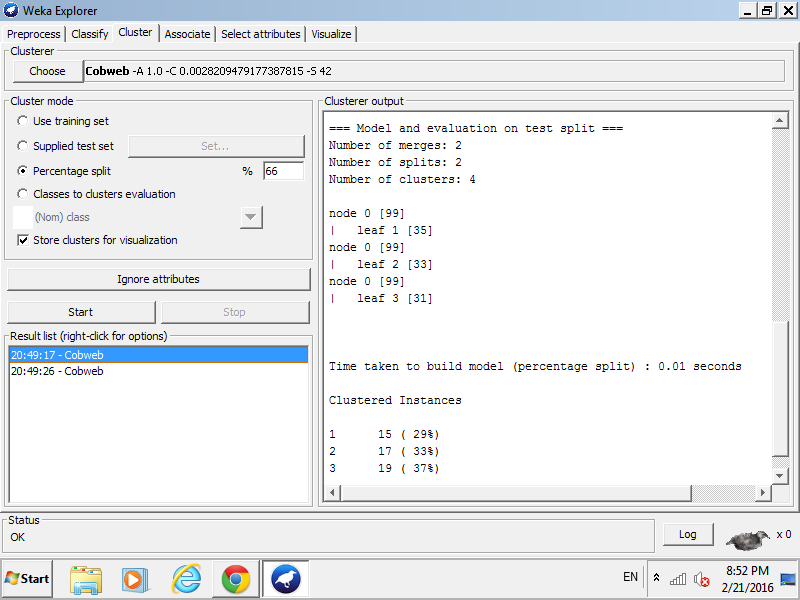


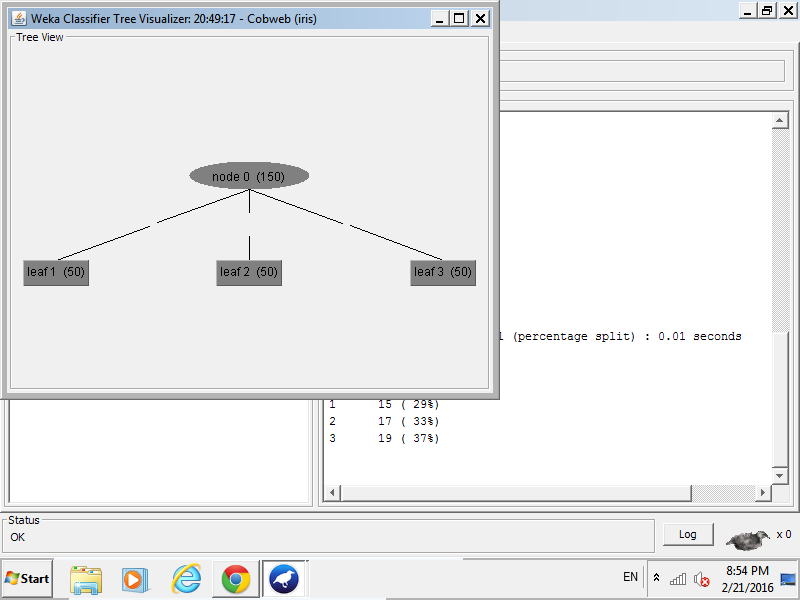
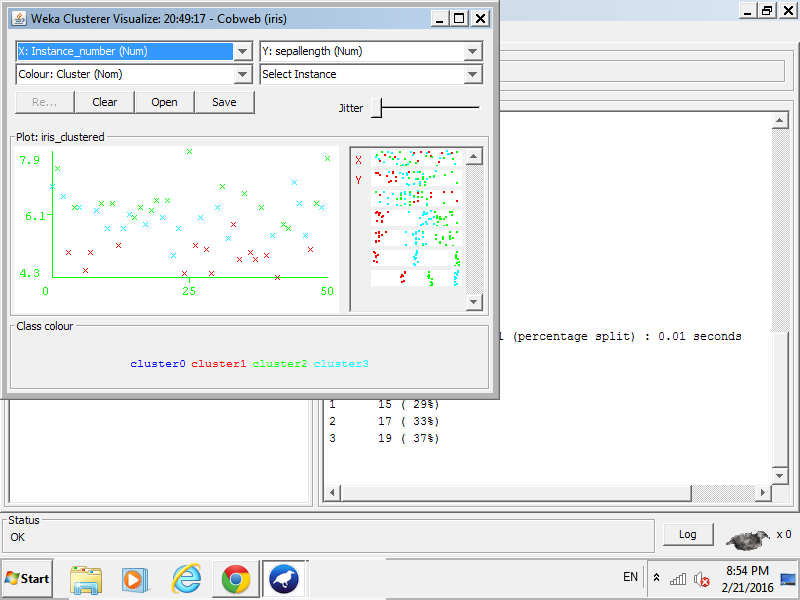
C. Utilizing the “Visualize” tab, clusters for attribute values can be seen with the most apparent clusters seen for petalwidth vs. sepalwidth/petallength and petallength vs. sepallength/sepalwidth/petalwidth. These attributes will be utilized in order to determine the class (Iris-setosa, Iris-versicolor, or Iris-virginica).



D. Under Cluster/COBWEB, the “Percentage split” at 66% was used to test the data set. COBWEB, utilizes incremental clustering on nominal values. All of the values in the iris data set are nominal. The analysis was run first with the “class” attribute as well as ignoring the attribute “class”:

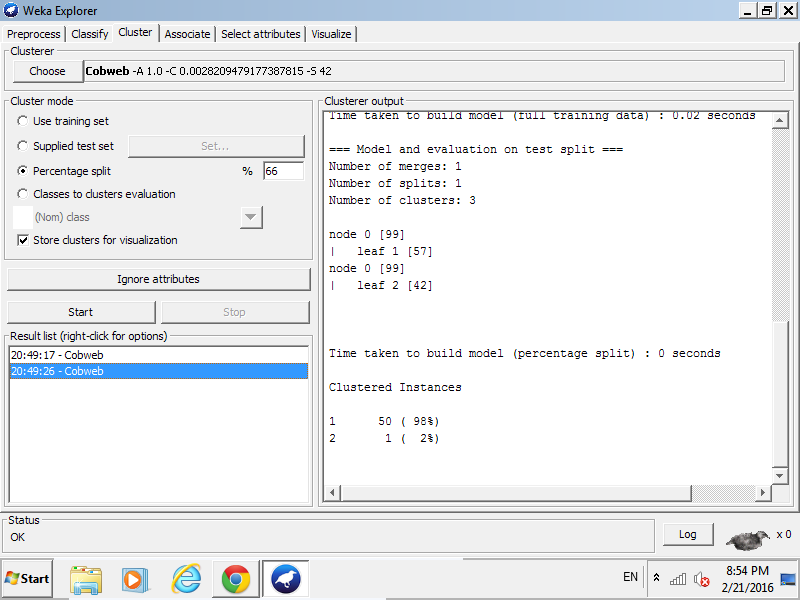
With the “class” attribute:

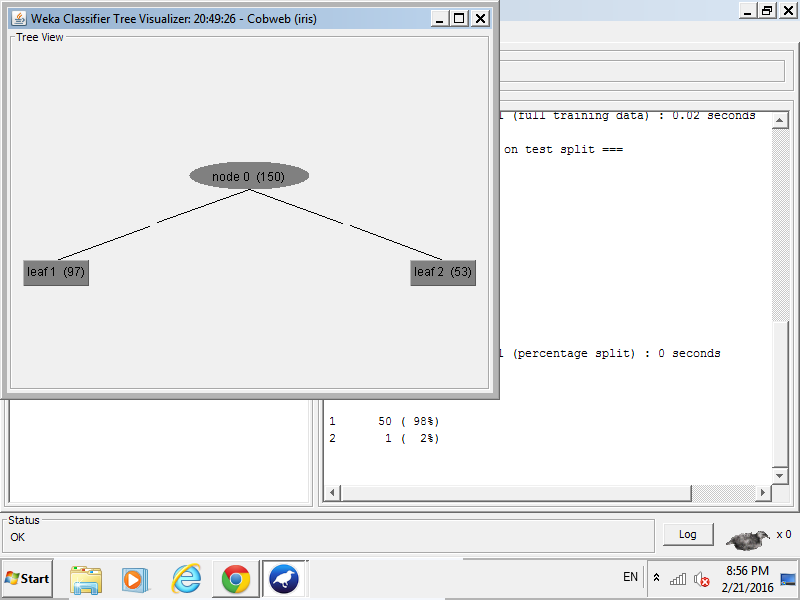
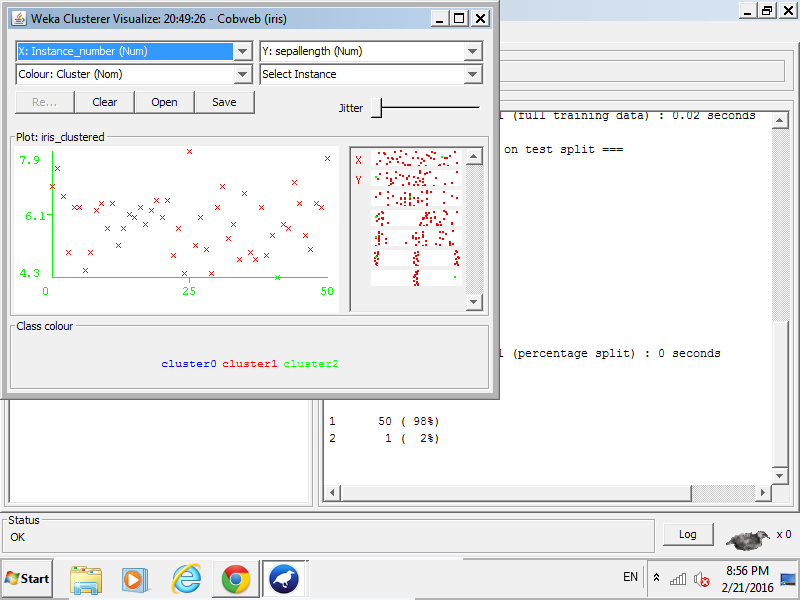




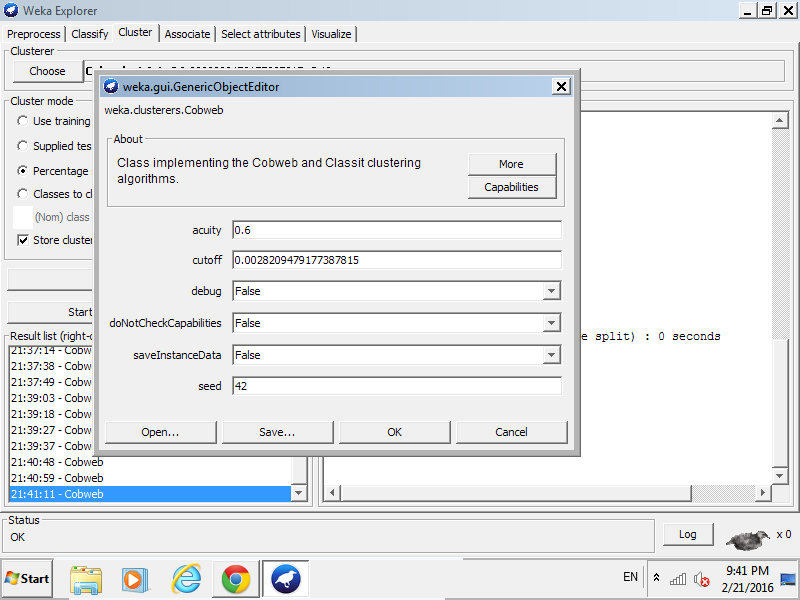
There are three leafs/clusters produced, which makes sense since the class attribute has 3 different values (Iris-setosa, Iris-versicolor, or Iris-virginica at 50:50:50).

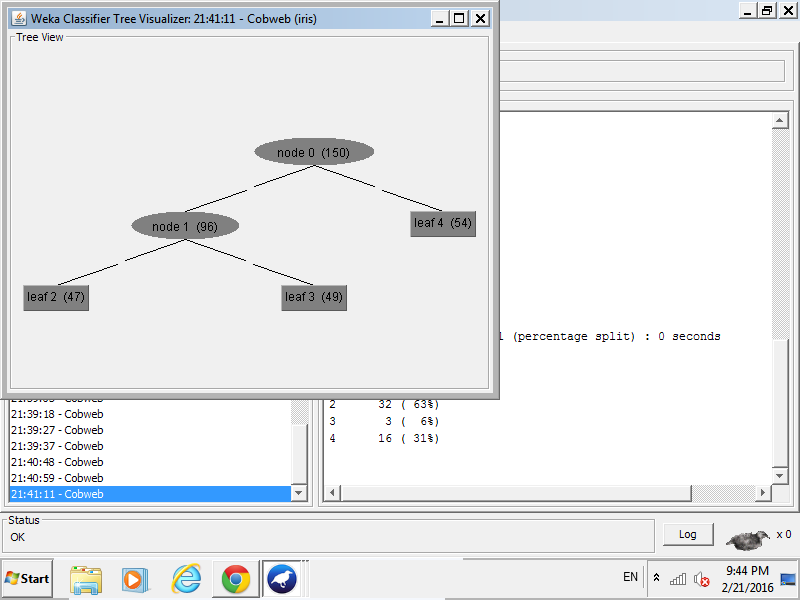
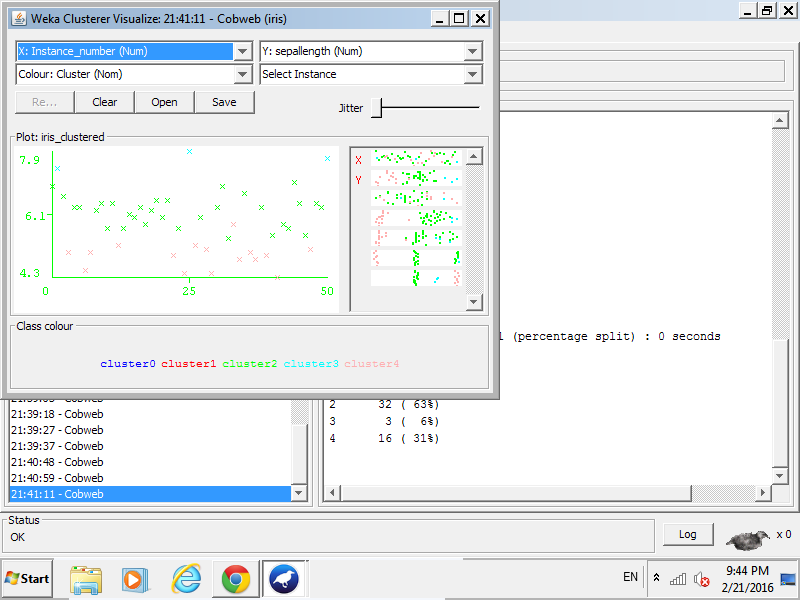
Ignoring the “class” attribute:





When ignoring the class attribute, only 2 cluster/leafs are generated with a 97:53 ratio which does not clearly cluster the data as well as the clustering above.

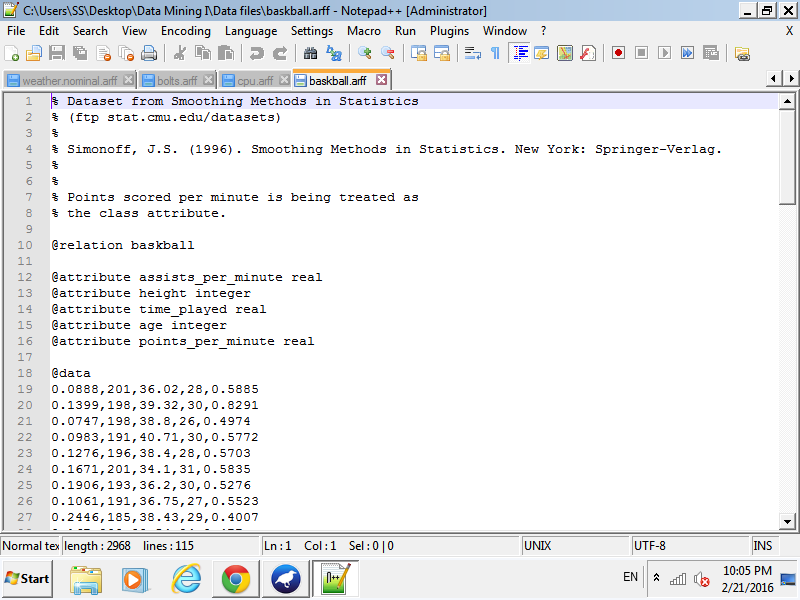
Ignoring the class, further changes to the acuity and cutoff parameters will be made to try to cluster the Iris types together and get a result as similar as possible to when the class is not ignored. Through systematic experimentation with acuity and cutoff parameters, it was found that utilizing a acuity of 0.6 and maintaining the cutoff at the default 0.0028209479177387815, gives us the closest clustering to that of the clustering performed when class is not ignored. 



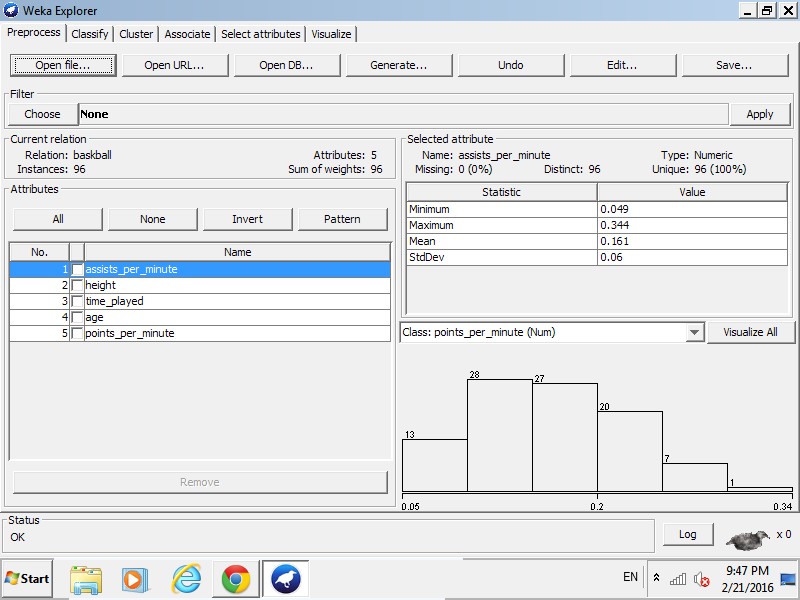
The ratio of the clusters/leaves are 47:49:54, which is the closest to the 50:50:50 ration seen previously.

1. Use the EM clustering method on either the basketball or the cloud data set. How many clusters did the algorithm decide to make? If you change from “Use Training set “ to “Percentage evaluation split – 66% train and 33% test” - how does the evaluation change? Examine your findings/understanding of the produced results.

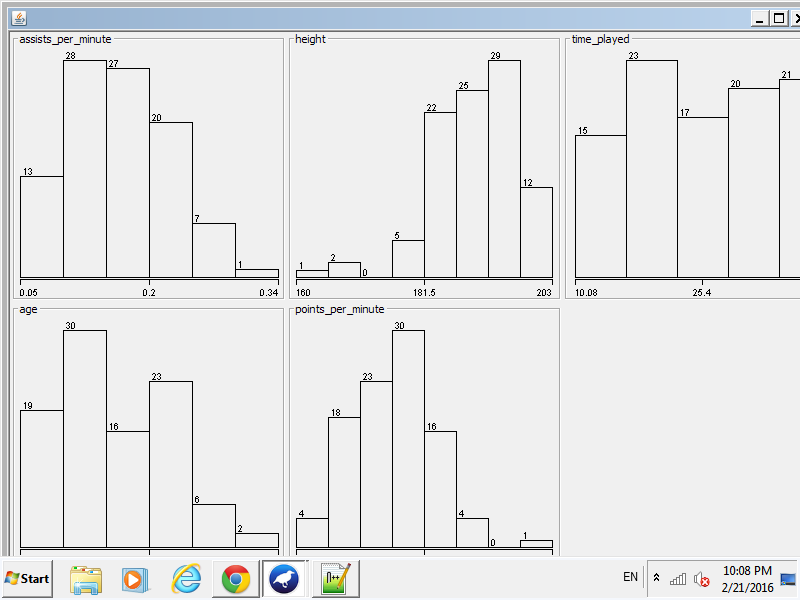
A. The basketball.arff was opened in Notepad, to understand the raw data. The attribute values are either real or integer values and the points scored per minutes is considered as the class attribute.



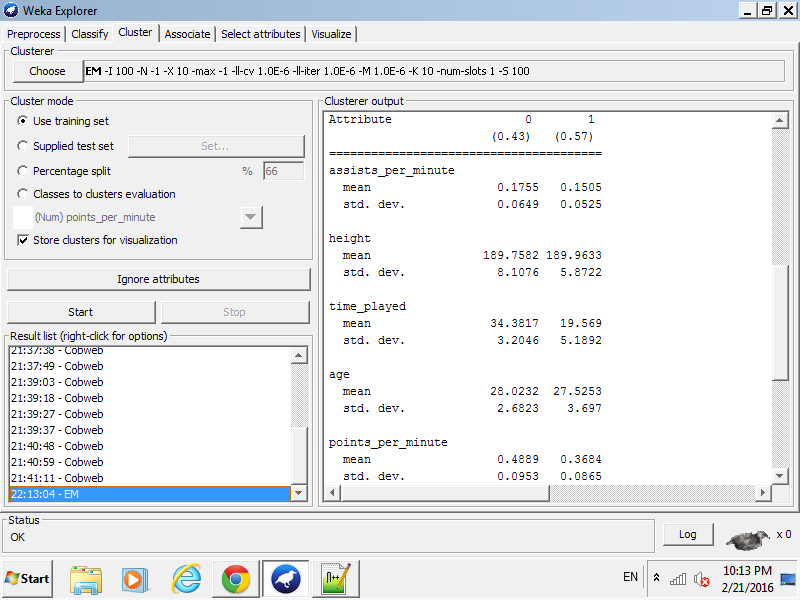
B. The basketball.arff was opened in Weka. No further filters were made during the “Preprocess phase”.

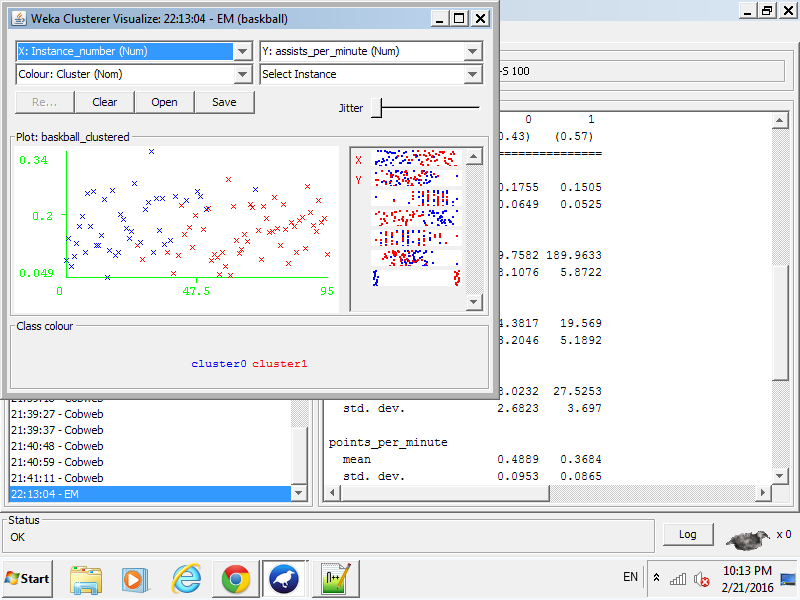


C. Selecting “Visualize all” showed the attribute distributions for all of the instances. Note: the attribute “points\_per\_minute” looks like a Gaussian distribution skewed left and all of the attributes has a mean, standard deviation, etc. associated.



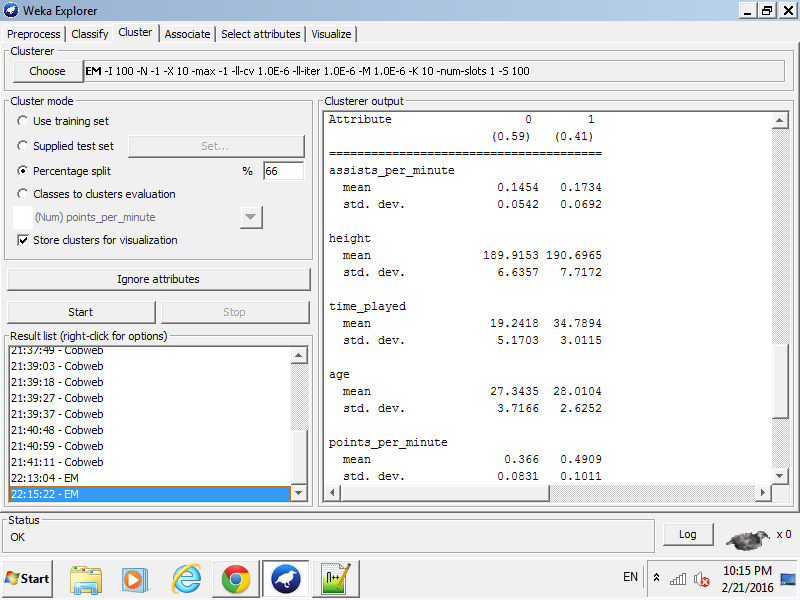
D. Under Cluster/EM, EM clustering is performed first using “Use Training set” and then using “Percentage evaluation split - 66% train and 33% test” to compare for understanding.

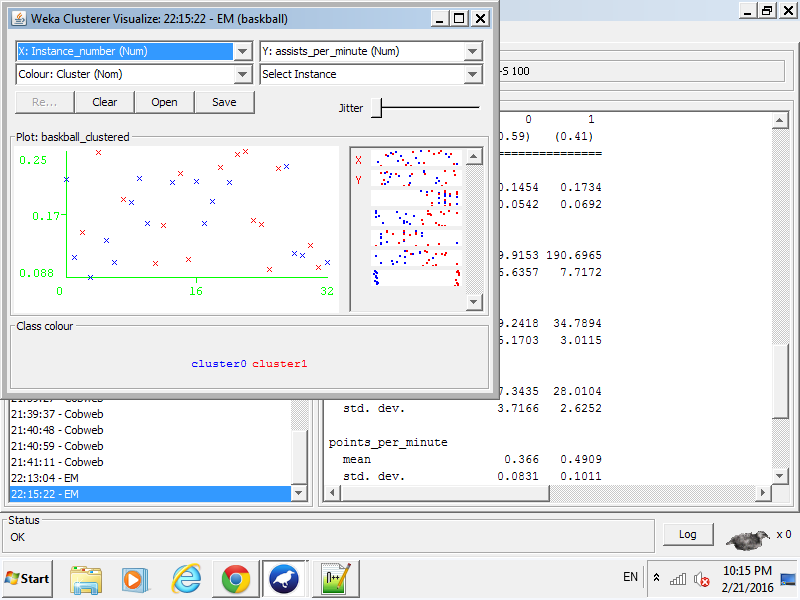
“Use training set”:  




Two clusters were created with cluster 0 44% of the instances and cluster 1 containing the remaining 56% of the instances.

“Percentage evaluation split - 66% train and 33% test”:





In this analysis, 2 clusters are also formed but with 52% of the instances belonging to cluster 0 and 48% of the instances belonging to cluster 1.

Discussion of difference between data used to create mode:

Looking back at the Notepad view of the data set, there are 95 instances to classify. For the training set, all of the 95 instances can be including in the 100 seeded while for the percentage split only 65 instances can be included in the 100 seeded and used to create the model. Differences arrive in the clusters, as viewed in the “Clusterer Visualize”, with the training set data split into right, left clusters while the percentage split data is not as clearly split up in visualization. Note: this is a small data set and although percentage split is often good to use for larger datasets as some instances can be used to train while others to test, the training data provides “neater” clusters as all of the instances are considered when created the model.