QUESTION 1:

* What is the size of training set?

The size of the total training set is 286 instances

* How many attributes exist in the training set?

The total attributes that exist in the training set is 10 that includes age, menopause, tumor size, inv-nodes, node-caps, deg-malig, breast, breast-quad, irradiat, Class.

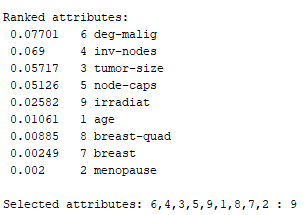
* How many instances are positive (Enjoy = yes) and how many negative?

In my case the no-recurrence-events is negative having 201 instances and recurrence-events is positive having 85 instances.

* Which attribute best separates the data?

The attribute the best separates the data is the one having the highest information gain.

In my case the attribute having highest information gain is node-caps.

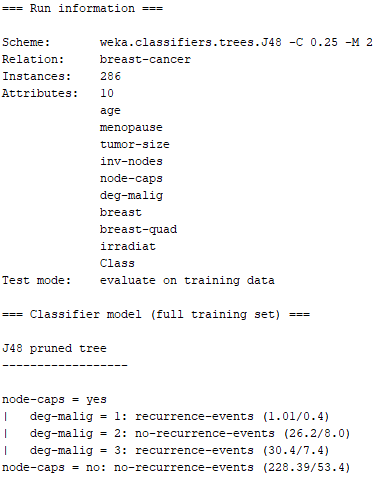


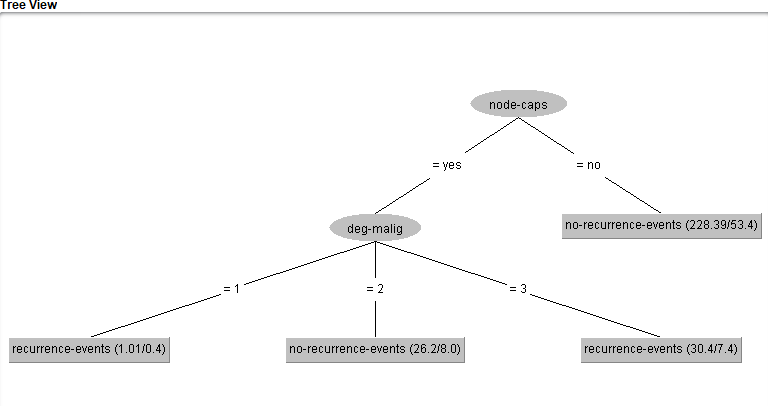
* How many elements from the data set have the node-caps attribute set as high?

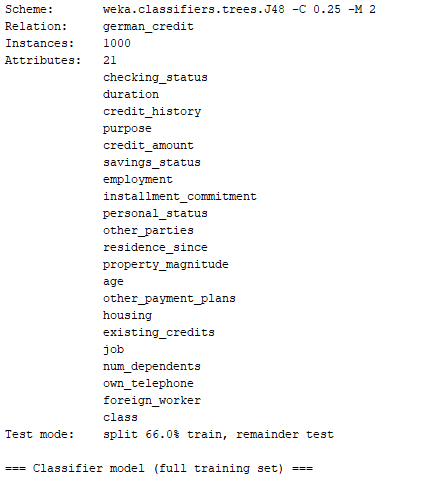
In my case I have selected node-caps attribute which has elements 56 set as high.

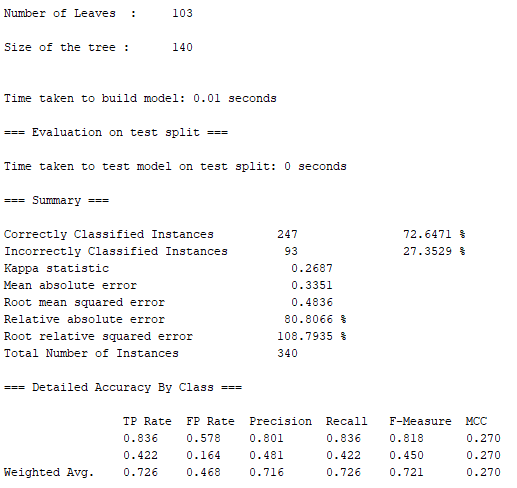
Question 2:

* By using the J48 classifier Analyzing the output result :
* It creates decision trees by recursively partitioning data based on attribute values. The classification algorithm used is shown in Scheme. The number of leaves and the size of the tree describes the decision tree. Also, Time is taken to build the model is also identified.



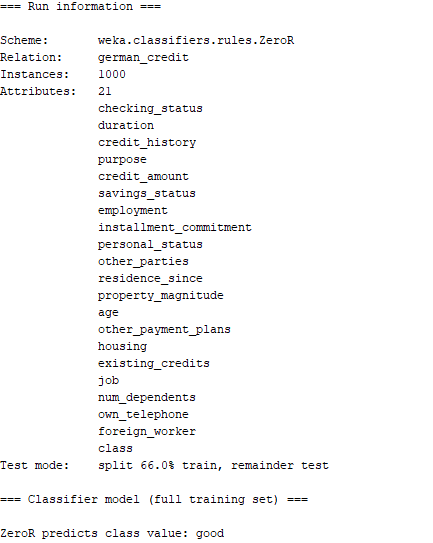
* Decision Tree Is shown as :
* Using Percentage Split in j48 Algorithm:

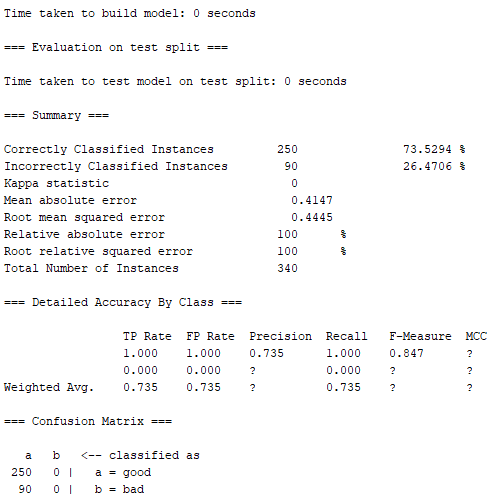




The percentage of correctly classified instances are 72.64% which is a pretty decent value but still as our dataset involves credit scores then it should have a pretty high value of correctly classified instances.

* Using Percentage Split using ZeroR:



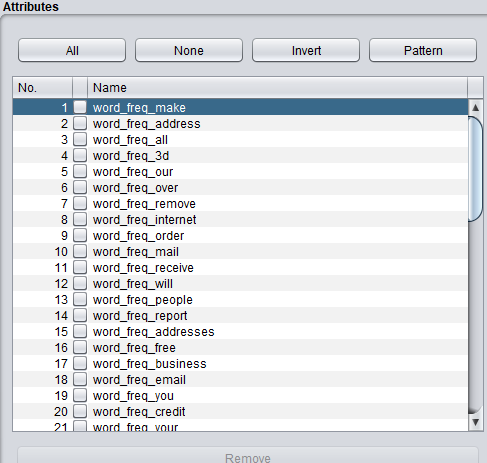


The percentage of correctly classified instances is 73.52% which is way more then the J48 algorithm .So the results obtained by the ZeroR algorithm is way better performing than the J48 algorithm.

* The algorithm that indiscriminately predicts whether a user is reliable or not, regardless of the actual values of attributes, is unlikely to be effective. Such an algorithm would essentially be making predictions based on a fixed or arbitrary criterion, ignoring the valuable information present in the input features.
* Taking a look on the data before the classification task is mandatory as we could get knowledge about the inconsistent data, dealing with class imbalance can be addressable.

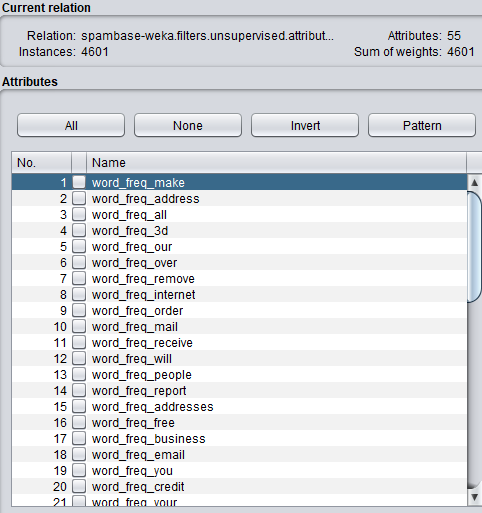
Naïve Bayes :

* Data set is shown as:

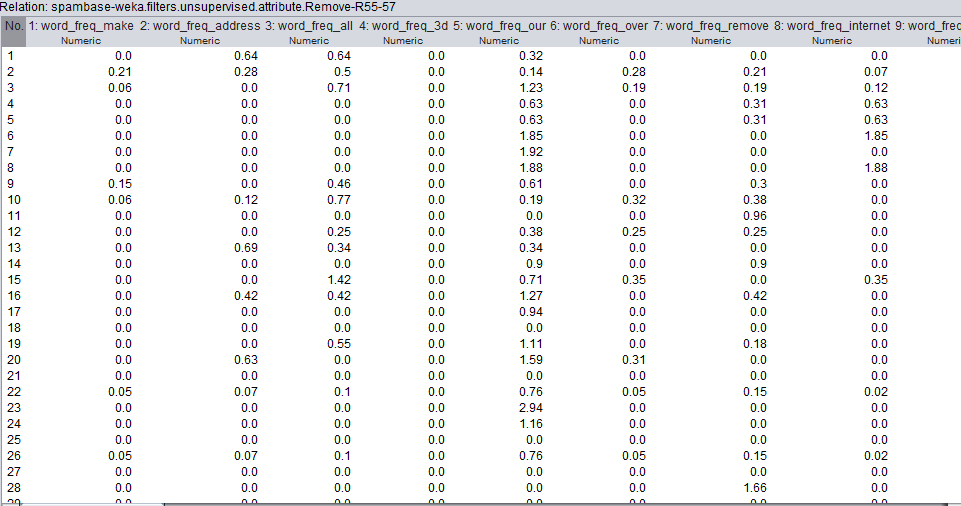


* Removing attributes capital\_run\_length\_average, capital\_run\_length\_longest and

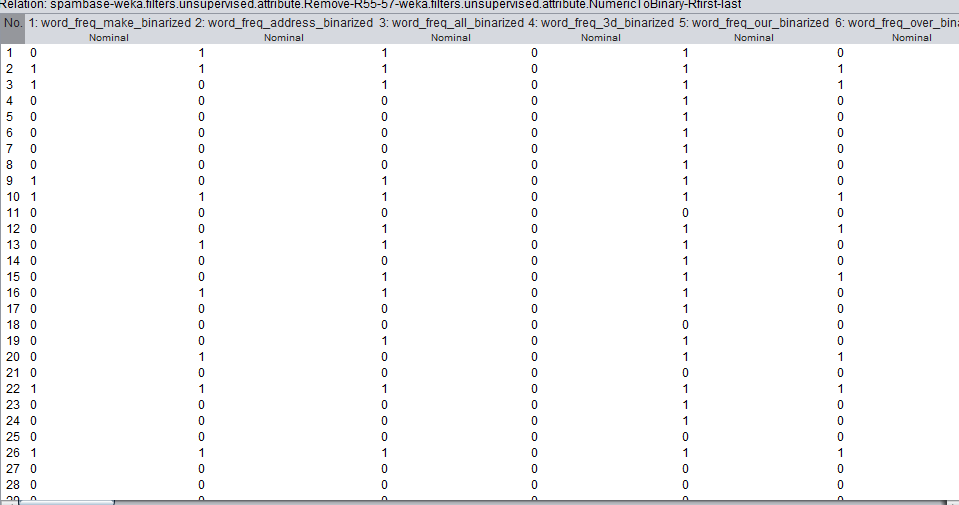
capital\_run\_length\_total attributes:



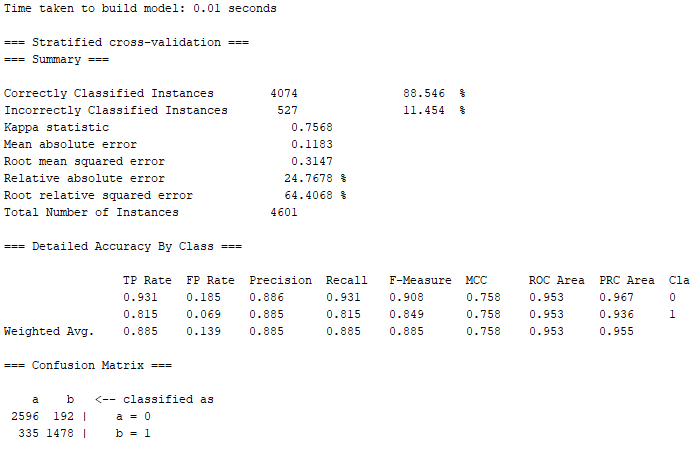
* Before relative frequency to Boolean :



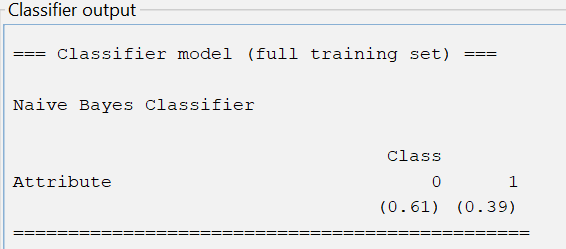
* After converting to Boolean :



* Applying Naïve bayes with the default settings of Cross fold as the test options:



* Reason for good performance could be that the Naive Bayes classifier assumes that features are conditionally independent given the class label and if we were not to take this assumption then the model would become complex which in turn would make the model leading to overfitting.
* The time taken to build this model is 0.07 which is relatively high in terms of training time for the given dataset. I think that Naïve Bayes is very much scalable when large datasets are involved due to there simple computation based on conditional independence.
* Prior Probabilities of the Naïve Bayes Classifier :



Naïve Bayes computes the probability of an email belonging to a particular class using Bayes' Theorem and the assumption of conditional independence among features.