MASTER THESIS

NIKOLAOS PAPAMANOLIOUDAKIS

MACHINE LEARNING ALGORITHMS COMPARISON

SUPERVISOR: DR TOCATLIDOU ATHENA

Contents

Introduction

The current thesis is a first approach on Machine Learning from the writer with very fundamental knowledge on this subject. On the other hand the writer having a great experience on Databases considered the transition natural (albeit not a smooth one). Like history, Machine Learning looks on the past (in this case existing data) and attempts to predict future situations (based on new data) and likewise to fictional character Harry Seldon and his psychohistory on Isaac Asimov's best novel "Last Foundation" you cannot predict the notions and ideas of one person but you can predict how the masses will move even in the far future if the number of persons that are part of the under examination mass is sufficiently big. Generally speaking machine learning is the process which gives to machines/computers the possibility to learn without the intervention or explicit guidance of the developer.

How is organized

This thesis is organized in two parts:

The theoretical part which is the first part examines the theory behind various fundamental algorithms such as decision trees, naive Bayes, K means and artificial neuron networks.
On a practical level, a comparison of 3 different algorithms is conducted. These 3 algorithms are naïve Bayes, C4.5 and finally Multilayer Perceptron. The comparison focus on how effective and how fast they are these algorithms. The source of the data that used for this research is from the Introduction to Artificial Intelligence of Cyprus Open University (Dr. Katakis Ioannis and Dr. Loizou - Kleanthous Styliani). In this project this Master Thesis has been based upon

Theoretical Part

C4.5

Introduction

C4.5 is an algorithm that is used to produce decision trees. This algorithm is implemented by Ross Quinlan and is an extension of his previous ID3 algorithm. Algorithm C4.5 improves ID3 by dealing with both continuous and discrete attributes, missing attributes as well as providing tree pruning. (Tan, Steinbach, & Kumar, 2014) Since C4.5 is a supervised algorithm requires some training data upon which the classifier will be build. It should be noted here that each example (record) consists of a set of attributes which define the final attribute (class). Records with same attributes belong to the same class (or at least in theory) (Perl, 1988). The algorithm analyzes the training data and builds the classifier. Accordingly the classifier should be able to classify correctly the test data after it has been build with the training data and it has been fine-tuned with validation data. Test dataset is used to validate the correctness of algorithm.

Decision Trees' Theory

At this point it is necessary to examine the basics of C4.5 algorithm:

For any given node of the decision tree that is built there are 3 possibilities:

1. All the records belong to the same Class C_i. In that case the node is a leaf node with class C_i (obviously).
2. There are no records on the node to be examined. In that case the node is a leaf BUT the class is the most frequent class of the parent node
3. There is a mixture of classes (C_1 ... C_n) on the current node T. This is the most usual case and in that case heuristic functions are used, such as entropy or GINI (Tan, Steinbach, & Kumar, 2014) and information Gain in order to split the current node T in children nodes in an effort to reach nodes that they have only one class C_i. Specifically for C4.5 the heuristic function used is entropy. It should be mentioned here that Entropy and Gini are used to measure the impurity of our data. For example in Entropy's case the values that it takes is between 0 and 1. If entropy is equal to 0.5 then the system is on the highest impurity while when on 0 or 1 the system has data that totally belong to either one or the other class.

In order for someone to understand better how C4.5 works, an example is necessary. The widely used example for this algorithm is the one examining if a game of football will take place or not, based on a series of weather parameters such as humidity, sun presence / outlook, temperature etc.

More specifically the attributes are as follow:

Outlook	Nominal	overcast,rainy, sunny
Temperature	Nominal	hot, cool,mild
Humidity	Ordinal	normal, high
Windy	Nominal	True, False

While the categorical attribute is the **Play: Yes / No**

Eventually the Training set consists of the following records:

Outlook	Temperature	Humidity	Windy	Play
overcast	hot	high	FALSE	yes
overcast	cool	normal	TRUE	yes
overcast	mild	high	TRUE	yes
overcast	hot	normal	FALSE	yes
rainy	mild	high	FALSE	yes
rainy	cool	normal	FALSE	yes
rainy	cool	normal	TRUE	no
rainy	mild	normal	FALSE	Yes
rainy	mild	high	TRUE	No
sunny	hot	high	FALSE	No
sunny	hot	high	TRUE	no
sunny	mild	high	FALSE	no
sunny	cool	normal	FALSE	yes
sunny	mild	normal	TRUE	yes

The final class Play has both Yes and No. As a result a further refinement is required. For that reason a choice of the most valid attribute must be made. Since this is the C4.5 algorithm the metrics of entropy and information gain will be used. The two mathematical types for Entropy "E" and Information Gain "IG" are as follow

$$E(S) = -p_1 log_2 p_1 - p_2 log_2 p_2 \dots - p_n log_2 p_n = -\sum_{i=1}^{n} p_i \log_2 p_i$$

We use here the Entropy E for a Set S of example (training set). Similarly $p_1 \dots p_n$ is the probability of each distinct value in set S. In our case for the final class n=2, p_1= 5/14 (TRUE), p_2 = 9/14 (False) and therefore:

$$E(S) = -\left(\frac{5}{14}\right) * log_2\left(\frac{5}{14}\right) - \left(\frac{9}{14}\right) * log_2\left(\frac{9}{14}\right) =$$

$$-(0{,}357 * -1{,}485) - (0{,}642 * -0{,}637) = 0{,}939$$

Similarly, the Entropy Et for each of the values that define the Attribute Temperature is as follow:
For the value Cool (4/14) 3 Yes and 1 No
For the value Hot (4/14) 2 Yes and 2 No
For the value Mild (6/14) 4 Yes and 2 No

Thus the Entropy for each value (Cool, Hot and Mild) is:

$$E(S_{Cool}) = -\left(\frac{3}{4}\right) * log_2\left(\frac{3}{4}\right) - \left(\frac{1}{4}\right) * log_2\left(\frac{1}{4}\right) =$$

$$-(0{,}75 * -0{,}415) - (0{,}25 * -2) = 0{,}811$$

$$E(S_{Hot}) = -\left(\frac{2}{4}\right) * log_2\left(\frac{2}{4}\right) - \left(\frac{2}{4}\right) * log_2\left(\frac{2}{4}\right) =$$

$$-(0{,}5 * -1) - (0{,}5 * -1) = 1$$

$$E(S_{Mild}) = -\left(\frac{2}{6}\right) * log_2\left(\frac{2}{6}\right) - \left(\frac{4}{6}\right) * log_2\left(\frac{4}{6}\right) =$$

$$-(0{,}333 * -1{,}585) - (0{,}667 * -0{,}585) = 0{,}918$$

We observe from the above that the entropy tends to 1 when we have equal division of the classes, and to 0.5 when we have clear classification (only one class).

As it is mentioned above the idea of Entropy alone is not enough to give the whole picture for the split since Entropy by itself is not sufficient to reach more than one final value of an attribute thus bringing into the table a very interesting problem: Which attribute gives the best split and find a way to choose this exactly attribute by calculating the best split. In that context the concept of **Information Gain** has been introduced (Tan, Steinbach, & Kumar, 2014). The Information gain is used in conjunction with the weighted average of the entropy of each attribute's Value. It can be easily seen that the Weighted Average WA is an expansion of the Entropy concept. More specifically the Weighted Entropy E_w of an attribute A in a Training Set S is described as follows:

$$E_w(S, A) = \sum_i \frac{S_i}{S} E(S_i)$$

Thus the Weighted Entropy of the Attribute Temperature T is as follows:

$$E_w(T) = \frac{4}{14} E(S_{Hot}) + \frac{4}{14} E(S_{Cool}) + \frac{6}{14} E(S_{Mild}) =$$

$$E_w(T) = \frac{4}{14} 1 + \frac{4}{14} 0{,}811 + \frac{6}{14} 0{,}918 = 0{,}911$$

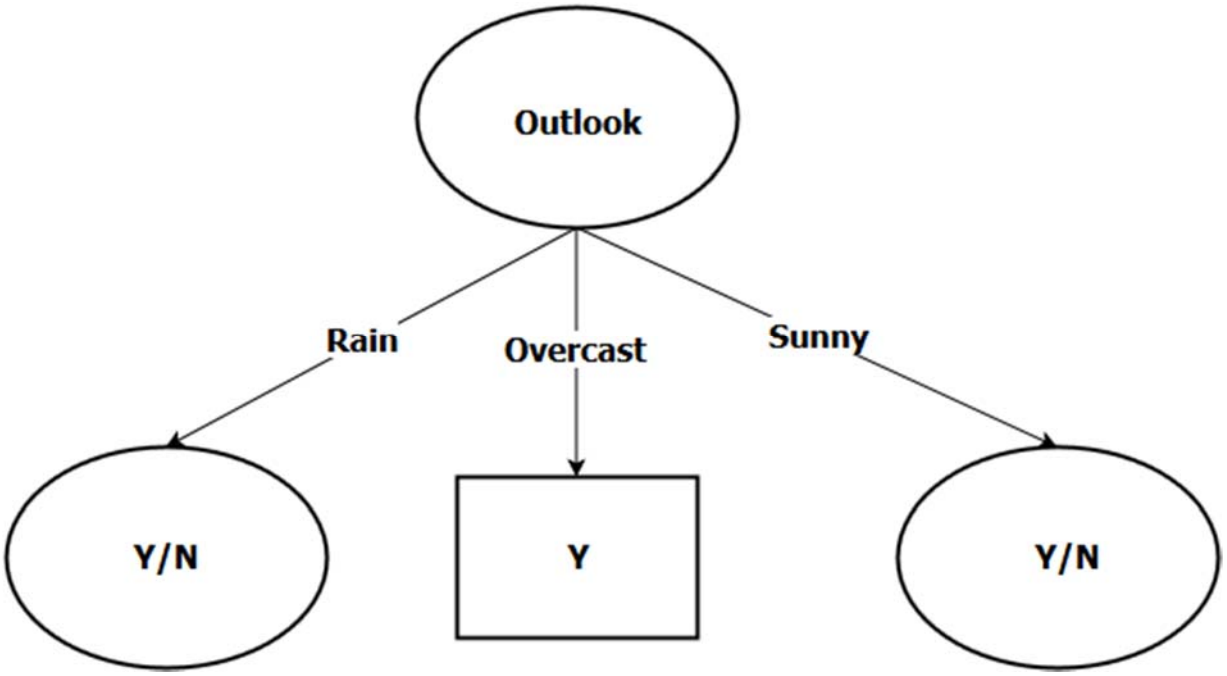
The Information Gain G over an Attribute A on a Training Set S is calculated from the following formula:

$$Gain(A, S) = E(S) - E_w(S, A) = E(S) - \sum_i \frac{S_i}{S} E(S_i)$$

Weighted Entropy Calculation		
Outlook	5/14*0,971+4/14*0+5/14*0,971	0,694
Temperature	4/14*0,811+6/14*0,918+4/14*1	0,911
Humidity	7/14*0,985+7/14*0,592	0,789
Wind	6/14*1+8/14*0,811	0,892

Note: The Calculation of each Attribute's Entropy is displayed on an attached Excel sheet (C45.xls)

From the above it is obvious that is sufficient to select the Attribute with the smallest Weighted Entropy (here outlook 0,694) since the first part of subtraction, which in that case is the entropy of the class attribute, is always constant (0,939). As a result the first attribute which is selected and will be the root of the tree is Outlook, since this is the attribute with the biggest Information Gain. On the next page the first step for the tree creation is presented.



Picture 1 - First step on Tree creation

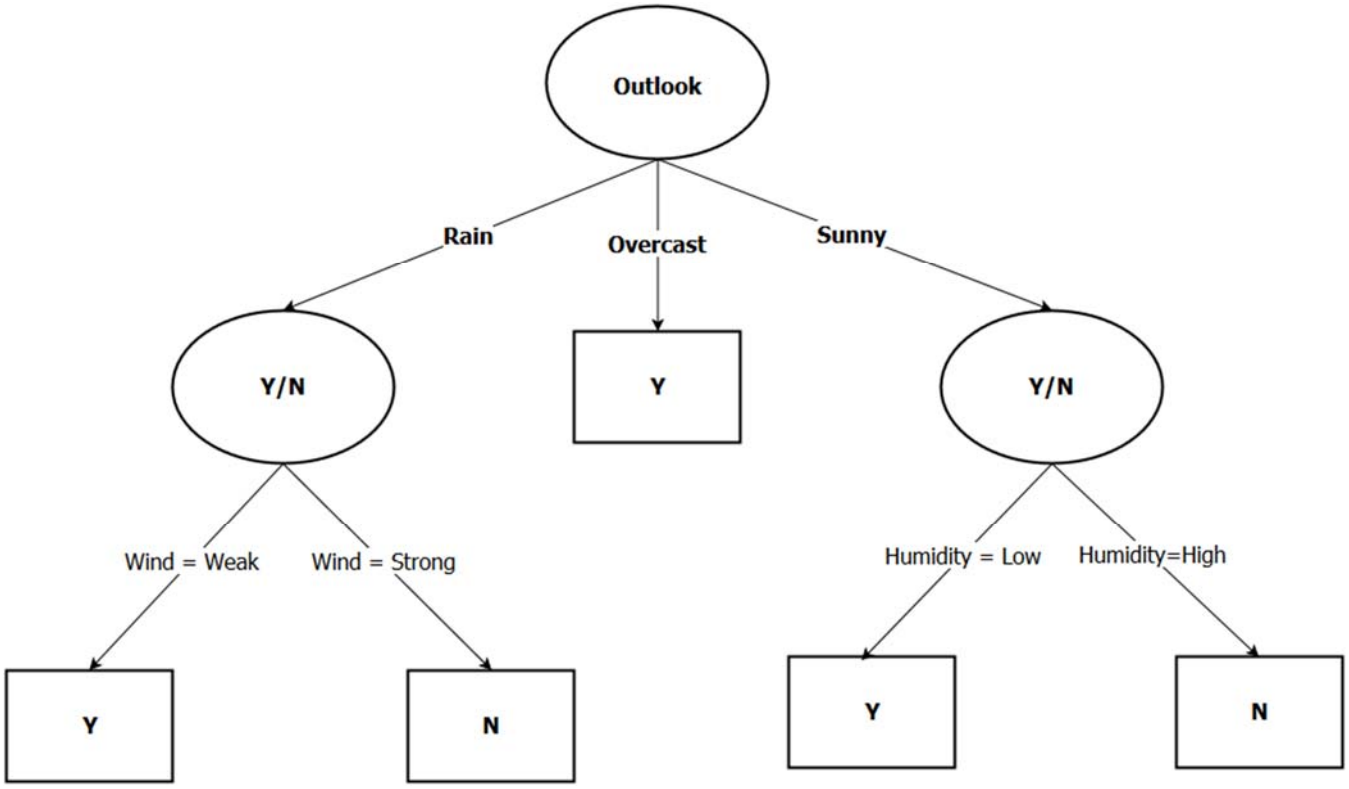
While the Outcast is a terminal Node (leaf) with all the records belonging to the same class (play ball ='Yes') this is not true for the other two values (Rain and Sunny) of the attribute Outlook. It should be reminded that the true aim here is to reach terminal nodes which belong to either one of the two values (Y/N) of the class attribute. As a result a new iteration of the algorithm and calculation of the Info gain for the two intermediate Nodes (Rain and Sunny)

On the next two tables someone can see the calculation of the Information Gain for the split on the Rain and Sunny values of the Outlook attribute

Weighted Entropy Calculation (Rain)		
Temperature	2/5*1+3/5*0,918	0,951
Humidity	2/5*1+3/5*0,918	0,951
Wind	3/5*0+2/5*0	0,000

Weighted Entropy Calculation (Sunny)		
Temperature	1/5*0+2/5*1+2/5*0	0,400
Humidity	3/5*0+2/5*0	0,000

Eventually from the above tables we conclude in the following tree:



Picture 2 - Second Step on Tree Creation

It is obvious from the above that if a Day ID existed, someone may be tempted to make a split based on this attribute. This is obvious wrong. Each ID uniquely identifies each record and while the combination Outlook = sunny, Temperature=hot, Humidity=high, Windy = false may appear again the Day ID 10 or the tenth Day of our observation will never appears again. Thus by choosing this attribute we will have 2 unwanted effects: 1) Very fragmented datasets and 2) The phenomenon of Overfitting (Tan, Steinbach, & Kumar, 2014) which will be described below. A very nasty side effect of selecting this type of attributes is that they always give the maximum Information Gain thus confusing this algorithm. In order for this effect to be countered the term of Intrinsic Information of the Attribute has been invented which basically describes the entropy when an attribute splits on branches. Intrinsic Information is calculated by the following formula:

$$IntI(S,A) = -\sum_i \frac{|S_i|}{|S|} log_2 \frac{|S_i|}{|S|}$$

Thus for the Day ID the Intrinsic Information IntI is:

$$IntI(DayID) = -\sum_i \frac{|1|}{|14|} log_2 \frac{|1|}{|14|} = 3{,}80$$

The Intrinsic Information by itself alone is not enough to decide for the split what is needed. It is the combination the Intrinsic Information in conjunction with the Information Gain. In order to counter the effect of selecting this kind of attributes a new term called Gain Ratio has been invented. The formula that describes the idea of Gain Ratio GR of an Attribute A in a Set S is described below:

$$GR(S,A) = \frac{IG(S,A)}{IntI(S,A)}$$

On the table below we can see the Gain Ratios of various attributes if we try to split from the class attribute First Step. As we have calculated above the Entropy for the Class is at 0,939. The Information Gain for the other 4 attributes is as follow:

Information Gain		
Outlook	0,939-0,694	0,247
Temperature	0,939-0,911	0,029
Humidity	0,939-0,789	0,152
DayID	0,939-0	0,939
Wind	0,939-0,892	0,048

Similarly the Intrinsic Information is as follow:

Intrinsic Information		
Outlook	-5/14*LOG2(5/14) - 5/14*LOG2(5/14) - 4/14*LOG2(4/14)	1,577
Temperature	-4/14*LOG2(4/14) - 4/14*LOG2(4/14) - 6/14*LOG2(6/14)	1,557
Humidity	-7/14*LOG2(7/14) - 7/14*LOG2(7/14)	1
DayID	14* LOG2(1/14)	3,80
Wind	-6/14*LOG2(6/14) - 8/14*LOG2(8/14)	0,985

And finally the Gain Ratio is:

Gain Ratio		
Outlook	0,247/1,577	0,157
Temperature	0,029/1,557	0,018
Humidity	0,029/1	0,029
Day ID	0,939/3,8	0,247
Wind	0,048/0,985	0,049

From the above it is obvious that Day ID attribute would still be the “winner”. On the other hand Gain Ratio is much more reliable criterion than the Information Gain. As a rule of thumb an attribute that uniquely identifies a record it should not be picked up and participate on the split procedure

Differences between C4.5 and Decision Trees

What has been examined up to now was only the theory of Decision Tree. In this context the three major contributions of C4.5 are that allow:

- Numeric Values
- Missing Values
- Pruning

The concepts of Numeric Values and Pruning deserves further analysis

Pruning

One of the biggest challenges (if not the biggest) is the decision on how big will become the decision tree. Is it going to have 10 nodes, 100 nodes or 1000 nodes? It has been observed that trees that have a small number of nodes, tend to give poor results both on training data as well as the testing data, while the trees with abundant nodes give good results on training data (since they have been built upon them) while have difficulties on test data. The former phenomenon is called model underfitting while the latter is called model overfitting. There are mainly three reasons for model overfitting:

1. The complexity of the model
2. The lack of representative samples on training data
3. The presence of noise on training data

In order to counter the effects of model fitting two methods of tree induction have been developed: **Prepruning** and **Postpruning**. (Witten, Frank, & Hall)

In the prepruning process the growth of the tree is stopped before the tree is build. Thus we choose a threshold based on the impurity gain e.g. the impurity gain falls below a certain level. The biggest challenge that can be faced (which is by the way the biggest disadvantage of choosing this method) is to find the correct threshold which will halt the further expansion of the tree. If the tree's growth stops too premature by choosing a high enough threshold then there is the danger to have the underfitting situation while if a low threshold is chosen then eventually the phenomenon of overfitting will not be avoided.

On the other hand in the case of postpruning the algorithm builds the whole tree and the tree is pruned afterwards. The mechanism of post pruning is based either on replacing the whole subtree with that leaf which has the majority of the records OR by replacing a bigger subtree with that branch that is used more. The former method is called Subtree Replacement while the latter is called Subtree Raising. (Witten, Frank, & Hall)

From the above it is rather obvious that the method of postpruning is giving better results because we have afterwards knowledge and we can prune specific non important branches and subtree. On the other hand the extra effort of calculating the whole tree it may be a burden if bigger tree are calculated and some parts of the calculations will be wasted during the process of postpruning.

Missing Values

Someone must examine how the missing values will be treated in the 3 stages of the algorithm

1. **In finding the best split** (through either Info Gain or Gain Ratio). In that stage the missing data are ignored and the best split is calculated with the rest of the data which they have actual values.
2. **In building the tree from the training data.** Here the data are sent to the children nodes according to the proportion of the records with existing values. For example if a decision has to be made over the attribute A and there are 20 records with value X and 10 records with value Y then the records with unknown values will be added by 2/3 to X and 1/3 to attribute Y
3. **And finally evaluating the tree accuracy.** In that stage all the possibilities are explored and a distribution is build. Eventually as final class for the prediction is the class with the highest distribution (Witten, Frank, & Hall).

K Nearest Neighbor Algorithm

Introduction

The next algorithm to be examined is the K Nearest Neighbor Algorithm (KNN). Similarly to the previous algorithm is a supervised learning algorithm, meaning that the algorithm is based on already known examples / output and it is on these known outputs that the new record/records are classified. Contrary to the decision trees, which is an eager algorithm, this is a lazy algorithm which means that the generalization happens with every new instance. On the previous algorithm (C4.5) the generalization has already been happened from the beginning. As a synopsis KNN Algorithm is a supervised lazy algorithm.

How it works

KNN Algorithm is a very simple algorithm. Basically what it does is to classify any new case based on the proximity with the already known cases of the database and that's the reason that is called Nearest Neighbor. K is the number of closest neighbors that we will select for the classification. It has been observed that if K is between 3 and 10 best results are achieved. Below 3 nearest neighbors underfitting issues have been observed, and similarly above 10 nearest neighbors overfitting issues occur. In order for the class of the query point to be determined it is necessary to define a measure between the query point and the already classified points. The usual choice for measuring distance is the Euclidean distance, based on the Pythagorean Theorem (Ancient Greeks to the rescue, again). The formula of the Euclidean distance (between points ……) is presented below:

$$\sum_{i=1}^{k} \sqrt{(x_i - y_i)^2}$$

Besides Euclidean Distance other distance functions can be used such as Manhattan or Minkowski in case that we have continuous variables. In case that we have a mixture of variables (continuous and nominal) OR nominal only then other distance measures should be used such as Hamming. As a rule of thumb Hamming distance is used mainly for measuring the difference between nominal variables. For example if one variable is Male and the second Female then the distance is 1 while if two variables are both Male or both Female then the distance is 0 (pretty much can be considered similar to a XOR Gate).

Example

In order for the algorithm to be more understandable the following case will be examined: An evaluation examining the possibility for a customer with a Loan of 142 K aged 48 to default based on the following table using the kNN algorithm

Age	Loan	Default
25	40.000	N
35	60.000	N
45	80.000	N
20	20.000	N
35	120.000	N
52	18.000	N
23	95.000	Y
40	62.000	Y
60	100.000	Y
48	220.000	Y
33	150.000	Y

Applying the Euclidean Distance Formula we get the following table:

Age	Loan	Default	Distance
33	150.000	Y	8.000,014
35	120.000	N	22.000,004
60	100.000	Y	42.000,002
23	95.000	Y	47.000,007
45	80.000	N	62.000,000
48	220.000	Y	78.000,000
40	62.000	Y	80.000,000
35	60.000	N	82.000,001
25	40.000	N	102.000,003
20	20.000	N	122.000,003
52	18.000	N	124.000,000

In the above table the data have been sorted based on the distance, thus producing the following table:

k	Yes	No	Default
1	1	0	Y
2	1	1	?
3	2	1	Y
4	3	1	Y
5	3	2	Y
6	4	2	Y
7	5	2	Y
8	5	3	Y
9	5	4	Y
10	5	5	?
11	5	6	N

It is rather obvious from the above that in the majority of the cases this loan will default, particularly in the area of k that has been described above (3 to 10).

There is an issue though with the calculation on Euclidean distance. A keen observant can see that the contribution of the Age attribute is practically nonexistent and the calculation is based almost exclusively on the Loan attribute. Practically it is the absolute difference between the query record and each record of the dataset. As a result an approach is needed that will consider also the Age Factor. For that reason the term of standardized variable is used in general and will be used here. Standardized variable is basically the distance of the original value from the mean of the variables that consist the attribute in terms of standard deviation. In other words a standardized variable x*of 1 means that the variable x is 1 standard deviation σ away from the mean μ of the data. All this is expressed with the following formula:

$$x^* = \frac{x - \mu}{\sigma}$$

The mean μ of a dataset $a_1, a_2, \ldots, a_i, \ldots, a_n$ is given by the following formula:

$$\mu = \frac{1}{n}\sum_{i=1}^{n} a_i$$

While the standard deviation σ for the above dataset is as follows:

$$\sigma = \sqrt{\frac{1}{n}\left(\sum_{i=1}^{n}(a_i - \mu)\right)^2}$$

Thus the first table with the raw data becomes as follows:

Age	Loan	Default
-1,035	-0,797	N
-0,227	-0,463	N
0,580	-0,129	N
-1,438	-1,131	N
-0,227	0,539	N
1,145	-1,165	N
-1,196	0,121	Y
0,176	-0,430	Y
1,790	0,205	Y
0,822	2,210	Y
-0,389	0,907	Y

And the calculation of the Euclidean distance based on the standardized variable is presented on the table that follows:

Age	Loan	Default	Distance
1,145	-1,165	N	0,441
0,176	-0,430	Y	1,220
0,580	-0,129	N	1,358
-0,227	-0,463	N	1,451
1,790	0,205	Y	1,931
-1,035	-0,797	N	1,973
-0,227	0,539	N	2,263
-1,438	-1,131	N	2,284
-1,196	0,121	Y	2,567
-0,389	0,907	Y	2,663
0,822	2,210	Y	3,675

Similarly here the data have been sorted based on the Euclidean distance (from smallest to largest). Eventually based on the above table the final table that calculates if the loan will default or not, is presented below:

k	Yes	No	Default
1	0	1	N
2	1	1	?
3	1	2	N
4	1	3	N
5	2	3	N
6	2	4	N
7	2	5	N
8	2	6	N
9	3	6	N
10	4	6	N
11	5	6	N

What is really interesting from the above table is the fact that now that the age attribute is contributing on equal measure to the calculation of the distance the prediction if the loan will default switched from Yes to No.

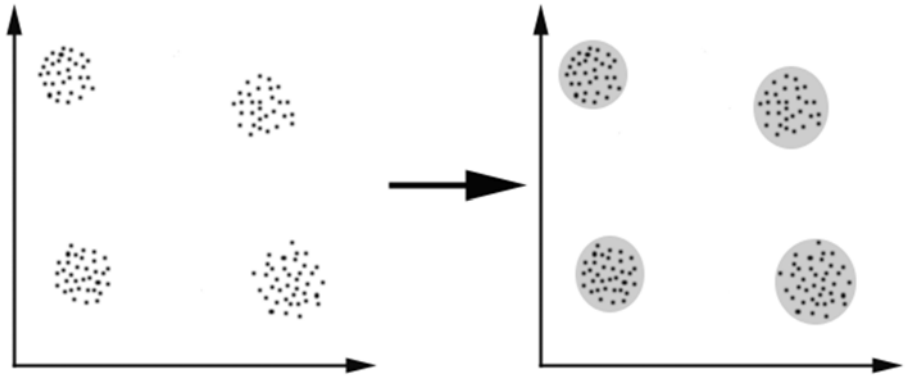
Conclusions about kNN

From the above is rather obvious that kNN is a very easy to use and understand algorithm. Additionally to that the results that is offering are of high value and predictability.

k-Means Algorithm

Introduction

The algorithm that will be examined on this session is a clustering algorithm (Witten, Frank, & Hall). This means that on the contrary with the algorithms that examined on the two previous sessions where some already known cases existed and based on these cases each new one was taking a label (classification), in this algorithm the records are grouped based on their attributes and the similarity in one cluster. According to Wikipedia clustering *is the task of grouping a set of objects in such a way that objects in the same group (called a **cluster**) are more similar (in some sense or another) to each other than to those in other groups (clusters).*It is considered to be the most important unsupervised algorithms. Clustering in general can be based on the distance between the members of one cluster and the members of other clusters, or other characteristics can be used such as sex (M, F), Age, Income etc. In the following picture someone can see an example of clustering based on distance. In the case that is examined here (k-Means) the number of clusters (groups) is equal to K



How it works

The K-Means Algorithm consists of the following steps:

1. Randomly select K initial centroids
2. Assign the other members of Dataset to the Initial Centroid forming K numbers of clusters.
3. Calculate the distance between the selected centroids and all the members of the cluster
4. If there is a smaller distance between a member of a cluster and a centroid that belongs to another cluster
   a. Then : move this record to the new cluster and repeat from step 2
   b. Else : if there is no change END

The procedure of reducing the intra-cluster variance can be expressed with the following formula:

$$J = \sum_{j=1}^{k} \sum_{i=1}^{n} \left(x_i^j - c_j\right)^2$$

Where

- k is the number of clusters
- n is the number of cases
- x_i^j is the case i for cluster j
- c_j is the centroid for cluster j

As it is the case with the other algorithms that examined up to now in order for the algorithm to be more understandable an example is in order.

Again here the example data with Age Income attributes that was used on K NN algorithm will be used here. In this example though the standardized variable will be used directly. Additionally the data is sorted by loan amount and the unknown case has also been included (with ID 12):

ID	Age	Loan
1	0,822	-1,465
2	1,145	-1,165
3	-1,438	-1,131
4	-1,035	-0,797
5	-0,227	-0,463
6	0,176	-0,43
7	0,58	-0,129
8	-1,196	0,121
9	1,79	0,205
10	-0,227	0,539
11	-0,389	0,907
12	0,822	2,21

Initially 3 groups will be selected. One group will be the one with ID 1 to 4, the second with ID 5 to 8 and the third one with ID 9 to 12. Thus the 3 centroids are as follow:

Cent ID	Age	Loan
1	-0,127	-1,14
2	-0,167	-0,225
3	0,499	0,9653

And the data have been grouped initially to 3 groups as shown below:

ID	Age	Loan
1	0,822	-1,465
2	1,145	-1,165
3	-1,438	-1,131
4	-1,035	-0,797
5	-0,227	-0,463
6	0,176	-0,43
7	0,58	-0,129
8	-1,196	0,121
9	1,79	0,205
10	-0,227	0,539
11	-0,389	0,907
12	0,822	2,21

The first calculation of the Euclidean distance gives the following metrics:

ID	Cent 1	Cent 2	Cent 3
1	1,003	1,586	2,452
2	1,272	1,614	2,226
3	1,312	1,561	2,854
4	0,971	1,040	2,336
5	0,684	0,245	1,602
6	0,771	0,399	1,432
7	1,233	0,753	1,097
8	1,653	1,086	1,894
9	2,341	2,003	1,498
10	1,682	0,767	0,842
11	2,063	1,154	0,890
12	3,481	2,628	1,286

From the above it is obvious that the record with ID 10 belongs to the second group thus reshaping the data as follows:

ID	Age	Loan
1	0,822	-1,465
2	1,145	-1,165
3	-1,438	-1,131
4	-1,035	-0,797
5	-0,227	-0,463
6	0,176	-0,43
7	0,58	-0,129
8	-1,196	0,121
10	-0,227	0,539
9	1,79	0,205
11	-0,389	0,907
12	0,822	2,21

While the centroids are now configured like this:

Cent ID	Age	Loan
1	-0,127	-1,14
2	-0,179	-0,072
3	0,741	1,107

By calculating the Euclidean distance for each member of the dataset for 2nd time the following table is obtained:

ID	Cent 1	Cent 2	Cent 3
1	1,003	1,715	2,574
2	1,272	1,716	2,308
3	1,312	1,645	3,124
4	0,971	1,122	2,604
5	0,684	0,394	1,845
6	0,771	0,504	1,638
7	1,233	0,761	1,247
8	1,653	1,035	2,174
10	1,682	0,613	1,123
9	2,341	1,988	1,384
11	2,063	1,002	1,148
12	3,481	2,492	1,106

From the above it is obvious that the record with ID 11 must be moved to 2nd group restructuring the data as follows:

ID	Age	Loan
1	0,822	-1,465
2	1,145	-1,165
3	-1,438	-1,131
4	-1,035	-0,797
5	-0,227	-0,463
6	0,176	-0,43
7	0,58	-0,129
8	-1,196	0,121
10	-0,227	0,539
11	-0,389	0,907
9	1,79	0,205
12	0,822	2,21

The Centroids are now becoming like this:

Cent ID	Age	Loan
1	-0,127	-1,140
2	-0,214	0,091
3	1,306	1,208

While calculating the Euclidean Distance for third iteration:

ID	Cent 1	Cent 2	Cent 3
1	1,003	1,869	2,716
2	1,272	1,850	2,378
3	1,312	1,730	3,605
4	0,971	1,209	3,082
5	0,684	0,554	2,267
6	0,771	0,651	1,990
7	1,233	0,824	1,521
8	1,653	0,983	2,728
10	1,682	0,448	1,672
11	2,063	0,835	1,721
9	2,341	2,007	1,113
12	3,481	2,359	1,113

The above makes obvious that there is no any other shift on the placement of records in clusters thus stopping our algorithm.

Generally speaking it is rather obvious that the choice to use standardized variables instead of the initial variables is a correct one. If only the initial variables had been used then there will be no shift to place of records on clusters and the final group will be similar with the initial Loan sorting. Additionally someone can observe that the 3 groups/clusters that initially were selected categorize satisfactory the Age/Income attributes. For example on the group that has the most records (number 2/green) there are the records that lie around the median while for the other 2 groups either have low to very loans (number 1/yellow) or have either very high loan (case 12) or very high age (case 9). A study based on clustering on Age would have particular interest for someone to discover if the Age would produce the same results.

Conclusions about k-Means

Similar to the other algorithms that have been examined up to here the K-Means is an easily understandable and easy to implement algorithm. It can be used as a first step for other algorithms or to get an insight to the data. Its complexity is O(Knt) where K are the clusters, n are the records count and t the iterations and because usually K, t << N it depends basically on the Number N. It should be reminded here that Big O notation is used for demonstrating how fast an algorithm is.

On the other hand it is rather obvious that the choice of K and a small amount of data may lead to undesired effects. Additionally to the above the initial selection of K and the structure of data (as it is in the above case the sorting based on Loan) can also affect negatively the algorithm.

Naïve Bayes Algorithm

Naïve Bayes Algorithm (Perl, 1988) is an algorithm that is based on the theorem of Thomas Bayes an English statistician who owes his fame on this exactly theorem. Before the famous formula presented it is imperative for an example to be presented

Assume that the probability that a man at the age of 50 has prostate cancer is 5%
The probability of a man WITH prostate's cancer that has conducted the prostate cancer screening process with positive results is 80% (aka true positive)
And finally the probability that a man WITHOUT prostate cancer process that had positive results is 20 % (aka false positive).
Assume the following situation: A patient takes the exam, the test shows that has cancer, what is the probability to actually have cancer? The answer obviously is not 80%

The above problem can be rephrased as follows assuming that we have a typical sample of 1000 men aged 50:

1. Men at age of 50 that have cancer: 50
2. Men at age of 50 that DO NOT have cancer: 950

a) Men at age of 50 that took the test, test was positive and had cancer 0,8 * 50 = 40
b) Men at age of 50 that took the test, test was positive and had not cancer 0,2 * 950 = 190
c) Men at age of 50 that took the test, test was negative and had cancer is 0,2 * 50 = 10
d) Men at age of 50 that took the test, test was negative and had not cancer is 0,8 * 950 = 760

The part that is relevant are the cases A and B, people that actually had positive test (regardless of having or not cancer meaning 230 people) and from those only these men that actually have cancer (40). The possibility of someone having cancer after the test is positive is eventually 40/230 = 0,17 or in other words 1 out of 6 men that have taken the test and had positive results will eventually have cancer.

It is time for the above example to be generalized and in accordance notations can be made.
Let X the fact that is expressed on statement 1 meaning P(X) = 0,05
From the above it is induced that P(1-X)=0,95 or P(¬X)=0,95
Assuming that A is the probability of having a positive test then according to statement a : P(A|X)=0,8 ==>P(A|X)*P(X) = 0,04 or 4%
Similarly P(A|¬X)P(¬X)=0,19 or 19% (Positive Test but Men without Cancer),
P(¬A|X) *P(X) = 0,01 or 1% (Negative Test but Men with Cancer),
P(¬A|¬X) * P(¬X) = 0,76 or 76%. (Negative Test and Men without Cancer),
Someone can observe that the sum of all the conditional probabilities multiplied by the initial (prior) probability is equal to 1 or 100%.
Generalizing from the above example it is obvious that:

$$P(X|A) = \frac{P(A|X) * P(X)}{P(A|X) * P(X) + \mathrm{P}(\neg \mathrm{A}|\mathrm{X}) * P(\neg X)}$$

The above is the Bayes Theorem in an extended form. A more compact formula is the following:

$$P(X|A) = \frac{P(A|X) * P(X)}{P(A)}$$

Note : The transition from the complex formula to its simplified form is rather easy and its prove can be found on the respective page of Wikipedia (Bayes' theorem, n.d.)

The interesting part is the Bayes Theorem's use for classification. The application of Bayes Theorem to classification is based on the assumption that each attribute is independent from any other. What is basically needed here is to calculate the probability of a Class C based on a series of Attributes (X_1, X_2, X_3,...X_n) or in other words :

$$P(C_k|X)$$

Where C_k is an instance of the class (Yes|No, Male|Female etc, not necessarily binary type classes though) and X is a vector consisting of various attributes (in our case only nominal and ordinal). It is rather difficult or even impossible, if for example the number of the data volume is big, to calculate this probability and that's why the Bayes Theorem is used. It is much easier to calculate the probability of each attribute based on the class, than the other way around. The above type can easily be transformed with the help of Bayes Theorem:

$$P(C_K|X) = \frac{P(X|C_k) * P(C_k)}{P(X)}$$

And eventually using the fact of conditional independence someone can reach the following type that summarizes the Naive Bayes Classifier:

$$P(C_k|X) = \frac{P(C_k) \prod_{i=1}^{n} P(x_i|C_k)}{P(X)}$$

In the above formula we calculate the possibility of the class Ck based on vector X ($P(C_k|X)$). Basically what this type says is that the $P(C_k|X)$ is based on the products (Π) of each attribute based on class Ck multiplied by the possibility of Class Ck and divided by the possibility of the vector X (p(X)). It is obvious here that we want to maximize the quantity $P(C_k|X)$ for every k. Since the P(x) is always stable we can safely assume that

$$Max(P(C_k|X)) = Max\left(p(C_k) \prod_{i=1}^{n} p(x_i|C_k)\right)$$

Again in order for the power of Naive Bayes Classifier to be fully understandable an example is in order. The well-known example of the football game in regard to weather condition will be used for one more time.

Outlook	Temperature	Humidity	Windy	Play
overcast	hot	high	FALSE	yes
overcast	cool	normal	TRUE	yes
overcast	mild	high	TRUE	yes
overcast	hot	normal	FALSE	yes
rainy	mild	high	FALSE	yes
rainy	cool	normal	FALSE	yes
rainy	cool	normal	TRUE	no
rainy	mild	normal	FALSE	yes
rainy	mild	high	TRUE	no
sunny	hot	high	FALSE	no
sunny	hot	high	TRUE	no
sunny	mild	high	FALSE	no
sunny	cool	normal	FALSE	yes
sunny	mild	normal	TRUE	yes

While on C4.5 in order for a new record to be classified trees were used, in Naive Bayes Classifier frequency tables are constructed. First of all the probability of class must be determined. These probabilities are as follow:

Probability of the game to be conducted P(Cy) = 9/14
Probability of the game NOT to be conducted P(Cn) = 5/14

The next step is to build the frequency tables for each attribute:

		Play Ball	
		YES	NO
Outlook	Sunny	2/9	3/5
	Overcast	4/9	0/5
	Rainy	3/9	2/5

		Play Ball	
		YES	NO
Temperature	Cool	3/9	1/5
	Mild	4/9	2/5
	Hot	2/9	2/5

		Play Ball	
		YES	NO
Humidity	High	3/9	4/9
	Normal	6/9	1/9

		PlayBall	
		YES	NO
Windy	TRUE	3/9	3/5
	FALSE	6/9	2/5

A new record comes with the following characteristics:
Outlook: Rainy
Temperature: Mild
Humidity: Normal
Windy: True

In order for a decision to be reached if the game will be conducted, the probabilities of P(Cy|X) and P(Cn|X) must be calculated or eventually

$$P(C_y|X) = \frac{P(Rainy|Yes) * P(Mild|Yes) * P(High|Yes) * P(True|Yes)}{P(Rainy) * P(Mild) * P(High) * P(True)} * P(Yes)$$

$$P(C_n|X) = \frac{P(Rainy|No) * P(Mild|No) * P(High|No) * P(True|No)}{P(Rainy) * P(Mild) * P(High) * P(True)} * P(No)$$

In both cases the denominator is the same and as a result can be omitted. The above formulas can be replaced with numbers as follow:

$$P(C_Y|X) = \frac{3}{9} * \frac{4}{9} * \frac{6}{9} * \frac{3}{9} * \frac{9}{14} = \frac{1944}{91854} = 0{,}0211$$

$$P(C_N|X) = \frac{2}{5} * \frac{2}{5} * \frac{1}{5} * \frac{3}{5} * \frac{5}{14} = \frac{60}{15750} = 0{,}0038$$

And after Normalization

$$P(Yes) = \frac{0{,}0211}{0{,}0211 + 0{,}0038} = 75{,}529\%$$

and respectively

$$P(No) = \frac{0{,}0038}{0{,}0211 + 0{,}0038} = 24.471\%$$

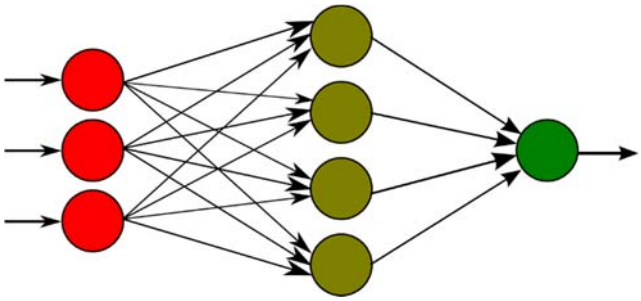
Since P (yes) > P (No) the game will take place

A very interesting problem can occur in the case that there is no observation for a particular case. Here for example what will happen if someone wants to calculate the possibility of overcast = NO? In that case the addition of 1 in all observations can eliminate this problem offering a workaround. Making for example the outlook table as follow:

		PlayBall	
		YES	NO
Outlook	Sunny	3/12	4/8
	Overcast	5/12	1/8
	Rainy	4/12	3/8

Artificial Neural Networks (Single-layer / Multi-Layer Perceptron)

Artificial Neural Networks is one of those ideas in which man tries to imitate nature. ANN imitate the most wonderful and frightening entity in nature: the human brain. Similarly to the human brain the Artificial Neuron Network consists of neurons. These neurons are receiving inputs, they process them and they forward them to the next neuron or to the output. In this context we will examine two manifestations of ANN: the single and the multilayer Perceptron. The biggest and most important difference between the single and multi-layer perceptron is, as it is obvious by the name, the extra layer that exists between the input and the output layer. A multilayer perceptron is depicted below:



(Image is from Wikipedia)

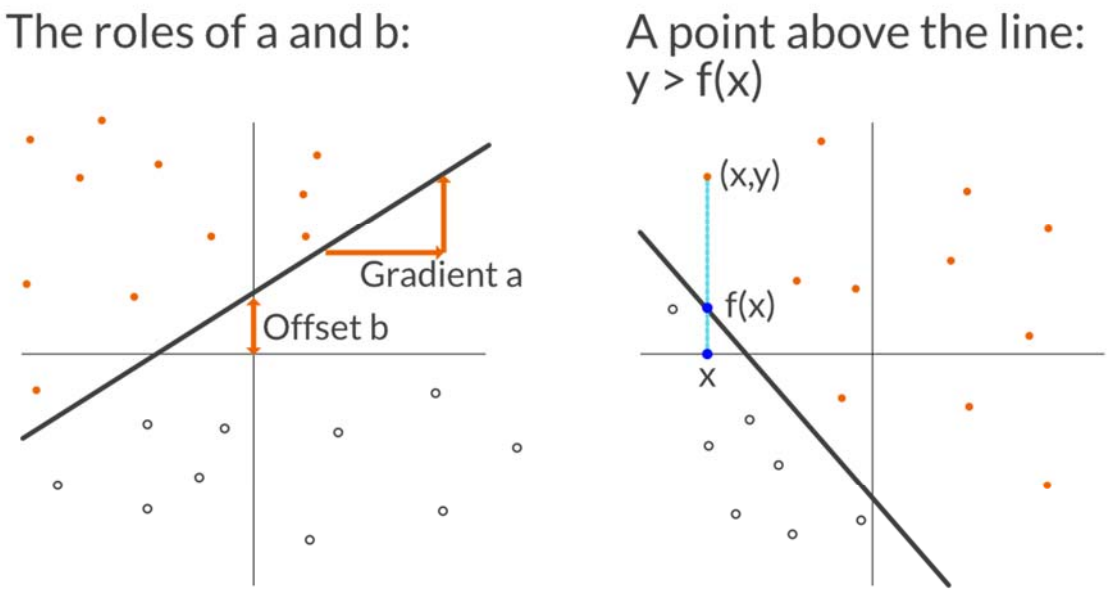
Functionality of Perceptron

The difference in functionality of a single and a multi-layer perceptron lies on the fact that single layer perceptrons can only classify linearly separable data while multi-layer perceptrons can classify any type of data (linear and nonlinear). The functionality of a perceptron is performed through 3 steps:

1. **Signals coming in**: Initially signals from other perceptrons or from the environment are coming in. For each signal a weight is assigned. If for example there are 2 inputs x_1, $x_{2\ 3}$ then there are also 2 weights (w_1, w_2) for the inputs. In that way inputs eventually become w_1x_1 and w_2x_2. It should be mentioned here that initially the weights are totally random. It is the responsibility of the learning methodology to adjust the weights in order for the classification to be as successful as possible.
2. **Signals summed up:** On the second step the weights are summed up and an offset is added. This offset is called bias and is symbolized with the Greek letter theta ϑ. So eventually the final outcome becomes $w_1x_1+w_2x_2+\vartheta$. Someone with basic mathematic knowledge can easily transform the previous sum to the well-known linear function y=ax+b where $b=\vartheta/w_2$ and $ax= w_1x_1$ or eventually

$$x_2 = -\frac{\theta}{w_2} - w_1x_1$$

3. **Activation**: In the third step the final sum is processed from a function which is called activation or transferred function. There are various functions that can be used such as the Heaviside Step function, sigmoid function or even the normal distribution function (Gauss Curve). The function that is used most often is the Heaviside Step function which results in 0 if the input is negative and 1 if the input is positive.



(Picture from https://appliedgo.net)

Since the function of a single layer perceptron is to linearly separate the two classes, points of which majority is "above" the line belong to the first class, while points that belong to the second class are "below" the line. In the case that a point belongs to the second class for example, and is above the line, the learning process is coming to game modifying bias and weights with the factor that is called learning rate (n). Obviously the smaller this n factor is, the more precise can be the learning process. In order for this to be clearer an example is necessary

Let's assume that we have the following data:

$X_1 = 2$ $X_2 = -2$ and this point belongs to the second class. At the same time this point is "above" the line. Additionally to the above we have $w_0 = 0$, $w_1 = 1$, and $w_2 = 0.5$ (these are the initial weights). Finally the learning rate n is equal to 0.2. The formula that produces the weight W_k on i_{th} iteration is as follow

$$W_k^i = W_k^{i-1} \pm nx_k^i$$

It should be mentioned here that if the point is above the line and belongs to the second class then there is subtraction (the $\pm$ becomes -). In the opposite case that a point is below the line and belongs to the first class then addition will be chosen (the $\pm$ becomes +) in order to produce the new weight. In the case that is examined the weights eventually become as follow:

$W_0 = 0 - 0.2*1 = -0.2$
$W_1 = 1 - 0.2*2 = 0.6$
$W_2 = 0.5 - 0.2 * (-2) = 0.9$

Research and Implementation Part

Introduction

In the next sections the algorithms that have been described on a theoretical level will be examined also with a more practical approach. In that context the well-known Machine Learning tool called Weka [R] will be used. Additionally 3 of the most frequently used algorithms will be examined.

These are: The Multilayer Perceptron, the Naive Bayes and finally the workhorse of the Machine Learning Community the C4.5 the algorithm of tree construction, not necessarily in that order. The material that will be used for the algorithms evaluation is a questionnaire that was used during the Cypriot elections of 2011. The machine that will be used for test is described by the following specifications:

Processor	Intel Core i5-4200 CPU @ 2.50
Number of procs	4
Memory	8 GB
OS	Windows 7 Professional

Data Preprocessing

The Data are provided in a csv (comma separated values) file, in which the first line is the column description. All the other lines (rows) are the answer of each user. Each row consist of the following fields:

1. Visitor id : record id
2. Date : Actual date the questionnaire was filled in
3. Time: Actual time the questionnaire was filled in
4. Sex : 1. Male, 2. Female, 0. Did not supply information
5. Date of Birth : The year of birth, 0. Did not supply information
6. Refugee: 0. Yes, 1. No, 99. Did not supply information
7. Q4 = Educational attainment 1. Primary School, 2. Secondary School, 3. Lyceum, 4. University degree, 5. Post-Graduate degree, 0. Did not supply information
8. Q5 = Party_id : In which party the visitor belongs : 1. AKEL , 2. DISY , 3. DIKO , 4. EDEK, 5. EUROKO, 6. Ecologist-Environmentalist Movement 7. ELAM, 8. Movement of Social Ecology, 9. ZYGOS 10. KY.PRO.S 11. LASOK 12. Other 13. None 0. Did not supply information
9. Q6: Vote_intention: If elections would have taken place now what would you have voted : 1. AKEL , 2. DISY , 3. DIKO , 4. EDEK, 5. EUROKO, 6. Ecologist-Environmentalist Movement EEM 7. ELAM, 8. Movement of Social Ecology MSE, 9. ZYGOS 10. KY.PRO.S 11. LASOK 12. Other 13. None 0. Did not supply information. This, by the way, will be the target class
10. Q7: Vote reason : Which of the following is for you the most fundamental criterion for senator choice : 1. The personality of the party leader, 2. The ideological identity of a party, 3. A party's knowledge of the problems, 4. The party's program, 5. The party candidates, 6. My personal relationship with party members 7. Other, 0. Did not supply information
11. Total time = Total length of time taken by the respondent to complete the questionnaire
12. Attempt = Number of times web browser was used to visit the site
13. Policy statements q1-30
    1) In the negotiations for the Cyprus' problem, the Government has made unacceptable concessions.
    2) A bi-zonal, bi-communal federation with one sovereignty and citizenship is an acceptable solution.
    3) The current process of resolving the Cyprus problem should be abandoned by replacing it with a five-party conference (Greek-Cypriots, Turk-Cypriots, Greece, Turkey and the EU).
    4) Even if an agreed settlement is achieved, with the Turkish Cypriots we will have to live separately rather than together.
    5) Turkey should join the EU because this will help solve the Cyprus problem.

6) The mobilization of Turkish-Cypriots for being released from Turkey is important in helping the prospect of a final settlement of the Cyprus problem.
7) Cyprus does not have to demand at the present moment the abolishing of the British bases.
8) The position of Cyprus is in NATO.
9) Cyprus must apply for membership in the program "Partnership for Peace".
10) The Cypriot economy has not benefited from EU membership.
11) The corporate profits tax should be increased for two years from 10% to 11%.
12) The tax of property should be increased for the 5,000 biggest land owners.
13) The semi-governmental organizations in general should be privatized.
14) The economy functions better as far as the state intervenes less and companies are provided with more liberty to operate.
15) The institution of the ATA is outdated and should be abolished.
16) The fiscal deficit should be covered largely by additional taxation of wealth.
17) It is better to increase taxes than to cut spending on education and health.
18) Foreign investments, like the one by the state of Qatar, should be facilitated by the government.
19) Military spending should be increased even if this means increasing the public deficit.
20) The system of social welfare (social security and pensions) should be rather managed by the private sector.
21) The entry of foreign workers generally harms rather than benefits the Cypriot economy.
22) There should be found a way so that political asylum seekers and political refugees do not get any financial benefits.
23) The increasing presence of foreign immigrants strengthens the multicultural identity of Cyprus.
24) The institutionalization of pupils' unionism will help the more active involvement of youth in politics.
25) Cyprus should follow the example of other European countries and allow civil partnerships between homosexual couples.
26) The views of the Church of Cyprus should be seriously taken into account regarding the formulation of the country's policy-making.
27) To increase the sense of security, civil and political liberties should be limited.
28) Criminality is due to the large number of immigrants who are in Cyprus.
29) The protection of the environment should not be an obstacle for economic development.
30) The owners of golf courts contribute vitally to the economy and this why they should not be further burdened with additional "green" taxes on excessive water consumption.

14. Policy statements q1-q30 Answer categories : 1 = Completely Disagree, 2 = Disagree, 3 = Neither agree, neither disagree , 4 = Agree, 5 = Completely agree, 99 = No opinion
15. Policy statement t1-t30 :Time taken to answer each question in seconds

From the above it is rather obvious that most of the fields must be eliminated / excluded from the list. Eventually only the 30 answers (field 14/q1-q30 as well as field 9/Q6 will be used). Additionally because C4.5 as well as Naive Bayes require either nominal or binary class further transformation is required. In that context the data from the initial files have been imported to .XLS the proper modifications have been made and two new .CSV's have been produced, one with the name NominalClass.csv, with the final class transformed to nominal data and a second file with the name NumericClass.csv, with the final class as is (numeric). These two .csv's files will be used in WEKA

Methodology

The experiment and the comparison of the algorithms will take place in 3 steps

1. The first step will be a comparison between the 3 algorithms with default values from weka (Basic Algorithm Comparison)
2. The second step will be an optimization of the above in order to find the optimal configuration for this data and finally
3. The third step will be to further exploit each algorithm again in conjunction with those data particularly of C4.5

As already explained all the above will be made possible via WEKA version 3.8 (Weka Dowonloading, n.d.)

Basic Algorithm Comparison

As already have explained, in this section 3 algorithms will be examined C4.5 for the construction of trees, Naive Bayes which implements the Bayes theorem and finally Multilayer Perceptron a representation of artificial neural networks. The comparison will be in 2 levels initially how accurate will be the algorithm and on the second level how fast it is. Additionally the comparison will be made with the default values using the Cross Validation with 10 folds. This test option is again the default.

C4.5

For C4.5 the default values are:
Confidence Factor: 0.25
Minimum Number of Objects: 2

A small explanation about these two terms seems necessary:
Confidence factor shows how much pruning the tree will take with smaller values indicating more pruning while bigger values result on less pruning. Confidence factor ranges from 0 to 1. On the next section it will be examined taking values from 0.1 up to 1. Sufficient to say that it has to do how confident is the algorithm that the error rate that is observed here is the actual error that will be observed elsewhere, (values near to 1), or not (values closer to 0).

Minimum Number of Object shows the minimum instances that exist on each leaf. It can be 2 or 20 or 200 or 2000 but NOT below 2. In the next section it will take values from 2 up to 2500 (a little more than half the number of the examined records)

Using the default values the results are as follow:

Correctly Classified Instances	1980	40,808%
Incorrectly Classified Instances	2872	59,192%
Kappa statistic	0,2676	

It took 6 seconds to produce results while the time to build the model was less than one second. While the two first rows are rather obvious the 3rd one, needs some explanation: Fundamentally it is the agreement between two independent and mutual exclusive raters either humans or machine learning algorithms. To elaborate further compares the **Observed Accuracy** with an **Expected Accuracy**. To put it simply an Expected Accuracy of 65% is much more impressive than an **Observed Accuracy** of 75%.More can be found on the usual suspect, Wikipedia, and even better explanation can be found on Stack Exchange here (http://stats.stackexchange.com/questions/82162/kappa-statistic-in-plain-english)

Naive Bayes

Due to the nature of the algorithm there are no default values

The results that were observed during the implementation of the algorithm are as follow:

Correctly Classified Instances	2204	45,425%
Incorrectly Classified Instances	2648	54,575%
Kappa Statistic	0,3376	

The duration of the execution was 1 second something which is expected given the fact that Naive Bayes on the core is nothing but simple calculations (mostly divisions) and moderns PC do not have any issues with those. What is

important here is that Naive Bayes was 5 % more successful on its prediction and it had been a better Kappa statistic of almost 7 points

Multilayer Perceptron

Again here we run with the default values which are:

-L 0.3: Learning Rate for the back-propagation algorithm. It controls the size of weight and bias changes in learning of the training algorithm . It takes the values between 0 and 1. The default is 0.3.

-M 0.2: Is the Momentum Rate again for the back-propagation algorithm. Very close to the previous and it shows the information that is transferred from each weight to the next weight

-N 500: Number of epochs or how many times the data (the entire dataset) will be passed from the entire neural network system

-V 0: It is the size of validation set as percentage to be used to terminate training. If this is different than 0 it can prevent the previous parameter (number of epochs) reaching its maximum

-S 0: The value used to seed the random number generator (Value should be >= 0 and a long, Default = 0)

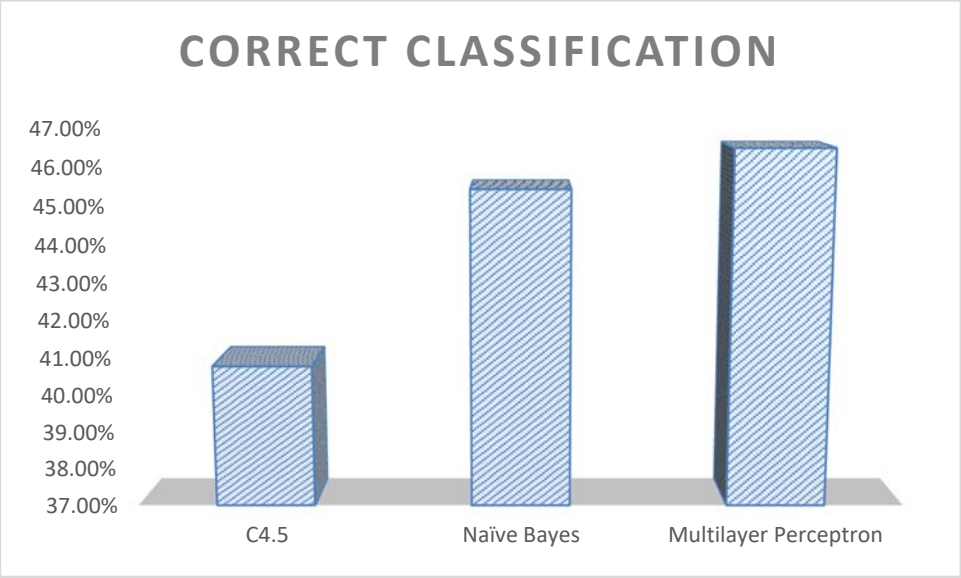
-E 20: The consecutive number of errors allowed for validation testing before the network terminates or in other words if the errors (continuously) surpass this threshold the algorithm will stop (Value should be > 0, Default = 20).

-H a: It shows how many nodes on the hidden layers will be created. It can be natural numbers (1, 2, 3 …) or predefined letters such as “a” (attributes and classes)/2, “I” (only attributes), or “o”(attributes + classes)

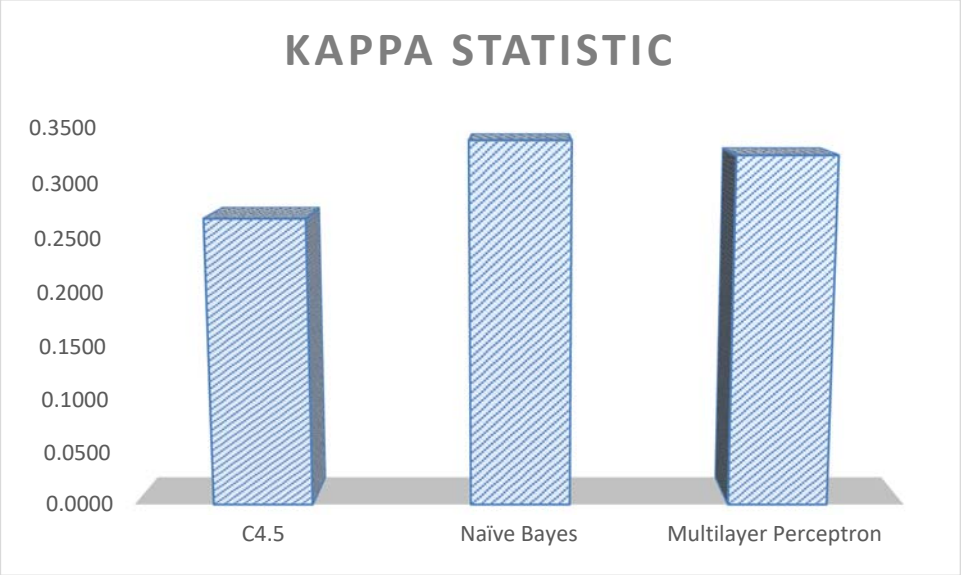
Again on the following table we can see the results from the execution of the following

Correctly Classified Instances	2255	46,476%
Incorrectly Classified Instances	2597	53,524%
Kappa statistics	0,3243	

In the case of Multilayer Perceptron it took 10 minutes (!!!) to finish the classification and it gave comparable results to the Naïve Bayes. The results are depicted on the following diagram.



Respectively for the Kappa Statistic:



From the above it is rather obvious that Naïve Bayes produced better results in conjunction with the time it took to finish the classification. On the next section the same algorithms will be examined on a machine with lower technical capabilities

Executing the same algorithms on a "lower" machine the following results have been observed:

1. For C4.5

Correctly Classified Instances	1980	40,81%
Incorrectly Classified Instances	2872	59,19%
Kappa statistic	0,2676	

And the time was 5 seconds. From the above it's obvious that the same exact behavior have been observed between the two machines

2. For Naïve Bayes

Correctly Classified Instances	2204	45,42%
Incorrectly Classified Instances	2648	54,58%
Kappa statistic	0,3376	

3. And Finally for Multilayer Perceptron

Correctly Classified Instances	2255	46,48%
Incorrectly Classified Instances	2597	53,52%
Kappa statistic	0,3243	

While the Multilayer Perceptron classification was close enough to other two methods almost identical to Naïve Bayes and less than 5% better off the C4.5, Multilayer Perceptron's time was disappointing compared to the other two classifications. This time it took 7 minutes to finish all the 10 folds in comparison to 1 second of Naïve Bayes and 5 seconds of C4.5

Advanced Algorithm Comparison

On the previous section we have examined the algorithm's performance in a very basic setup. In the next pages we will examine how changing various parameters on different levels (algorithm parameters, testing parameters, different data format) will affect the algorithm's performance and duration.

C4.5

In these series of test there we will consider the modification of two parameters:

1. Confidence factor C: It will take the following values {0.1, 0.2, 0.25, 0.38, 0.50, 0.75 and 1}. It should be reminded here that the lower the value the more the pruning it occurs
2. Minimum Number of the objects on tree's leafs M: It will take the values {2, 5, 10, 20, 50, 100, 250, 500, 1000, 2000}. Obviously here the lower the value the more branches and leafs the tree will have.

Initially the first parameter (C) will be increased keeping the second stable. As long as a full circle (from 0.1 to 1) of the first parameter has been completed, the second (M) will also be increased. Additionally to the previous the cross-validation method will take place with values {5, 10, 15, and 20} and there will be no training set as well as no percentage split.
There will be initially an extensive analysis of the methodology of the algorithm using the most fundamental setup with no prune and lowest minimum objects per leaf, one.

Unpruned Tree

The first test will be done by changing the algorithm parameters to unpruned tree (parameter –U) and minimum number of voters per leaf node to 1 (M1). In all the comparisons from now on, the test will be conducted on 3 testing options meaning Training Set, Cross-validation with 5, 10, 15, 20 folds and finally percentage split. A little explanation about each term is necessary at this point:

a) Use of training Set: In that option the training set that is used for building the tree is used for validation also. As it is obvious this setup can't be reliable, particularly with those algorithm parameters, since any deviation on the real data will eventually fail.
b) Cross validation: Under this testing option the data set will be split to 5, 10, 15, 20 subsets and from them 4, 9, 14, 19 respectively will be used for training and the one that is left outside will be used for the validation. An interesting choice here is the stratified cross-validation. In stratified cross-validation there is always a similar distribution of the class in each subset. In other words there will always be 26,1% of data that have as DI.SY their preferred selection and there will always be 18,8% of data that have as AKEL its preference. Similarly for the other classes
c) Percentage split. In that case the data are split into two parts. The bigger part (66%) is usually the training data and the rest (34%) is used for validation

Use Training set

Correctly Classified Instances	4.723	97,34%
Incorrectly Classified Instances	129	2,66%
Kappa statistic	0,9675	

The above results are simply too good to be true. There is no chance with this type of data to succeed such classification

Cross Validations with 5 folds

Correctly Classified Instances	1.762	36,31%
Incorrectly Classified Instances	3.090	63,69%
Kappa statistic	0,2253	

Cross Validations with 10 folds

Correctly Classified Instances	1.736	35,78%
Incorrectly Classified Instances	3.116	64,22%
Kappa statistic	0,2204	

Cross Validations with 15 folds

Correctly Classified Instances	1.739	35,84%
Incorrectly Classified Instances	3.113	64,16%
Kappa statistic	0,2219	

Cross Validations with 20 folds

Correctly Classified Instances	1.759	36,25%
Incorrectly Classified Instances	3.093	63,75%
Kappa statistic	0,2253	

Percentage split.

Correctly Classified Instances	439	26,50%
Incorrectly Classified Instances	1217	73,49%
Kappa statistic	0	

The reason for the number of Instances being equal to 1656 and not 4900 is because the percentage split counts only the validation data in other words 33% of the initial data

Results Discussion/Evaluation

The results on this setup is basically plagued by the dreaded phenomenon of overfitting giving pretty much the same classification (around 35 %) on Cross Validation and 25% on percentage split with 33% as validation data. Tinkering with the number of objects per leaf as well as the pruning eventually will produce better results.

Pruned Tree

As it has been already mentioned on the following series of tests there will be only cross validation as a testing method. It has been already proved that the option of using the training set as validation produces unreal results while with the percentage split the classification that is achieved is much lower (about 10 %). They were measurements in 3 dimensions and parameters:

1. How successful was the classification
2. How big was the resulting tree
3. How fast was the algorithm

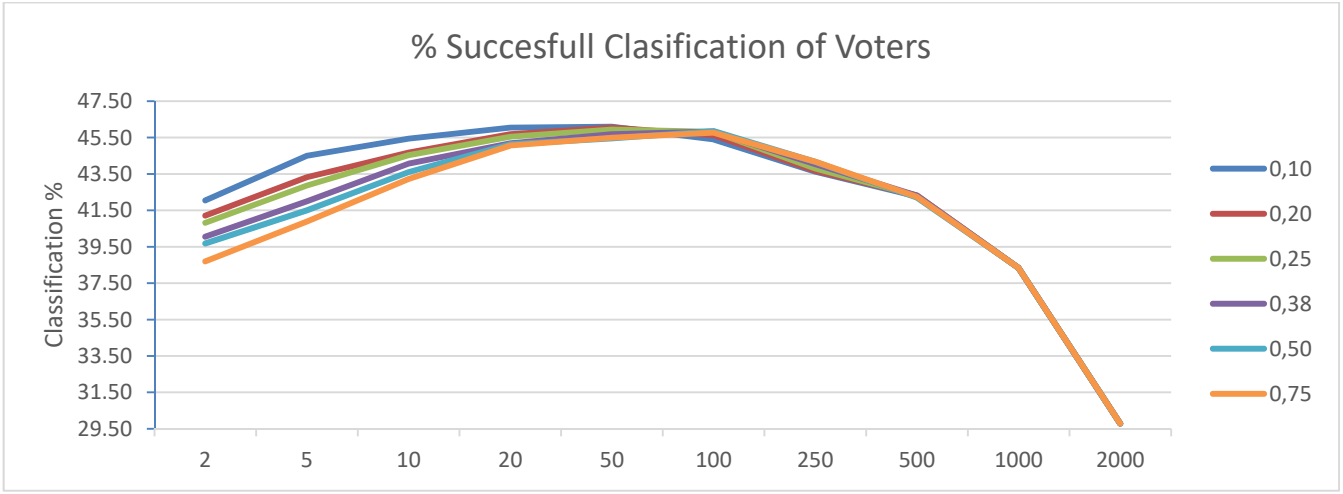
The two parameters, as has already mentioned, that were examined was:

1. The Minimum Number of Voters per Leaf
2. The Confidence Factor

In the following tables the results of the measurements are presented.

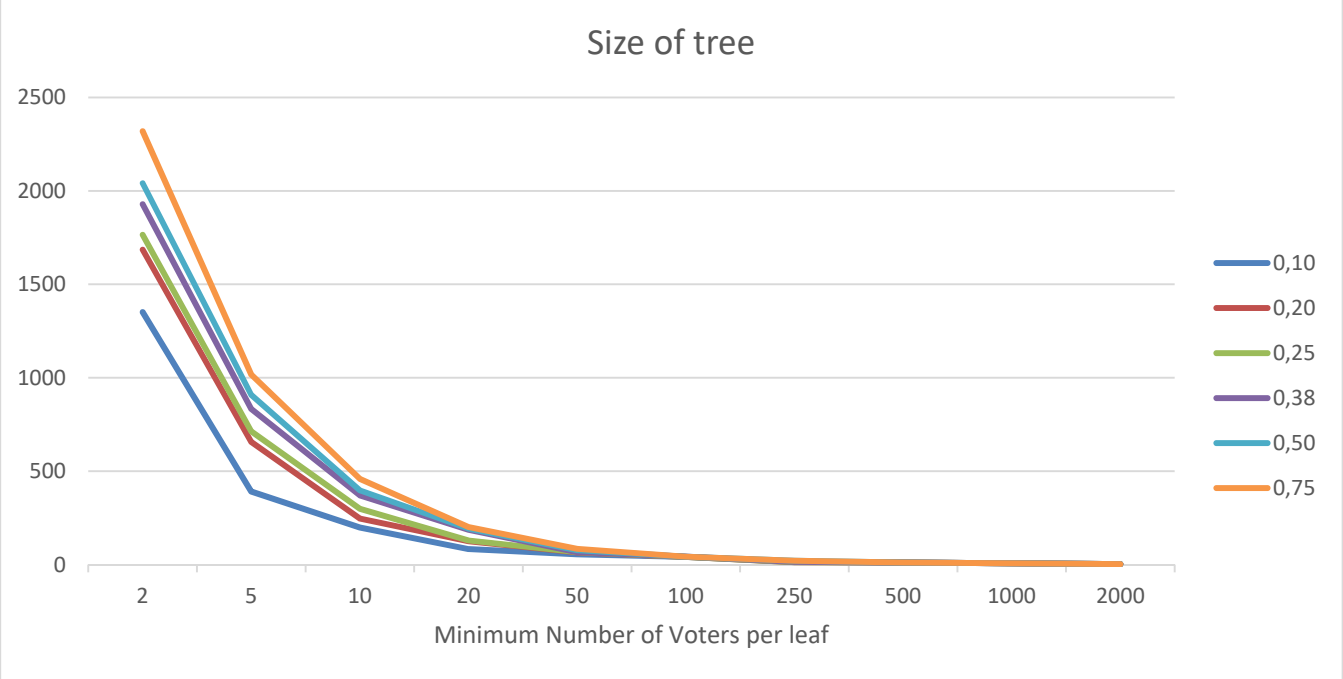
		Minimum Number of Voters per leaf									
		2	5	10	20	50	100	250	500	1000	2000
Confidence Factor	0,10	42,05	44,50	45,45	46,05	46,10	45,40	43,63	42,27	38,34	29,79
	0,20	41,22	43,32	44,68	45,68	46,06	45,63	43,66	42,32	38,34	29,79
	0,25	40,81	42,87	44,54	45,55	45,95	45,81	43,80	42,30	38,34	29,79
	0,38	40,05	41,99	44,07	45,19	45,69	45,83	44,05	42,31	38,34	29,79
	0,50	39,68	41,50	43,61	45,13	45,45	45,86	44,16	42,20	38,34	29,79
	0,75	38,70	40,89	43,23	45,07	45,49	45,77	44,18	42,24	38,34	29,79
	1,00	38,70	40,80	43,23	45,07	45,57	45,77	44,18	42,24	38,34	29,79

In the above table the success for C4.5 is presented. From a very first reading it is obvious that the best results are achieved when we have a small confidence factor 0.1 and number of voters per leaf is between 10 and 100 which is around 0.2% and 2% of the total data (4900). Another interesting observation is that when the confidence factor is low, better results are achieved when the number of customers is small and medium (up to 50 customers per node) as the number of customers moves toward bigger numbers (100 and 250) this phenomenon tends to mitigate having very small differences between small and big confidence factor. Generally speaking when the number of voters is small the phenomenon of overfitting is observed and as the number of voters is growing we are moving towards the phenomenon of underfitting passing from the area of "sweet spot" thus making the factor of voters per leaf (and in general the number of objects per leaf) the most important factor. A representation of previous results is depicted in the next graph.



		Minimum Number of Voters per leaf									
		2	5	10	20	50	100	250	500	1000	2000
Confidence Factor	0,10	1351	391	199	83	55	41	15	11	7	3
	0,20	1685	657	247	125	65	41	13	11	7	3
	0,25	1765	713	299	129	67	41	13	11	7	3
	0,38	1929	833	371	187	67	43	15	11	7	3
	0,50	2041	909	397	193	77	43	21	11	7	3
	0,75	2319	1017	459	201	85	43	21	11	7	3
	1,00	2319	1017	459	201	85	43	21	11	7	3

Similarly another aspect of the C4.5 is evaluated: The size of tree (leafs and intermediate nodes) varying from 3 when the minimum voters per node is equal to 2000 up to 2319 when the minimum voters per node is equal to 2000 and the confidence factor is equal to 1. Additionally to the above there is a huge variation between the size of the tree when the number of voter per leaf is minimal (2) between 1351 for CF = 0.1 and 2319 for CF = 1.0. Respectively there is no variation when the number of voters becomes maximum (2000). In that case the size of the tree is always equal to 3. In intermediate stages there is a dwindling of this variation as well as of the size of the tree (obviously). The results are presented again on the following diagram:



In the above diagram the results for the case where the confidence factor is equal to 1 have been omitted (similarly to previous diagram) since they are exactly the same as with value 0.75

Finally a generated tree from the Weka is presented on the next screen. This tree has been created with confidence factor equal to 0.38 and minimum number of customers per leaf equal to 100: